Models, To Model, and Modelling
-
Towards a Theory of Models, especially Conceptual Models and Modelling
-
Third Collection of Recent Papers (2018-2019)

Bernhard Thalheim

Christian-Albrechts-University Kiel, Department of Computer Science, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

A **model** *is a well-formed, adequate, and dependable instrument that represents origins*.

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its *community of practice* within some *context* and correspond to the *functions* that a model fulfills in *utilisation scenarios*.

As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Content of This Third Collection

Remark: See our previous work on the ***entity-relationship approach to modelling***, e.g. [ET11,Tha00,MST09b,ST15], on semantics [ST13,Tha11b], on programming, on the theory of databases and information systems and on the technology of information systems at
http://dblp.uni-trier.de/pers/hd/t/Thalheim:Bernhard
or https://www.researchgate.net/profile/Bernhard_Thalheim/publications or
https://scholar.google.com/citations?user=lkH3h9gAAAAJ or
http://independent.academia.edu/BernhardThalheim or ...

Content of The Second Collection

available through research gate and academia

Table of Content of the Second Collection

Remark: page numbers from the second collection in the sequel.

The First Collection

available through research gate and academia

Table of Content of the First Collection

Remark: page numbers from the first collection in the sequel.

References

[ADT12a] S. Amarakoon, A. Dahanayake, and B. Thalheim. Domain requirements modeling framework for cross-disciplinary service systems development. In *ICCNDT*, pages 152–166. Gulf University, Bahrain, 2012.

[ADT12b] S. Amarakoon, A. Dahanayake, and B. Thalheim. A framework for modelling medical diagnosis and decision support services. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, 2(4):7–26, 2012.

[AFT09] Sabah S. Al-Fedaghi and Bernhard Thalheim. Personal information databases. *IJCSIS*, 5(1):11–20, 2009.

[AGK+19] V.P.J. Arponen, S. Grimm, L. Käppel, K. Ott, B. Thalheim, Y. Kropp, K. Kittig, J. Brinkmann, and A. Ribeiro. Between natural and human sciences: On the role and character of theory in socio-environmental archaeology. *The Holocene*, 29(10, Special Issue "Scales of Transformation: Human-Environmental Interaction in Prehistoric and Archaic Societies"):tba, October 2019.

[BFA+16] M. Bichler, U. Frank, D. Avison, J. Malaurent, P. Fettke, D. Hovorka, J. Krämer, D. Schnurr, B. Müller, L. Suhl, and B. Thalheim. Theories in business and information systems engineering. *Business & Information Systems Engineering*, pages 1–29, 2016.

[BST09] E. Börger, O. Sörensen, and B. Thalheim. On defining the behavior of or-joins in business process models. *Journal of Universal Computer Science*, 15(1):3–32, 2009.

[BT12] P. Broman and B. Thalheim. Interactive data extraction from semi-structured text. In *Information Modelling and Knowledge Bases XXII*, volume 237 of *Frontiers in Artificial Intelligence and Applications*, pages 1–19. IOS Press, 2012.

[DT10a] A. Dahanayake and B. Thalheim. Co-evolution of (information) system models. In *EMMSAD 2010*, volume 50 of *LNBIP*, pages 314–326. Springer, 2010.

[DT10b] A. Dahanayake and B. Thalheim. Towards a framework for emergent modeling. In *ER Workshops*, volume 6413 of *Lecture Notes in Computer Science*, pages 128–137. Springer, 2010.

[DT11] A. Dahanayake and B. Thalheim. Enriching conceptual modelling practices through design science. In *BMMDS/EMMSAD*, volume 81 of *Lecture Notes in Business Information Processing*, pages 497–510. Springer, 2011.

[DT12] A. Dahanayake and B. Thalheim. A conceptual model for IT service systems. *Journal of Universal Computer Science*, 18(17):2452–2473, 2012.

[DT13a] A. Dahanayake and B. Thalheim. The conception of a service model. In *Proc. ICNVICT'2013*, pages 36–49, Amman, 2013. IEEE.

[DT13b] A. Dahanayake and B. Thalheim. Continuous database engineering. *Int. Journal Business Inf. Syst. (IJBIS)*, 11(4):26–58, 2013.

[DT15] A. Dahanayake and B. Thalheim. W*H: The conceptual model for services. In *Correct Software in Web Applications and Web Services*, Texts & Monographs in Symbolic Computation, pages 145–176, Wien, 2015. Springer.

[DT18] A. Dahanayake and B. Thalheim. Development of conceptual models and the knowledge background provided by the rigor cycle in design science. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 3–28. College Publications, 2018.

[ET11] D. Embley and B. Thalheim, editors. *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*. Springer, 2011.

[HFHT18] J. Hedtrich, E. Fabritz, C. Henning, and B. Thalheim. Digital playground for policy decision making. In *Proc. XX Int. conf., DAMDID-RCDL'18*, pages 248–254, 2018.

[JHMT19] H. Jaakkola, J. Henno, J. Mäkelä, and B. Thalheim. Artificial intelligence forever - again and again. In *Submitted to MiPRO*, page tba. IEEE, 2019.

[JHWD+16] H. Jaakkola, J. Henno, T. Welzer-Družovec, J. Mäkelä, and B. Thalheim. Why information systems modelling is so difficult. In *SQAMIA'2016*, pages 29–39, Budapest, 2016. CEUR Workshop Proceedings.

[JP10] K. Jannaschk and T. Polomski. A data mining design framework - A preview. In *Advances in Databases and Information Systems - 14th East European Conference, ADBIS 2010, Novi Sad, Serbia, September 20-24, 2010. Proceedings*, volume 6295 of *Lecture Notes in Computer Science*, pages 571–574. Springer, 2010.

[JT10] H. Jaakkola and B. Thalheim. Framework for high-quality software design and development: a systematic approach. *IET Software*, 4(2):105–118, 2010.

[JT11] H. Jaakkola and B. Thalheim. Architecture-driven modelling methodologies. In *Information Modelling and Knowledge Bases*, volume XXII, pages 97–116. IOS Press, 2011.

[JT18] H. Jaakkola and B. Thalheim. Modelling cultures. In Endrjukaite T., Jaakkola H., Thalheim B., and Kiyoki Y., editors, *Proc. 28'th EJC*, pages 33–52, Riga, Latvia, 2018. Transport and Telecommunication Institute.

[JT19a] H. Jaakkola and B. Thalheim. Cultures in information systems development. In *Information Modelling and Knowledge Bases XXX*, pages 61–80. IOS Press, 2019.

[JT19b] H. Jaakkola and B. Thalheim. Models as programs: The envisioned and principal key to true fifth generation programming. In *Proc. 29'th EJC*, pages 170–189, Lappeenranta, Finland, 2019. LUT, Finland.

[JT20] H. Jaakkola and B. Thalheim. Model-based fifth generation programming. In *Information Modelling and Knowledge Bases Vol. XXXI*, Frontiers in Artificial Intelligence and Applications, 312, pages 377–396. IOS Press, 2020.

[JTH+18] H. Jaakkola, B. Thalheim, J. Henno, J. Mäkelä, and H. Keto. Role of the user in information systems development. In *MiPRO*, pages 701–708. IEEE, 2018.

[KST09] M. Kirchberg, O. Sörensen, and B. Thalheim. A BPMN case study: Paper review and submission system. In *GI Jahrestagung*, volume 154 of *LNI*, pages 4067–4081. GI, 2009.

[KT17a] F. Kramer and B. Thalheim. Metadata as support for data provenance. In *Information Modelling and Knowledge Bases XXVIII*, Frontiers in Artificial Intelligence and Applications, 280, pages 195–214. IOS Press, 2017.

[KT17b] Y. Kropp and B. Thalheim. Data mining design and systematic modelling. In *Proc. DAMDID/RCDL'17*, pages 349–356, Moscov, 2017. FRC CSC RAS.

[KT18a] Y. Kropp and B. Thalheim. Viewpoint-oriented data management in collaborating research projects. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 146–174. College Publications, 2018.

[KT18b] Y. O. Kropp and B. Thalheim. Deep model guided data analysis. In *DAMDID/RCDL 2017, Revised Selected Papers*, volume 822 of *Communications in Computer and Information Science*, pages 3–18. Springer, 2018.

[KT19] Y. Kropp and B. Thalheim. Model-based interface generation. In *Proc. 29'th EJC*, page 70, Lappeenranta, Finland, 2019. LUT, Finland.

[KT20] Y. Kropp and B. Thalheim. Conceptual modelling and humanities. In *Proc. Modellierung*, LNI, page forthcoming. Springer, 2020.

[KTD+] Y. Kiyoki, B. Thalheim, M. Duzi, H.Jaakkola, P. Chawakitchareon, and A. Heimbürger. Towards a great design of conceptual modelling. pages 530–543.

[MST09a] Hui Ma, K.-D. Schewe, and B. Thalheim. Modelling and maintenance of very large database schemata using meta-structures. In *UNISCON*, volume 20 of *Lecture Notes in Business Information Processing*, pages 17–28. Springer, 2009.

[MST09b] Hui Ma, Klaus-Dieter Schewe, and Bernhard Thalheim. Geometrically enhanced conceptual modelling. In *ER*, volume 5829 of *Lecture Notes in Computer Science*, pages 219–233. Springer, 2009.

[MSTQ09] Hui Ma, Klaus-Dieter Schewe, Bernhard Thalheim, and Q.Wang. A service-oriented approach to web warehousing. In *iiWAS*. ACM, 2009.

[MSTW09a] H. Ma, K.-D. Schewe, B. Thalheim, and Q. Wang. A theory of data-intensive software services. *Service Oriented Computing and Applications*, 3(4):263–283, 2009.

[MSTW09b] H. Ma, K.-D. Schewe, B. Thalheim, and Q. Wang. A theory of data-intensive software services. *Service Oriented Computing and Applications*, 3(4):263–283, 2009.

[MSTW11] H. Ma, K.-D. Schewe, B. Thalheim, and Q. Wang. Cloud warehousing. *Journal of Universal Computer Science*, 17(8):1183–1201, 2011.

[MSTW12] Hui Ma, K.-D. Schewe, B. Thalheim, and Q. Wang. Conceptual modelling of services. *Journal of Universal Computer Science*, 18(17):2361–2363, 2012.

[MT19] A. Molnar and B. Thalheim. Usage models mapped to programs. In *Proc. M2P – New Trends in Database and Information Systems*, Bled, 2019. Springer, CCIS 1064.

[NKT] I. Nissen, F. Kramer, and B. Thalheim. Underwater cooperation and coordination of manned and unmanned platforms using S-BPM. In *Information Modelling and Knowledge Bases XXX,*, pages 137–146.

[NT14] R. Noack and B. Thalheim. Multi-dimensional schema composition for conceptual modelling in the large. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 25–44. IOS Press, 2014.

[NT15] I. Nissen and B. Thalheim. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter The Notion of a Model, pages 615–618. De Gryuter, Boston, 2015.

[ST10] M. Skusa and B. Thalheim. Kohärente Softwareentwicklung - Grundlagen, Arbeitsumgebungen, Vorgehensweisen. Preprint 1018, Department of Computer Science, Kiel University, December 2010.

[ST12] E. Sivogolovko and B. Thalheim. Semantic approach to cluster validity notion. In *ADBIS (2)*, volume 186 of *Advances in Intelligent Systems and Computing*, pages 229–239. Springer, 2012.

[ST13] O. Sörensen and B. Thalheim. Semantics and pragmatics of integrity constraints. In *SDKB'11, LNCS 7693*, pages 1–17. Springer, 2013.

[ST15] K.-D. Schewe and B. Thalheim. Co-design of web information systems. Texts & Monographs in Symbolic Computation, pages 293–332, Wien, 2015. Springer.

[ST17] V. Storey and B. Thalheim. Conceptual modeling: Enhancement through semiotics. In *Proc. ER'17*, LNCS, 10650, pages 182–190, Cham, 2017. Springer.

[ST19] K.-D. Schewe and B. Thalheim. *Design and development of web information systems*. Springer, Chur, 2019.

[TD15] B. Thalheim and A. Dahanayake. A conceptual model for services. In *Invited Keynote, CMS 2015, ER 2015 workshop*, LNCS 9382, pages 51–61, Berlin, 2015. Springer.

[TD16] B. Thalheim and A. Dahanayake. Comprehending a service by informative models. *T. Large-Scale Data- and Knowledge-Centered Systems*, 30:87–108, 2016.

[TELT14] S. Torge, W. Esswein, S. Lehrmann, and B. Thalheim. Categories for description of reference models. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 229–240. IOS Press, 2014.

[TFTL+14] M. Tropmann-Frick, B. Thalheim, D. Leber, G. Czech, and C. Liehr. Generic workflows - a utility to govern disastrous situations. In *Information Modelling and Knowledge Bases*, volume XXVI of *Frontiers in Artificial Intelligence and Applications, 272*, pages 417–428. IOS Press, 2014.

[Tha00] B. Thalheim. *Entity-relationship modeling – Foundations of database technology*. Springer, Berlin, 2000.

[Tha09a] B. Thalheim. Abstraction. In *Encyclopedia of Database Systems*, pages 6–7. Springer US, 2009.

[Tha09b] B. Thalheim. Extended entity-relationship model. In *Encyclopedia of Database Systems*, pages 1083–1091. Springer US, 2009.

[Tha09c] B. Thalheim. Specialization and generalization. In *Encyclopedia of Database Systems*, pages 2746–2747. Springer US, 2009.

[Tha09d] B. Thalheim. Towards a theory of conceptual modelling. In *ER Workshops*, volume 5833 of *Lecture Notes in Computer Science*, pages 45–54. Springer, 2009.

[Tha10a] B. Thalheim. Model suites for multi-layered database modelling. In *Information Modelling and Knowledge Bases XXI*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 116–134. IOS Press, 2010.

[Tha10b] B. Thalheim. Towards a theory of conceptual modelling. *Journal of Universal Computer Science*, 16(20):3102–3137, 2010. http://www.jucs.org/jucs_16_20/towards_a_theory_of.

[Tha11a] B. Thalheim. *Anwendungsorientierte Organisationsgestaltung*, chapter The Culture and Art of Conceptual Modelling, pages 127–144. baar, Hamburg, 2011.

[Tha11b] B. Thalheim. Integrity constraints in (conceptual) database models. In *The Evolution of Conceptual Modeling*, volume 6520 of *Lecture Notes in Computer Science*, pages 42–67, Berlin, 2011. Springer.

[Tha11c] B. Thalheim. The science of conceptual modelling. In *Proc. DEXA 2011*, volume 6860 of *LNCS*, pages 12–26, Berlin, 2011. Springer.

[Tha11d] B. Thalheim. The theory of conceptual models, the theory of conceptual modelling and foundations of conceptual modelling. In *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*, chapter 17, pages 547–580. Springer, Berlin, 2011.

[Tha12a] B. Thalheim. The art of conceptual modelling. In *Information Modelling and Knowledge Bases XXII*, volume 237 of *Frontiers in Artificial Intelligence and Applications*, pages 149–168. IOS Press, 2012.

[Tha12b] B. Thalheim. The notion of a conceptual model. In N. Seyff and A. Koziolek, editors, *Modelling and Quality in Requirements Engineering*, pages 21–30. Verlagshaus Monsenstein und Vannerdat, 2012. ISBN 978-3-86991-740-5.

[Tha12c] B. Thalheim. The science and art of conceptual modelling. In A. Hameurlain et al., editor, *TLDKS VI*, LNCS 7600, pages 76–105. Springer, Heidelberg, 2012.

[Tha12d] B. Thalheim. Syntax, semantics and pragmatics of conceptual modelling. In *NLDB*, volume 7337 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2012.

[Tha13a] B. Thalheim. The definition of the (conceptual) model. In *Proc. EJC 2013*, pages 256–269, Nara, Japan, 2013.

[Tha13b] B. Thalheim. Open problems of information systems research and technology. In *Invited Keynote, BIR'2013. LNBIB 158*, pages 10–18. Springer, 2013.

[Tha14] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[Tha15] B. Thalheim. Das Modell des Modelles. *Erwägen-Wissen-Ethik*, EWE-Heft, Heft 3, 2015, 26. Jg.:407–409, 2015.

[Tha17a] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[Tha17b] B. Thalheim. Model-based engineering for database system development. In *Conceptual Modeling Perspectives*, pages 137–153, Cham, 2017. Springer.

[Tha18a] B. Thalheim. Conceptual model notions - a matter of controversy; conceptual modelling and its lacunas. *EMISA International Journal on Conceptual Modeling*, February:9–27, 2018.

[Tha18b] B. Thalheim. Model adequacy. In *Joint Proceedings of the Workshops at Modellierung 2018 co-located with Modellierung 2018, Braunschweig, Germany, February 21, 2018.*, volume 2060 of *CEUR Workshop Proceedings*, pages 11–18. CEUR-WS.org, 2018.

[Tha18c] B. Thalheim. Normal models and their modelling matrix. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 44–72. College Publications, 2018.

[Tha18d] B. Thalheim. Qualitative and quantitative models. In *Proc. XX Int. conf., DAMDID-RCDL'18*, pages 112–119, 2018.

[Tha19a] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. Springer US, 2019.

[Tha19b] B. Thalheim. Conceptual models and their foundations. In *Proc. MEDI2019, LNCS 11815*, pages 123–139. Springer, 2019.

[Tha19c] B. Thalheim. Models and their foundational framework. *Studia Metodologiczne*, 2019.

[Tha19d] B. Thalheim. Models for communication, understanding, search, and analysis. In *Proc. XXI (DAMDID/RCDL 2019), CEUR Workshop Proceedings, vol. 2523, Kazan, Russia, October 15-18, 2019.*, pages 19–34, 2019.

[TJ19] B. Thalheim and H. Jaakkola. Models and their functions. In *Proc. 29'th EJC*, pages 150–169, Lappeenranta, Finland, 2019. LUT, Finland.

[TJ20] B. Thalheim and H. Jaakkola. Functions of models and their maturity. In *Information Modelling and Knowledge Bases Vol. XXXI*, Frontiers in Artificial Intelligence and Applications, 312, pages 265–284. IOS Press, 2020.

[TN15] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[TSF19] B. Thalheim, A. Sotnikov, and I. Fiodorov. Models: The main tool of true fifth generation programming. In *Proc. EEKM 2019 – Enterprise Engineering and Knowledge Management*, pages 161–170, Moscov, 2019. CEUR workshop poceedings, Vol. 2413.

[TT13] M. Tropmann and B. Thalheim. Mini story composition for generic workflows in support of disaster management. In *DEXA 2013*, pages 36–40. IEEE Computer Society, 2013.

[TTF15] B. Thalheim and M. Tropmann-Frick. The conception of the conceptual database model. In *ER 2015*, LNCS 9381, pages 603–611, Berlin, 2015. Springer.

[TTF16a] B. Thalheim and M. Tropmann-Frick. Model capsules for research and engineering networks. In *New Trends in Databases and Information Systems*, volume 637 of *CCIS*, pages 189–200. Springer, 2016.

[TTF16b] B. Thalheim and M. Tropmann-Frick. Models and their capability. In C. Beierle, G. Brewka, and M. Thimm, editors, *Computational Models of Rationality*, volume 29 of *College Publications Series*, pages 34–56. College Publications, 2016.

[TTF16c] B. Thalheim and M. Tropmann-Frick. Wherefore models are used and accepted? The model functions as a quality instrument in utilisation scenarios. In I. Comyn-Wattiau, C. du Mouza, and N. Prat, editors, *Ingénierie Management des Systèmes d'Information*, pages 131–143. Cépaduès, 2016.

[TTFZ14] B. Thalheim, M. Tropmann-Frick, and T. Ziebermayr. Application of generic workflows for disaster management. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 64–81. IOS Press, 2014.

Model Adequacy

Bernhard Thalheim
Christian-Albrechts University at Kiel, Department of Computer Science, D-24098 Kiel

December 28, 2017

Abstract

Models, modeling languages, modeling frameworks and their background have dominated research on information systems engineering for last four decades. Models are mainly used as mediators between the application world and the implementation or system world. Modelling is still conducted as the work of an artisan and workmanship. While a general notion of the model and of the conceptual model has already been developed, the modelling process is not investigated so well.

Modelling has to be based on principles and a general theory of modelling activities. One of the lacunas is still a proper understanding of adequacy of models, adequacy of modelling and deployment methods, and a theory of adequacy. We will concentrate on the first issue.

Keywords: model notion; model adequacy; analogy; focus/truncation/abstraction; purposeful; well-formed model; model dependability

1 Models, Modelling Activities, Systematic Modelling

Models are principle and central *instruments* in mathematics, data analysis, modern computer engineering (CE), in teaching any kind of computer technology, and also modern computer science (CS). They are built, applied, revised and manufactured in many CE&CS sub-disciplines in a large variety of application cases with different purposes and context for different communities of practice. CE&CS expressively use the conception of model for daily work. Modelling is one of their four central paradigms beside structures (in the small and large), evolution or transformation (in the small and large), and collaboration (based on communication, cooperation, and coordination). It is now well understood that models are something different from theories. They are often intuitive, visualisable, and ideally capture the essence of an understanding within some community of practice and some context. At the same time, they are limited in scope, context and the applicability. Models have been considered to be somewhere in the middle between the perception and understanding of the state of affairs (world, situations, data etc.) and theories (concepts and conceptions, statements, beliefs, etc.) since they may describe certain aspects of a situation and may represent parts of a theory. Models should thus be considered to be the third dimension of science [2, 50, 52][1]. Other disciplines (see for instance [50]) have developed a different understanding of the notion of model, of the function of models in scientific research and of the purpose of the model. Models are often considered to be artifacts where also virtual models are considered beside real one. Models might also be mental models and thought concepts. Models are used as instruments in *utilisation scenarios*. They function in these scenarios.

2 The Notion of the Model

There is however a general notion of a model and of a conception of the model:

A **model** *is a well-formed, adequate, and dependable instrument that represents origins.* (see [8, 45, 47])

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

[1] The title of the book [4] has inspired this observation.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*.

Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics (see [40]) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

3 Adequacy as a Generalisation of Mapping, Truncation, and Pragmatic Properties

Following H. Stachowiak (see, for instance, [33, 34]), a model is often defined in a phenomenalistic way based on three properties:

(1) *Mapping* property: the model has an origin and can be based on a mapping from the origin to the instrument.

(2) *Truncation* (*reduction*) property: the model lacks some of the ascriptions made to the origin.

(3) *Pragmatic* property: the model use is only justified for particular model users, the tools of investigation, and the period of time.

We observe however that these properties do not qualify a representation as a model. The mapping and truncation properties are far too strict and need further investigation. A model must not be a mapping from some origin. Homomorphism is a nice property but far too strict in most applications. We might use representations that are not images of mappings such as a Turing machine, a system architecture, or development strategies. Furthermore, we might use representations that are not reducts of origins such as (conceptual) information system models for the variety of viewpoints users of databases might have. Truncation (or abstraction) considers a model to be an Aristotelian one by abstraction by disregarding the irrelevant. The relevance criterion is based on the purpose (or goal or function) of a model. So, truncation is far too fuzzy. Models are developed by a community of practice for utilisation by a community of practice and in a context. The utilisation depends on the intentions of users and their context. So, we observe that the utilisation of models determines (a) the kind of model, (b) the governing purposes or goals of utilisation of the model, (c) the properties of a model, (d) the amplification a model provides with extensions, (e) the idealisation by scoping the model to the ideal state of affairs, (f) the divergence by deliberately diverging from reality in order to simplify salient properties of interest, and (g) the added value of a model. The seven additional statements are combined in the *mission* a model has. The mission clarifies how the model functions well within its intended scenarios of usage according to its capacity and potential. The mission must be coherent with the context, the determination or specific basis of conduct or utilisation of the model, and must be acceptable for the users or – more concrete – the community of practice. Therefore, the mission clarifies the functions (and anti-functions or forbidden ones), purposes and goals of the utilisation, the potential and the capacity of the model.

4 An Agenda: Towards Adequacy of Modelling Methods

The theory of modelling is still struggling with a number of research challenges (see [40]): Adjustable selection of principles depending on modelling goals; model suites with explicit model association; development of a language culture; models 2.0; explicit treatment of model value; coexistence of theory, languages, and tools; adequate representation variants of models; compiler development for models; model families and variants. These challenges are the background behind the consternation that has been summarised at Modellierung 208 by W. Hesse (see also [11, 12]): ... but they do not know what they do ...; Babylonian language confusion and muddle; “it’s not a bug, it’s a feature” and other statements for de-facto-standards and lobbyists; why I should cope with what was the state of art yesterday; each day a new wheel, new buzzwords without any sense, and a new trend; without consideration of the value of the model; competition is a feature, inhomogeneity; Laokoon forever; dreams about a sound mathematical foundation; take but don’t think - take it only without critics; academia in the ivory tower without executable models; where is the Ariadne thread through.

This consternation and the challenges can be summarised by a research agenda, e.g. with the following problems:

- *Can be develop a simple notion of adequateness that still covers the approaches we are used in our subdiscipline?*
- *Do we need this broad coverage for models? Or is there any specific treatment of dependability for subdisciplines or specific deployment scenarios?*
- *Which modelling methods are purposeful within which setting?*
- *Which model deployment methods are properly supporting the function of a model within a utilisation scenario?*
- *How does the given notion of model match with other understandings and approaches to modelling in computer science and engineering?*
- *What is the background of modelling, especially the basis that can be changed depending on the function that a model plays in some utilisation scenario?*
- *Language matters, enables, restricts and biases (see [54]). What is the role of languages in modelling?*
- *Which modelling context results in which modelling approach?*
- *What is the difference between the modelling process that is performed in daily practice and systematic and well-founded modelling?*
- *Are we really modelling reality or are we only modelling our perception and our agreement about reality?*
- *What is the influence of the modeller's community and schools of thought?*

5 The Storyline for this Keynote

In this keynote we discuss mainly the first element of the research agenda: adequateness of models, modelling methods, and modelling as a systematic activity. So far, the adequateness notion is far too fuzzy and too wide. The keynote is based on a large body of knowledge developed on models, modelling activities, and systematic modelling[2] The basis of our understanding of adequacy and dependability is the case study in the Kiel compendium of models, modelling activities and systematic modelling (see [50]). This MMM approach to modelling has been investigated for models in agriculture, archaeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports.

The introduction is based on a discussion of adequacy for two modelling methods widely used in our area. The specific utility of models follow the line given in [19, 20]. We are going to introduce a general and formal notion of adequacy. Since adequacy cannot be separated from dependability we have also to investigate it for the two modelling methods. Finally, the keynote ends with a collection of open problems on adequacy of modelling methods.

References

[1] C. Batini, S. Ceri, and S. Navathe. *Conceptual database design (an entity-relationship approach)*. Benjamin/Cummings, Redwood City, 1992.

[2] M. Bichler, U. Frank, D. Avison, J. Malaurent, P. Fettke, D. Hovorka, J. Krämer, D. Schnurr, B. Müller, L. Suhl, and B Thalheim. Theories in business and information systems engineering. *Business & Information Systems Engineering*, pages 1–29, 2016.

[3] M. Bjekovic, H. A. Proper, and J.-S. Sottet. Embracing pragmatics. In *Proc. ER 2014*, volume 8824 of *Lecture Notes in Computer Science*, pages 431–444. Springer, 2014.

[4] S. Chadarevian and N. Hopwood, editors. *Models - The third dimension of science*. Stanford University Press, Stanford, California, 2004.

[5] P.P. Chen, J. Akoka, H. Kangassalo, and B. Thalheim, editors. *Conceptual Modeling, Current Issues and Future Directions, Selected Papers from the Symposium on Conceptual Modeling 1997*, volume 1565 of *Lecture Notes in Computer Science*. Springer, 1999.

[6] A. Dahanayake and B. Thalheim. W*h: The conceptual model for services. In *Correct Software in Web Applications and Web Services*, Texts & Monographs in Symbolic Computation, pages 145–176, Wien, 2015. Springer.

[7] F. de Saussure. *Cours de Linguistique Générale*. Payot, 1995.

[8] D. Embley and B. Thalheim, editors. *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*. Springer, 2011.

[9] U. Frank and S. Strecker. Open Reference Models - Community-driven collaboration to promote development and dissemination of reference models. *Enterprise Modelling and Information Systems Architectures*, 2(2):32–41, 2007.

[2] For details and classical database design books we refer to [1, 5, 17, 21, 22, 26, 31, 37, 38].
For details on language theory we refer to [3, 7, 18, 27, 28, 36, 43, 56].
For details of design science research we refer to [13, 15, 30, 55].
Formalisation also includes approaches to a general theory of modelling such as [9, 10, 16, 23, 24, 25, 29, 32, 35, 57].
For details of our work we refer to [2, 6, 8, 14, 39, 41, 42, 43, 44, 45, 46, 48, 49, 51, 52, 53].

[10] I.A. Halloun. *Modeling Theory in Science Education*. Springer, Berlin, 2006.

[11] W. Hesse. Modelle - Janusköpfe der Software-Entwicklung - oder: Mit Janus von der A- zur S-Klasse. In *Modellierung 2006*, volume 82 of *LNI*, pages 99–113. GI, 2006.

[12] W. Hesse and H. C. Mayr. Modellierung in der Softwaretechnik: eine Bestandsaufnahme. *Informatik Spektrum*, 31(5):377–393, 2008.

[13] A. Hevner, S. March, J. Park, and S. Ram. Design science in information systems research. *MIS Quaterly*, 28(1):75–105, 2004.

[14] H. Jaakkola, J. Henno, T. Welzer-Družovec, J. Mäkelä, and B. Thalheim. Why information systems modelling is so difficult. In *SQAMIA'2016*, pages 29–39, Budapest, 2016. CEUR Workshop Proceedings.

[15] P. Johannesson and E. Perjons. *An introduction to design science*. Springer, Cham, 2014.

[16] R. Kaschek. *Konzeptionelle Modellierung*. PhD thesis, University Klagenfurt, 2003. Habilitationsschrift.

[17] M. Klettke and B. Thalheim. Evolution and migration of information systems. In *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*, chapter 12, pages 381–420. Springer, Berlin, 2011.

[18] B. Kralemann and C. Lattmann. Models as icons: modeling models in the semiotic framework of peirce's theory of signs. *Synthese*, 190(16):3397–3420, 2013.

[19] F. Kramer and B. Thalheim. Holistic conceptual and logical database structure modelling with adoxx. In D. Karagiannis, H.C. Mayr, and J. Mylopoulos, editors, *Domain-specific conceptual model*, pages 269–290, Cham, 2016. Springer.

[20] Y. Kropp and B. Thalheim. Data mining design and systematic modelling. In *Proc. DAMDID/RCDL'17*, pages 349–356, Moscov, 2017. FRC CSC RAS.

[21] P. Lockemann and J. W. Schmidt. *Datenbankhandbuch*. Springer, 1987.

[22] P.C. Lockemann and H.C. Mayr. *Rechnergestützte Informationssysteme*. Springer, 1978.

[23] P. Loos, P. Fettke, B. E. Weißenberger, S. Zelewski, A. Heinzl, U. Frank, and J. Iivari. Welche Rolle spielen eigentlich stilisierte Fakten in der Grundlagenforschung der Wirtschaftsinformatik? *Wirtschaftsinformatik*, 53(2):109–121, 2011.

[24] B. Mahr. On judgements and propositions. *ECEASST*, 26, 2010.

[25] B. Mahr. Modelle und ihre Befragbarkeit - Grundlagen einer allgemeinen Modelltheorie. *Erwägen-Wissen-Ethik (EWE)*, Vol. 26, Issue 3:329–342, 2015.

[26] H. Mannila and K.-J. Räihä. *The design of relational databases*. Addison-Wesley, Wokingham, England, 1992.

[27] S. Overbeek, U. Frank, and C. Köhling. A language for multi-perspective goal modelling: Challenges, requirements and solutions. *Computer Standards & Interfaces*, 38:1–16, 2015.

[28] C.S. Peirce. What is a sign? In Peirce Edition Project, editor, *The essential Peirce: selected philosophical writings*, volume 2, pages 4 – 10. Indiana University Press, Bloomington, Indiana, 1998.

[29] A. Rutherford. *Mathematical Modelling Techniques*. Dover publications, 1995.

[30] H.A. Simon. *The science of the artificial*. MIT Press, Cambridge, 1996.

[31] G. Simsion. *Data modeling essentials - Analysis, design and innovation*. Van Nonstrand Reinhold, New York, 1994.

[32] B.Ja. Sovetov and S.A. Jakovlev. *Systems Modelling*. Vysschaja Schkola, 2005. In Russian.

[33] H. Stachowiak. *Allgemeine Modelltheorie*. Springer, 1973.

[34] H. Stachowiak. Modell. In Helmut Seiffert and Gerard Radnitzky, editors, *Handlexikon zur Wissenschaftstheorie*, pages 219–222. Deutscher Taschenbuch Verlag GmbH & Co. KG, München, 1992.

[35] W. Steinmüller. *Informationstechnologie und Gesellschaft: Einführung in die Angewandte Informatik*. Wissenschaftliche Buchgesellschaft, Darmstadt, 1993.

[36] V. Storey and B. Thalheim. Conceptual modeling: Enhancement through semiotics. In *Proc. ER'17*, LNCS, 10650, page forthcoming, Cham, 2017. Springer.

[37] T. J. Teorey. *Database modeling and design: The entity-relationship approach*. Morgan Kaufmann, San Mateo, 1989.

[38] B. Thalheim. *Entity-relationship modeling – Foundations of database technology*. Springer, Berlin, 2000.

[39] B. Thalheim. *The Conceptual Framework to Multi-Layered Database Modelling based on Model Suites*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 116–134. IOS Press, 2010.

[40] B. Thalheim. Towards a theory of conceptual modelling. *Journal of Universal Computer Science*, 16(20):3102–3137, 2010. http://www.jucs.org/jucs_16_20/towards_a_theory_of.

[41] B. Thalheim. The art of conceptual modelling. In *Information Modelling and Knowledge Bases XXII*, volume 237 of *Frontiers in Artificial Intelligence and Applications*, pages 149–168. IOS Press, 2012.

[42] B. Thalheim. The science and art of conceptual modelling. In A. Hameurlain et al., editor, *TLDKS VI*, LNCS 7600, pages 76–105. Springer, Heidelberg, 2012.

[43] B. Thalheim. Syntax, semantics and pragmatics of conceptual modelling. In *NLDB*, volume 7337 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2012.

[44] B. Thalheim. The conception of the model. In *BIS*, volume 157 of *Lecture Notes in Business Information Processing*, pages 113–124. Springer, 2013.

[45] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[46] B. Thalheim. Das Modell des Modelles. *Erwägen-Wissen-Ethik*, EWE-Heft, Heft 3, 2015, 26. Jg.:407–409, 2015.

[47] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. 2017.

[48] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[49] B. Thalheim and A. Dahanayake. Comprehending a service by informative models. In *Conceptual Modeling of Services*, LNCS 10130, pages 87–108, Berlin, 2016. Springer.

[50] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[51] B. Thalheim and M. Tropmann-Frick. The conception of the conceptual database model. In *ER 2015*, LNCS 9381, pages 603–611, Berlin, 2015. Springer.

[52] B. Thalheim and M. Tropmann-Frick. Models and their capability. In C. Beierle, G. Brewka, and M. Thimm, editors, *Computational Models of Rationality*, volume 29 of *College Publications Series*, pages 34–56. College Publications, 2016.

[53] B. Thalheim and M. Tropmann-Frick. Wherefore models are used and accepted? The model functions as a quality instrument in utilisation scenarios. In I. Comyn-Wattiau, C. du Mouza, and N. Prat, editors, *Ingénierie Management des Systèmes d'Information*, pages 131–143. Cépaduès, 2016.

[54] B.L. Whorf. *Lost generation theories of mind, language, and religion*. Popular Culture Association, University Microfilms International, Ann Arbor, Mich., 1980.

[55] R. Wieringa. *Design science methodology for information systems and software engineering*. Springer, Heidelberg, 2014.

[56] Z. Zarwin, M. Bjekovic, J.-M. Favre, J.-S.. Sottet, and H. A. Proper. Natural modelling. *Journal of Object Technology*, 13(3):4: 1–36, 2014.

[57] S. Zelewski. Kann Wissenschaftstheorie behilflich für die Publikationspraxis sein? In F. Lehner and S. Zelewski, editors, *Wissenschaftstheoretische Fundierung und wissenschaftliche Orientierung der Wirtschaftsinformatik*, pages 71–120. GTO, 2007.

Conceptual Model Notions - A Matter of Controversy

Conceptual Modelling and its Lacunas

Bernhard Thalheim
Christian-Albrechts University at Kiel, Department of Computer Science, D-24098 Kiel

December 10, 2017

Abstract

The conception of a conceptual model is differently defined in Computer Science and Engineering as well as in other sciences. There is no common notion of this conception yet. The same is valid for the understanding of the notion of model. One notion is: *A model is a well-formed, adequate, and dependable instrument that represents origins and functions in some utilisation scenario.* The *conceptual model of an information system* consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema. In a nutshell, a *conceptual model* is an enhancement of a model by concepts from a concept(ion) space.
The variety of notions for conceptual model is rather broad. We analyse some of the notions, systematise these notions, and discuss essential ingredients of conceptual models. This discussion allows to derive a research program in our area.

Keywords: Model, Conceptual model, Concept and notion of a model, Art of modelling.

1 What is a Conceptual Model

Modelling is a topic that has already been in the center of research in computer engineering and computer science since its beginnings. It is an old subdiscipline of most natural sciences with a history of more than 2.500 years. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope. Meanwhile it became a branch in the Philosophy of Science. The number of papers devoted to modelling doubles each year since the early 2000's.

It is often claimed that there cannot be a common notion of model that can be used in sciences, engineering, and daily life. The following notion covers all known so far notions in agriculture, archaeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments that are deployed in certain scenarios (see [39]). A commonly acceptable statement for a general model notion is the following one[1]:
A model is a *well-formed*, *adequate*, and *dependable* instrument that represents *origins* and *functions* in some utilisation *scenario*. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its *community of practice* within some *context* and correspond to the *functions* that a model fulfills in *utilisation scenarios*. The function determines the purposes and goals.

CS-conceptual modelling[2] is often related back to the introduction of the entity-relationship model(ling language) for information systems development. It surprises nowadays that there is no commonly accepted notion of conceptual model yet. There have been several trials but none of them was sufficient and was able to cover the idea of the conceptual model.

The database and information systems research communities are extensively using the term "conceptual model"[3]. The notion of conceptual model still needs some clarification: what is a conceptual model and what not; which application scenario use which kind of conceptual model; is conceptual modelling only database modelling; do we need to have an understanding of modelling; is a conceptual database model only a reflection of a logical database model; is a conceptual model a model or not; etc. Let us illustrate the wide spread and understanding of conceptual models, the activity of conceptual modelling, and the modelling as a scientific and engineering process by some examples[4,5]:

Reality and world description: Conceptual modelling is the activity of formally describing some aspects of

[1] We refer to the model-to_model-modelling compendium (see [39]) for notions that are not introduced in this paper.

[2] In the paper we restrict ourselves to this kind of conceptual model and thus omit the CS acronym. In general, a conceptual model is a representation of a system in its widest sense on the basis of concept(ion)s that allow people to consciously act and being guided in certain situations of their systems.

[3] Facetted search for the term "conceptual model" in DBLP results in more than 5.000 hits for titles in papers (normal DBLP search also above 3.400 titles).

[4] The notion of conceptualisation, conceptual models, and concepts are far older than considered in computer science. The earliest contribution to models and conceptualisations we are aware of is pre-socratic philosophy.

[5] Wikiquote (see [44]) lists almost 40 notions. We add our list to this list.

EMISA International Journal on Conceptual Modeling, Febr. 2018, 9-27

the physical and social world around us for purposes of understanding and communication. Such descriptions, often referred as conceptual schemata, require the adoption of a formal notation, a conceptual model in our termininology[6]. (see [25])

Community description : Conceptual modeling is about describing the semantics of software applications at a high level of abstraction[7].
Specifically, conceptual modelers (1) describe structure models in terms of entities, relationships, and constraints; (2) describe behavior or functional models in terms of states, transitions among states, and actions performed in states and transitions; and (3) describe interactions and user interfaces in terms of messages sent and received and information exchanged. In their typical usage, conceptual-model diagrams are high-level abstractions that enable clients and analysts to understand one another, enable analysts to communicate successfully with application programmers, and in some cases automatically generate (parts of) the software application. (see [12])

Conceptual database modelling : A data model is a collection of concepts that can be used to describe a set of data and operations to manipulate the data. When a data model describes a set of concepts from a given reality, we call it a conceptual model. (see [2, 10][8])

Instance-integrating conceptual modelling: A conceptual model consists of a conceptual schema and an information base. A conceptual schema provides a language for reasoning about an object system, and it specifies rules for the structure and the behaviour of the system. A description of a particular state is given in an information base, which is a set of type and attribute statements expressed in the language of the conceptual schema. (see [4])

System-representation models: A conceptual model is a descriptive model of a system based on qualitative assumptions about its elements, their interrelationships, and system boundaries. (see [7])

Representational models: A conceptual model is a type of diagram which shows of a set of relationships between factors that are believed to impact or lead to a target condition; a diagram that defines theoretical entities, objects, or conditions of a system and the relationships between them. (see [8])

Enterprise modelling and conceptual modelling : A conceptual is a model which represents a conceptual understanding (i.e. conceptualisation) of some domain for a particular purpose. A model is an artefact acknowledged by the observer as representing some domain for a particular purpose. (see [3])

Holistic view : In most cases, a model is also a conceptual model [9]. (see [28])

Conceptual models as a result of an activity: We use the name of conceptual modeling for the activity that elicits and describes general knowledge a particular information system needs to know. The main objective of conceptual modeling is to obtain that description, which is called a conceptual schema. (see [26])

Purpose-oriented modelling: Conceptual modelling is about abstracting a model that is fit-for-purpose and by this we mean a model that is valid, credible, feasible and useful. (see [31])

Documentation-oriented conceptual model: A conceptual data model is a summary-level data model that is most often used on strategic data projects. It typically describes an entire enterprise. Due to its highly abstract nature, it may be referred to as a conceptual model. (see [17])

Semiotics viewpoint: Conceptual modeling is about describing syntax, and semantics (potentially also pragmatics) of software applications at a high level of abstraction. (see [11])

Documentation and understanding viewpoint: A conceptual model of an application is the model of the application that the designers want users to understand. By using the application, talking with other users, and reading the documentation, users build a model in their minds of how to use the application. Hopefully, the model that users build in their minds is close to the one the designers intended. (see [18])

[6] And continuing: These terms are introduced by analogy to data models and database schemata. The reader may want to think of data models as special conceptual models where the intended matter consists of data structures and associated operations.

[7] Some research challenges in conceptual modeling: Provide the right set of modeling constructs at the right level of abstraction to enable successfully communication among clients, analysts, and application programmers. Formalize conceptual-modeling abstractions so that they retain their ease-of-communication property and yet are able to (partially or even fully) generate functioning application software. Make conceptual modeling serve as analysis and development tools for exotic applications such as: modeling the computational features of DNA-level life to improve human genome understanding, annotating text conceptually in order to superimpose a web of knowledge over document collections, leveraging conceptual models to integrate data (virtually or actually) providing users with a unified view of a collection of data, extending conceptual-modeling to support geometric and spatial modeling, and managing the evolution and migration information systems. Develop a theory of conceptual models and conceptual modeling and establish a formal foundation of conceptual modeling.

[8] Another version is the following one: The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. A conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. A high-level data model or an implementation data model can be used at this level.

[9] The slides of the keynote talk state: A conceptual model is a simplification of a system built with an intended goal in mind.
An abstraction of a system to reason about it (either a physical system or a real or language-based system). A description of specification of a system and its environment for some purpose. One main conclusion that we can reach is that the distinction between "model" and "conceptual model" is not always as precise as it should be.

Conceptualisations of models : Conceptual models are nothing else as models that incorporate concepts and conceptions which are denoted by names in a given name space. A concept space [10] consists of concepts (see [24]) as basic elements, constructors for inductive construction of complex elements called conceptions, a number of relations among elements that satisfy a number of axioms, and functions defined on elements. (see [38])

At the ER'2017 conference a special brainstorming and discussion session has been organised with the task to coin the notion of a conceptual model. It seems to be surprising that there is no commonly accepted notion of a conceptual model after more than 40 years of introduction of this concept into database research. One proposal of the brainstorming discussion was:

ER 2017 discussion proposal: A conceptual model is a partial representation of a domain that can answer a question.

As for a model, the purpose dimension determines the quality characteristics and the properties of a model.

In a nutshell, a *conceptual model* is an enhancement of a model by concepts from a concept(ion) space. It is formulated in a language that allows well-structured formulations, is based on mental/perception/domain-situation models with their embedded concept(ion)s, and is oriented on a modelling matrix that is a common consensus within its community of practice.

We thus meet a good number of challenges, e.g. the following ones: is there any acceptable and general notion of conceptual model; do conceptual models really provide an added and sustainable value; what are the differences between conceptual models and models; what is a model; what means conceptualisation; how to support language-based conceptual modelling; etc. This paper is oriented on these questions and tries to develop an answer to them. We restrict the investigation to conceptual models in computer science and computer engineering and thus do not consider conceptual modelling for product design, service design, other system's design, natural and social sciences. Physical conceptual models are also left out of scope.

2 Revisiting Conceptual Modelling

2.1 State-Of-Art and State-Of-Needs

Modelling offers the benefit of producing better and understandable systems. It is based on a higher level of abstraction compared to most programming languages. Whether a model must be formal is an open question. The best approach is to consider model suites (or ensembles) that consist of a coherent collection of models which are representing different points of view and attention. We observe a resurgence in domain specific approaches that are challenged by technical, organisational and especially language design problems. UML is not the solution yet because UML Models aren't executable but MDA needs them to be. The vast majority of UML models we have seen in industrial project are mere sketches and are informal and incomplete. They are not yet a viable basis for precise and executable models. Without precise models, no formal checking can take place. Therefore, these issues must be addressed either if modelling is well-accepted and gains significant presence in applications.

From the other side, the large body of knowledge on conceptual modelling in computer science is a results of hundreds of research papers over the last three-score years although different names have been used for it. Modelling is often based on a finalised-model-of-the-real-world paradigm despite the constant change in applications. Model quality has already been considered in a dozen papers. Modelling literacy is rarely addressed in education. Models must however be reliable, refinable, and translatable artifacts in software processes.

Conceptual modelling is supported by a large variety of tools. e.g. (see [21]). However, few of them support executable models. Of that few, far fewer still are actually rewarding to use. Conceptual models are acknowledged as mediators in the software development process. However, they are used and then not evolving with the evolution of the software. Reuse, migration, adaptation, and integration of models is still a lacuna. The lack of robust, evolution-prone and convenient translators is one reason. An environment as a constituent part for modelling and translation into a consistent, easy-to-use and -revise, seamless, and industry-quality tools is still on the agenda. Information and software systems become eco-systems. Modelling eco-systems are not yet properly addressed.

Models are also used for communication based on some injection of a name space while the community of practice uses a wealth of terms and terminology with which they express their nuances of viewpoints. So, we need a number of representation models beside the singleton graphical representation. At the same time, models must be properly formal and based on rules strictly to be followed or else having a risk of making illogical statements. Modelling must thus be based on methodologies.

2.2 Myths of (Conceptual) Modelling

Modelling and especially conceptual modelling is not yet well understood and misinterpreted in a variety of ways. It has brought a good number of myths similar to those known for software development (see [1]):

1. *Modelling is mainly for documentation.* The introduction of the conceptual modelling for database systems has been motivated by documentation scenario. A conclusion might be that modelling is a superfluous

[10] We follow R.T. White (see [37, 42]) and distinguish between concepts, conceptual, conceptional, and conceptions.

activity, especially in the case that documentation is not an issue.

2. *Modelling is finished with the use of the model and an initial phase*. Historic development of software started with requirements which were frozen afterwards and with modelling and specifications that were complete and became frozen before realisation begins.

3. *Modelling is only useful for heavyweight V-style software development*. Modelling and especially conceptual modelling is abandoned due to its burden and the discovery of the complexity of the software that is targeted.

4. *The collection of origins must be "frozen" before starting with modelling*. Models should be plastic and stable (one of the justification and thus dependability properties), i.e. the collection of origins to be modelled could change.

5. *The model is carved in stone and changes only from time to time if at all*. The realisation becomes 'alive' and thus meets continuous change requests. The model can have some faults, errors, misconceptions, misses etc. Extensions and additional services are common for systems. So, the model has to change as well.

6. *Modelling is starts with selecting and accommodating a CASE tool*. Although CASE tools are useful they impose their own philosophy, language, and treatment. Moreover, CASE tools allow to become too detailed. Instead, conceptual modelling should allow to create the model that is simple as possible and as detailed as necessary.

7. *Conceptual modelling is a waste of time*. Developers are interested in quick success and have their own perception model in mind. It seems to be superfluous to model and better to focus solely on how to write the code.

8. *Conceptual data modelling is a primary concern*. Data- and structure-driven development without consideration of the usage of the data in applications results in 'optimal' or 'normalised' data structure models and bad database performance. One must keep in mind the usage of the data, i.e. use a co-design method, e.g. (see [34]).

9. *The community of practice has a common understanding how to conceptually model*. Modelling skills evolve over years and are based on modelling practice and experience. Further, conceptual models are based on a common domain-situation model that has to be shared within the community of practice. So, the perception models of modellers should match.

10. *Modelling is independent on the language*. Modelling cannot be performed in any language environment. Language matters, enables, restricts and biases (see [43]).

Understanding these and other myths allows to better understand the modelling process and the models. One way to overcome them is the development of sophisticated and acknowledged frameworks. Model-centred development (see [23]) uses models as a kernel for development of systems. Conceptual modelling ist still taught as modelling in the small whereas modelling in the large is the real challenge.

2.3 Specifics of Notions

Let us return to the list of notions given in Section 1. Each of these notions has its graces, biases, orientations, applicability, acceptability, and specifics.

Scopes of conceptual models may vary from very general models to fine-grained models. General models allow to reason on system properties whereas fine-grained models serve as a blueprint for development.

Result-oriented viewpoint: Conceptual models can be seen as the final result and documentation of an activity that follows a certain development strategy such as agile, extreme, waterfall etc. methodologies.

Communication viewpoint: Conceptual models are a means for communication and negotiation among different stakeholders.

System construction orientation: Database, information and software system development is becoming more complex, more voluminous, requires higher variety, and changes with higher velocity. So a quick and parsimonious comprehension becomes essential and supports higher veracity and an added value for the system itself.

Perception and domain-situation models are specific mental models either of one member or of the community of practice within one application area. It is not the real world or the reality what is represented. It is the common consensus, world view and perception what is represented.

Conceptual models as documentation: Models provide also quality in use, i.e. they allow to survey, to understand, to negotiate, and to communicate.

Conceptual modelling with prototypes: Models can be enhanced by prototypes or sample populations. A typical approach is sample-based development (see [16]).

Visualisation issues: Conceptual models may be combined with representation models, e.g. visualisation models on the basis of diagrammatic languages.

Biased conceptual modelling approaches: Conceptual models are often models with a hidden background, especially hidden assumptions that are commonly accepted in a community of practice in a given context and utilisation scenario.

Semiotics and semiology of conceptual modelling: Conceptual models are often language-based. The language selection is predetermined and not a matter of consideration in the modelling process.

Quality models: Conceptual models should be well-formed and satisfy quality requirements depending on their function in utilisation scenarios.

Concepts, conceptions: The elements in a conceptual models are annotated by names from some name space. These names provide a reference to the meaning, i.e. a reference to concepts and conceptions in a concept space.

Conceptual model suites: Models can be holistic or consist of several associated models where in the latter case each of them represents different viewpoints. For instance, a conceptual database model consists of a schema and a number of derived views which represent viewpoints of business users.

Normal models: Conceptual models represent only certain aspects and are considered to be intentionally enhanced by elements that stem from commonsense, consensuses, and contexts.

A normal models (called 'lumped' model in [45]) is a part of the model that is considered to be essential and absolutely necessary. The *normal model* has a context, a community of practice that puts up with it, a utilisation scenario for which is is minimally sufficient, and a latent – or better deep – model on which it is based (see [45] for 'base' model). The *deep model* combines the unchangeable part of a model and is determined by the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and the basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. The (modelling) *matrix* consists of the deep model and the modelling scenarios. The last ones are typically stereotyped in dependence on the chosen *modelling method*.

This variety of viewpoints to conceptual models illustrates the different requirements and objectives of models. So, we might ask whether a common notion of a conceptual model exists or whether we should use different notions.

2.4 Problems and Challenges

Conceptual modelling techniques suffer from a number of weaknesses. These weaknesses are are mainly caused by concentration on database modelling and by non-consideration of application domain problems that must be solved by information systems. We follow the state-of-the-art analysis of A. van Lamsweerde (see [40, 41]) who gave a critical insight into software specification and arrive with the following general weaknesses for conceptual modelling of information and database systems:

Limited scope. The vast majority of techniques are limited to the specification of data structuring, that is, properties about what the schema oft he database system is expected to do. Classical functional and nonfunctional properties are in general left outside or delayed until coding.

Poor separation of concerns. Most modelling approaches provide no support for making a clear separation between (a) intended properties of the system considered, (b) assumptions about the environment of this system, and (c) properties of the application domain

Low-level schematology. The concepts in terms of which problems have to be structured and formalized are concepts of modelling in the small - most often, data types and some operations. It is time to raise the level of abstraction and conceptual richness found in application domains.

Isolation. Database modelling approaches are isolated from other software products and processes both vertically and horizontally. They neither pay attention to what upstream products in the software might provide or require nor pay attention to what companion products should support nor provide a link to application domain description.

Poor guidance. The main emphasis in the database modelling literature has been on suitable sets of notations and on a posteriori analysis of database schemata written using such notations. Constructive methods for building correct models for complex database or information systems in a safe, systematic, incremental way are by and large non-existent.

Cost. Many information systems modelling approaches require high expertise in database systems and in the white-box use of tools.

Poor tool feedback. Many database system development tools are effective at pointing out problems, but in general they do a poor job of (a) suggesting causes at the root of such problems, and (b) proposing better modelling solutions.

Modern modelling approaches must not start from scratch. We can reuse achievements of database modelling in a systematic form and thus maintain theories and technologies while supporting new paradigms.

Constructiveness. Models of information systems can be built incrementally from higher-level ones in a way that guarantees high quality by construction. A method, is typically made of a collection of model building strategies, paradigm and high-level solution selection rules, model refinement rules, guidelines, and heuristics. Some oft hem might be domain-independent, some others might be domain-specific.

Support for comparative analysis. Database models depend on the experience oft he developer, the background or reference solutions on hand, and on preferences of developers. Therefore, the results within a team of developers might need a revision or a transformation to a holistic solution. Beyond the modelling qualities we may develop precise criteria and measures for assessing models and comparing their relative merits.

Integration. Tomorrow's modelling should care for the vertical and horizontal integration of models within the entire analysis, design, development, deployment and maintenance life cycle - from high-level goals to be supported by appropriate architectures, from informal formulation of information system models to conceptual models , and from conceptual models to implementation models and their integration into deployment of information systems.

Higher level of abstraction. Information systems modelling should move from infological design to holistic co-design of structuring, functionality, interactivity and distribution. These techniques must additionally be error-prone due tot he complexity of modern information systems. These abstraction techniques may be combined with refinement techniques similar to those that have been developed fort he abstract state machines.

Richer structuring mechanisms. Most modelling paradigms of the modelling-in-the-small approach available so far for modularising large database schemata have been lifted from software engineering approaches, e.g., component development. Problem-oriented constructs be developed as well model suites that provide a means for handling a variety of models and viewpoints.

Extended scope. Information system development approaches need to be extended in order to cope with the co-design of structuring, functionality, interactivity and distribution despite an explicit treatment of quality or non-functional properties.

Separation of concerns. Information system modelling languages should enforce a strict separation between descriptive and prescriptive properties, to be exploited by analysis tools accordingly.

Lightweight techniques. The use of novel modelling paradigms should not require deep theoretical background or a deep insight into information systems technology. The results or models should be compiled to appropriate implementations.

Multi-paradigm modelling. Complex information systems have multiple facets. Since no single modelling paradigm or universal language will ever serve all purposes of a system. The various facets then need to be linked to each other in a coherent way.

Multilevel reasoning and analysis. A multi-paradigm framework should support different levels of modelling, analysis, design and development - from abstract and general to deep-level analysis and repairing of detected deficiencies.

Multi-format modelling. To enhance the communicability and collaboration within a development and support team the same model fragment must be provided in a number of formats in a coherent and consistent way.

Reasoning in spite of errors. Many modelling approaches require that the model must be complete before the analysis can start. We claim that is should be made possible to start analysis and model reasoning much earlier and incrementally.

Constructive feedback from tools. Instead of just pointing out problems, future tools should assist in resolving them.

Support for evolution. In general, applications keep evolving due to changes in the application domain, to changes of technology, changes in information systems purposes etc. A more constructive approach should also help managing the evolution of models.

Support for reuse. Problems in the application domain considered are more likely to be similar than solutions. Models reuse should therefore be even more promising than code reuse.

Measurability of modelling progress. To be more convincing, the benefits of using information models should be measurable as well as their deficiencies.

This list of theories, solutions and methodological approaches is not exhaustive. It demonstrates, however, that modelling in the large and modern information systems modelling require specific approaches beyond integration of architectures into the analysis, design and development process.

2.5 The Research Issue

Let us reconsider the notions presented in Section 1. Table 1 compares essential properties of models. Missing model elements are denoted by n(ot).g(iven).

We observe that dependability is often either implicit or not considered in the model notion. Implicitness is mainly based on the orientation to normal models. The model matrix and especially the deep model are considered to be agreed before developing the model.

The origin is too wide in most cases. Models are not oriented towards representing some reality or the world. They are typically based on some kind of agreement made within a community of practice and according to some context, i.e.

Table 1. Orientation of notions of conceptual models according model properties

version	adequate	dependable	origin	function	scenario	concepts
reality, world	reflection, truncation	formal, reflection	world	describe	communication, understanding	n.g.
community	abstraction, mapping	semantic invariance	software application	describe	construction	n.g.
conceptual database	mapping, homomorphy	n.g.	data, operations	describe	construction, documentation	reality concepts
system & instance	mapping, abstraction	n.g.	system, objects	n.g.	construction	n.g.
system representation	reflection,	qualitative assumptions	system	describe	representation	system concepts
represen-tional	mapping	n.g.	relation-ships	represent	visualisation	impact factors
enterprise	mapping, abstraction	faithful	domain	purpose-determined	understanding	concept space
result of activity	mapping,	n.g.	system knowledge	describe	acquisition, elicitation	domain knowledge
purpose-oriented	abstraction purposeful	viable, fit	any	elicitate	n.g.	n.g.
documen-tation	summary, abstraction	n.g.	data system	represent, survey	strategy development	n.g.
semiotics	syntax abstraction	semantics, pragmatics	software application	describe	representation	n.g.
document understand	mapping	closeness	application	understand by users	design	n.g.
conceptualise	formal representation	semantics	any	describe	representation	concept(ion) space
ad-hoc	selective mapping	n.g.	domain	consider problem	solving	n.g.

they reflect some domain-situation model[11] or more generally some mental model[12]. They might represent a perception model of some members of the community practice. They say what the phenomena in the given domain are like.

Table 1 directs to a conclusion that the function is mainly oriented towards description and partially prescription for systems development. The notion of the conceptual model has, however, mainly considered in system construction scenarios.

Concepts are often hidden behind the curtain of conceptual models. A conceptual model does not reflect the reality. Instead it reflects the mental understanding within its utilisation scenario.

These observations show now directly some open issues that should be solved within a theory and practice of conceptual modelling. Let us state some of them.

[11] We restrict consideration to our field and thus to domain-oriented models. These models describe the application domain and more specifically the understanding, observation, and perception of an application domain that is accepted within a community of practice. In general, a situation model is a mental representation of a described or experienced situation in a real or imaginary world (see [30]).

[12] Mental models are out-of-scope in this paper. Those consist of an evolving model suite with small-scale and parsimonious models carried in human head (see [13, 19]). They support various kinds of observation, information acquisition and filtering, reasoning, storage and information (de)coding, and communication. They are dependent on the observations, imaginations, and comprehension a human has made. Unlike conceptual models, mental models must neither be accurate, nor complete, and not consistent.

Research question 1. What are the *origins* for conceptual models? Are these mainly domain-situation and perception models from one side and systems on the other side?

Research question 2. How tightly conceptual models are bound to their *modelling matrix* and especially their *deep model*? To what extent conceptual models are normal models that are intentionally combined with their deep models?

Research question 3. Which functions have conceptual models in which utilisation scenarios? Which properties must be satisfied by conceptual models in these scenarios? Which purposes and goals can be derived?

Research question 4. What is the role of the *community of practice* in conceptual modelling? Which kind of model supports which community in which context?

Research question 5. Conceptual modelling is less automated and more human dependent than any other development, analysis, and design process for information systems. It is a highly creative process. Is there any formalisation and foundation for this process?

Research question 6. Since models must not be conceptual models (see models in [39]), we might ask whether there exists a set of characteristics or criteria that separate

a conceptual model from a model that is not conceptual. What are the concept space that can be used for an enhancement of a model by concepts or conceptions?

3 The Nature of Models

3.1 The Notion of a (Conceptual) Model

The model is an utterance and also an imagination. As already stated above (see also [39]), a model is a *well-formed*, *adequate*, and *dependable* instrument that represents *origins* and *functions* in some utilisation *scenario*. A model is a representation of some origins and may consist of many expressions such as sentences. *Adequacy* is based on satisfaction of the purpose or function or goal, analogy to the origins it represents and the focus under which the model is used. *Dependability* is based on a *justification* for its usage as a model and on a *quality certificate*. Models can be evaluated by one of the evaluation frameworks. A model is *functional* if methods for its development and for its deployment are given. A model is *effective* if it can be deployed according to its portfolio, i.e. according to the tasks assigned to the model. Deployment is often using some deployment macro-model, e.g. for explanation, exploration, construction, documentation, description and prescription.

Models *function* as *instruments* or tools. Typically, instruments come in a variety of forms and fulfill many different functions. Instruments are partially independent or autonomous of the thing they operate on. Models are however special instruments. They are used with a specific intention within a utilisation scenario. The quality of a model becomes apparent in the context of this scenario.

Model development is often targeted on normal models and implicitly accepts the deep model. A model is developed for some modelling scenarios and thus biased by its modelling matrix. The deep model and the matrix thus 'infect' the normal model.

Within the scope of this paper, we concentrate on representation models as proxies. So, a model of a collection of origins, within some context, for some utilisation scenario and corresponding functions within these scenarios, and for a community of practice is

- a relatively enduring,
- accessible
- but limited
- internal and at the same time external
- representation of the collection of origins.

The model becomes *conceptual* by incorporation of concepts and conceptions commonly accepted, of ideas provided by members from the community of practice, or of general well-understood language-like semiotic components. One main utilisation scenario for conceptual database model is system construction[13]. In this case, the conceptual model thus becomes predictively accurate for the system envisioned and technologically fruitful. The model is an utterance and also an imagination. Other scenarios for conceptual models are: system modernisation, explanation, exploration, communication, negotiation, problem solving, supplantation, documentation, and even theory development.

Conceptual models must not limited to representation of static aspects of systems. They can also be used for representation of dynamic aspects such as business stories, business processes, and system behaviour. The carrier of representation is often some language. In this case, a conceptual model can be considered to be an utterance with a collection speech acts. The model itself can be then build on well-formedness rules for its syntax, semantics, and pragmatics, or more general of semiotics and semiology. According to J. Searle (see [33]), a speech act consists of uttering elements, referring and predicating, requesting activities, and causing an effect. Whether at all and which language is going to be used is a matter of controversy too.

3.2 Facets of a Conceptual Model

1. The conceptual model is a result of a perception and negotiation process. The conceptual model represents mental models, especially domain-situation models or a number of perception models. Domain-situation models represent a settled perception within a context, especially an application. Perception models might differ from the domain-situation model. They are personal perceptions and judgements of a member of the community of practice. Maturity of conceptual models is reached after the community of practice negotiated different viewpoints and has found an agreement.

2. The conceptual model represents its collection of origins. Considerations about what to model and what not to model are expressed via the adequacy criteria, especially for analogy to its origins, for focusing on specifics of the origins, and also on well-formedness of the model. The conceptual model does not represent a real world or a problem domain. It is already based on perception models of users about this problem domain or on domain-situation models of a user community on this problem domain.

3. The conceptual model is an instrument. The conceptual model is used in some utilisation scenario by its users. So it functions in this utilisation scenario. It should describe in a more abstract way compared to the origins how the user conceives it and thus does not target on describing the origins.

4. The deep model underpins the conceptual model. The deep model consists of all elements that are taken for granted, are considered to be fixed, and are common within the context for the community of practice. Elements of this model are symbolic generalizations as formal or readily formalisable components or laws or law schemata, beliefs in particular heuristic and ontological models or analogies

[13] Notice however that the first introduction of conceptual data models has been oriented on a documentation scenario.

supplying the group with preferred or permissible analogies and metaphors, and values shared by the community of practice as an integral part and supporting the choice between incompatible ways of practicing their discipline. There is no need to redevelop this model. So, the normal model only display those elements that are additionally introduced for the model.

5. The conceptual modelling matrix. The modelling matrix combines the deep model with the typical utilisation scenarios that are accepted by a community of practice in a given context. It specifies a guiding question as a principal concern or scientific interest that motivates the development of a theory, and techniques as the methods an developer uses to persuade the members of the community of practice to his point of view. Although often not explicitly stated, the model matrix consists of a number of components: the objectives, inputs (or experimental factors), outputs (or responses), content requests, grounding, basis, and simplifications. The matrix sets a definitional frame for the normal model. It might support modelling by model stereotypes. The agenda of the modelling method is derived from the matrix. The matrix determines also a spefic treatment of adequacy and dependability for a model.

6. The performance and quality criteria. The model is a persistent and justified artifact that satisfy a number of conditions according to its function such as empirical corroboration according to modelling objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability, and by stability and plasticity within a collection of origins. The quality characteristics bound the model to be valid, credible, feasible, parsimonious, useful, and at the same time as simple as possible and as complex as necessary.

7. The model is the main ingredient of a modelling method. Sciences and technologies have developed their specific deployment of models within their investigation, analysis, development, design etc. processes. The deep model and the matrix are often agreed. The central element of all modelling methods is the model that is used as an instrument in scenarios which have been stereotyped for the given modelling method. The modelling method typically also includes design of a representation model (or a number of such). The representation model of the (conceptual) model may be based on approaches such as diagramming and visualisation. It uses a set of predefined signs: icons, symbols, or indexes in the sense of Peirce.

3.3 Sources for Conceptual Models: Domain-Situation and Perception Models

The domain-situation model is build by a community of practice on a semantical level. It refers to the world-as-described-and-conceived-by-the-deep-model. It thus forms the deep understanding behind the conceptual model. This deep internal structure of the conceptualisation is commonly shared in the community, abstracts from accidental origins, uses a partial interpretation, exhibits (structural) hidden similarities of all origins under consideration, and presents the common understanding in the community. It gives thus a literal meaning to the domain. The context for the conceptual model is typically governed by domain-situation models. The domain-situation model is thus one source for the conceptual model.

A domain-situation model might or might not exist. It shapes, however, what is seen in an application domain and how to reason about what is seen. They represent some common negotiated understanding in the application domain. It may represent the application domain as it is or the application domain as it makes sense to be characterised, categorised or classified in one way rather than another given certain interests and aptitudes or more generally given certain background.

The second source for conceptual models is a collection of perception models that are provided and acknowledged by members of this community of practice. A perception model is one kind of epistemological mental model with its verbal, visual and other information compiled on the basis of cognitive schemata. It organises, identifies, and interprets observations made by the member. It does not need to know the deep facts or essential properties of the origins in order to succeed in communicating about them or to reason. The perception model typically follows the situation that it represents. It is however often underdetermined and thus may also partially contradictious. So it parallels and imitates parts of the reality ('Gestalt' notion of the model). They provide a partial understanding, refer to some aspect, may use competing sub-models about the same stuff, and may set alternatives on meaning. It is build by intuitive, discursive and evidence-backed perception, by imagination, and by comprehension. It is shaped by learning, memorisation, expectation, and attention. Perception models serve as an add-on beyond domain-situation models.

These two sources for conceptual models depend on the community of practice. So, different communities might use different kinds of verbal and nonverbal representation. Although they provide a literal meaning to the conceptual model they must not be explicitly stated within the conceptual model. They serve as the origin for the conceptual model and thus might not be explicitly incorporated into the conceptual model. The conceptual model may have its deep background, i.e. its basis and especially its grounding.

Both origins are not complete. Typically the scope of both models is not explicit. There are unknown assumptions applied for description, unknown restrictions of the model, undocumented preferences and background of the community of practice, and unknown limitations of the modelling language. Classically we observe for members of a community of practice that

- they base their design decisions on a "partial reality", i.e. on a number of observed properties within a part of the application,

- they develop their models within a certain context,
- they reuse their experience gained in former projects and solutions known for their reference models, and
- they use a number of theories with a certain exactness and rigidity.

The conceptual model to be developed is deeply influenced by these four hidden factors.

4 Conceptualisation of Models

The domain-situation model and also partially the perception model are commonly using concepts. Conceptual models reuse such concepts from these origins and thus inherit semantics and pragmatics from these models. Further, conceptualisation may also be implicit and may use some kind of lexical semantics of these models, e.g. word semantics, within a commonly agreed name space.

4.1 Concepts and Conceptions

Various notions of concept has been introduced, for instance, by J. Akoka, P. Chen, H. Kangassalo, R. Kauppi, A. Paivio, and R. Wille (see [6, 14, 22, 20, 27]). Artificial intelligence and mathematical logics use concept frames. Ontologies combine lexicology and lexicography. Concepts are used in daily life as a communication vehicle and as a result of perception, reasoning, and comprehension. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions etc. Some version may be preferred over others, may be time-dependent, may have a level of rigidity, is typically usage-dependent, has levels of validity, and can only be used within certain restrictions. We also may use a large variety of semantics (see [32]), e.g., lexical or ontological, logical, or reflective.

We distinguish two different meanings of the word 'concept' (see [42]):

1. Concepts are general categories and thing of interest that are used for classification. Concepts thus have fuzzy boundaries. Additionally, classification depends on the context and deployment.
2. Concepts are all the knowledge that the person has, and associates with, the concept's name. They are reasonable complete in terms of the business.

Conceptions (see [42]) are systems of explanation. They are thus more difficult to describe.

The typical definition frame we observed is based on definition items. These items can also be classified by the kind of definition. Concepts may simultaneously have different descriptions. Competing description may differently represent the same concept depending on context (e.g. time, space), validity, usage, and preferences of members of the community of practice. A concept may have elements that are necessary or sufficient, that may be of certain rigidity, importance, relevance, typicality, or Fuzziness. Based on the generalisations of the approach that has been proposed by G.L. Murphy (see [24, 35]), concepts are defined in a more sophisticated form as a tree-structured structural expression.

SpecOrderedTree(StructuralTreeExpression
(DefinitionItem, Modality(Sufficiency, Necessity),
Fuzziness, Importance, Rigidity,
Relevance, GraduationWithinExpression, Category))) .

Concept may be regarded as the descriptive and epistemic core units of perception and domain-situation models. These origins govern the way how a concept can be understood, defined, and used in a conceptual model. The conceptual model inherits thus concepts and their structuring within a concept space, i.e. conceptions.

4.2 Conceptualise

Conceptualisation and semantification are orthogonal concerns in modelling. *Conceptual modelling* is based on concepts that are used for classification of things. Concepts have fuzzy boundaries. Additionally, classification depends on the context and deployment. Conceptual[14] modelling uses *conceptions* which are systems of explanation.

Semantification (see [9]) improves comprehensibility of models and explicit reasoning on elements used in models. It is based on name spaces or ontologies that are commonly accepted in the application domain. Conceptual models are models enhanced by concepts and integrated in a space of conceptions.

Conceptualisation injects concepts or conceptions into models. These enriched models reflect those concepts from commonly accepted concept space. The concept space consists of a system of conceptions (concepts, theoretical statements (axioms, laws, theorems, definitions), models, theories, and tools). A concept space also may include procedures, conceptual (knowledge) tools, and associated norms resp. rules. Is is based on paradigms which are corroborated.

4.3 Dependability of Conceptual Models

Models must be dependable, i.e. justified from one side and and qualitatively certified from the other side. Justification can be based on the domain-situation and perception models and the relation of the conceptual models to these models. If however such models are not available or of low quality then justification will become an issue. Quality certification is an issue of pragmatism and of added value of the conceptual model. So, we target on a high quality conceptualisation. Conceptualisation may be based on the seven principles of Universal Design (see [29]). Typical mandatory principles are usefulness, flexibility, sim-

[14] Conceptual modelling is performed by a modeller that directs the process based on his/her experience, education, understanding, intention and attitude. Conceptual models are using/incorporating/integrating concepts (see [42]) Conceptional modelling aims at development of concepts.

plicity, realisability, and rationality. Optional conceptualisation principles are perceptability, error-proneness, and parsimony.

The *principle of conceptualisation* is considered to be one -if not the main - of the seven fundamental principles for conceptual modelling (see [15]). The other six principles are: Helsinki, Universe of discourse, searchlight, 100%, onion, and three level architecture principles. They can be questioned further. These principles can be enhanced by the principles of understanding, of abstraction, of definition, of refinement, evaluation, and of construction (see [36]). Conceptualisation can be considered to be completed if: A conceptual schema should only include conceptually relevant aspects, both static and dynamic, of the universe of discourse, thus excluding all aspects of (external or internal) data representation, physical data organization and access, as well as all aspects of particular external user representation such as message formats, data structures, etc.
Based on Section 3.3, the principle of conceptualisation can be stated as follows:
A conceptual model should only include conceptually relevant aspects of the domain-situation and perception models. It does not consider neither aspects of realisation nor of representation. It includes, however, different viewpoints of business users and concepts from the common concept space.

5 Conclusion: Towards a Notational Frame for Conceptual Models

Conceptual modelling is not yet a science or culture. It is rather a craft or even an art. It can be learned similar to craft learning. It is however based on understanding and abstraction througout the perception and domain-situation models, i.e. of mental models in general. Perception is dependent on deep models and thus incomplete, revisable, time-restricted, activity-driven, and context-dependent.

5.1 Slim, Light, and Concise Versions for Conceptual Models

Conceptual models are widely used in system construction scenarios. They function as description of the phenomena of interest within the context for its community of practice. So, conceptual models are normal models with rather specific modelling matrices and deep models. A slim notion of a conceptual model is should only reflect such normal models and refer to a specific modelling matrix. A light version needs to refer to some elements of the basis and to some context. A concise version must explicitly represent all the hidden details of a model, especially its relationships to the concept space, to the perception of this space by members of the community of practice, and to the utilisation scenario.

5.2 A Proposal for a Light Version: Conceptual Model $\sqsupseteq$ Model $\bigoplus$ Concepts

Conceptual modelling is not yet a common method in science (see [31]). Systems can be build without any conceptual model. It seems that there is no need for a formal conceptual modelling process. It seems to be too restrictive to require a full conceptual model. Performance and quality criteria are not commonly agreed. The science of conceptual modelling is still missing.

The main bottleneck is however the missing notion of a conceptual model. The conceptual model is a specific model and is based on conceptualisation. It might be language-bound. It is probably the most important aspect of system construction in computer science and computer engineering. It is however the most difficult and least understood. Minimal justification characteristics of models are classical viability, i.e. corroboration, validity, credibility, rational coherent and conform, falsifiable, stability against origin collection change. Minimal quality characteristics of models are the one for quality in use (e.g. usability, aptness for the function and purpose, value for the utilisation scenario, feasibility). Minimal performance characteristics are timely, elegant and feasible usage within the given context for their community of practice according to their utilisation scenario and their competencies or more general their profiles.

So, we might conclude for a *light version*: A conceptual model is a well-formed, adequate and dependable instrument that functions within its specific utilisation scenario, that represents origins, and that is enhanced by concepts from a concept(ion) space.

Therefore, the incorporation of concepts and the conceptions is one main difference to the model.

5.3 Lacunas of Conceptual Modelling

Since conceptual modelling is still more an art than a science and a culture of conceptual modelling is still beyond the horizons, we need

- an understanding of the area of conceptual modelling;
- a theory, techniques, and engineering of conceptualisation;
- an integrated multi-view approach for the needs and the capabilities of the members of the community of practice;
- a refinable definition of the conceptual model with all three versions, i.e. a simplified version, a fully fledged version, and an assessable version;
- a working approach with intentional and thus latent matrices and deep models for daily practice; and
- an understanding of language use in conceptual modelling.

These lacunas do not limit usability, usefulness, and utility of conceptual models. Conceptual database models improve from one side system comprehension. They allow to indicate associations among system elements, reduce the effect of bad implementation, provide abstraction mechanisms, support prediction of system behaviour, provide an elegant and adequate overview of the system at various levels of abstraction, support the construction of different user views, and cross-reference multiple viewpoints. From the other side, the reduce the developers, maintainers and programmers overhead. They support a simple and free navigation through components of the database system, provide an easy deduction of various viewpoints that represent the needs of business users, support concentration and focusing in evolution and maintenance phases, display the decisions made during development, indicate opportunities for further development and system maintenance, reduce the effort by reuse of design and development decisions that have already been made, and use a comfortable and effective visualisation. So, conceptual models are not restricted to construction scenarios or to database modelling.

We realise that the development and the acceptance of a notion of conceptual model follows the 13 Commandments stated (see [5]):

1. Thou shalt choose an appropriate notation.
2. Thou shalt formalise but not overformalise.
3. Thou shalt estimate costs.
4. Thou shalt have a formal methods guru on call.
5. Thou shalt not abandon thy traditional development methods.
6. Thou shalt document sufficiently.
7. Thou shalt not compromise thy quality standards.
8. Thou shalt not be dogmatic.
9. Thou shalt test, test, and test again.
10. Thou shalt reuse.
11. Thou shall meet intentions of all members of the community of practice
12. Thou shall provide a usable notation, i.e. for verification, validation, explanation, elaboration, and evolution.
13. Thou shall be robust against misinterpretation, errors, etc.

References

[1] S.W. Ambler and P.J. Sadalage. *Refactoring databases - Evolutionary database design*. Addison-Wesley, 2006.

[2] C. Batini, S. Ceri, and S. Navathe. *Conceptual database design (an entity-relationship approach)*. Benjamin/Cummings, Redwood City, 1992.

[3] M. Bjeković. *Pragmatics of Enterprise Modelling Languages: A Framework for Understanding and Explaining*. PhD thesis, Radboud University Nijmegen, 2017.

[4] M. Boman, J.A. Bubenko Jr., P. Johannesson, and B. Wangler. *Conceptual modelling*. Prentice Hall, London, 1997.

[5] J. P. Bowen and M. G. Hinchey. Ten commandments ten years on: Lessons for asm, b, z and vsr-net. In *Rigorous Methods for Software Construction and Analysis*, volume 5115 of *Lecture Notes in Computer Science*, pages 219–233. Springer, 2009.

[6] P.P. Chen, J. Akoka, Kangassalo H, and B. Thalheim, editors. *Conceptual Modeling: Current Issues and Future Directions*, volume 1565 of *LNCS 1565*. Springer, 1998.

[7] Business dictionary. Conceptual model. http://www.businessdictionary.com/definition/conceptual-model.html, 2017. Assessed Nov. 21, 2011.

[8] WordNet dictionary. Conceptual model. http://www.dictionary.com/browse/conceptual-model, 2017. Assessed Nov. 21, 2011.

[9] M. Duží, A. Heimburger, T. Tokuda, P. Vojtas, and N. Yoshida. Multi-agent knowledge modelling. In *Information Modelling and Knowledge Bases XX*, Frontiers in Artificial Intelligence and Applications, 190, pages 411–428. IOS Press, 2009.

[10] R. Elmasri and S. Navathe. *Fundamentals of database systems*. Addison-Wesley, Reading, 2000.

[11] D. Embley and B. Thalheim. Preface. In *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*, pages v–ix. Springer, Berlin, 2011.

[12] ER community. Homepage. http://conceptualmodeling.org, 2017. Accessed Oct. 29, 2017.

[13] J.W. Forrester. *Collected papers of J.W. Forrester*, chapter Counterintuitive behaviour of social systems, pages 211–244. Wright-Allen Press, Cambridge, 1971.

[14] B. Ganter and R. Wille. *Formal concept analysis - Mathematical foundations*. Springer, Berlin, 1998.

[15] J. J. Van Griethuysen. The Orange report ISO TR9007 (1982 - 1987) Grandparent of the business rules approach and sbvr part 2 - The seven very fundamental principles. *Business Rules Journal*, 10(8), August 2009. http://www.brcommunity.com/a2009/b495.html.

[16] T. A. Halpin. Object-role modeling. In *Encyclopedia of Database Systems*, pages 1941–1946. Springer US, 2009.

[17] InfoAdvisors. What are conceptual, logical and physical data models? http://www.datamodel.com/index.php/articles/what-are-conceptual-logical-and-physical-data-models/, 2017. Assessed Nov. 21, 2011.

[18] J. Johnson and A. Henderson. Conceptual modelling in a nutshell. http://boxesandarrows.com, 2013. Accessed Jan. 29, 2013.

[19] P.N. Johnson-Laird. *Mental models: Towards a cognitive science of language, inference, and consciousness.* Cambridge University Press, London, 1983.

[20] H. Kangassalo and J. Palomäki. Defintional conceptual schema - the core for thinking, learning, and communication. Keynote given at 25th EJC Conference, Maribor, Slovenia, June 2015.

[21] D. Karagiannis, H. C. Mayr, and J. Mylopoulos, editors. *Domain-Specific Conceptual Modeling, Concepts, Methods and Tools*. Springer, 2016.

[22] R. Kauppi. Einführung in die Theorie der Begriffssysteme. *Acta Universitatis Tamperensis, Ser. A, Vol. 15, Tampereen yliopisto, Tampere*, 1967.

[23] H. C. Mayr, J. Michael, S. Ranasinghe, V. A. Shekhovtsov, and C. Steinberger. Model centered architecture. In *Conceptual Modeling Perspectives.*, pages 85–104. Springer, 2017.

[24] G. L. Murphy. *The big book of concepts*. MIT Press, 2001.

[25] J. Mylopoulos. Conceptual modeling and Telos. In *Conceptual modeling, databases, and CASE: An integrated view of information systems development*, LNCS 10509, pages 49–68, Chichester, 1992. Wiley.

[26] A. Olivé. *Conceptual modeling of information systems*. Springer, Berlin, 2007.

[27] A. Paivio. *Mental representations: A dual coding approach*. Oxford University Press, Oxford, 1986.

[28] O. Pastor. Conceptual modeling of life: Beyond the homo sapiens. http://er2016.cs.titech.ac.jp/assets/slides/ER2016-keynote2-slides.pdf, 2016. Assessed Nov. 15, 2016.

[29] B. Patil, K. Maetzel, and E. J. Neuhold. Design of international textual languages: A universal design framework. In *IWIPS*, pages 41–56. Product & Systems Internationalisation, Inc., 2003.

[30] G.A. Radvansky and R.T. Zacks. *Cognitive models of memory*, chapter The retrieval of situation-specific information, pages 173–213. MIT Press, Cambridge, 1997.

[31] S. Robinson. Conceptual modelling: Who needs it? *SCS M & S Magazine*, (2):1–7, 2010.

[32] K.-D. Schewe and B. Thalheim. Semantics in data and knowledge bases. In *SDKB 2008*, LNCS 4925, pages 1–25, Berlin, 2008. Springer.

[33] J. Searle. *Speech Acts: An essay in the philosophy of language*. Cambridge University Press, Cambridge, England, 1969.

[34] B. Thalheim. *Entity-relationship modeling – Foundations of database technology*. Springer, Berlin, 2000.

[35] B. Thalheim. The conceptual framework to user-oriented content management. In *Information Modelling and Knowledge Bases*, volume XVIII of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2007.

[36] B. Thalheim. Towards a theory of conceptual modelling. *Journal of Universal Computer Science*, 16(20):3102–3137, 2010. http://www.jucs.org/jucs_16_20/towards_a_theory_of.

[37] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[38] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[39] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[40] A. van Lamsweerde. Formal specification: a roadmap. In *ICSE - Future of SE Track*, pages 147–159, 2000.

[41] A. van Lamsweerde. Requirements engineering: from craft to discipline. In *SIGSOFT FSE*, pages 238–249. ACM, 2008.

[42] R.T. White. Commentary: Conceptual and conceptional change. *Learning and instruction*, 4:117–121, 1994.

[43] B.L. Whorf. *Lost generation theories of mind, language, and religion*. Popular Culture Association, University Microfilms International, Ann Arbor, Mich., 1980.

[44] Wikiquote. Conceptual model. https://en.wikiquote.org/wiki/Conceptual_model, 2017. Assessed Nov. 21, 2017.

[45] B. P. Zeigler, T. G. Kim, and H. Praehofer. *Theory of Modeling and Simulation*. Elsevier Academic Press, 2000.

Qualitative and Quantitative Models in Data Science

Bernhard Thalheim
Christian Albrechts University Kiel, Department of Computer Science, D-24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract

Data are considered to be the oil of the 21th century. They are also a rich source for many sciences, especially those that use observational data for development of an understanding behind the data. They are used to gain an insight into the discipline based on observations. This insight may result in a quantitative theory offer. The main target is however a theory that explains the data. We develop a model-backed approach to theory development based on quantitative theory offers. Models are becoming the mediator between quantitative and qualitative theories. Models can be systematically developed based on a layering approach.

1 Introduction

1.1 From Empiric Sciences to Data Science

Data science is considered to be a new stage of scientific research. Data science is based on analysis of data resources. The analysis asks the right questions with efficient processing algorithms, machine learning and cognitive computing techniques, refined statistical models, and innovative visions of how to more effectively extract the relevant data assets and scrutinise them fast with more sophisticated results. It goes beyond empirical sciences, theory-oriented research, and computational research [12]. Disciplines often use a combination of empirical research that mainly describes natural phenomena, of theory-oriented research that develops concept worlds, of computational research that simulates complex phenomena and of data exploration research. Thus, Figure 1 distinguishes four stages of sciences according to [12].

Data science discovers pattern and generates insights in data sources or data proxies. It is based on raw data and build these insights based on knowledge from the scientific discipline and application domain. It provides models, recommendations, and potential

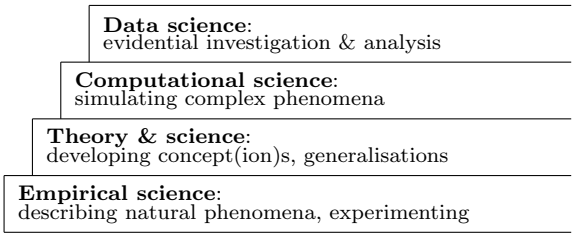

Figure 1: The four stages of scientific research

theories on how to interpret the data. It is based on a process of organising data for analysis including data proliferation, data collection organisation, cleaning, application of tools, and analysis. It may consider huge data collections as well as small data sets. The proxy data are compiled and may become 'smart' quantitative data for quantitative research. Data science is essentially the 'science' that is turning data proxies into narrative and into quantitative data. It thus develops an understanding of the data itself.

In our case, we investigate rather thin data sets. The picture is however similar to the one with very large data sets.

1.2 Data Science: From Proxy-Based Investigation to Theories

Explorative and investigative theory development (e.g. [1, 18, 20]) starts with an investigation of data sources and develop some proxy-based observation concepts and a theory offer. A theory offer is a scientific, explicit and systematic discussion of foundations and methods, with critical reflection, and a system of assured conceptions providing a holistic understanding. A theory offer is understood as the underpinning of technology and science similar to architecture theory [23] and the approaches by Vitruvius [32] and L.B. Alberti [2]. A (scientific) theory is a "systematic ideational structure of broad scope, conceived by the human imagination, that encompasses a family of empirical (experiential) laws regarding regularities existing in objects and events, both observed and posited. A scientific theory is a structure suggested by these laws and is devised to explain them in a scientifically rational manner. In attempting to explain things and events, the scientist employs (1) careful observation or experiments, (2)

Proceedings of the XIX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 09-12, 2018

reports of regularities, and (3) systematic explanatory schemes (theories)." [6] Figure 6 displays this

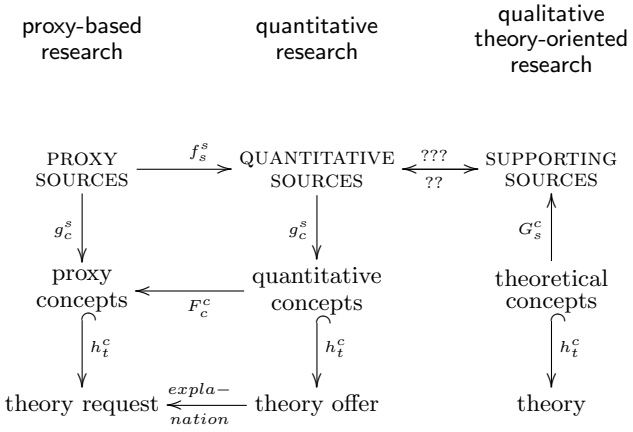

Figure 2: The current state-of-the-art in the data science

situation. We start with some data, e.g. proxy data. We (g_c^s-)derive proxy concepts (or concepts) and form some proposals for (h_t^c-)formation of a proposal of a potential explaning theory, i.e. a theory request (or theory offer). Proxy sources can be aggregated and (f_s^s-)condensed and thus become quantitative sources which are the basis for (g_c^s-)formation of quantitative concepts. These quantitative concepts are (h_t^c-)embedded into theory offers (or, resp., theory requests for proxy requests) and are the basis for a theory offer that serves as an explanation for the theory request. Quantitative concepts can be (F_c^c-)mapped back to proxy concepts. Proxy-based research and quantitative research is well-integrated if the diagram is commuting, i.e. $F_c^c(g_c^s(f_s^s(sources))) = g_c^s(sources)$.

Theories can be build on the basis of theoretical concepts which are supported by sources. Quantitative concepts should be associated with qualitative concepts. The association can only be developed in the case when the association among the data has been clarified. So far, the explanations that can be generated are mainly developed for explaining the observations made on the basis of proxies. We arrive therefore with the following problem:

***Research challenge**: How we can close the gap between quantitative theory offers and qualitative theories within the setting of data science?*

1.3 A Typical Data Science Application

Investigative modelling at CRC 1266 [1, 16] aims at exploring and explaining transformations in societies as "processes leading to a substantial and enduring re-organisation " [1] of any or all aspects of the human social, cultural, economic, and environmental relations. Proxies are observations for main concepts in Figure 3[1]. These main concepts need however a quantitative underpinning and a number of theoretical concepts.

Figure 3: Theoretical concepts to be investigated in the CRC 1266

1.4 The Storyline of the Paper

We develop an approach to data science based on models. The ladder in Figure 1 is thus supported by models in the form depicted in Figure 4

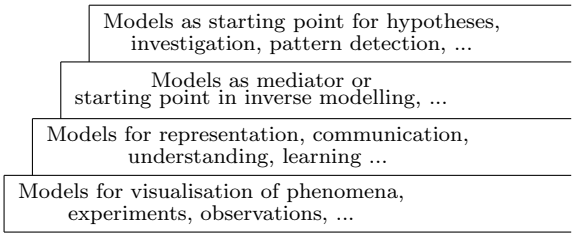

Figure 4: The four kinds of models in scientific research

Models are instruments that function in utilisation scenarios. One of these scenarios might be the development of a theory for a theory offer. We will show in the sequel how this approach can be systematically applied to development of mediating models that close the gap in Figure 6. We start with a notion of model in Section 2. Six research questions are developed which are answered in Sections 3 and 4. Next we develop a model construction approach in Section 3. Finally, we apply this approach to data science and use models as mediators in Section 4.

2 Models

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the

[1]We restrict the mindmap to main concepts and do not display the full concept network. For details see the website of the project.

solution computation of problems, accurate at least within the scope, and must function within an application scenario.

Research question 1*: Can models be used for resolving the gap between theory offers in quantitative research and theories in qualitative research?*

Consider for instance the CRC 1266 application: Transformation is considered in this context as a phenomenon that requires detailed description of features and hence quantitative data are necessary for descriptions by empirical models and simulations. Models mediate between quantitative theories and qualitative theories. Models are applied in hypothetical and investigative scenarios, should support causal reasoning as well as network-oriented reasoning, and are developed in an empiric framework.

2.1 The Notion of Model

Let us first briefly repeat our approach to the notion of model:

A **model** *is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.* [10, 27, 28]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*.

Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics [26] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

2.2 Functions of Models

Models are used as instruments in certain utilisation scenarios such as as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. They have to fulfill a number of specific functions in these scenarios. Typical functions of models as instruments in scenarios are (a) cognition, (b) explanation and demonstration, (c) indication, (d) variation and optimisation, (e) projection and construction, (f) control, (g) substitution, and (h) experimentation [31].

2.3 Model $\stackrel{!}{=}$ Normal Model $\ltimes$ Deep Model

A model consists of a normal model that is combined with some deep model similar to the visible (or exterior) and invisible parts of an iceberg [16, 29, 30]. The deep model reflects (α) the intentions of the problem world, (β) the accepted understanding within the community of practice, (γ) the context of the application domain, (ϵ) the background that is commonly accepted in the problem and application domain, and (ε) the general restrictions to the origins that might be considered. The deep model allows to derive a part of the justification and adequacy of a model.

The normal model reflects the collection of origins that are currently under consideration. Both the deep and the normal model are dependent on the functions that a model should play in application scenarios. Development of models is often restricted to development of a normal model under the assumption that the deep model is given by the modelling method, the context, the community of practice, and the function that the model has to play in a given scenario. The modelling methods also determined the methods that are used for model development. It might also include the utilisation methods.

Research question 2*: Can we separate the deep model from the normal model in such a way that the model can be composed of the deep model and of the normal model?*

If the answer to this question is positive then we might try to consider the model as an enhancement of the deep model. In this case, the development of a model can be layered.

Research question 3*: How can be development of a model layered into the development of a deep model followed by the development of the normal model?*

We may now ask us whether this approach is universal. The answer will be negative if the notion of model also includes models with intractable deep models, e.g. for metaphors, parables, or physical representations. We might however concentrate on models in sciences and technology.

2.4 Model Suites

Models may be given as a holistic instrument that combines all aspects into one model. The approach is often too challenging. A simpler approach is the consideration of a model as a model suite (or model ensemble) [8, 25] that consists of a coherent collection of models which are representing different points of view and attention. It is extended by an explicit association or collaboration schema among the models, controllers that maintain consistency or coherence of the model suite, application schemata for explicit maintenance and evolution of the model suite, and tracers for the establishment of the coherence.

Research question 4*: Exists there a systematic approach to model development that is based on a co-development of normal models and deep models? Which additional models should be integrated into the model suite?*

2.5 Generic and Specific Models

Model development does not start from scratch. We often start with generic models. A *generic model* [16] is a model which broadly satisfies the purpose and broadly functions in the given utilisation scenario. It is later tailored to suit the particular purpose and function. Generic models can be calibrated to specific models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

Research question 5*: Can we develop normal models starting with a generic model and are they still integratable with the deep model?*

If the answer is positive then generic normal models can be calibrated to specific normal models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

2.6 Data Mining as a Success Story

In [16], we developed the V-model to data mining based on a separation of the data mining process into the domain perspective with its domain world of users from a community of practice, the modelling perspective with a model world, and the data perspective in a data world. Users are interested in solution of certain problems an application world, share the context and also the scientific and technological background. The classical data model mining process uses these perspectives for a stepwise development of a model that allows to solve their problems, e.g. (1) by modelling the problem and the issues under consideration, (2) by preparing the data world for development and enhancement of models, (3) by applying data mining algorithmics for pattern detection and model development, and (4) by using the model for development of some solution for the problems and thus augmenting the application domain world. The model development process itself can be understood a multi-iterative guided procedure that has its flow of activities.

This approach extend the classical CRISP framework [4] and other approaches to systematic data mining, e.g. [15]. Each of these approaches has its capacity and potential as well as its threats and limitations. The question is now:

Research question 6*: Can we generalise a data mining setting to model development for data science in such a way that models mediate between theory offers and theories?*

Data analysis and model suite development currently inherit success stories in a similar application. These success stories follow some kind of a metapattern and result in a specific data mining process as an example of a modelling method or modelling mould. Data mining starts with exploring and understanding the data mining project, its data, and a general setting of principles of modelling. After the project and the nature of the data is understood, data are preprocessed and prepared for the application of algorithms. Next pattern within the data are investigated. This pattern analysis results in clusters, maps, association rules, and some deviation analysis, i.e. we develop a model on the data space. This model is then used for development of explanations, e.g. via decision trees, (Bayesian) classifiers, regression, and (rule) learning approaches. We develop a second model on top of the first model. Next, the data space is considered in a general form by prediction analysis, e.g. nearest neighbour predictors, (artificial) neural networks, support vector machines, or other ensemble methods. The result is another model. Finally, the models are evaluated and potentially deployed. If the evaluation does show that the models satisfyquality criteria, we revise the models.

3 A General Model Composition Approach

Engineering and software engineering (e.g. [13, 22]) distinguish between the five primary development dimensions:

Activities ('how') describe the way how the work is performed and the practises for the work.

Work products ('what') are the result of the specification and are used during specification.

Roles ('who') describe obligations and permissions, the involvement of actors in the specification process.

Aspects ('where') are used for separation of concern during the specification process.

Resources ('on which basis') are the basis for the specification.

Since a model (suite) is also a work product we may refine this approach to model engineering. The modelling method we will use is similar to the modelling method in mathematics [3].

3.1 Separating Modelling into Layers

Model suite development results in a number of models: deep, generic, specific, and normal models. Since any model has its deep elements we may start with development of this deep model. In many cases, we might use reference model or generic models (or tactical model frames like those we use in data mining and analysis). Let us investigate whether a layered approach can be applied within a five-layered separation of concern and aspects. The separation into layers generalises the approaches used in mathematics, e.g. separation by Craig's interpolation theorems.

Classical modelling often intentionally presupposes the initialisation and intrinsic layers and assumes that these layers cannot be reconsidered and specifically changed according to the functions. We loose, however, the understanding of the model and cannot understand why the model is dependent without an understanding of these layers.

(I) The Initialisation Layer

The W*H specification pattern [9]. can be applied to model *initialisation* as well an includes then the following set of statements:

- a plan, function, and purpose dimension (model as a conception: 'wherefore', 'why', 'to what place or end', 'for when', 'for which reason') within a scenario in which the model is going to be used as an instrument,
- a user or CoP dimension ('who', 'by whom', 'to whom', 'whichever') that describes the task portfolio in the CoP and profile of users including beliefs, desires and intentions,
- an application and a problem dimension ('in what particular or respect' , 'from which', 'for what', 'where', 'whence'), and
- additionally, the added value dimension (evaluation).

The initialisation layer may also be enhanced by a contrast space for user-related separation of a model and a relevance space that is dependent on the user [11]. The contrast and relevance spaces as a form of mind-setting also define what is not of interest.

(II) The Enabling Setup Layer

The *intrinsic setup* defines the opportunity space and the infrastructure for the model. The results will be from one side a deep model and from the other side a modelling framework or modelling mould that guides and govern next activities. We define the *context* and the most of the *background* (the grounding (paradigms, postulates, restrictions, theories, culture, foundations) and the basis (assumptions, concept world, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense)) of the model. The context, extrinsic, and strategic dimension answers question like 'at or towards which', 'where about', 'to what place or situation', and 'when'.

Additionally, we decide which methodology and environment seem to be the most effective and purposeful. The development and deployment dimension ('how', 'whence', 'what in', 'what out', 'where') defines the modelling methodology, i.e the *modelling mould*.

(III) The Extrinsic Source Reflection Layer

strategem We separate the deep model elements from elements of the *normal model*. According to the model function, the normal model represents *extrinsic* elements of potential origins based on their content and thus answers questions such as 'what', 'with which', and 'by means of which'. It reflects the extrinsic theory essentials that are necessarily to be represented, e.g. conceptions or pre-conceptions from the theory that is underpinning the application. The normal model can be build from scratch ('greenfield' modelling). It is more usual based on the experience gained so far. The latter case thus starts with a generic or reference model that might incorporate parameters. The extrinsic source reflection layer can be understood as a tactical layer.

(IV) The Operational Customisation Layer

Generic or general normal models are adjusted to those that a best fitted to those origins that are considered for the application. The *operational customisation layer* is sometimes holistically handled with extrinsic reflection. Inverse modelling is the general case however. It instantiates parameters, adapts the normal model to those origins (or data sources) that are really under consideration, prepares the model for the special use and to the special - most appropriate - solution, and integrates the deep model with the normal model. The normal model is typically pruned in order to become simpler (Solomonoff and Occam principled) more deviation- or error-prone. The (normal) model might be enhanced by concepts and thus become a *conceptual model*.

(V) The Delivery and Product Layer

The *final result* of the modelling process is a model suite that is adequate for origins, properly justified, and sufficient. We cannot expect that one singleton model is the best instrument for all members of the community of practice. A sophisticated model that integrates deep and specific normal models is delivered to some members. An informative model that is derived from this model can be better for other CoP members. Models delivered in the finalisation space are often enhanced by additional annotations, e.g. relating the model to the demands for members of the CoP by answering the 'with', 'by which', 'by whom', 'to whom', 'whichever', 'what in', and 'what

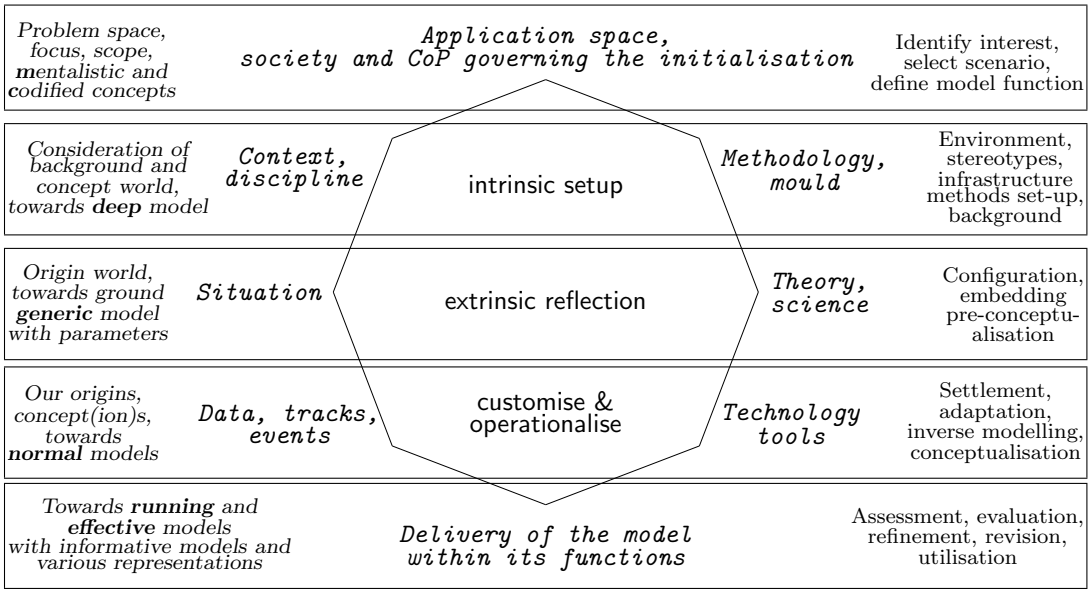

Figure 5: The layered model development framework

out' questions. At the delivery and product layer we thus generate a number of associated models.

3.2 Systematic Model Development

We combine now the five layers in Figure 5. At the left side we represent the issues for the model. The right side displays the activities and methods for the development. The corners of the octagon represent the starting and final stages as well as sources and enablers of the intermediate stages. We restrict the picture to the layered model development process.

4 Models as Mediating Instruments

Model-backed reasoning is thus some kind of revisable reasoning depending on the stages of knowledge. Modelling as a process starting with suites of generic models and revisable refinement according to data on hand. It should support handling of uncertainties and incompleteness of any kind and must thus make use of an integrated data management. Therefore, model-backed reasoning is properly based on layered model development.

4.1 Towards Models as Mediators between Theory Offers and Theories

Models can be used to render the theory offer. At the same time models may also render a theory. We claim that these two views can be integrated. The model functions thus as mediator [17]. The rendering procedures are however different.

We envision that this integration can be based on the mappings in Figure 6. Models can be understood as being composed of model concepts that are supported by data sources. We can now distinguish f^x_y-mappings at the same level, g^x_y-mappings between sources and concept, G^x_y-mappings from concepts to supporting sources, and h^x_y-embedding mappings from concepts to theory offers, models, or theories. Quantitative concepts are indicators or general quantitative properties. Model concepts are already abstractions from those quantitative concepts. Theoretical concepts in Figure 3 are elements of a theory that is currently under development. The research task is the harmonisation of the two mappings $f^{(c,q)}_{(c,m)}$ and $f^{(c,t)}_{(c,m)}$. This harmonisation can be based on the mappings for supporting resources $f^{(s,q)}_{(s,m)}$ and $f^{(s,q)}_{(s,t)}$ if some commuting diagram properties are valid for model concepts and the model.

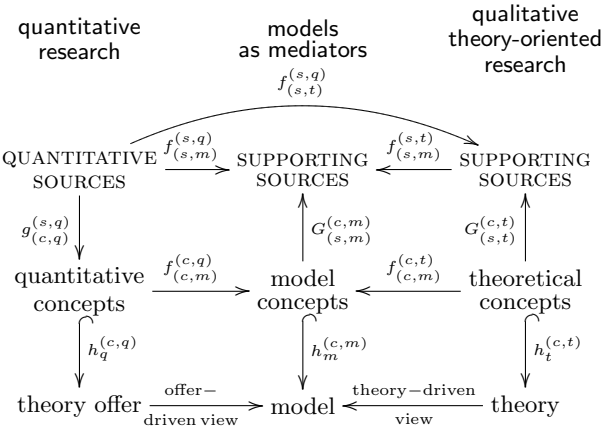

Figure 6: Models as integrating and mediating instrument for data science

We thus use the model as some kind of twofold medium which has 'Janus' head behaviour: it is both (I) a view of the theory-offer and (II) a view of the theory. It is (i) a model *for* the theory offer as a reflective "epistemic thing" [21] of discovery and a presupposition and (ii) a model *of* a theory as a specific viewpoint representation. It thus comprehends what has been developed for theory offers and supports explanations of the theory. The development of

a model has however also a feedback turn both on the theory offer and the theory. The model is then at the same time an instrument, a mediator, a companion, a middle, and a medium [5]. The model thus becomes an investigation instrument.

4.2 Evidence-Based Reasoning in Data Science

Let us finally discuss an obstacle of quantitative research that results in some obstinacy of models. Theory and model development are in both cases evidence-based due to the way how they are derived from proxies. The O(bservation)-C(laims/Hypotheses)-E(vidence)-R(easoning) pattern [7] starts with some observations and detection of hypotheses about these observations. Hypotheses are transformed into claims and research questions that form a research agenda. Evidences are then either systematically elicited from data, from previous investigations, or from the belief and knowledge space. Reasoning should then connect evidences to the claims. The results is some kind of Bayesian formulas representing the claim with the evidence. Evidence-based reasoning combines therefore inductive and abductive reasoning. It is enhanced by Occam's razor approaches [19] that allow to finalise the model development. It can be combined with Solomonoff induction [24] that enhances a result (1) by conduction of experiments that will test the claims and (2) by provisionally accepting the claim if the experiment confirms the claims. The reasoning schema follows the *induction/abduction-retrospection-observation_concepts-theory_offer* pattern. It can be combined with Epicurus' principle of keeping multiple explanations that allowing consideration of several models and theories as long as they are consistent with the observations.

OCER pattern are the basis for evidence-based proxy reasoning (e.g. in the CRC 1266 [1]) that follow positive evidences. Evidence-based reasoning is based on the following principles:

1. Models represent only acceptable possibilities (each model captures a distinct set of possibilities to which the current description refers) which are consistent with the premises and the knowledge gained so far what makes them intrinsically uncertain because they mirror only some properties they represent.
2. Models are proxy-driven (the structure of the model corresponds to the proxies it represents. They might also include abstractions such as negation.
3. Models represent only what has been observed and not what is false in contrast to fully explicit models (that represent too what is false).
4. The more proxies that are considered, and the richer those models are, the more accurate the world view is.
5. We use pragmatic reasoning schemata (e.g. A causes B; B prevents C; therefore, A prevents C).

Evidence-based reasoning thus makes a difference between deterministic conclusions (A cause B to occur: given A then B occurs) and ordered sets of possibilities (A enables B to occur: given A then it is possible for B to occur).

5 Conclusion

Models are one of the main instruments in science and technology. They support reasoning in various forms, e.g. by systematic revisable modelling based on data and as an associated collection of models;

This paper develops an approach for development of fully fledged models (a) with extrinsic parts similar to usual (normal) models and (b) with intrinsic parts which are typically hidden in the modelling approach, in the background and context of the model, and in the intentions behind the model. While making this explicit, we are able to use a model as a problem description and to compute the solution of the problem under consideration directly from the model. The paper presents the first part of this solution. The development of corresponding tools is the topic of a forthcoming paper.

The presented layered approach should not be applied as a 1-2-3-4-5 waterfall sequence of activities. Rather, model development and model utilisation use an evolutionary approach that returns to previous steps whenever sufficiency characteristics of models become problematic within the application domain. The layers can however be considered as phases of development. We notice that our layered approach also supports model revision and model evolution. It can also be used for model migration and model reengineering. The layered approach seems to be combinable with modelling cultures, e.g. those that can be observed for our first case study [14]. Figure 5 represents the 'greenfield' or glassbox development with model development from scratch. A similar picture can be given for 'brownfield' model development, i.e. model redevelopment, revision, and migration. Blackbox model representation uses only the first and the last layers.

The layered approach is based on a separation of concern within an initialisation layer, within an intrinsic and this implicit setup layer, within an extrinsic and thus explicit source reflection layer, within an operational customisation layer, and finally with a model delivery layer.

We conjecture that a similar layered approach can be developed for model utilisation. The layers will be different but oriented on the usage of a model as an instrument. It can also follow the solution story *initialise-prepare-investigate-do-deliver* that is used for layered model development.

The main result of the paper is a systematic approach that closes the gap between theory offers and theories in data science. The approach is based on the mediating function of models between theories and theory offers.

References

[1] CRC 1266. Scales of transformation - Human-environmental interaction in prehistoric and archaic societies. Collaborative Research Centre. http://www.sfb1266.uni-kiel.de/en/, accessed May 13, 2018.

[2] L.B. Alberti. *On the art of building in ten books*. MIT Press, Cambridge. Promulgated in 1475, published in 1485, 1988.

[3] R. Berghammer and B. Thalheim. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Methodenbasierte mathematische Modellierung mit Relationenalgebren, pages 67–106. De Gryuter, Boston, 2015.

[4] M.R. Berthold, C. Borgelt, F. Höppner, and F. Klawonn. *Guide to intelligent data analysis*. Springer, London, 2010.

[5] C. Blättler. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Das Modell als Medium. Wissenschaftsphilosophische Überlegungen, pages 107–137. De Gruyter, Boston, 2015.

[6] S. Bosco, L. Braucher, and M. Wiechec. *Encyclopedia Britannica, Ultimate Reference Suite*. Merriam-Webster, 2015.

[7] E. Brunsell. Claims, evidence and reasoning. www.explainthatstuff.com, accessed May 13, 2018.

[8] A. Dahanayake and B. Thalheim. Co-evolution of (information) system models. In *EMMSAD 2010*, volume 50 of *LNBIP*, pages 314–326. Springer, 2010.

[9] A. Dahanayake and B. Thalheim. Development of conceptual models and the knowledge background provided by the rigor cycle in design science. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 3–28. College Publications, 2018.

[10] D. Embley and B. Thalheim, editors. *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*. Springer, 2011.

[11] B. Van Fraassen. *The scientific image*. Clarendon Press, Oxford, 1980.

[12] J. Gray. eScience: A transformed scientific method. Technical report, Talk given Jan 11, 2007. Edited by T. Hey, S. Tansley, and K. Tolle. http://research.microsoft.com/en-us/um/people/gray/talks/NRC-CSTB_eScience.ppt, Microsoft Research Publications, 2007.

[13] ISO/IEC. Information technology - process assessment - part 5: An exemplar process assessment model. FCD 15504-5:2004, 2004. not publicly available.

[14] H. Jaakkola and B. Thalheim. Modelling cultures. In *EJC*, forthcoming, 2018.

[15] K. Jannaschk. *Infrastruktur für ein Data Mining Design Framework*. PhD thesis, Christian-Albrechts University, Kiel, 2017.

[16] Y. Kropp and B. Thalheim. Data mining design and systematic modelling. In *Proc. DAMDID/RCDL'17*, pages 349–356, Moscov, 2017. FRC CSC RAS.

[17] M.S. Morgan and M. Morrison, editors. *Models as mediators*. Cambridge Press, 1999.

[18] O. Nakoinz and D. Knitter. *Modelling Human Behaviour in Landscapes*. Springer, 2016.

[19] W. Ockham. *Philosophical writings: A selection*. Hackett Publishing Company, Indianapolis, translated and edited from writings, early 1300 edition, 1990.

[20] A. Raab-Düsterhöft. Integrating social media information into the digital forensiv investigation process. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 29–43. College Publications, 2018.

[21] H.-J. Rheinberger. *Experiment - Differenz - Schrift*. Basilisken-Presse, Marburg an der Lahn, 1992.

[22] A. Samuel and J. Weir. *Introduction to Engineering: Modelling, Synthesis and Problem Solving Strategies*. Elsevier, Amsterdam, 2000.

[23] G. Semper. *Die vier Elemente der Baukunst*. Braunschweig, 1851.

[24] R.J. Solomonoff. Complexity-based induction systems: Comparisons and convergence theorems. *IEEE Transactions on Information Theory*, IT-24:422–432, 1978.

[25] B. Thalheim. *The Conceptual Framework to Multi-Layered Database Modelling based on Model Suites*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 116–134. IOS Press, 2010.

[26] B. Thalheim. Towards a theory of conceptual modelling. *Journal of Universal Computer Science*, 16(20):3102–3137, 2010. http://www.jucs.org/jucs_16_20/towards_a_theory_of.

[27] B. Thalheim. The conceptual model ≡ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[28] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. Springer US, 2017.

[29] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[30] B. Thalheim. Normal models and their modelling matrix. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 44–72. College Publications, 2018.

[31] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[32] Vitruvius. *The ten books on architecture (De re aedificatoria)*. Oxford University Press, London, 1914.

Modelling Cultures

Hannu JAAKKOLA [a,1] and Bernhard THALHEIM [b,2]
[a] *Tampere University of Technology, P.O.Box 300, FI-28101 Pori, Finland*
[b] *Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany*

Abstract. Modelling is an essential part of information systems development. Models are used for communication between interest groups and inside development teams. Models are also used for transferring baseline artefacts between development phases. Models are mainly developed by humans, which represent certain cultures - national, enterprise, professional, team, project etc. Because of that we claim that models, as well as many other information systems related aretfacts are culture dependent. The models are born in certain context and these must be also interpreted by taking the original context into account. In our earlier studies we have analysed the effect of culture in information systems development: culture related aspects in general level, in information search and interaction and in web information systems. These studies acts as a basement for this paper having focus in modelling. Because of that we shortly answer to the question "How cultures differ from each other". This reviews and synthesisis generally accepted frameworks for cultural analysis. In addition we shortly open the results of our earlier studies. Because modelling is a human activity, as well as information systems are used by humans, we feel important to build a model of humans in information systems development an use context. The findings of culture analysis are transferred to modelling practices via our framework that defines model as an instrument transferring elements of its development context to the models - we discuss about the roles of normal models, deep models and modelling matrix. Finally we will concentrate in the problems of cross-cultural modelling using selected national cultures as an example.

Keywords. culture-dependence of modelling; deep model, modelling matrix; multi-cultural system development;

1. Cultural Differences

1.1. The Layered and Dimensions Approaches

G. Hofstede [12] defines culture adapting the layered structure of Maslov pyramide (Figure 1). He uses the term "mental program" to describe the characteristics of each layer. The lowest level - operating system - is common for all humans. The second layer - collective program - is learned and remains same in a collective group of people and indicates the culture. The "onion model" on the right side of the Figure 1 describes the main elements of the culture. Values are the core of the culture. Rituals are collective

[1] hannu.jaakkola@tut.fi
[2] thalheim@is.informatik.uni-kiel.de

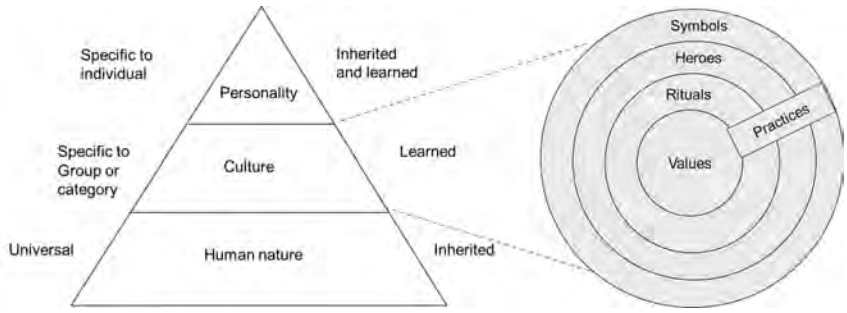

Figure 1. The layered structure of culture

activities that are essential in a culture and indicate the membership of the group. Heroes are highly prized examples in a culture and indicate positive values of it. Symbols are words, gestures and objects that are common for those share the culture. Practices are manifestations of all other elements of a culture.

R. Lewis [21,22] applies the pyramid model in his culture analysis (Figure 2). As seen in the Figure the culture layers between Finland and Germany are different. Both countries in Lewis' classification (discussed later in this paper) are close to each other and belong to the group of linear-active and data-oriented cultures. In spite of that, the values and core beliefs - the core of the culture - are different.

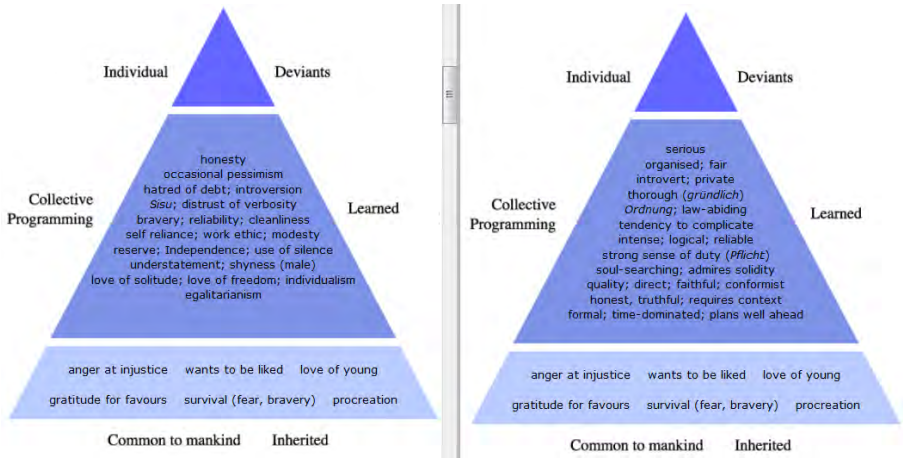

Figure 2. Values and core beliefs of Finland and Germany according to [22]

In our paper [17] we have introduced three methods to be used in recognizing cultural differences cultural differences: the 6D model of Hofstede, Lewis "triangle model" and Hall's high/low context culture model. In addition to these there are several other ones; most of these overlap with the former ones and do not provide additional value to the analysis. Figure 4 illustrates the classification principles of Hofstede's model.

The model of Hofstede basis on the analysis of six cultural dimensions [11,12]:

- Power Distance (PDI): the extent to which power differences are accepted.

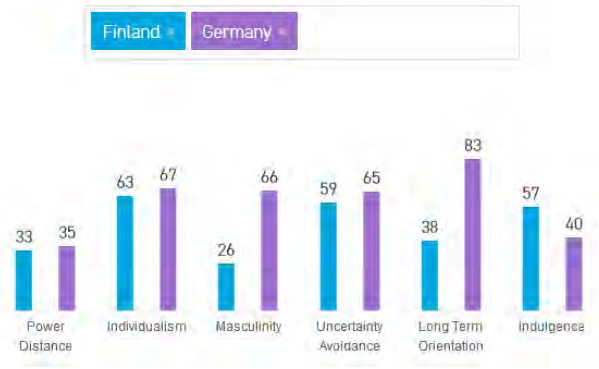

Figure 3. The Hofstede 6D model [11]

- Individualism / Collectivism (IDV): the extent to which a society emphasises the individual or the group.
- Masculinity / Femininity (MAS): refers to the general values in the society - hard/ soft values.
- Uncertainty avoidance (UAI): refers to the extent that individuals in a culture are comfortable (or uncomfortable) with unstructured situations.
- Long-term / Short term orientation (LTO): refers to the extent to which the delayed grati?cation of material, social, and emotional needs are accepted.
- Indulgence / Restraint (IVR): acceptance of enjoying life and having fun vs. controlling the life by strict social norms.

The country comparison tool[3] provides access to the database collected by Hofstede during the decades. The data covers culture values of most countries in the world. The left side of Figure 4 indicates the similarity of Finland and Germany. Meaningful difference is in LTO and MAS values, slight difference in IDG. The German work to reach results, which are more long-range than the Finns (LTO). They appreciate material values more than Finns (MAS) and live in the atmosphere, which is more puritan than in Finland (IVR).

1.2. Interaction and Collaboration

The Lewis' model [21,22] focuses in analyzing interaction and collaboration activities of people. The nationalities locate in the corners and sides of a triangle. The corners represent the basic stereotypes: linear-active, multi-active and reactive. Linear-active cultures are data oriented (decisions are based on facts and official sources) and can be descried by terms cool, factual and decisive planners. Reactive cultures are "listeners" - they base their behavior in more rich base of information sources (oral information from social networks, family, friends, ...) than people in data oriented cultures do. In communication they are not active members of the interaction; listening and reacting is typical to them in dialogues. Terms courteous, amiable, accommodating, compromiser and good listener describe people in reactive cultures. They are also usually members of collective cultures in Hofstede's classification. Multi-active cultures are dialogue oriented. Like

[3] https://www.hofstede-insights.com/product/compare-countries/

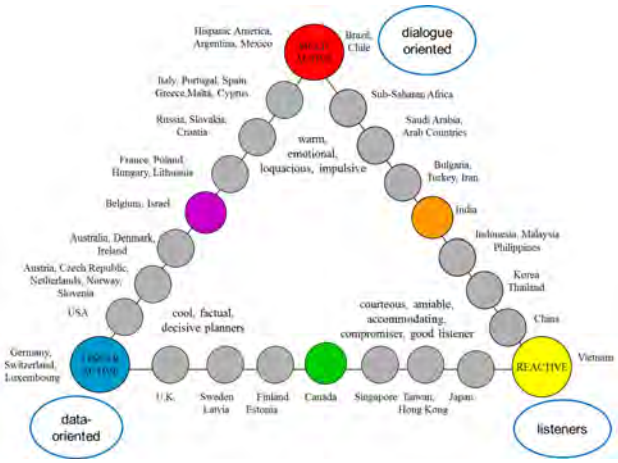

Figure 4. The Lewis' triangle model (modified from [21])

people in reactive cultures they use rich set of information sources, but especially prefer oral information. Multi-active characteristic means doing several things at once, being extrovert and being active member of dialogues. Terms warm, emotional, loquacious and impulsive describes them.

1.3. Context Dependence

The Hall's [9] model divides cultures according to the importance of context recognition in communication. Context means the extent of "wordless" communication included in the messages. High importance of context in communication indicates the importance of the membership in a collective group - i.e. collectivistic group culture in Hofstede's classification. In high context cultures, the meaning of the message relates to the context in which it is presented. The group members know a variety of details included in the message without explicit messaging. The low context cultures are opposite. These cultures prefer punctual and clear messaging. All information is clearly included in the message and need for knowing the context is minimal. Low importance of context indicates highly individualistic society in Hofstede's classification. Finland, as well as all Scandinavian countries, and Germany belong to the category of low context cultures; Japan and Arab counties instead are typical high context cultures. Somewhere in the middle of the continuum are USA and England, in which it is also typical to use words and sentences with hidden meaning.

1.4. The Storyline of the Paper

In this paper we start with an investigation whether cultures have an impact on models, i.e. on model development and model utilisation. Models can, for instance, be used as a means for communication. A hypothesis could be that models are a stability kernel among different people. The opposite hypothesis states that models are culture dependent. Section 2 discusses the relationship between modelling and cultures. Section 3 introduces a general model notion and a separation between the normal model and the deep model. We illustrate the cultural dependence for different kinds of schemata. Section 4

investigates the cultural differences for modelling styles. It is shown that the models developed in different cultures might also be different although the application is the same. Culture thus determines how models are developed and how models are used.

2. Cultures Influence Human Modelling Activities

2.1. Modelling is Different Worldwide

While zapping through textbooks from different countries on database analysis, design, and development we observe that the same topics and the same application tasks result in rather different database schemata. So, what causes these differences? What styles are preferred where? Under which circumstances one model is better than the other? Which detailedness is the best? Shall we concentrate on typical structures only? How exceptional cases are handled?

We observe also different modelling pattern and styles beside language differences (ER-like, UML. ORM, NIAM, IDEF, etc.). Students who got first lectures in object-orientation develop completely different schemata than those who got introduced to functional or procedural paradigms. Some companies like Ploenzke or SAP have a completely different way of representing the same application.

Moreover, the same language paradigm is often modified and extended. For instance, there are more than 50 extensions of the entity-relationship approach. Most of them are actually incompatible. Many modelling languages exist in a large variety of dialects what makes knowledge transfer and communication difficult. One reason might be that the ways of thinking, of modelling, of controlling, of working, and of supporting are different in different communities and thus result in different environments and thus in different cultures. Another reason might be the insufficiency of a language. In this case, we can use language pluralism and develop model suites [29].

Modelling might also follow different paradigms and postulates. It might use different not combinable theories. Modelling is biased by the developers and their educational and professional background [33]. It is typically laden[4] by concepts that are to be represented, by its community, by the context into which the model is set, and by the way of utilising a model. If we compare these factors influencing modelling with the culture notion then we realise that all these factors are culture-driven.

So, we may conclude that organisation, professional, educational, and finally national cultures influence the outlook, the content, the adequacy and dependability of a model. It is not only the behaviour of people that is governed by the cultures but also the development and utilisation of tools that is governed by the culture. Models are instruments that are used in utilisation scenarios. Communication is one of the main scenarios. Models are used similar to utterances in natural languages in this scenario.

2.2. Culture Sensitivity in Information Systems Development

We have handled the topics related to information systems (IS) development in multicultural context in several papers. The papers handle information systems development from different points of view. Culture related aspects affect in both the development and

[4]This concept has been considered in detail by H. Kangassalo (and J. Palomäki) in the EJC'15 keynote "Definitional conceptual schema - The core for thinking, learning, and communication" at June 11, 2015.

the use of IS. The development work is made in multi-cultural distributed teams, in which it is important to recognize the dynamics of the team in decisions related organizing the work, leading the team and managing the development project. Transfer towards cloud based ecosystems and web information systems (WIS) makes recognition of the end-user base more difficult. In requirements engineering phase we have more and more often "faceless" clients from different cultures and from different parts of the world that must be served by the WIS [15].

Our earlier studies cover general aspects in IS development [14], information and query-answer related aspects [17] and web information systems design related aspects including database design and conceptual modelling [16]. These papers provide a "handbook" type list of findings to support IS development in multi-cultural context. Our analysis applies the interpretations of human behavior using Hofstede's dimensions and Lewis analysis. It also acts as an evidence to the applicability of culture analysis and stereotyping methods to guide IS development for multi-cultural context. The realization of the findings is included in the requirements engineering phase, which transfers them to non-functional requirements in the requirements specification of the IS.

We have approached the topic via Hofstede's and Lewis' models. Hall's model provides some new aspects to the analysis, which are worth of more studies. Low context cultures are tended to demand exact communication. Our hypothesis is that IS development in such cultures indicate clear and unambiguous user interface, whereas high context cultures are tended to accept some ambiguities and complexity in it. Low context indicates linearity, high context multi-activity.

2.3. Information Systems Modelling and Culture

In IS projects models are means for communication - transferring duties and work items trough the life cycle of the IS and supporting interaction between the interest groups. We defined modelling to be a kind of solution to the problems of communication. Modelling languages are culture independent unlike natural languages. However, our hypothesis is that the use of them and the structure of the models indicates culture of its user. In IS development models transfer system related knowledge between interest groups. Because most of the modelling techniques used in practical work are semi-formal, the lacking exactness opens door for misunderstandings. In addition the sender's and receiver's ability to interpret the model may vary; one of the reasons is culture. Interpretation of the models is also context sensitive (i.e. in different contexts the interpretation may vary). The model itself is a construction of concepts and individuals according to their internal concept handling mechanism interpret it. In our paper [13] we introduced a hypothesis that also this mechanism is culture dependent - that what a Finn finds in a (conceptual) model would be different to the findings of a German or Japanese. In the same paper we have listed problems related to communication and collaboration in multi-cultural context: (1) behavioral patterns of people are different, (2) concept creation and handling is different, (3) language of communication is different, (4) communication includes opportunity to serious misunderstandings and (5) transformations (transferring the message from one language to other) may change the meaning of the message. All these problems fit to IS modelling, too.

2.4. Modelling of a Human Being, Team Dynamics and Organization Culture

In culture adaptable information systems development context there have been efforts to model its user. In adaptable IS the system includes a model of the user. If this "user model" is equal to the real behavior of the user, the system may adapt its operations according to the expectations of different users. In culture adaptable IS this model includes culture related factors.

One of the best-known model is MOCCA environment developed by K. Reinecke [27]. MOCCA is an application that can adapt ten different aspects of its UI with 39366 combination possibilities altogether. MOCCA acts also as an example of the technical implementation of the flexible user interface in information systems design. The user model takes into account the cultural background of the user including user's cultural adaptation because of the influence of foreign cultures. The user profile basis on the following parameter: MAS, UAI, PDI, LTO, IDV, year of birth, political orientation, social structure, religion, education level, familiarity to certain form of education, computer literacy and gender. External dependencies cover nationality of the person, his/her mother's and father's nationality and language skills (mother tongue, foreign languages). Dynamics of the model basis on the former length of stay under the effect of the foreign culture.

M. Phaedra and M. Permanand [26] have introduced a student model that takes into account his/her demographic factor values. The person (student) has simple arguments: identification, age and gender. The dimensions of the model fall into five categories that describe particular contextual categories: geographical aspects, religion, ethnic background, education level (including school - note the importance of school as a root of an important source of information in dialogue oriented and reactive cultures), and particular physical environment settings and terrains. External properties cover - as in MOCCA - parent data including their occupation (social group) and native language. The model does not include any aspects creating dynamics, if changes in the parameter values not counted. The model neither includes any cultural factors derived from stereotype models.

G. Dafoulas and L. Macaulay [6] have modelled the dynamics of multi-cultural virtual software development teams. The model lists a variety of factors that to take into account in management of the team and organizing the work in it. They emphasize that each individual is a member of multiple cultures (*Cultural profile* category): one or more national/ethnic cultures, one or more professional cultures, a functional culture, a corporate culture, and a team culture, among others combined to individual (personal) characteristics. They have seen, especially in multi-cultural distributed teamwork the importance of professional and functional culture: "software professionals worldwide belong the computing subculture, which is stronger than any other culture". A Russian software engineer (professional culture) would be more similar to an American peer than to a Russian marketing manager (functional culture).The model of cultural dimensions in virtual software teams does not specify the properties of an individual (professional) but a roadmap to manage the team. *Human resources* category includes PDI, UAI, IDV, time difference between members, trust level between team members, concept of space (Lewis) and material power (goods that create or indicate power). The required skills interact with human resources. The required skills category covers communication skills, participation activity, leadership, conflict resolution, problem solving, decision-making, goal setting and motivation. *Team development* category covers the improvement of required skills by taking into account first the team profile (diversity level), the role profile

(preferences) and finally task profile (requirements); the improvement is a continuous iterative process. Although this model belongs more into the category of "management and leadership models" it points out important aspects that indicate personal properties to be included in the model of a software engineer.

Our paper [14] includes a simple user model structured as a mind map. This model indicates the important factors of an individual to be taken into account in developing adaptive information systems. The personal properties category of the model covers personality profile (nine general factors and three dialogue preference related factors), work profile (six factors) and education profile (three factors). The portfolio category includes task related parameters, user involvement description, type of collaboration and restrictions to take into account.

E.G. Blanchard et al. [5] use very similar approach in their conference paper related to intercultural communication. They have found a remarkable (literature based) evidence which shows that the way people interpret and react to their environment significantly differs from one culture to another and that wide range of human activities and situations influenced by culture. In spite of that, the human-related technologies have not accounted for culture. Western context dominates in design and solutions, which are tested and validated on Western samples. In their paper Blanchard et al. (2013) introduce a simplified conceptual model of intercultural communication. Cultural elements concept class in the model covers cognitive cultural elements and cultural non-verbal communication (body language) related aspects. Non-cultural (innate) elements concept class includes behavioral primitives (gestures, postures, facial expressions) and some innate non-verbal communication elements. Additional concept classes cover the role of context, culture and cultural group, enculturated individual aspects, cultural group cohesion and a variety of descriptors.

B.S. Parumasur [25] handles the problems related to organizational development (OD). The paper states that American and European consultants have developed most of the OD practices. Because of that, cultures collide in different cultures. Contextualized and customized approach is needed: The skills readiness acquired at school varies (abstract thinking, team skills, entrepreneurship, technical, language, ...), motivation factors vary between cultures (emergent/mature), gap to the welfare plays an important role. In all change and improvement processes gap between local values and proposed interventions must be recognized. However, the evolution of the political and economic climate changes the values rapidly; globalisation leads to adoption of foreign influence (see [27]). The paper concludes to a model, which indicates organisation's readiness to changes; the model applies Hofstede's 6D model in the following way. PDI: High PDI indicates acceptance of social and economic gaps, acceptance of inequality, acceptance of centralization, valuation and respect to authorities and hierarchical relationships. In high PDI cultures close supervision is needed. PDI indicates also suitability of participative / non participative decision making. Large PDI is associated with collectivism and (lower national wealth), small to individualism (and greater national wealth). UAI: High UAI indicates resistance to changes. Combined with high PDI it reflects the responsibility of an organization instead of individuals. UAI links to formalization, the need for formal rules and specialization. IDV: IDV indicates the acceptance of person level benefits (salary differentiation based on productivity) and the importance of interpersonal relationships. High IDV leads to a need to explain every act in terms of self-interest. IDV relates also to the directness / informality of feedback (performance improvement vs. de-

stroying the harmony in certain conflict situations) and the aim to avoid face loosing (in a group. MAS: Masculinity relates to career advancement and salary vs. social aspects of work. It indicates also differentiation of gender roles. Relatively high MAS and Weak UAI justifies the high level of achievement motivation.

2.5. Revisiting Cultural Studies

The different approaches to dimensions of cultures have been combined, systematised, and generalised in [17]. The result is a 11-dimensional Kiviat graph in Figure 5. The dimensions could be used to derive guidelines for web information system development what has been illustrated by the dimension values for Finland, Germany, and Japan.

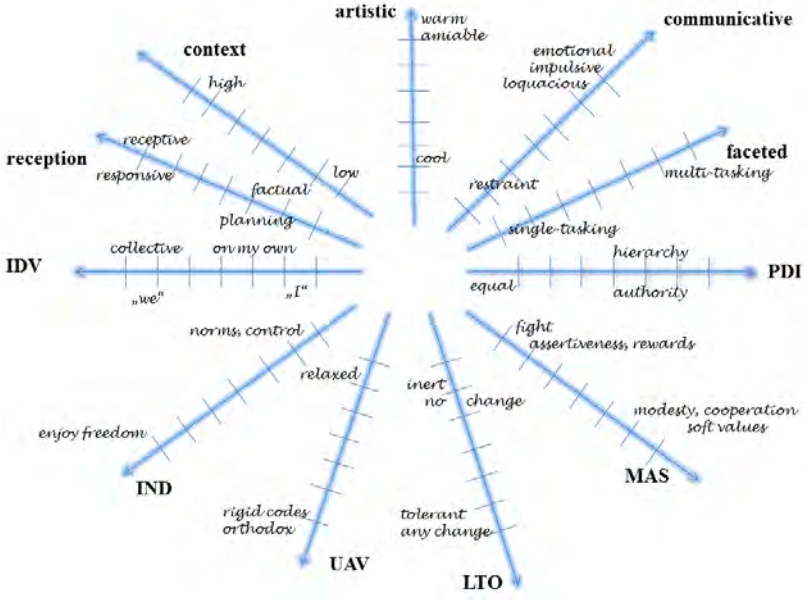

Figure 5. The Kiviat graph of the cultural dimensions of people

The graph can be extended by dimensions from other culture models. Instead we can use this combination also for derivation of other properties. The models by Hall, Hofstede, and Lewis cannot be solely considered. Additionally, combined properties cannot be derived. For instance, the triangle model [21] does not allow to reason on the cultural distance. The cultural distance is classically the differences of cultural values and is expressed as a function of differences in values of some of these dimensions, e.g. Euclidian or Mahalanobis distances. The triangle model also mixes three dimensions in Figure 5: the kind of being active, the kind of reacting on the partner and the way how tasks are performed. If we compare the distances between German and Japan people from one side and between Japan and European Russian people then first one is small in the triangle Lewis model whereas it is larger in Figure 6. The distances between Northern German and Japan people and between Japan and European Russian people in Figure 6 match far better with observations in normal life.

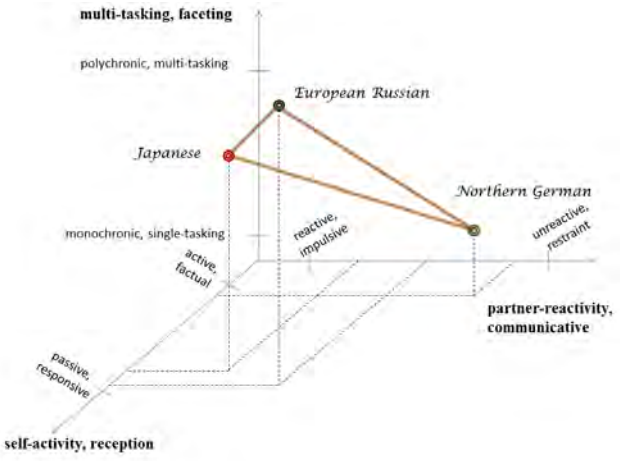

Figure 6. Three-parameter space of cultural dimensions with distances

So, the eleven dimensions in Figure 5 provide a better means for supporting also cross cultures. They can be easily extended by Victor's model [23]. Since some of the dimensions are important for some aspects and not relevant for others, we can use views for these aspects. For instance, if we concentrate on the communication and interaction level then the Lewis triangle or the three-dimensional characterisation together with high and low contexts should be taken into account. Models are also developed for communication scenario. Therefore, we can abstract from Hofstede's approach in this case. If we consider however the modelling activities then Hofstede's dimensions become more central.

3. Models and Modelling

3.1. A Model is an Adequate and Dependable Instrument

Modelling is a topic that has already been in the center of research in computer engineering since its beginnings. It is an old subdiscipline of most natural sciences with a history of more than 2.500 years. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope.
A **model** *is a well-formed, adequate, and dependable instrument that represents origins* [31,32]. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its *community of practice* within some *context* and correspond to the *functions* that a model fulfills in *utilisation scenarios*.

As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and

it sufficiently satisfies its *purpose*. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics. Models are used in

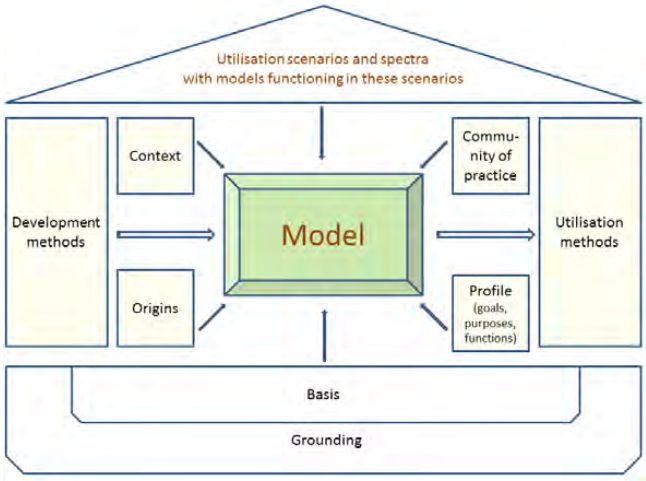

Figure 7. The model as an instrument that is adequate and dependable for its driving directives (origins, profile (functions, purposes, goals), community of practice, context) within its background (grounding, basis) and that properly functions in utilisation scenarios as a deputy of its origins

various scenarios, e.g. *communication*, perception, system construction, analysis, forecasting, documentation, system modernisation and optimisation, control, management, and simulation. Let us in the sequel concentrate on the first scenario.

3.2. Database Modelling and Cultures

We already developed a number of *stereotypes for database schemata* [15]:

(a) strictly hierarchical (ER-like) database schemata,
(b) schemata with local viewpoints that reflect the needs of some stakeholders (local-as-view approach),
(c) variants of XML-schemata, Bachman diagrams,
(d) sets of local database schemata with the requirement that the corresponding database schemata is simply the union of the set (global-as-view based on local viewpoints),
(e) sets of personalised views based on local database schemata with some kind of coherence constraint among all views (rigid global-as-view) etc.

These schema stereotypes ca directly be associated with stereotypes as shown in table 1.

Table 1. Cultural stereotypes, kinds of database schemata that are potentially preferred, and potentially useful database schema stereotypes [15]

Cultural stereotype	Preferences	Schema
High Power Distance	completely specified and well-formed, easy to understand and persistent database schema	(a)
Low Power Distance	freely configurable database schemata that is adaptable to current needs and preferences	(d)
Individualism	my own database schema according to my and only my preferences (work profile, education profile, personality profile, security profile)	(e)
Collectivism	commonly agreed database schema reflecting all elements within a group according to the collaboration style	(b)
Masculinity	restriction to essential elements and only those, strict structuring	(a)
Femininity	schema with additional and optional elements, with exploration opportunities, personalised schemata	(e)
Uncertainty avoidance	complete schema with all elements, hierarchical structuring, more linear, well-scoped sub-schemata with simple reference to main schema	(a),(d)
Uncertainty tolerance	extensible schema, flexible schema style, web-like schemata	(c),(e)
Long-term culture	all potential elements are reflected as well as all viewpoints, focused (oil stain) schemata	(a), (b)
Short term culture	handy schemata depending on current use and smooth integration of them, decomposable schemata	(e)
Indulgence	schema with a central part containing all necessary elements and further elements that might of use in future	(e),(c)
Restraint	puritanical schemata without any non-essential elements	(a)
Linear-active culture	schemata with step-wise exploration of all its aspects	(b)
Multi-active culture	different variants of the global schema for parallel integrated work	(d),(c)
Reactive culture	completely fledged schemata with all details and views for later work	(d)

3.3. The Normal Model, the Deep Model, and the Modelling Matrix

Model development is typically based on an explicit and rather quick description of the 'surface' or normal model and on the mostly unconditional acceptance of a deep model [18]. The latter one directs the modelling process and the surface or normal model. Modelling itself is often understood as development and design of the normal model. The deep model is taken for granted and accepted for a number of normal models.

The *deep model* can be understood as the common basis for a number of models. It consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. It uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice. Education on modelling starts, for instance, directly with the deep model. In this case, the deep model has to be accepted and is thus hidden and latent.

This separation into normal model and deep model provides a means to distinguish two different logical theories behind: entailment or logical consequence for normal models and semantic presupposition for deep models. The pragmatic presupposition additionally consider the relation between a model developer or user and the appropriateness

of a model in a context. Inferences become then context- and scenario-dependent. Models are thus also evaluated based on their added value and not mainly evaluated based on their validity or correctness[5].

A (modelling) *matrix* is something within or from which something else originates, develops, or takes from. The matrix is assumed to be correct for normal models. It consists of the deep model and the modelling scenarios. The modelling agenda is derived from the modelling scenario and the utilization scenarios. The modelling scenario and the deep model serve as a part of the definitional frame within a model development process. They define also the capacity and potential of a model whenever it is utilized. Deep models and the modelling matrix also define some frame for adequacy and dependability. This frame is enhanced for specific normal models. It is then used

3.4. Why Conceptual Modelling is (Not) Acceptable

A *conceptual model* is an adequate and dependable artifact or instrument that

- is enhanced by concepts from a concept(ion) space,
- is formulated in a language that allows well-structured formulations,
- is based on mental/perception/situation models with their embedded concept(ion)s, and
- is oriented on a matrix that is commonly accepted.

The conceptual model of an information system consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema [35]. Conceptual modelling is either the activity of developing a conceptual model or the systematic and coherent collection of approaches to model, to utilise models, etc.

Conceptual modelling is not in the center of development activities in all countries. Observing the history of the ER-conferences on conceptual modelling for three decades, we discover that it is still a central and attracting topic in Europe with a movement from North to South over three decades, did not change in Middle East, lost its attraction in Northern America and partially also Southern America, and has not been a central issue in the rest of Asia. So, one might ask why this attention and changes happened. One answer could be the loosing interest and importance in this approach. Another answer could be however that development is based on rather different styles in different countries, i.e. is culture-dependent. A third answer might be that models are latent and not explicitly stated what is also culture-dependent.

4. Cultures in Modelling

4.1. Models, Languages, and the Background

P.P. Chen [3] made the observation that the entity-relationship modelling language follows specific construction rules of the Old Egyptian and the Chinese language. This modelling language can only represent simple English sentences. Later, [10] could show that the extended ER modelling language HERM [28] covers the main categories in the English language. So, languages enable and hinder modelling.

[5] "All models are wrong. But some of them are useful." (often cited as a phrase by G.E.P. Box [2])

The background and especially the grounding are often incorporated into the deep model that is not explicitly communicated. For instance, the grounding for information system models includes DBMS and CE paradigms and postulates, set semantics, database theory, DBMS solution layering, DBMS technology (theory and culture), graphics and diagrammed canonical representation, ER canon, data-first-methods-second paradigm, database-approach-as-guide, etc. The basis of the model house in Figure 7 includes also a number of specific assumptions and commonly accepted practices. For instance, database modelling can be based on specific extended ER language, hidden basic types, views as derived (algebra) expressions, concept fields, extended ER thought style, parametric generic concept field, Indo-European utterance composition, extended ER development methods, extended ER heuristic rules, transformation techniques to other deep models, and extended ER tools. Additional assumptions are Salami slice tactics, the believe that functionality comes later, a rigid separation into firstness of syntax and secondness of semantics, visualisation, well-formedness (including lazy normalisation), extended ER pattern, reuse of experienced solutions (as exemplars), and flat two-dimensional schema representation. Global-as-design is commonly accepted in the database community. The global schema is the main result. Views are then defined on top of the schema by algebraic expressions in oder to cope with user viewpoints. Global-as-design has its limitations. The combination with local-as-design, e.g. for BPMN diagram suites, becomes rather difficult.

The language as an essential part of the basis, the grounding and also the other choices in the basis are acceptable in one community of practice and might be completely unacceptable for others. So, the deep model and partially also the matrix are part of the cultural setting. It is often claimed that the organisation and education cultures rule this setting. The other dimensions of culture [14] are, however, not less important.

4.2. Models as a Sufficient and Necessary Means of Communication

Communication or exchange of data/knowledge/information is one of the main scenarios where models function as a content that is communicated. Models support learning, description, prescription, prognosis scenarios as well. Communication involves several partners with their own background and culture and is based on a relationship between these partners. Each of these partners also interprets the model in a specific way based on hidden background and the specific treatment of the four directives, i.e. presupposes a specific conditional framework against which the model makes sense. The explicit part of the model is the normal model. The implicit or pragmatic part is the deep model. The matrix of the model combines the deep model and the specific ways of model usage according to the considered scenarios. We shall see in the sequel that the pragmatic part is interwoven with the culture.

Models for communication must follow felicity resp. appropriateness conditions, i.e. conditions on well-formedness. Models and especially representation models must be developed on the principles of visual communication, of visual cognition and of visual design [15,24,30]. The culture of modelling is based on a clear and well-defined design, on visual features, on ordering, effect, and delivery, and on familiarity within a user community.

The meaning of models is typically combining four parts: (1) the literal model meaning (“what is said”), (2) the conveyed model developer meaning, (3) the model user meaning, and (4) and the implicated meaning (“what is implicated”). The implicated

meaning might be conventional or non-conventional. Non-conventionality of models includes what is the implicated content within the model and what has been left aside (non-conversationally). The first one can be general or particular. These different kinds of model content influence the *model informativeness*. The first part is triggered by the meaning of the model constructs and the model design as a statement. The second, third and fourth meanings are human related and thus depend on the culture of the people involved. The model itself should have a holistic interpretation.

Models in communication scenarios have to follow general principles and a set of rules called ***maxims of model communication***. They are, in general, communication implicatures from [7,8].

Cooperative principle: Make your model such as is required, at the stage at which it occurs, by the accepted purpose of the model within the communication scenarios in which it is are deployed. This general principle has several sub-principles called "Maximes of Model Communication"

- **Maxim of quantity:** Make the model as informative as is required for the current purpose of the model do not make the model more informative than is required.
- **Maxim of quality:** Try to make model as valid as possible do not incorporate aspect that are invalid. All constructs need an adequate evidence.
- **Maxim of relation/relevance:** The model and all its elements must be relevant.
- **Maxim of manner:** The model should be parsimonious and perspicuous, i.e. economic and well-formed. Any obscurity of expression and ambiguity are avoided.

Implicated maxim of efficiency: The maxim of quantity requires that a model should be sufficient for an understanding by the model user (I(nformation)-principle [20]. From the other side it requires that the model should contain all necessary elements for an understanding by the model user (R(elevance)-principle). The model represents as much as the modeller can and must. The M(odality)-principle assumes that non-normal, non-stereotypical situations by the model that contrast to normal situations are given in an explicit and understandable form. The P(recision)-principle requires that a model is only at a precision level according to purposefulness. The B(revity)-principle prefers smaller models over longer, complex ones even though it has to be interpreted in a vague way. In some cases, vague models might serve better its function.

These maxims are explicitly stated by the sufficiency characteristics which allow to evaluate the quality in use, the external quality and the internal quality. Based on the modelling style we are able to reason on negation. A typical, however, often impractical approach with the strongest interpretation is the closed-world assumption in modelling that allows to conclude about the meaning of missing parts in the model. This assumption follows [4] ("Dire et ne pas dire"). The maxim of efficiency is often based on 'hidden' sub-models (called in the sequel 'deep model') which are taken for granted within a context and background by a community of practice.

We observe that these maxims are accepted in different cultures in a different way. So, the pragmatics of models depends on the culture. Moreover the deep model is governed by this pragmatics. The principles cannot be satisfied at the same time. Which principle is preferred also depends on the community of practice and thus on their culture imprinting.

4.3. Cross-Culture in Modelling

The adequacy of models has been handled in a strict or flexible way. Some model notions require a mapping property as a strict form of analogy. At the same time truncation or abstraction is required instead of focus. Also well-formedness is often taken more tolerant. Purposefulness is however commonly accepted. A similar observation can be made for dependability of models which is often only implicitly assumed. All model notions analysed in [34] use an implicit deep model that is undisputable. A rather surprising difference is the explicit statement on quality characteristics which have to be satisfied. At the first glance it seems that the list is random.

Let us, however, analyse[6] the German database or information system books which are often used for teaching and papers and books from US where the first are published in the ER conferences since 1992 [7]. We observe that there are common properties applicable to both. There are also properties than can be only observed for one side. some properties are out of scope or out of style although they are important for information systems.

To make these different style more clear we shall use Lewis' horizons of communication [21] in Figure 8. There are general properties that are commonly accepted by the two communities. There are also properties that are out of style or out of scope. There are also typical German and US properties that can only be observed for one of the communities. For instance, the US approach to development is often based on an 80:20 principle, i.e. the schema is left open for further development. The normal case is mainly considered. The opposite is observable for the German style. The schema must be complete. Whatever is not explicitly stated in the schema is not relevant in the application.

Therefore, cross-culture projects often result in a complete mismatch although the orientation to global-as-design and the deep model are commonly accepted. In oder to come to a common solution, the principles to modelling must be agreed in advance. This agreement may start with an agreement of the maxims (of quantity, quality, relevance, manner) and on the R-, I-, M-, B-principles. According to [1], the choice of the language is influenced by effectiveness (cost-effectiveness, representation effectiveness), infrastructure (especially tools), resource availability, knowledge capitalisation, and - what she calls - political factors. The latter are cultural factors.

4.4. Deep Models are Governed by Culture

The deep model combines the unchangeable part of a model and is determined by the *grounding for modelling* (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the *outer directives* (context and community of practice), and the *basis* (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling.

Let us consider information systems development: The grounding includes DBMS and Computer Engineering paradigms and postulates, set semantics, database theory,

[6] These observations only cover partially the specific styles and should be extended with other material as well. Since we are interested in the general culture-dependence of modelling and not in a complete empirical study we restrict ourselves.

[7] A similar observation has been made by M. Bjeković [1] for selection of enterprise modelling languages. She investigated the role of the purpose in modelling, the choice of modelling languages, and the factors for preferring one language above the other.

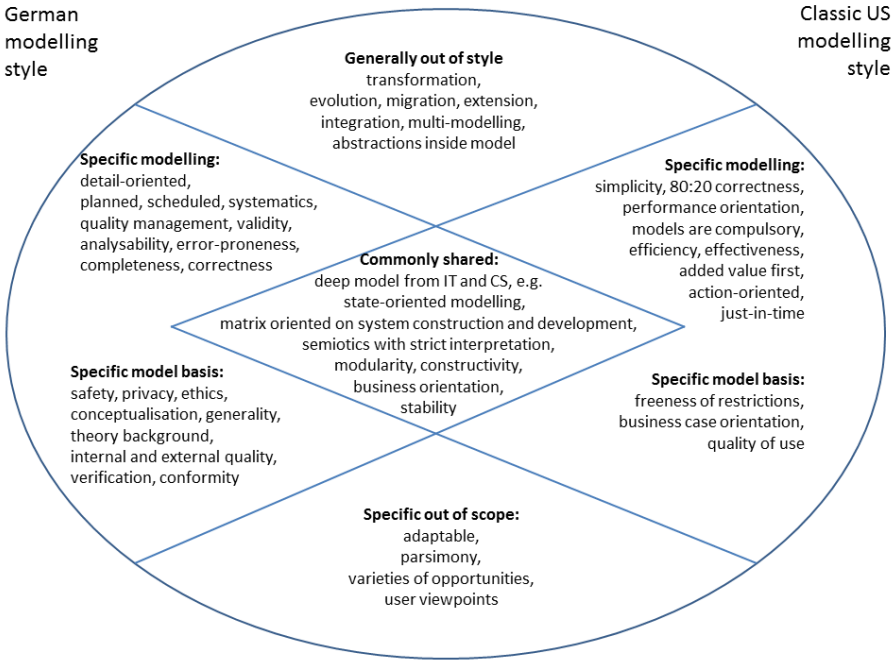

Figure 8. Modelling styles in two cultures

DBMS solution layering, DBMS technology (theory and culture), graphics and diagrammed canonical representation, ER canon, data-first-methods-second paradigm, and DBS guiding question. The basis can be build on specific an extended ER language, on hidden basic types, on views as derived (algebra) expressions, on concept fields, on a specific extended ER thought style, on parametric generic concept field, on Indo-European utterance composition, on extended ER development methods, on extended ER heuristic rules, on transformation techniques to other deep models, and on ER tools. Typical commonsense and common practices that are applied are: global-as-design, Salami slice, functionality comes later, rigid separation into firstness syntax and secondness semantics, visualisation, well-formedness (lazy normalisation), ER pattern, experienced solutions (as exemplars), and an orientation on flat schemata. Deep adequacy uses a specific analogy, specific focus, specific purpose. Deep dependability is based on arguments from the origins, on coherence inherited, on a rigid stability, and on sufficiency on the basis of extended ER quality criteria. The deep model is extended by the four directives: (i) Perception and situation models with lexicology and lexicography (e.g. an ontology as cut-out in the concept fields); (ii) a specific communication-oriented profile; (iii) the context typical for current IT or Business Informatics; (iv) the ER community of practice. The deep model provides the interpretation and the make of the normal model.

We realise that all components of the deep model are governed by the culture of the community of practice. This culture must be accepted and is the basis for a smooth communication within this community. The culture includes the acceptance of several principles: (1) the community uses a common vocabulary (*Helsinki principle*); (2) the 'what' and 'why' of modelling is agreed (*principled universe of discourse, environment,*

and information system); (3) an individual can have more than one viewpoint, one for each subject in which he is interested or has to deal with (*searchlight principle*); (4) all relevant general static and dynamic aspects, i.e., all rules, laws, etc., of the universe of discourse should be described in the conceptual schema (*100 % principle*); (5) a conceptual schema should only include conceptually relevant aspects, both static and dynamic, of the universe of discourse, thus excluding all aspects of (external or internal) data representation, physical data organization and access, as well as all aspects of particular external user representation such as message formats, data structures, etc. (*conceptualisation principle*); (6) the conceptual schema for an information system in practice can be perceived as being built up like some sort of onion the inner layer of the onion being formed by the minimal conceptual schema based on the fundamentals of logic, the extensions representing the layers of the onion (*onion principle*); (7) development is concentrated on the *what about what* with paying attention to the *how with what* we do it (e.g. conceptual level, external level, internal level) (*x-level architecture principle*).

4.5. Model Matrices are Driven by Culture

According to [19], a disciplinary matrix consists of (I) symbolic generalizations as formal or readily formalisable components or laws or law schemata, (II) beliefs in particular heuristic and ontological models or analogies supplying the group with preferred or permissible analogies and metaphors, (III) values shared by the community of practice as an integral part and supporting the choice between incompatible ways of practicing their discipline, and (IV) exemplars for concrete problem solutions similar to Polya's theory for puzzle-solving (see also Wittgenstein 'Game' [36]). Additionally we consider (V) a guiding question as a principal concern or scientific interest that motivates the development of a theory, and (VI) techniques as the methods an developer uses to persuade the members of the community of practice to his point of view. So, the modelling matrix includes the deep model ((I),(II),(III)) which already culture-governed and additionally.

The modelling matrix is a specific disciplinary matrix and consists of the deep model and the modelling scenarios with specific stereotypes. So it governs the development of the 'rest' of the model development and model utilisation. The agenda is derived from the modelling scenario and the utilisation scenarios.The modelling matrix thus provides also a specific understanding of adequacy and dependability of models.

We may now derive specific modelling matrices for information system models — mainly for the development of the normal model. The matrix is assumed to be correct for normal models. Normal modelling involves showing how systems and their models can be fitted into the elements the matrix provides. Most of this work is detail-oriented. The matrix itself is thus driven by the culture accepted by the community of practice within the given context.

5. Conclusion

The aim of this paper was to analyse the culture sensitivity of modelling and models. We see modelling as a human activity and because of that it is as culture sensitive as human behavior in general. Worldwide we use same modelling techniques and tools, which for their part unify modelling practices and models. There are also several studies that criticize the use of tools and practices developed in powerful cultures in foreign

culture context. The kernel of the criticism is that these tools and techniques transfer the elements of the origin to the culture where these are used. Big gap can be seen between Western and Eastern cultures, as well as between mature (welfare) and emergent (more poor) cultures.

In our paper we have approached the topic from the direction of culture analysis in general level and applying the results of our findings in (cross-cultural) modelling context. A modelling framework — "the model house" — is used as a basic structure. The role of normal and deep models, as well as the role of modelling matrix are parts of this framework. Culture dependent aspects in conceptual modelling provide and comparison of modelling styles of two cultures are used as applications.

Our conclusion is that we found a lot of culture sensitive aspects in modelling. Models include a lot of "wordless" information that have source in modelling languages. In this context we want to make analogy to Hall's high and low context cultures discussed in section 1 of this paper. Modelling languages — because of the semi-formal character — leave a lot of gaps to exact specification. This gap includes always some amount of culture related aspects. In addition normal information system requirements specification includes a lot of non-functional features that are defined by still less formal language, like natural language. These features are culture sensitive and also in most cases impossible to test by normal testing practices and verification; instead of tests human validation is used - again one culture sensitive step more.

References

[1] M. Bjeković. *Pragmatics of Enterprise Modelling Languages: A Framework for Understanding and Explaining*. PhD thesis, Radboud University Nijmegen, 2017.

[2] G. E. P. Box. Science and statistics. *Journal of the American Statistical Association*, 71(356):791–799, 1976.

[3] P. P. Chen. From ancient egyptian language to future conceptual modeling. In *Conceptual Modeling, Current Issues and Future Directions*, volume 1565 of *Lecture Notes in Computer Science*, pages 56–64. Springer, 1997.

[4] R.J. Fogelin. *Pyrrhonian Reflections on Knowledge and Justification*. Oxford University Press, Oxford, 1967.

[5] Blanchard E. G., Karanasios S., and Dimitrova V. A conceptual model of intercultural communication: Challenges, development method and achievements. In *Proc. CATS 2013, Vol. 5, AIED 2013 Workshops*, volume 1009, pages 1–10. CEUR Workshop Proceedings, 2013.

[6] Dafoulas G. and Macaulay L. Investigating cultural differences in virtual software teams. *The Electronic Journal on Information Systems in Developing Countries*, 7(4):1–14, 2001.

[7] P. Grice. *Logic and conversation, Syntax and Semantics 3: Speech Acts*. Academic Press, New York, 1975.

[8] P. Grice. *Studies in the Way of Words*. Harvard University Press, 1989.

[9] E.T. Hall. *Beyond Culture*, volume 222. Anchor Books, New York, 1976.

[10] S. Hartmann and S. Link. English sentence structures and EER modeling. In *Proc. APCCM 2007*, volume 67 of *CRPIT*, pages 27–35. Australian Computer Society, 2007.

[11] G. Hofstede. Hofstede insights - the 6 dimensions of national culture. Available in https://www.hofstede-insights.com/models/national-culture/, 2017. Retrieved October 18th, 2017.

[12] G. Hofstede, G.J. Hofstede, and M. Minkow. *Cultures and Organizations: Software of the Mind: Intercultural Cooperation and Its Importance for Survival*. McGraw-Hill, New York, 2010.

[13] H. Jaakkola, J. Henno, B. Thalheim, and J. Mäkelä. Collaboration, distribution and culture challenges for communication. In *MiPRO*, pages 657–664. IEEE, 2015.

[14] H. Jaakkola and B. Thalheim. Multicultural adaptive systems. In *Information Modelling and Knowledge Bases*, volume XXVI of *Frontiers in Artificial Intelligence and Applications, 272*, pages 172–191. IOS Press, 2014.

[15] H. Jaakkola and B. Thalheim. Culture-adaptable web information systems. In *Information Modelling and Knowledge Bases XXVII*, Frontiers in Artificial Intelligence and Applications, 280, pages 77–94. IOS Press, 2016.

[16] H. Jaakkola and B. Thalheim. Supporting culture-aware information search. In *Information Modelling and Knowledge Bases XXVIII*, Frontiers in Artificial Intelligence and Applications, 280, pages 161–181. IOS Press, 2017.

[17] H. Jaakkola and B. Thalheim. Web information systems for high and low context cultures. In *Information Modelling and Knowledge Bases XXIX*, forthcoming. IOS Press, 2018.

[18] Y. Kropp and B. Thalheim. Data mining design and systematic modelling. In *Proc. DAMDID/RCDL'17*, pages 349–356, Moscov, 2017. FRC CSC RAS.

[19] T. Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, Chicago, Illinois, 2nd, enlarged, with postscript edition, 1970.

[20] S. Levinson. *Pragmatics*. Cambridge University Press, 1983.

[21] R.D. Lewis. *When Cultures Collide. Managing Successfully Across Cultures*. Nicholas Brealey, London, 3rd edition, 2011.

[22] R.D. Lewis. Cultureactive — country profiles. https://secure.cultureactive.com/, 2017. Retrieved October 18th, 2017.

[23] M. Limaye and D.A. Victor. Cross-cultural business communication research: state of the art and hypotheses for the 1990s. *The Journal of Business Communication*, 28(3):277299, 1991.

[24] T. Moritz. *Visuelle Gestaltungsraster interaktiver Informationssysteme als integrativer Bestandteil des immersiven Bildraumes*. PhD thesis, HFF Berlin-Babelsberg, 2006.

[25] B.S. Parumasur. The effect of organisational context on organisational development (OD) interventions. *SA Journal of Industrial Psychology*, 38(1), Art. # 1017 2012.

[26] M. Phaedra and M. Permanand. Contextualised student modelling for enculturated systems. In *AIED 2013 Workshops Proceedings, Volume 5*, number 1009, pages 20–29. CEUR Workshop Proceedings, 2013.

[27] K. Reinecke and A. Bernstein. Improving performance, perceived usability, and aesthetics with culturally adaptive user interfaces. *ACM Transaction on Computer-Human Interaction*, 18(2):8, 2011.

[28] B. Thalheim. *Entity-relationship modeling – Foundations of database technology*. Springer, Berlin, 2000.

[29] B. Thalheim. *The Conceptual Framework to Multi-Layered Database Modelling based on Model Suites*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 116–134. IOS Press, 2010.

[30] B. Thalheim. Syntax, semantics and pragmatics of conceptual modelling. In *NLDB*, volume 7337 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2012.

[31] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[32] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. 2017.

[33] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[34] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[35] B. Thalheim and M. Tropmann-Frick. The conception of the conceptual database model. In *ER 2015*, LNCS 9381, pages 603–611, Berlin, 2015. Springer.

[36] L. Wittgenstein. *Philosophical Investigations*. Basil Blackwell, Oxford, 1958.

Models and their Foundational Framework

Bernhard Thalheim
Christian-Albrechts University at Kiel, Department of Computer Science, D-24098 Kiel

Abstract

The term model is mainly used in two meanings which are considered to be different: a model of a problem domain as a conceptualisation; a model of a set of formulas as an interpretation in which every formula within this set is true. A general theory of models has not yet been developed. H. Stachowiak proposes a phenomenal approach and 'defines' models by their properties of mapping, truncation and pragmatics. Meanwhile, a notion of the model has been developed. At the same time, it seems that there are rather different understandings of model in sciences and especially Mathematical Logics. Sciences treat models as reflections of origins. Mathematical logics considers models as an instantiation in which a set of statements is valid. So, mathematical model theory is often considered to be a completely different approach to modelling. We realise however that mathematical model theory is only a specific kind of modelling. We show that the treatment of models in logics and in sciences can be embedded into a more general framework. So, the theory of models is based on a separation of concern or orientation.

1 Introduction

Modelling is a topic that has implicitly been in the center of research in science and engineering since its beginnings. It has been considered as a side issue for long time. During the last 40 years it has gained more attention and becomes nowadays a subdiscipline in many disciplines. The compendium [TN15] introduces models in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments used in certain scenarios. So, essentially it is an old subdiscipline of most natural sciences with a history of more than 2.500 years [Mül16][1]. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope.

The *modelling method* is a specific science method that uses models as instruments with certain intention or goal, e.g. for solving a task. The model represents or deputes origins. The model is used instead of the origin due to its properties, esp. adequacy and dependability. The modelling method thus consists (i) of the development of 'good' models, (ii) of the utilisation of the model according to the goal, (iii) of the compilation of the experience gained through model utilisation according to the goal, and finally (iv) of generalisation of the experience back to the origins. So, a model must be well-build for this goal, must be enhanced by methods that support its successful deployment, and must support to draw conclusions to the world of its origins.

1.1 A Model is an Adequate and Dependable Instrument

A **model** *is a well-formed, adequate, and dependable instrument that represents origins* [Tha14, Tha17a].

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its *community of practice* within some *context* and correspond to the *functions* that a model fulfills in *utilisation scenarios*.

As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

[1] The earliest source of systematic model consideration we know is Heraklit with his λόγος (logos). Model development and model deployment is almost as old as the mankind, however.

Studia Metodologiczne, 2019, Poznan

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). Model functions determine which justification is required and which sufficiency characteristics are important. A well-formed instrument is called *dependable* if it is sufficient and is justified for justification properties and sufficiency characteristics.

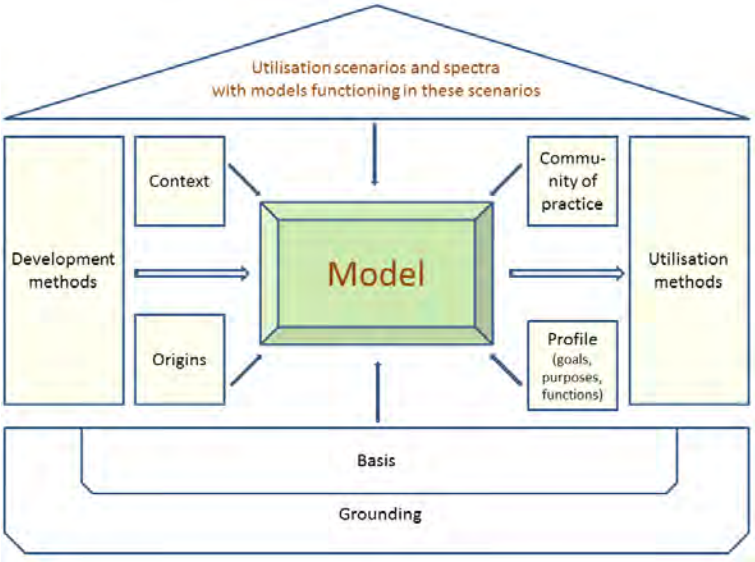

Figure 1: The model as an instrument that is adequate and dependable for its driving directives (origins, profile (functions, purposes, goals), community of practice, context) within its background (grounding, basis) and that properly functions in utilisation scenarios as a deputy of its origins

Figure 1 represents a model of the model. The development and utilisation methods form the enabling aspects of the modelling method. Driving directives are (1) origins to be represented by the model, (2) purposes or goals or functions of models, (3) the community of its users and developers, i.e. the community of practice, and (4) the context into which the model is embedded. Models function as instruments in application or utilisation scenario. Typical functions of models are (a) cognition, (b) explanation and demonstration, (c) indication, (d) variation and optimisation, (e) projection and construction, (f) control, (g) substitution, and (h) experimentation. A model is not built on its own. It has an undisputable grounding that has to be accepted. The basis of the model - similar to the cellar - can however be disputed. Grounding and basis form the background of a model. We observe that the background is often given only in an implicit form. The same kind of concealment can also be observed for the utilisation scenario which are implicitly given by sample and generalisable case studies for the utilisation frame.

The model is not simply an image of its origins. The mapping property [Kas03, Mah09, Mah15, Sta73] might be too restrictive for models. Instead, we use analogy. Models can also be material artifacts. A model can be a model of another model. Models might follow different structuring and behaviour than the origins. Usefulness and utility according to goals govern the selection of a model instead of quality characteristics such as validity. Finally, a model comes with its background. It cannot be properly understood and used if the background is concealed. Let us distinguish the concepts of goal, of purpose, and function in the sequel. The goal of a model is in general the association between a current state and the target state that is accepted by stakeholders or – more general – by members of a community. The purpose enhances the goal by means that allow to reach the target state, e.g. methods for model development and utilisation. The function extends the purpose by practices or – more systematically – by

scenarios in which the model is used. A typical scenario is the modelling method and its specific forms.

1.2 Models in Science and Daily Life versus Models in Mathematical Logics

Models in sciences and model theory in mathematical logic are often considered to be completely different issues [Bal16]. This point of view is correct as long there is no consolidated understanding of a notion of a model. Models in model theory are instantiations of a set formulas. This set of formulas is satisfied by a model according to a logical definition frame. The model is a structure that is defined with the same signature as the set of formulas.

So, we might come to the conclusion that there are at least three different understandings of the model. We will oppose this conclusion in the sequel. It is only true for the Fuzzy or phenomenalistic view.

Models in science typically follow the modelling methods. They may be composed of a number of models and be based on other models. A model must not be true. It should however be coherent to some extent within its discipline

The origin in science is not limited to material origins. The origin itself can be virtual or be again a model, e.g. a mental model. So, the modelling methods may also be iteratively applied.

Models often used in daily life. One kind are metaphors or parables. The typical kind is, however, a pattern for explanation, negotiation, and communication. Models carry a meaning. It is often debated whether a fashion model or a diagram or a visualisation can be considered as a specific kind of a model.

The modelling method presented so far is associated with its origins. We might however also use models for construction of other origins or models. In this case, the model is not generalised but used as a blueprint for another artifact. So, we observe that the modelling method must be extended.

1.3 Models and their Utilisation Scenarios

Models are used in various scenarios, e.g. communication, system construction, perception, analysis, forecasting, documentation, system modernisation and optimisation, control, management, and simulation. Let us in the sequel concentrate on the first three scenarios.

The extended modelling method is embedded into a more general form of activities, i.e. scenarios. The model itself is used as an instrument in a scenario or a bundle of scenarios which we call usage spectrum. It has a function or a number of functions in these scenarios. This functioning must be effectively supported by utilisation methods and is used by members of a community of practice in most cases. For instance, models of situations/states/data are often used for structuring, description, prescription, hypothetic investigation, and analysis. So, we observe that the function (or simpler the purpose or the goal) of the model is determined by the concrete way how a model is used.

A model might be oriented towards this community of practice. It can however also represent the scenarios themselves. It might represent the context of these scenarios, e.g. the scientific or engineering background, the relation to time and space, the application area insight, and the knowledge accepted by the community. It might also be oriented to representation of either a situation and state under consideration or a evolutionary change process.

The different orientations is the basis to distinguish the six concerns for models: community of practice, background/knowlegde/context, application scenario and stories of model utilisation with their specific frames, situation-/state/data, dynamics/evolution/change/operations, and models as representations and instruments.

Figure 2 shows the relation between the concerns and the functions a model might have[2].

1.4 The Storyline of the Paper

A general theory of models, of modelling activities and of systematic modelling has not yet been developed although modelling has already attracted a large body of knowledge and research[3]. The notion of the model is not yet commonly accepted. Instead we know a large variety of rather different notions. Model development activities have been a concern in engineering. The process of model development has not yet attracted a lot of research. Model deployment also needs a deeper investigation. The model is mainly used as an instrument in certain application scenarios and must thus function in these scenarios. So, a model is a medium.

[2]Modified and revised from [Tha17c].

[3]It is not our purpose to develop a bibliography of model research. Instead we refer to bibliographies in [TN15] and the more than 5.000 entries in R. Müller's website, e.g. [Mül16].

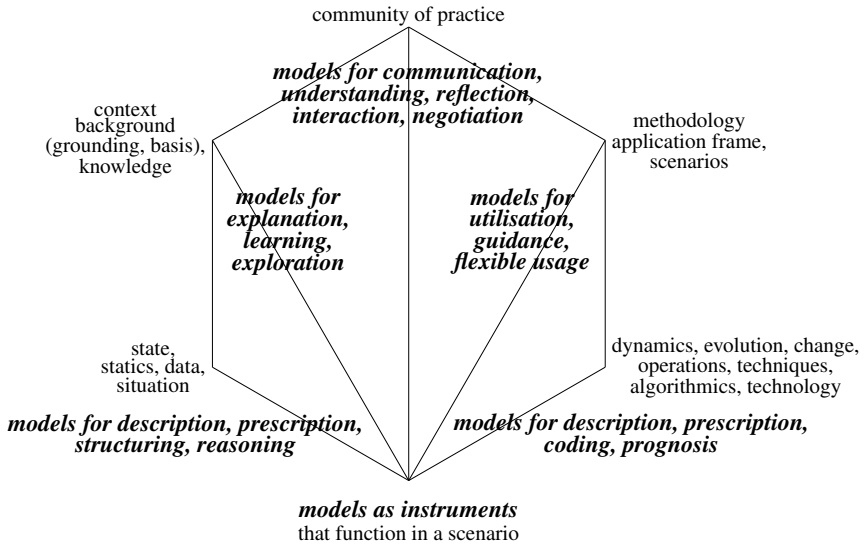

Figure 2: Models and the five concerns in model-based reasoning, investigation, and engineering

We have already introduced the general notion of a model as a starting point. The next step could be the development of a general theory of modelling. It is often claimed that modelling is rather different in science and engineering. So, we might conclude that there is no general theory of modelling. This paper is going to show that there is a general theory of modelling. We start with a case study in Section 2. These lessons gained in this cases study are a starting point for a general theory of models, of modelling activities, and of systematic modelling. In Section 3 first elements of this theory are developed.

2 Models in Everyday Life and Sciences: A Case Study

Analysing model notions we realise that there are at least four different approaches:

1. The general phenomenalistic definition uses properties such as mapping, truncation and pragmatic properties for the association between origins and models. Most research on models starts with this approach.
2. The axiomatic definition follows frames used in Mathematical Logics and defines models as exemplifications of formal systems and formal theories. Models thus depute and represent a certain part of reality.
3. The mapping-based definition is based on a direct homomorphic mapping between origin and model. We might have another mapping between model and implemented system that is a realisation of the model.
4. The construction-oriented definition defines a model as being a result of a modelling process by some community of practice.

There is a fifth approach to models which simply uses artifacts as models without any definition, e.g. in human communication and also in sciences[4]. The definition given above follows, however, the mathematical way of defining things through definitional extensions.

Models are used as (a) perception models reflecting someone's understanding, (b) mental models that combine various perception models and that make use of cognitive structures and operations in common use, (c) domain-situation models representing a commonly accepted understanding of a state of affairs within some application

[4]One of the prominent definition is given by John von Neumann [vN55]: "*The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work - that is correctly to describe phenomena from a reasonably wide area. Furthermore, it must satisfy certain esthetic criteria - that is, in relation to how much it describes, it must be rather simple. I think it is worthwhile insisting on these vague terms - for instance, on the use of the word rather. One cannot tell exactly how "simple" simple is. Some of these theories that we have adopted, some of the models with which we are very happy and of which we are proud would probably not impress someone exposed to them for the first time as being particularly simple.*"

domain, (d) experimentation models that guide experimentation, (e) formal model based on some kind of formalism, (f) mathematical models that are expressed in some mathematical language and based on some mathematical methods, (g) conceptual models which combine models with some concept and conception space, (h) computational models hat are based on some (semi-)algorithm , (i) informative models that used to inform potential users about origins, (j) inspiration models that provide an intuitive understanding of something, (k) physical models that use some physical instrument, (l) visualisation models that provide a visualisation, (m) representation models that represent things like other models, (n) diagrammatic models that are based on some diagram language with some kind of semantics, (o) exploration models for property discovery, (p) prototype models that represent a class of similar items, (q) mould models that are used for production of artefacts, (r) heuristic models that are based on some Fuzzy, probability, plausibility etc. relationship, etc. Although this categorisation provides an entry point for a discussion of model properties, the phenomenon of being a model can be properly investigated. Each category is rather broad and combines many different aspects at the same time. We already introduced a general notion of model. In this Section we will investigate whether the general definition covers all these kind of models for science and also daily life and whether it can be supported by a holistic treatment of models.

2.1 Models in Mathematical Logics

Let us consider only one kind of logics: classical Mathematical Logic based on first-order or higher-order predicate logics. Similar observations can be drawn for other mathematical logics as well. Mathematical logic has a long tradition of model research. Model theory became its branch and has a deep theoretical foundation. The main language is the first-order predicate logic. This language is applied in a rigid form [ST08] that became a canonical form of Mathematical Logics: It uses a canonical way of associating syntactic and semantic types. Additionally, the semantic type and the syntactic type have the same signature. The expressions of syntactic types are inductively constructed starting with some basic expressions of certain construct by application of expressions of some other construct. For instance, we may start with truth values and variables. Terms and formulas are then based on these basic expressions. The context is not considered. The world of potential structures is typically not restricted. The rigidity however allowed to gain a number of good properties. For this reason, first-order predicate logics became a first-class fundament for Computer Science.

In general, a model in Mathematical Logics is defined through its relationship to a set of formulas. These formulas are valid in the model. Additionally, axioms and rules of the first order predicate logics are valid in the model since they are valid in any structure of given signature. Models are thus instantiations (or exemplifications) for a set of statements. The theory of deduction is the main basis for reasoning. Therefore, the five concerns in Figure 2 have the specific peculiarity shown in Figure 3.

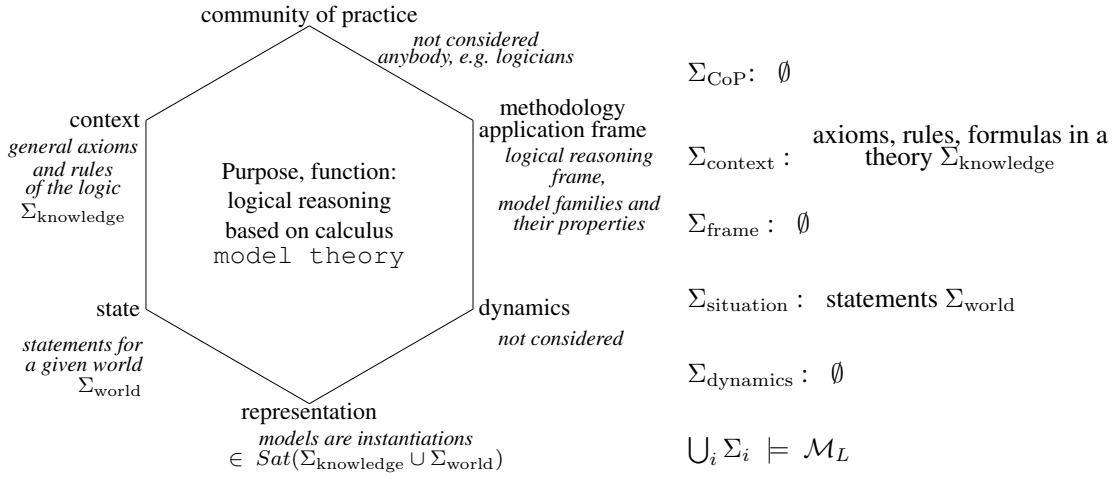

Figure 3: Models in logics for investigation of situations and expressible properties: axioms and rules form the context world; admissible states are characterised by a set of formulas; models are instances of potential systems that obey the system

The special side of the approach of Mathematical Logics to modelling is the consideration of the set of all potential models together with a given instantiation. This approach is however also taken into consideration for other model kinds as we shall in the sequel.

A model might become then an exemplar or prototype for a given theory. It can represent this theory and thus allows to reason on the given theory. It can be thus a final or an initial model (see the theory of abstract data types [Rei84, Wec92]) where the first one is the best and most detailed representation of the given theory and allows to reason on all potential negative statements as well.

We notice that classically the community of practice is not considered. Also, dynamics ist not an issue. There is not really defined any reasoning frame beside the calculus itself. We are free to choose Hilbert style or Gentzen style or any other derivation style for reasoning.

A specific decision within mathematical logics is the invariance of the signature, i.e. models as structures and logical languages for theory statements share the same signature. Therefore, there is a tight mapping between terms and formulas and the properties that can be stated on the model.

This specific mapping property has also been used for the phenomenal characterisation of models as structures that a based on a mapping from the origin to the model, e.g. [Bal82, Sta73, Ste66, Ste93]. We also observe that the truncation or abstraction property is a specific property of logical models.

2.2 Mathematical Models

Mathematical models are considered to be the most prominent kind of model. A mathematical representation of another 'donor' or *origin model* is based on the mathematical language. The mathematical model is used for solving of problems that have been formulated for the origin model. The association between the mathematical model and the origin model must be problem invariant. Solution faithfulness is often not given explicitly required, i.e. the solution obtained for the mathematical model must be faithful for the origin model. Mathematical modelling presumes the existence of this origin model. So, (1) it starts with an application analysis and a formulation of the problem to be considered in the application area. Next, (2) this formulation is transformed to the origin model which allows to describe the problem. (3) This origin model is then mapped to a mathematical model. (4) The fourth phase is the development of a solution of the problem within the mathematical model. (5) The solution is verified and will be validated for faithfulness within the origin model. Finally, (6) the solution is examined for its reflection in the application area. If the solution is not of the required quality then the phases are repeated. This 6-phase circular frame [GKBF13, Pol45] is a commonly accepted scenario for mathematical modelling.

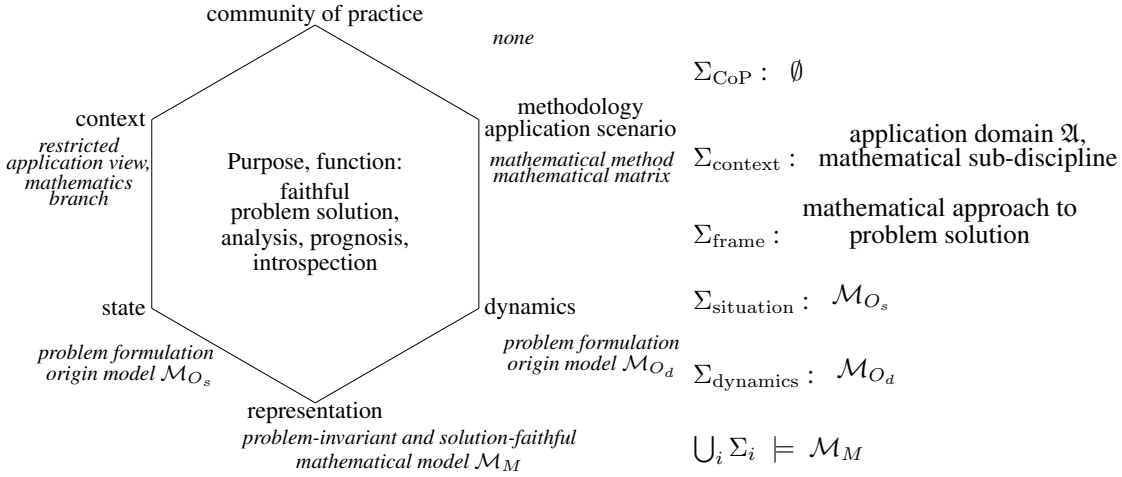

Figure 4: The mathematical model as a representation of a origin model within the mathematical frame

We observe that the mathematical frame is similar to the logical reasoning frame. Main quality requirements satisfaction of the problem solving purpose, adequacy of the mathematical model, robustness against minor changes,

and potential and capacity for problem solution. The community of practice should not influence the model properties. It may influence on the selection of various representation models. The situation and its dynamics determines the appropriatedness of the mathematical language. The mathematical model is determined by some mathematical method that has shown useful in the past.

Our model notion extends the model discussion by H. Hertz [Her84, vDGOG09]. He postulates that some artefact is a model due to its analogy to origins, its dependence within an application context, its purposefulness, its correctness, its simplicity, and its potentially only implicit given background. Models have thus a validity area.

Mathematical models are specific formal models. They are based on a formalisation that can be mapped to some mathematical language. The mapping from the formal model to the mathematical model should preserve the problem, i.e. it is invariant for the problem. The mapping should additionally also allow to associate the mathematical solution to the problem with a correct or better faithful solution in the formal model and for the origins, i.e. the model is solution-faithful [BT15]. The mathematical language has not only a capacity and potential. It also restricts and biases the solution space. The calculus used for the derivation of the model is any mathematical and not restricted to logical reasoning.

2.3 Science Models

All sciences widely use models. Typical main purposes are explanation, exploration, hypothesis and theory development, and learning. Models are mediators, explainers, shortcuts, etc. We can consider models as the third dimension of sciences [BFA$^+$16, TD16, TTF16][5]. Following [Gra07], sciences may combine empirical research that mainly describes natural phenomena, theory-oriented research that develops concept worlds, computational research that simulates complex phenomena and data exploration research that unifies theory, experiment, and simulation. Models are an essential instrument in all four kinds of research. Their function, however, is different as illustrated in Figure 5 [BFA$^+$16].

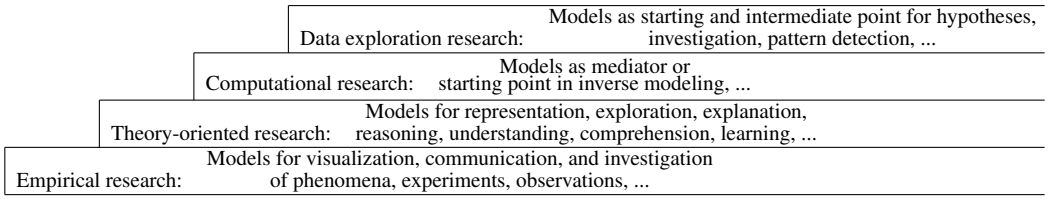

Figure 5: Some model functions according to the kind of scientific research

Empiric research also uses a canonical modelling mould. Beside an ad-hoc mould we might use a sophisticated one: (1) define a research question (based, for instance, on the rhetoric frame (who, what, when, where, why, in what way, by what means (Hermagoras of Temnos, also Augustine's De Rhetorica) or W*H framework [DT15]), (2) consider threats to the research, (3) choose a research model (e.g. positivistic), (4) develop an approach how facts become theories, (5) create a generic meta-model (with some level of abstraction, with independent and dependent parameters and indicators), (6) define analysis approaches (qualitative or quantitative), (7) define the research program and agenda as a specific research process, (8) select the research method, (9) analyse the capacity and potential of quantitative data, (10) design the experiement, (11) design the case study, and (12) design the outcomes survey.

The empirical research approach often combines qualitative and quantitative approaches. The quantitative approach is often oriented on observable data whereas the qualitative approach orients towards theory, on concepts and conceptions, and on a characterisation of the situations of interest. The quantitative theories are often 'phenotypical' approaches contrary to the 'genotypical' approaches used in qualitative approaches. A typical approach is used in the collaborative research centre 1266 [6]. It uses additionally an investigative reasoning approach. Figure 6 shows the differences between genotypical and phenotypical models. We use a planar representation of the three dimensions: (1) the composition dimension with sources, concepts, and theories; (2) the kind dimension with qualitative and quantitative reasoning, and (3) the model dimension that allows to concentrate on certain aspects of the first dimensions depending which function, purpose and goal the model should satisfy. A typical specific

[5] The title of the book [CH04] has inspired this observation.

[6] Scales of Transformation – Human-Environmental Interaction in Prehistoric and Archaic Societies: https://www.sfb1266.uni-kiel.de/en

treatment of concepts is applied in modelling. Since models orient on certain aspects and represent also combined representations, concepts used in models are often not directly derived from concepts in the theory. Additionally, we should distinguish between quantitative, investigative, and quantitative models. The model kind in Figure 6 uses investigative reasoning and lends some elements from quantitative and qualitative theories beside the theory offering that are used for investigative reasoning. The quantitative theory should also be reflected in the qualitative theory.

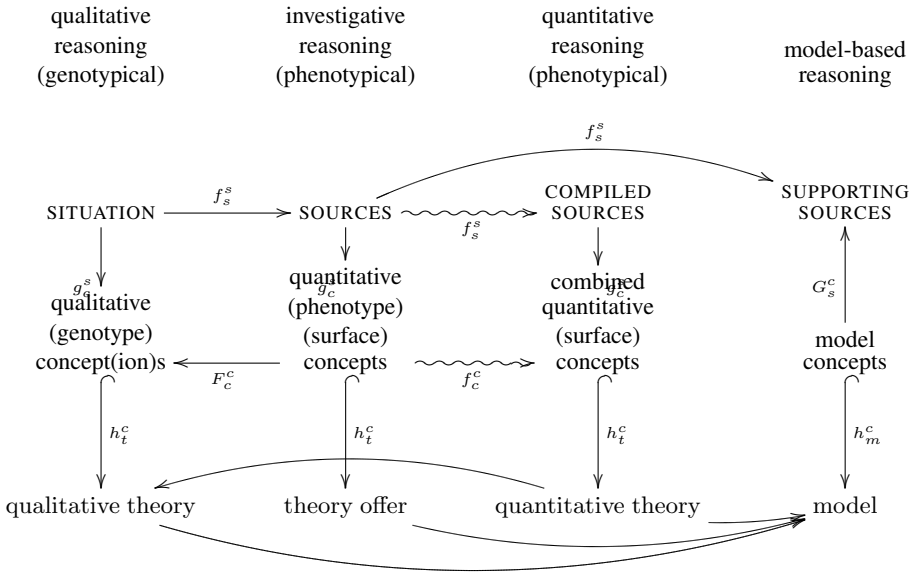

Figure 6: Models for investigative and quantitative reasoning in empirical research

A qualitative theory uses a concept or conception space that represents situations of interest (may be based on some mapping g_c^s). The situation can be observed and characterised by sources (may be based on some f_s^s mapping). Empirical research in sciences often differentiates between an investigative reasoning and quantitative reasoning. Both use phenotypical observations on proxies. Quantitative approaches aggregate and combine the source data and thus allow to reason on correlation, dependencies, time and spatial relationships. The first two reasoning approaches should be based on a commuting diagram, i.e. we assume $F_c^c(g_c^s(f_s^s(situation))) = g_c^s(situation)$ for any situation considered.

Evidence-based proxy modelling and reasoning treats models in a different way.

(α) Models represent only acceptable possibilities. Each model captures a distinct set of possibilities to which the current description refers) which are consistent with the premises and the knowledge gained so far what makes them intrinsically uncertain because they mirror only some properties they represent.

(β) Models are proxy-driven. The structure of the model corresponds to the proxies it represents.

(γ) Models represent only what has been observed and not what is false in each possibility in contrast to fully explicit models (also representing what is false).

(δ) The more proxies that are considered, and the richer those models are, the more accurate the world view is.

(ϵ) Additionally, we use pragmatic reasoning schemata, e.g. A causes B; B prevents C; therefore, A prevents C. The model themselves illustrate then concepts. Therefore, sources support concepts and conceptions what inverts the mapping (G_s^c instead of g_c^s).

Let us now consider the theory-oriented research. The frame for empirical research is similar to communication frames in Subsection 2.5. We neglect inverse modelling [Men89] although it is an important approach to science and it has been reconsidered and generalised under various other names, e.g. [ASG13, Noa09, SV05, BST06, TT13]. Data science approaches have been considered in [KT17].

So, we arrive with the hexagon in Figure 7. Models function as instruments within the science. They are vehicles for investigation, for analysis, for discovery of alternatives, for prognosis, for exploration, for explanation, for intellectual absorbtion, for learning, for understanding, for scoped and focussed comprehension, for representation of certain aspects, for discussion with partners within their background, for quick illustration, etc. They are supported by various kinds of reasoning. It seems that this variety is rather broad. If we however orient our investigation

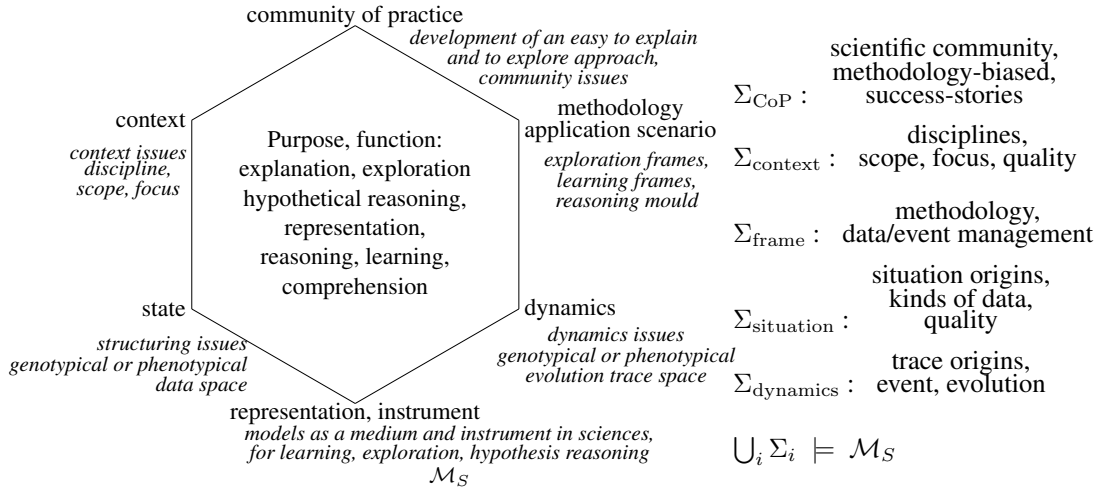

Figure 7: Models are used in natural, social, and other sciences as enhancements and contributions to sciences and as instruments: science contribution, explanation, exploration, learning, comprehension, intellectual absorption, simulation, and reasoning scenario

on the scenarios then we discover that the model utilisation scenarios determine the function of the model. At the same time, the background with the grounding and basis strikes through. Models are biased by their foundations, by their development and utilisation methods, their communities of interest, and their context. A specific context is the school of thought [Bab03, Fle11]. The concept space determines what could the content and the scope of a model. The MMM compendium [TN15] illustrates that models, the approach for to model, and modelling share a good number of common approaches.

2.4 Conceptual Models

Conceptual models are widely used in Computer Science and more specifically in Computer Engineering. In Computer Science and Computer Engineering, one main scenario is (1) the model-based construction of systems beside (2) the explanation and exploration of an application, (3) description of structure and behaviour of systems, and the (4) prognosis of system properties. Model-based construction might include conceptualisation. The application scenarios mainly follows the description-prescription frame. The model is used as a description of its origin and as a prescription of the system to be constructed. The notion of conceptual model is not commonly agreed however[7]. In a nutshell, a *conceptual model* is an language-determined enhancement of a model by concepts from a concept(ion) space.

The conceptual modelling method uses a canonical style of model development and utilisation. Models are instruments in perception and utilisation scenarios. They function is explicitly defined, e.g. models for design and synthesis. The scenario can incorporate a decision point that stops after understanding the perception and domain-situation models or that designs and synthesises the conceptual model after a preparation phase. The last stage support then evaluation and acceptance of the model.

So, Figure 8 displays the more specific way of conceptual modelling for information systems. The IS community with its actors $\{a\}$ shares an IT orientation. It might however be in conflict with the business users. They reason in a different way and are often using a local-as-viewpoint approach. The global-as-design approach might not provide an appropriate support. The model development and utilisation becomes canonical after the choice of the enabling language and the modelling method. The origin models such as the perception and domain-situation models follow the style accepted in these communities. The global-as-design approach must then provide appropriate aggregations and derivations for support of local viewpoints. The community also shares the assumption of strict

[7] We know almost threescore different notions what shows the wider controversy about this notion[Tha18a]. E.g., Wikiquote (see [Wik17]) lists almost 40 notions. Facetted search for the term "conceptual model" in DBLP results in more than 5.000 hits for titles in papers (normal DBLP search also above 3.400 titles)

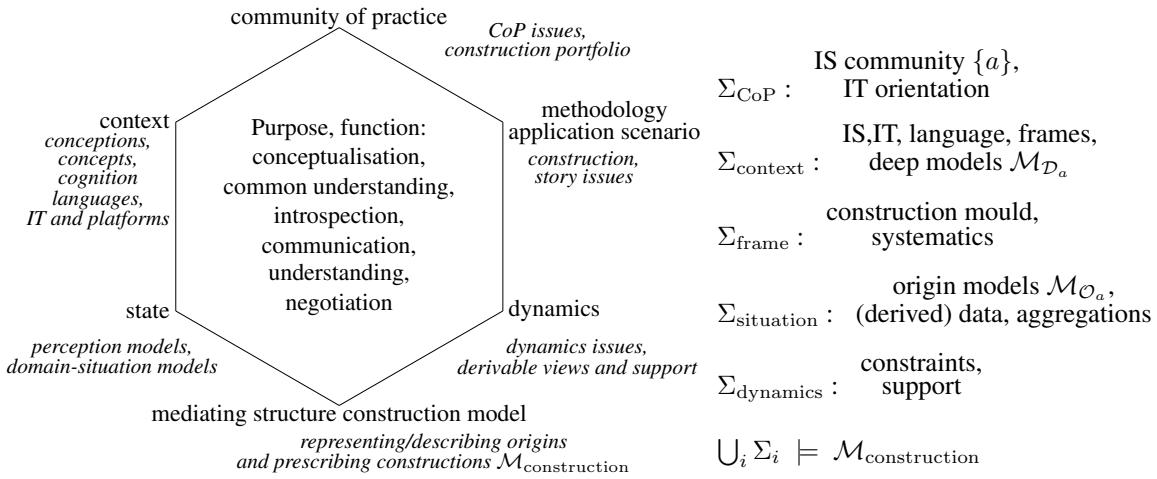

Figure 8: Conceptual models for IS structuring

separation of specification into syntax and semantics with the firstness paradigm [KL13, Pei98] for structures and the secondness [Cas55] of functions and views. The model to be developed inherits all the paradigms, assumptions, biases, conceptualisations, cultures, background theories, etc.

A typical example for conceptual modelling is entity-relationship modelling. [Tha18b] observed a large number of paradigms, postulates, specific modelling cultures, commonsense, practices, and assumptions such as global-as-design (with derivation of local viewpoints), Salami slice typing (for homogenisation of object structure within a class), set semantics (instead of multi-set semantics that is used for implementation), uniqueness of names within a schema, hidden implementation assumptions, specific styles for model composition one must follow, well-formedness conditions, etc. Some approaches add also requirements such as strict binarisation of all relationship types.

The notion of conceptualisation, conceptual models, and concepts are far older than considered in Computer Science. The earliest contribution to models and their conceptualisations we are aware of is pre-socratic philosophy and especially the work by Heraclitus [Leb14].

2.5 Models for Communication and Human Interaction

Human communication heavily uses models. They are often not called models. Some models might be metaphors or prototypes. Other models might be incomplete or not really coherent or consistent. They are however used for exchange of opinions among users. Models function in communication scenario as a medium. The communication itself determines the role and thus the function and therefore the purpose of the model. Models represent in this case a common understanding of the communication partners. They are biased by these partners. Communication is based on some common understanding about the topic that is under consideration. Partner have already agreed on some background. They use this agreement within their communication. This agreement is based a common reflection and some common model. This model is taken for granted and not further discussed in communication. So, partners agree on some background or deep model. Typically, deep models [KT17, Tha17b] are not explicitly communicated. We need however an understanding of a theory of deep model and return to it in the next Section. The model is used for a shared understanding, for sense making, for reflection, for derivation of open issues, and for negotiation.

The hexagon in Figure 9 shows the differences between models in Mathematical Logics or sciences and communication model. The main difference is the explicit community dependence of such models. Each of the partners or agents a has some understanding of the world. This understanding is the main ingredient of a personal model that we call below perception model. The perception model also reflects the setting of the agent, especially the orientation and the priming. The communication might also be based on some common understanding, i.e. on a situation model. The situation model represents the common world view, shared knowledge and beliefs, and shared opinion.

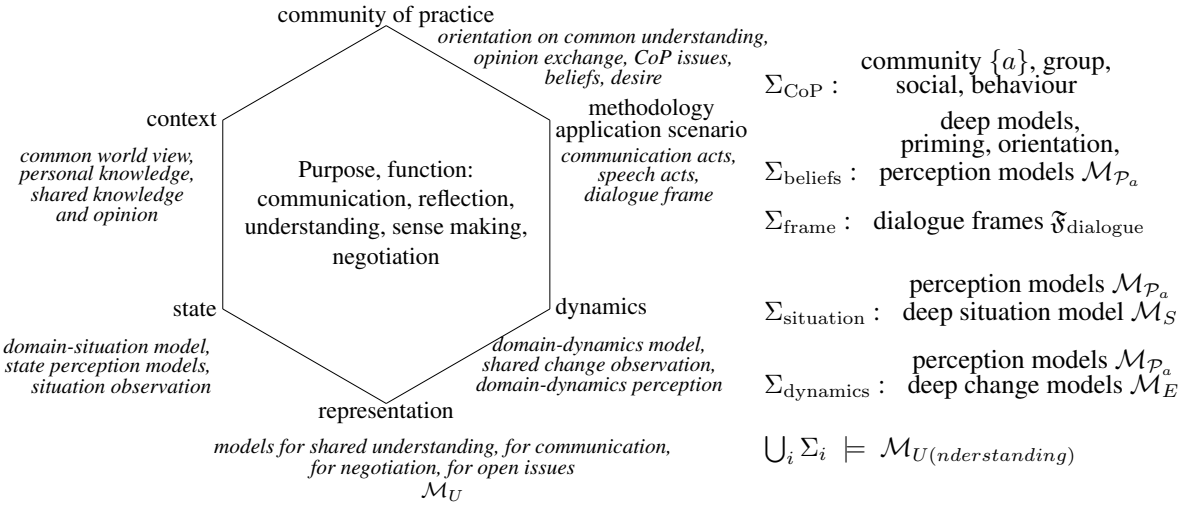

Figure 9: Models in human interaction: development of common understanding, exchange of opinion, communication, reflection, negotiation; context on the basis of commonalities in world views as deep models; scenario based on communication acts

The modelling methods is governed by communication and human interaction. So, we might base the frame on the dialogue and interaction frame. Models play a different role. They are used for common understanding. Typical specific models for human interaction are metaphors [Lak87].

Our second case shows the differences and also commonalities between Mathematical Logics and human interaction. The model must suffice all hidden agreements within the community of practice, the context, and the specific scope and focus taken by the agents. Therefore, the logics becomes now more advanced. Mathematical Logic as the opposite is oriented on general laws and thus not oriented on one model but rather on a family of models.

2.6 Lessons Learned with the Case Studies

We may now summarise the experience we gained:

- We realise by these case studies that there exists a common framework to models, to the activities of modelling and to modelling as a systematics reflection, for development of models, and for utilisation of models.
- Models are used to represent certain issues. They are more focused and must serve its purpose. The purpose and the focus determine which kind of adequacy is appropriate.
- Models do not exist on their own. They represent something in the world. The world under consideration depends again on the modelling frame. In most cases, mental models and perception or situation models are the origins which are reflected by the model.
- The justification must be given in a way that can be accepted by its community of practice. Models are developed by some members of this community and are utilised by some – may be other – members of this community of practice. So, models must be satisfying. Therefore, we need an explicit understanding of the sufficiency and thus quality of the given model.
- Models are composed of models that reflect their background and of models that represent specific states and situations within from one side and specific dynamics.
- Models are used as instruments in certain scenarios. They have a number of specific functions in these scenarios.
- Models are typically multi-models, i.e. an association of models which are reflecting specific sides of the same issue depending on the viewpoint that is actually considered. Since such models must be coherent we may bundle them within a model suites [DT10, Tha10].

- Model development and model utilisation typically follow canonical stories. An example is mathematical modelling that consists of a six-step procedure. similar procedures can be observed for most sciences that start with a research question, initialise a certain research agenda or problem solving program or schedule, adapt elements to be used to this program, and then solve a problem. Solution-faithfulness is assumed as a hidden quality characteristics beyond the problem invariance. Modelling is typically based on some specific method or methodology, e.g. the mathematical method. These methods are a mould for the modelling process itself, e.g. a pattern, template, stereotype, work-holding attachment, and an appliance. The method itself follows a macro-model.

- Modelling is still a big challenge to science and has a lot of lacunas. The biggest lacunas seems to be the missing support for combined model-based reasoning. Conceptual modelling uses a specific kind of layered model-based reasoning with changing reasoning methods depending on the stage of model development and model utilisation, e.g. in greenfield development of conceptual models: settlement of the context and the method, transfer of mentalistic concepts to codified ones with a concept expression language, transfer of domain-situation models to raw conceptual models, language-backed negotiation and agreement on a number of conceptual models that allow reflexion of different viewpoints, maturation of these conceptual models, and proper documentation. The reasoning method changes according to the stages. The integration of all these reasoning methods into a holistic one is not required.

3 Towards a General Theory of Models

3.1 Deep Models and the Modelling Matrix

The context and methodology layer determines the set-up of the model. It is often taken for granted and as given. It makes modelling more economical and also more reliable. A number of quality characteristics can be thus satisfied without any further consideration. Model development is typically based on an explicit and rather quick description of the 'surface' or normal model and on the mostly unconditional acceptance of this set-up. In reality, this setting becomes an essential part of the model. We call it deep model [Tha18b]. It directs the modelling process and the surface or normal model. Modelling itself is often understood as development and design of the normal model. This approach has the advantage that the deep model can be used as a basis for many models.

The set-up is the modelling matrix behind the model. It consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. It uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice.

The modelling matrix is often given as a stereotype one should follow while developing the normal model. Adequacy and dependability of is partially already defined by such stereotypes. The stereotype of a modelling process is based on a general modelling situation.

Stereotypes determine the model kind, the background and way of modelling activities. They persuade the activities of modelling.

3.2 The Five Concerns and a General Approach to Modelling

The case studies led us to the conclusion that there is a common three-layer setting in modelling:

(1) Community and scenario setting: The community governs the function that a model has to serve according to their issues and scenario.

Community of practice and application cases: The community of practice has its needs and desires. It faces a number of application cases. The application case consists of tasks that should be accomplished. These tasks form the community portfolio. The application cases can be solved by a model, i.e. the model functions as an instrument. Community members determine which model functions best. The community agrees on the issues for modelling.

(2) Guiding settings: The deep model and the matrix is commonly agreed according to the setting in the first layer.

Context: Modelling has its implicit and sometimes also its explicit context. Knowledge and disciplinary schools of thought and understanding are considered to be fixed. In a similar form, the background is fixed. This context forms the deep model that underpins the entire modelling process. A typical element of the deep model is the school of thought.

Modelling methodology and application mould: Modelling follows typically practices that are accepted within the community of practice. These practices are often stereotyped. The methods that are used for model development

(3) Origins and targets: Members of the community form their personal perception models and share their domain-situation model that characterises states and dynamics in the application domain that is of interest. These models are the origins on which the normal model is formed as an extension of the deep model.

The final result is a model that combines the normal and the deep models. The representation of the final model must not show all details of the deep model.

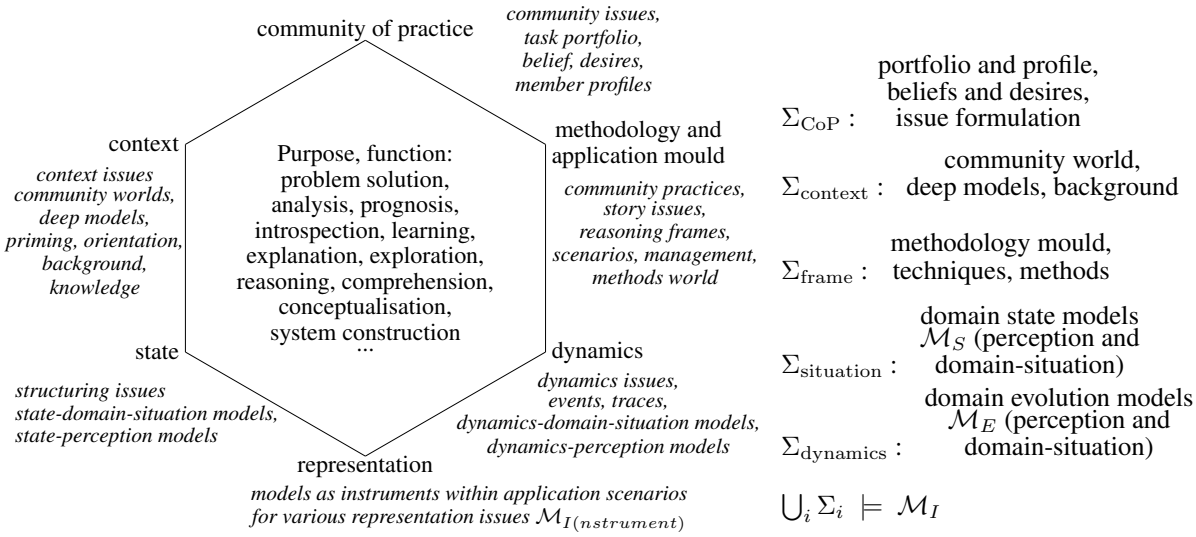

Figure 10: The five concerns for models as a kernel for a theory of models and of modelling

This general setting takes us back to the rhetorical frame[8] and its generalisation to the W*H specification framework [DT15]: In our case, the model ("what") incorporates the meaning of parties (semantical space; "who") during a discourse ('when') within some application with some purpose ("why") based on some modelling language.

We thus distinguish between five grounding and driving perspectives to models:

Community perspective: The community has intentionally set-up its application cases, its interests, its desires and its portfolio. The community communicates, knows languages, explains, recognizes, accept the grounding behind the models, has been introduced to the basis and is common with it. Models are used by, developed by and for, and gain a surplus value for a community of practice. They may have a different shape, form, and value for community members. They must, however, be acceptable for its community. Typical specialisations of this concern are 'by whom', 'to whom', 'whichever', and 'worthiness'.

Purpose, function, goal perspective: Models and model development serve a certain purpose in some utilisation scenarios. The model has to function in these scenarios and should thus be of certain quality. At the same time it is embedded into the context and is acceptable by a community of practice with its rules and understandings. We answer 'why' and 'for which reason' questions.

[8]It relates back to Hermagoras of Temnos or Cicero more than 2000 years ago., i.e. they are characterised through "who says what, when, where, why, in what way, by what means" (Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis).

Product perspective: Models are products that are requested, have been developed, are delivered according to the first perspective, are potentially applicable within the scenarios, and have their merits and problems. Typical purpose characteristics are answers to 'how-to-use', 'why', 'whereto', 'when', for which reason'' and 'wherewith' (carrier, e.g., language) questions.

Engineering perspective: Models are mastered within an engineering process based on some approaches to modelling activities and to utilisation of models. Modelling is a systematically performed process that uses methods, techniques, preparations, and experience already gained in former modelling processes. The modelling method is typically given in a canonical form. It guides and steers the model development and the model utilisation processes. This guidance can be derived from the scenarios in which the model functions.

Background and context perspective: Model development and utilisation is a systematic, well-founded process that allows one to reason on the capacity and potential of the model, to handle adequacy and dependability of models in a proper way, and the reason on the model and its origins that it represents. A modelling culture also answers the by-what-means question beside providing the background. The background is typically considered to be given and not explicitly explained. It consists of an undisputable grounding and of a disputable and adjustable basis. The context clarifies on which basis and especially on which grounding the model has been developed and must be restricted in its utilisation. Additional context characteristics are answers to questions about the 'whereat', 'whereabout', 'whither', and 'when'.

3.3 Model-Based Reasoning

The observation depicted in Figure 6 drives us to a multi-model approach. We build models in situations, concepts and theories in dependence on their function and purpose. The same situation-concept-theory may be the basis for a variety of models. A typical multi-model approach is the consideration of models in Physics. Models should

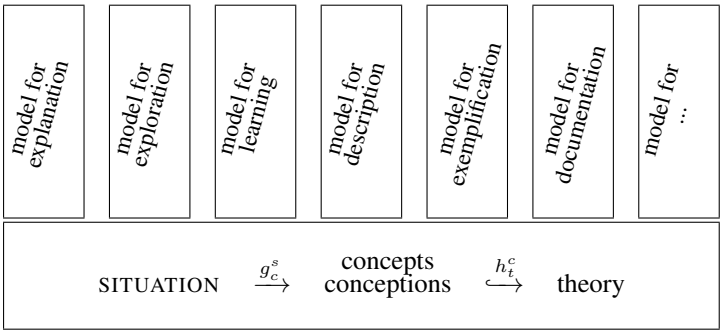

Figure 11: Models as specific representations of situations, concept(ion)s, and theories

thus be considered to be the third dimension of science [BFA$^+$16, TN15, TTF16]. Disciplines and also human communication, human interaction, and human collaboration have developed a different understanding of the notion of model, of the function of models in scientific research and communication. Models are often considered to be artifacts. In reality, they are however instruments that are used with a certain intention. Models might also be perception models that incorporate mentalistic concepts [Jac04]. Models are used in various *utilisation scenarios* such as construction of systems, verification, optimization, explanation, and documentation. *In these scenarios* they *function* as *instruments* and thus satisfy a number of properties.

Model-based reasoning [Bre10, Mag14] is enhances classical reasoning such as reasoning mathematical calculi or logical derivation. There are several kinds of reasoning that are more appropriate and widely used:

Evidence-based modelling and reasoning is one of the main approaches for quantitative models. Models only represent acceptable possibilities. Each model captures a distinct set of possibilities to which the current description refers. Possibilities are consistent with the premises and the knowledge gained so far what makes

them intrinsically uncertain because they mirror only some properties they represent. In investigative and quantitative modelling, models can be proxy-driven where the the structure of the model corresponds to the proxies it represents. They might also include abstractions such as negation which must be then stratified. Propositional evidence-based reasoning is based on monotone functions and specific interpretations for logical connectives. Models represent in this case only what has been observed and not what is false in each possibility what is different from fully explicit models which represent what is false. The more proxies are considered the richer those models are, the more accurate the world view is. Evidence-based modelling and reasoning uses pragmatic reasoning schemata, e.g. A causes B; B prevents C; therefore, A prevents C. The calculus may use several implication forms, e.g. deterministic conclusions (A cause B to occur: given A then B occurs) and ordered sets of possibilities (A enables B to occur: given A then it is possible for B to occur).

Hypothetical and investigative modelling considers different assumptions in order to see what follows from them, i.e. reasons about alternative possible worlds (i.e. states of the world), regardless of their resemblance to the actual world. Potential assumptions with their possible world conclusions and assertions are supported by a number of hypotheses (allowing to derive them). It is often combined with abductive reasoning. Evidence against hypothesis is performed by testing its logical consequences, i.e. exploring different alternative solutions in parallel to determine which approach or series of steps best solves a particular problem.

Causal reasoning and modelling is a specific variant of inductive reasoning and justification-backed truth maintenance with assertions (beliefs, background) and justifications within some context (current beliefs, justifications, arguments). It establishes the presence of causal relationships among events based on methods of agreement, difference, concomitant variation, and residues. It uses assumptions and thus avoids inconsistent sets ('nogood' environment). The environment consists of a set of assumptions, premises, assumed statements, and derived statements for the world view. Justifications (e.g. data-supported) represent cause.
Hypotheses are not derived from evidence but are added to evidence. They direct the search for evidence. They are tested by modus tollens ($(H \rightarrow I) \wedge \neg I \Rightarrow \neg H$).

Network reasoning uses models that are expressed as networks. Nodes carry justification (arguments) and status (in, out, believed, relevant, necessary, ...). Edges, hyperedges, or directed edges have an antecedent (support nodes) and conclusions. They may also be non-monotonic and enable backtracking for dependencies (causality, chronological, space, etc. Labels also express the degree of consistency and believability. Queries can be expressed as subgraphs and are evaluated by query embedding into the network.

Model-based reasoning is an interactive and iterative process that helps to digest a theory and to develop the theory. Therefore, model-based reasoning integrates many reasoning approaches, e.g. deduction, induction, abduction, Solomonoff induction, non-monotonic reasoning, and restrospective reasoning. Model refinement might also be based on inverse modelling approaches. Facets of the last one are inductive learning, data mining, data analysis, generic modelling, universal applications, systematic modelling, and pattern-based reasoning.

3.4 Towards Powerful Methodological Moulds

The hexagon picture and the consideration of the variety of different (reasoning) techniques might lead to the impression that a general treatment of models and a methodological support is infeasible. Sciences and humans have however developed their specific approaches and overcome the challenges of this complexity. We will illustrate resolution of complexity by two methods: Layered treatment and generic modelling. Both approaches are based on the separation of a model into a deep or core model and a normal model. A typical example of a methodology is the mathematical modelling method [BT15, GKBF13, vDGOG09, Pol45] (see Subsection 2.2). The CRISP cycle (data selection according to generic model, data preprocessing, data transformation, data mining, model development, interpretation/evaluation) [BBHK10] and classical investigation cycles (define issues and functions of the model, hypothetically predict model properties, experiment, (re)define model, apply and validate the model against the situation) are typical methodologies. Similar methodologies are known for data mining [Jan17], data analysis [BBHK10], and systematic mathematical problem solving [Pod01]. They use a variety of reasoning techniques and layer their application of these techniques according to the stage that is currently under consideration. These modelling methods and methodologies are used similar to moulds that are commonly used in manufacturing.

Data mining [Jan17], inverse modelling [RSS$^+$10], and generic modelling [TTFZ14] start with a generic model. A set of associated models (called model suite) is the result of a modelling process. We may develop a singleton model or a model suite. Figure 12 displays a variant that starts with an initialisation and setting of the modelling process. The initialisation is based on the issues that are important for the community of practice, the tasks that are on the agenda, and the injection of the context. The community of practice aims at completion of tasks from its portfolio and is bound by profiles of their members what also includes beliefs and desires shared in this community. At the same time, the methodology for modelling is already chosen. That means, the upper dimensions in Figure 10 governs the entire modelling process. A similar approach can be declared for model redevelopment model evolution instead of model development from scratch (greenfield modelling). The result of the first layer is a deep model and a matrix.

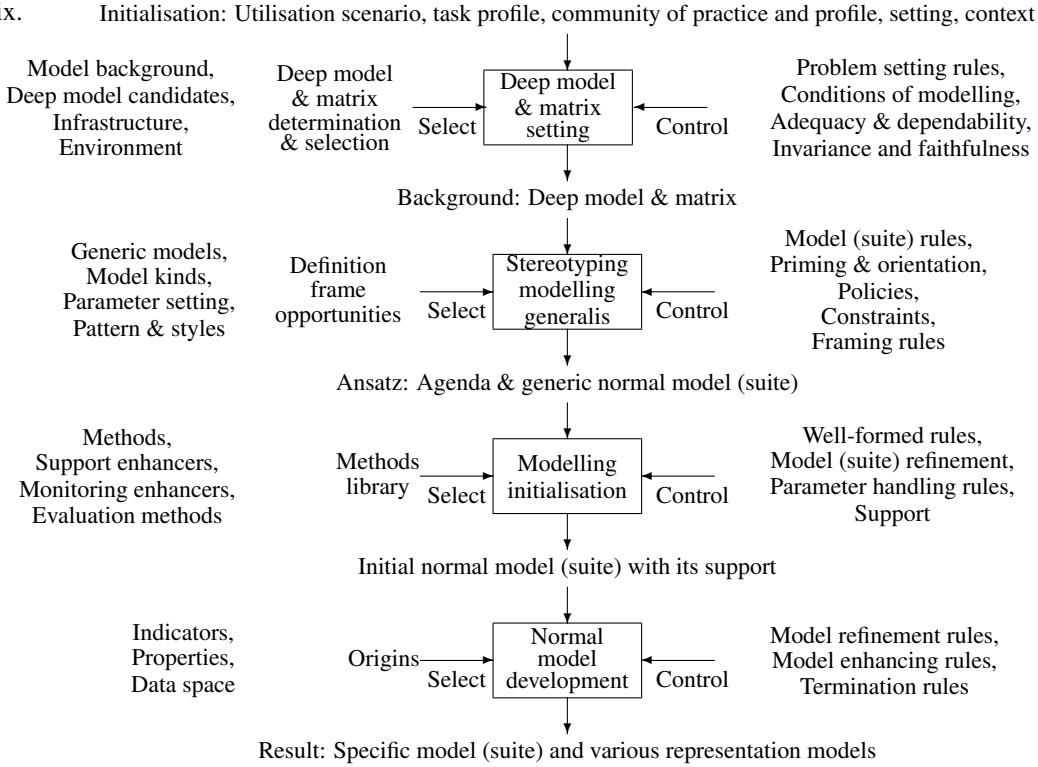

Figure 12: Layered model (suite) development (None-iterative form, greenfield variant)

The second layer or stage uses some kind of most general and refinable model as the initial model. A generic model [BST06, TF16] is a general model which can be used for the function within a given utilisation scenario and which is not optimally adapted to some specific origin collection. It is tailored in next steps to suit the particular purpose and function. It generally represents many origins under interest, provides means to establish adequacy and dependability of the model, and establishes focus and scope of the model. Modelling is often based on some experience. This experience can be systematically collected within a number of libraries. Libraries and collections are used for collecting the most appropriate setting and model. This selection is controlled or governed by rules, restrictions, conditions, and properties. The main results of the second layer are generic models and an agenda for the next modelling steps.

The third layer sets the environment for the development of the normal model. This environment prepares model development on the basis of the generic models and under inclusion of the deep model. The section of methods might also include the selection of parts and pieces from the context, e.g. from the background and especially from theories and knowledge. The fourth layer results then in the development of a normal model that can be neatly combined with the deep model. Representation models are developed for different members of the community of practice and for different functions the model must fulfill in the utilisation scenario.

This development process is often cut down to the fourth layer assuming the results of the first, second, and third

layer as already given. This kind of implicitness has often been assumed for language utterance. The government and binding approach [Cho82, BST06] made the two-step generation of sentences explicit: we intentionally prepare the deep model and then express ourselves by an explicit statement which is build similar to a combination of a normal model and of a cutout of the deep model.

4 Conclusion

A collection of modelling approaches has been presented in [TN15]. It seems that the variety of modelling approaches, the different utilisation of model, the broad span of underpinning theories, the variety of models themselves do not allow to develop a common setting for models. We often met the claim that models used in social and natural sciences, in mathematics, in logics and in daily life are so different that a common treatment cannot exist. From the first side, logicians provided a specific understanding of models that is easy and formally to handle. They inspired model research and the notion of model, e.g. [Bal16, Kas03, Mah09, Mah15, Sta73, Ste66, Ste93]. This notion has mainly been based on properties that a model should satisfy: mapping, truncation, and pragmatic properties as phenomenalistic characterisation of the notion. From the second side, models in all sciences have been used as an artifact for solution of problems, e.g. [BT15, Her84, vDGOG09, vN55]. The model notion has been enhanced by amplification, distortion, idealisation, carrier, added value, and purpose-preservation properties. From the third side, language- and concept-based foundations of models have been developed in philosophy of science and linguistics [Blfrm[o]–5, Bur15, Cas55, KL13, Lat15, Pei98]. From the fourth side, models in engineering [BFA$^+$16, LH15, TD16, TTF16] are instruments for system construction. From the sixth side, models are also instruments in human interaction. They are used as metaphors, for communication, for brief reference, for depiction, as prototype, etc. For instance, the question whether a picture or a photo is a model depends on their utilisation in some interaction scenarios. We thus may conclude that a common science and culture of modelling cannot exist.

The main claim in this paper is however that a common treatment of models in science and human interaction can be developed. We base our foundational framework on a separation of concern. This separation into five governors for models provides a common treatment of models and model utilisation. We base our framework on the observation that not all concerns are considered at the same time. So, we can use some kind of stepwise procedure for model development.

Utilisation of models as instruments in scenarios is the main driving property that distinguishes something from a model. The model functions in scenarios such as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. How the model functions has been illustrated in the case of model-based reasoning. Model-based reasoning goes far beyond model methods used in classical first-order predicate logics or mathematics. We use the layering approach also for model methods since the development of a general reasoning method is far beyond the horizon.

The meta-models of modelling concerns in Figures 3, 4, 7, 8, 9, 10 support the layered modelling method in Figure 12. Instead, we could separate the layers into communities and their application scenario, into background and methodology setting, into situation and theory setting, into origin calibration, and model delivery layers.

This paper has been centred around models, theories, communities, context, methodologies, state, and dynamics at the same level of abstraction. Model-driven development and architecture [MMR$^+$17, SV05] is an orthogonal approach to this paper. It distinguishes abstraction layers for models (M1), model frames (M2) [as meta-models], model frameworks (M3) [as meta-meta-models], and model framework setting (M4). The data/information and traces/events abstraction layer (M0) underpins models. Our approach has been mainly oriented on M1. We envision that the general M0-M1-M2-M3-M4 architecture can be integrated into our approach as well.

References

[ASG13] B. F. Albdaiwi, B. Szalkai, and V.I. Grolmusz. Finding combinatorial biomarkers for type 2 diabetes in the camd database. In *European Biophysics Journal with Biophyscs Letters*, volume 42, pages S195–S195. SPRINGER 233 SPRING ST, NEW YORK, NY 10013 USA, 2013.

[Bab03] B. E. Babich. From Fleck's Denkstil to Kuhn's paradigm: conceptual schemes and incommensurability. *International Studies in the Philosophy of Science*, 17(1):75–92, 2003.

[Bal82] W. Balzer. *Empirische Theorien: Modelle - Strukturen - Beispiele*. Vieweg-Teubner, 1982.

[Bal16] W. Balzer. *Die Wissenschaft und ihre Methoden*. Karl Alber, 2016.

[BBHK10] M.R. Berthold, C. Borgelt, F. Höppner, and F. Klawonn. *Guide to intelligent data analysis*. Springer, London, 2010.

[BFA+16] M. Bichler, U. Frank, D. Avison, J. Malaurent, P. Fettke, D. Hovorka, J. Krämer, D. Schnurr, B. Müller, L. Suhl, and B Thalheim. Theories in business and information systems engineering. *Business & Information Systems Engineering*, pages 1–29, 2016.

[Blfrm[o]–5] C. Blättler. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Das Modell als Medium, pages 107–137. De Gryuter, Boston, 2015.

[Bre10] J.E. Brenner. The logical process of model-based reasoning. In L. Magnani, W. Carnielli, and C. Pizzi, editors, *Model-based reasoning in science and technology*, pages 333–358. Springer, Heidelberg, 2010.

[BST06] A. Bienemann, K.-D. Schewe, and B. Thalheim. Towards a theory of genericity based on government and binding. In *Proc. ER'06, LNCS 4215*, pages 311–324. Springer, 2006.

[BT15] R. Berghammer and B. Thalheim. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Methodenbasierte mathematische Modellierung mit Relationenalgebren, pages 67–106. De Gryuter, Boston, 2015.

[Bur15] T. Burkard. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Der Blick des Philologen. Modelle 'Literatur als Text' in der Klassischen Philologie, pages 175–217. De Gryuter, Boston, 2015.

[Cas55] E. Cassirer. *The Philosophy of Symbolic Forms*, volume 1–3. Yale University Press, New Haven, 1955.

[CH04] S. Chadarevian and N. Hopwood, editors. *Models - The third dimension of science*. Stanford University Press, Stanford, California, 2004.

[Cho82] N. Chomsky. *Some concepts and consequences of the theory of government and binding*. MIT Press, 1982.

[DT10] A. Dahanayake and B. Thalheim. Co-evolution of (information) system models. In *EMMSAD 2010*, volume 50 of *LNBIP*, pages 314–326. Springer, 2010.

[DT15] A. Dahanayake and B. Thalheim. W*H: The conceptual model for services. In *Correct Software in Web Applications and Web Services*, Texts & Monographs in Symbolic Computation, pages 145–176, Wien, 2015. Springer.

[Fle11] L. Fleck. *Denkstile und Tatsachen, edited by S. Werner and C. Zittel*. Surkamp, 2011.

[GKBF13] G. Greefrath, G. Kaiser, W. Blum, and R. Borromeo Ferri. *Mathematisches Modellieren für Schule und Hochschule*, chapter Mathematisches Modellieren - Eine Einführung in theoretische und didaktische Hintergründe, pages 11–37. Springer, 2013.

[Gra07] J. Gray. eScience: A transformed scientific method. Technical report, Talk given Jan 11, 2007. Edited by T. Hey, S. Tansley, and K. Tolle. http://research.microsoft.com/en-us/um/people/gray/talks/NRC-CSTB_eScience.ppt, Microsoft Research Publications, 2007.

[Her84] H. Hertz. *Die Prinzipien der Mechanik in neuem Zusammenhange dargestellt*. Akad. Verl.-Ges. Geest und Portig, Leipzig, 2. aufl., nachdr. der ausg. edition, 1984.

[Jac04] R. Jackendoff. *Foundations of languages: Brain, meaning, grammar, evolution*. Oxford University Press, 2004.

[Jan17] K. Jannaschk. *Infrastruktur für ein Data Mining Design Framework*. PhD thesis, Christian-Albrechts University, Kiel, 2017.

[Kas03] R. Kaschek. *Konzeptionelle Modellierung*. PhD thesis, University Klagenfurt, 2003. Habilitationsschrift.

[KL13] B. Kralemann and C. Lattmann. Models as icons: modeling models in the semiotic framework of peirce's theory of signs. *Synthese*, 190(16):3397–3420, 2013.

[KT17] Y. Kropp and B. Thalheim. Data mining design and systematic modelling. In *Proc. DAMDID/RCDL'17*, pages 349–356, Moscov, 2017. FRC CSC RAS.

[Lak87] G. Lakoff. *Women, fire, and dangerous things - What categories reveal about the mind*. The University of Chicago Press, Chicago, 1987.

[Lat15] C. Lattmann. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Die Welt im Modell. Zur Geburt der systematischen Modellierung in der Antike, pages 307–327. De Gryuter, Boston, 2015.

[Leb14] A.V. Lebedev. *The Logos Heraclitus - A reconstruction of thoughts and words; full commented texts of fragments (in Russian)*. Nauka, 2014.

[LH15] J. Leibrich and P.A. Höher. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Modelle in der Kommunikationstechnik, pages 329–345. De Gryuter, Boston, 2015.

[Mag14] L. Magnani, editor. *Model-Based Reasoning in Science and Technology*. Springer, 2014.

[Mah09] B. Mahr. Information science and the logic of models. *Software and System Modeling*, 8(3):365–383, 2009.

[Mah15] B. Mahr. Modelle und ihre Befragbarkeit - Grundlagen einer allgemeinen Modelltheorie. *Erwägen-Wissen-Ethik (EWE)*, Vol. 26, Issue 3:329–342, 2015.

[Men89] W. Menke. *Geophysical Data Analysis: Discrete Inverse Theory*, volume 45 of *International Geophysics*. Academic Press Inc., 1989.

[MMR+17] H.C. Mayr, J. Michael, S. Ranasinghe, V.A. Shekhotsov, and C. Steinberger. Model centered architecture. In *Conceptual Modeling Perspectives*, pages 85–104, Cham, 2017. Springer.

[Mül16] R. Müller. Model history is culture history. From early man to cyberspace. http://www.muellerscience.com/ENGLISH/model.htm, 2016. Assessed Oct. 29,2017.

[Noa09] K. Noack. Technologische und methodische Grundlagen von SCOPELAND. White paper, www.scopeland.de, 2009.

[Pei98] C.S. Peirce. What is a sign? In Peirce Edition Project, editor, *The essential Peirce: selected philosophical writings*, volume 2, pages 4 – 10. Indiana University Press, Bloomington, Indiana, 1998.

[Pod01] A.S. Podkolsin. *Computer-based modelling of solution processes for mathematical tasks (in Russian)*. ZPI at Mech-Mat MGU, Moscov, 2001.

[Pol45] G. Polya. *How to solve it: A new aspect of mathematical method*. Princeton University Press, Princeton, 1945.

[Rei84] H. Reichel. *Structural induction on partial algebras*. Mathematical research, 18. Akademie-Verlag, Berlin, 1984.

[RSS+10] J. Rückelt, V. Sauerland, T. Slawig, A. Srivastav, B. Ward, and C. Patvardhan. Parameter optimization and uncertainty analysis in a model of oceanic co2-uptake using a hybrid algorithm and algorithmic differentiation. *Nonlinear Analysis B Real World Applications*, 11(5):3993–4009, 2010.

[ST08] K.-D. Schewe and B. Thalheim. Semantics in data and knowledge bases. In *SDKB 2008*, LNCS 4925, pages 1–25, Berlin, 2008. Springer.

[Sta73] H. Stachowiak. *Allgemeine Modelltheorie*. Springer, 1973.

[Ste66] W. Stegmüller. Eine modelltheoretische Präzisierung der Wittgensteinschen Bildtheorie. *Notre Dame Journal of Formal Logic*, 7(2):181–195, 1966.

[Ste93] W. Steinmüller. *Informationstechnologie und Gesellschaft: Einführung in die Angewandte Informatik*. Wissenschaftliche Buchgesellschaft, Darmstadt, 1993.

[SV05] T. Stahl and M. Völter. *Model-driven software architectures*. dPunkt, Heidelberg, 2005. (in German).

[TD16] B. Thalheim and A. Dahanayake. Comprehending a service by informative models. *T. Large-Scale Data- and Knowledge-Centered Systems*, 30:87–108, 2016.

[TF16] M. Tropmann-Frick. *Genericity in Process-Aware Information Systems*. PhD thesis, Christian-Albrechts University of Kiel, Technical Faculty, Kiel, 2016.

[Tha10] B. Thalheim. Model suites for multi-layered database modelling. In *Information Modelling and Knowledge Bases XXI*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 116–134. IOS Press, 2010.

[Tha14] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[Tha17a] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. Springer US, 2017.

[Tha17b] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[Tha17c] B. Thalheim. Model-based engineering for database system development. In *Conceptual Modeling Perspectives*, pages 137–153, Cham, 2017. Springer.

[Tha18a] B. Thalheim. Conceptual model notions - a matter of controversy; conceptual modelling and its lacunas. *EMISA International Journal on Conceptual Modeling*, February:9–27, 2018.

[Tha18b] B. Thalheim. Normal models and their modelling matrix. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 44–72. College Publications, 2018.

[TN15] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[TT13] M. Tropmann and B. Thalheim. Mini story composition for generic workflows in support of disaster management. In *DEXA 2013*, pages 36–40. IEEE Computer Society, 2013.

[TTF16] B. Thalheim and M. Tropmann-Frick. Models and their capability. In C. Beierle, G. Brewka, and M. Thimm, editors, *Computational Models of Rationality*, volume 29 of *College Publications Series*, pages 34–56. College Publications, 2016.

[TTFZ14] B. Thalheim, M. Tropmann-Frick, and T. Ziebermayr. Application of generic workflows for disaster management. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 64–81. IOS Press, 2014.

[vDGOG09] C. von Dresky, I. Gasser, C. P. Ortlieb, and S Günzel. *Mathematische Modellierung: Eine Einführung in zwölf Fallstudien*. Vieweg, 2009.

[vN55] J. von Neumann. *The Unity of Knowledge*, chapter Method in the Physical Sciences, page 628. ???, 1955.

[Wec92] W. Wechler. *Universal Algebra for Computer Scientists*. Springer-Verlag, Berlin, 1992.

[Wik17] Wikiquote. Conceptual model. https://en.wikiquote.org/wiki/Conceptual_model, 2017. Assessed Nov. 21, 2017.

Digital Playground for Policy Decision Making

© J. Hedtrich
© Prof. C. Henning
© E. Fabritz
© Prof. B. Thalheim

Kiel University
Department of Agro-Economy and Department of Computer Science
24098, Kiel, Germany

johannes.hedtrich@ae.uni-kiel.de
chenning@ae.uni-kiel.de
ef@is.informatik.uni-kiel.de
thalheim@is.informatik.uni-kiel.de

Abstract. Policy development is a complex and highly dimensional process. This complexity is very difficult to comprehend due to complexity of the parameter space, multi-dependence of parameters, and the nature of process. Therefore, policy makers should be supported while considering and evaluating various alternative decisions. This paper illustrates a modeling approach for advisory and assistance in decision making for political practitioners. We describe the corresponding advisory tool supporting the interactive decision process.

Keywords: Computer-based communication tool, interactive learning between scientific models and practitioners, political decision making support

1 Introduction

Policy decision making is a complex task which comprises the understanding of possible positive or negative consequences of decisions as well as a mechanism to restore consistency of a system in the case of inappropriate decisions. Thus, even policy experts often have only a vague understanding of how policies impact on relevant outcomes. Therefore, political practitioners use simple mental models (beliefs) to understand complex impacts of policies. For this reason, a technical solution for the simulation of policy impacts can be helpful, e. g. a graph displaying the impact of parameters. Our software will work as a digital playground system with relevant decision parameters as inputs and implied outcomes (consequences of the decision) as outputs.

Nowadays it is commonly accepted that good economic policy has to be evidence-based, i.e. rest on scientific knowledge and statistically proven evidence. However, scientific modeling is often criticized by political practitioners as a purely academic exercise that fails to provide practical tools for understanding or designing optimal real-life economic processes [5]. Accordingly, scholars promote participatory policy analysis that is characterized by an interaction between economic theory and political practice to combine the 'objective' knowledge derived from economic theories and empirical data with the 'subjective' knowledge of stakeholder organizations as political practitioners ([2], [9], [5]). Moreover, inadequate communication between scientific policy analysts and political actors is proposed to be a principal cause of the limited impact of research on policymaking. For example, the 'utilization of knowledge school' emphasizes the fact that policy analysts and policymakers live in two separate communities [5]. Hence, to become more efficient, the relationship between scientific experts and policy actors must be redefined.

Moreover, Stiglitz argues in his highly recognized book "Whither socialism?" [14] that the market-socialist experiences in Eastern Europe failed due to the incorrect beliefs of politicians in the Arrow-Debreu concept of real market economies as a complete set of competitive markets ([14], Chapter 11). Interestingly, Stiglitz's explanation of the failure of the market socialism experiment highlights an interesting general point: economics must be recast as something more than a constrained maximization problem to understand and design real economies. In other words, theoretical models provide a relevant benchmark for understanding real-life economic processes but require abstract scientific models and political praxis to actually change the world. Hence, as previously discussed in [6], [12], [7], identifying effective solutions for central economic problems appears to be a problem of linking abstract economic theory with feasible political practice. Accordingly, scholars of participatory policy analysis discussed innovative tools, such as participative modeling (see [5]) (i.e., improving communication in formal models by means of interactive or man-machine simulations [for example, see [1]] or decision seminars [10]).

Beyond interesting methodological ideas and concepts for assessing the role of relevant 'objective' scientific knowledge it is important to better understand

Proceedings of the XX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 9-12, 2018

and design the complex communication processes between science and political practitioners in a way that combines the knowledge of both worlds to generate advanced solutions to existing economic problems, such as the transformation to a sustainable bio-economy or reaching sustainable development goals.

In this context the paper develops a computer-based tool Policy-Lab that facilitates an interactive communication and learning between political practitioners and scientific models. Figure 1 shows a graphical presentation of positioning of scientific models' statements (scientific world), political practitioners' beliefs (stakeholder beliefs world) and the aspired communication between these two worlds.

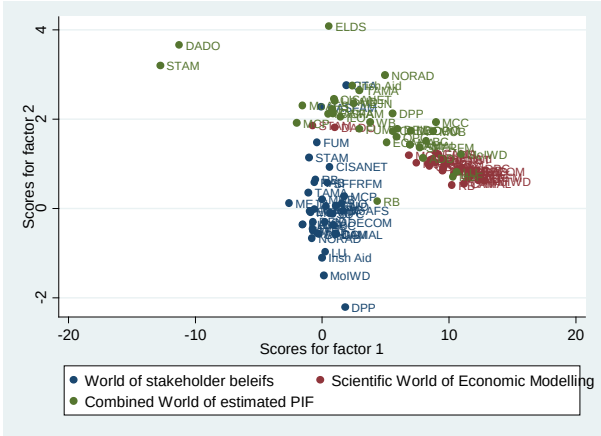

Figure 1

2 Policy-Lab tool

The Policy-Lab tool has to fulfill various tasks in order to effectively support the decision-making process and facilitate the learning of stakeholders. Those tasks can be categorized as follows:

- Input Device: Survey policy preferences, goals and beliefs using questionnaires.
- Report Device: Report surveyed data back to the group. This requires the dynamic application of statistical analysis of the data.
- Interactive Modelling Device: Users can simulate different policies and evaluate their impact on policy goals.
- Consensus Device: This device provides support in finding a potential political compromise.

An integral part of those devices is the simulation of scientific models. For example, typical formulas used in such simulation that political practitioners should understand in order to make a decision in the economy area look like the following one:

$$\gamma = [\gamma_i], i \in I = \{1, \dots, n_i\}; s \in S = \{1, \dots, n_s\};$$
$$j \in J = \{1, \dots, n_j\};\ t \in T = \{0, \dots, 9\}$$

$$Be_s(\gamma) = \eta_s \left(\sum_{i \in I} \mu_{i,s} \gamma_i^{-\rho} \right)^{-\frac{1}{\rho}}$$

$$tp_s(\gamma) = tp_s^{max} \frac{e^{a_s Be_s(\gamma) + b_s}}{e^{a_s Be_s(\gamma) + b_s} + 1}$$

$$wZ_j(\gamma) = \xi_j^{const} + \sum_{s \in S} tp_s(\gamma) \cdot \xi_{s,j}$$

$$Z_{j,t}(\gamma) = Z_{j,0} \cdot \left(1 + t \cdot wZ_j(\gamma)\right)$$

In this case developing a decision is rather difficult and some supporting technical solutions are necessary.

To support the simulation of these models technical methods and frameworks are used. The methods being used during the simulation are mathematical statistical methods (Bayesian model averaging, Meta Modelling) and programming languages for statistical computing (R) and optimization problems (GAMS).

In order to make models accessible to a wide range of users, who in this case are organizations or individuals, who are interested in the construction of economic policies (in our case these are agricultural policies [8]), an intuitive visualization is required. The visualization part of the tool should work as a playground for model simulation supporting expert learning, model learning, interactive learning (expert-model-expert exchange), and learning from collective decision (voting over policies or exchange games).

Thus, the Policy-Lab tool should work as an interactive input-output playground for the models' simulation and graphical visualization.

At the same time, the tool should process a large amount of model specific data: different kinds of input-output parameters and computational cores of the models. So an important issue during the tool development is the implementation of a suitable database structure.

The Policy-Lab tool will be implemented in the form of a web application. The tool is now in the creation phase, for this reason the main concepts of tool development, tool requirements, and its structure will be discussed further.

2.1 Theoretical concepts

Some theoretical concepts will be explained before the structure of the playground is going to be introduced.

1) What is a model from the tool's perspective?

In the sense of the current tool, a model is a computable unit with defined input parameters, computational core, and computed output parameters, which can be shown in a graphical form. A special sub-type of a model is a questionnaire, that has input parameters and computational core, which adds user input to a statistical model and recalculates its output. The output of recalculation is not shown to the users directly, but can be called from another view.

2) How model data will look like?

The computational core of a model is predefined by the model scientists. It can be written in R, GAMS or in other programming language. The input and output parameters depending on the language used are language specific character values, which can be saved in a database or in an external file. These parameters should be accessible to the playground.

2.2 Playground system requirements

The creation of the simulation tool begins with the comprehension of required features. Partly this information can be derived from the existing Policy-Lab tool prototype, partly from model scientists' requirements and user expectations.

The list of requirements for the simulation tool includes the following:

- clear and comprehendible software structure
- clear and comprehendible database structure
- scalability of the system
- maintainability of the system
- efficiency of the system
- run-time reciprocative input-output system
- user-friendliness of the system

Based on the analysis of system requirements the following issues can be defined during the development of the tool:

- How to implement interactive forms for user-input and output? Which interfaces are needed?
- How input and output parameters for the models look like and how they are saved?
- How the communication between the computational module and the web interface looks like?

2.3 Playground system structure

The simulation tool should serve as a web information system for model simulations, with interactive input-output mechanisms for users. The system should have a clear structured database, expandable for new entities, since the system will describe a varying amount of models. The system should visualize a list of models and its descriptions for users. Further the system should have views for input parameters from users and possibilities for the graphical presentation of computed output. Another integral part of the system is a computational module, where the computation of output takes place.

According to the system requirements the new system should have the following components:

- Web interface for users with possible use-cases' definition, user management functions, presentation of views related to models, including model-input-parameters and output graphics.
- Computational module with possible integration of R and GAMS sub-modules.
- Communicational interface: beside other functions web interface and computational module should be capable of interaction with each other.
- Database for the web interface
- Database for the computational module

Web interface
Web interface is a unit that contains common login, logout, and register functions, explanative use-cases, overview of present models, view for input parameters for the models, view for the output in graphical form. Moreover, there should be a separate view for administrators to allow user management.

Computational module
Computational module is a unit that can be connected to R or GAMS sub-modules or use some other language for computation. This module should communicate with the web interface: parse user-input-parameters, convert them to input-parameters in the format of computational language depending on the model, parse computed output back to the chosen web interface format (e.g. JSON).

Database for the web interface
Database for the web interface should contain all the information about users and their management, widgets shown in the interface, and shown model views. Furthermore, for the presentation of input and output this database should have information about input and output parameters of a model.

Diagram 1 shows a fragment of a possible ER-schema for the database:

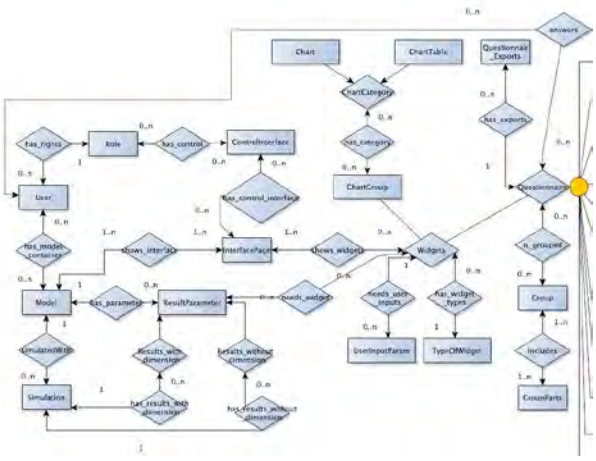

Diagram 1

The ER-schema describes users, their roles, and interfaces that depend on roles. Further, the schema

includes descriptions of models, their simulations and different types of simulation result parameters. Additionally, every interface page has specific widgets of different types depending on model being simulated, including charts and questionnaires.

Database for the computational module

In the case computation is produced in another application it needs its separate database.

The database for the computation should have information about models, their computational cores, and their input-output parameters.

If the computation module does not need its own database, analogical database entities are necessary.

A possible ER-schema of a computational module is shown in Diagram 2:

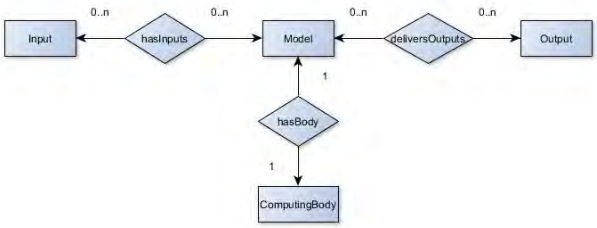

Diagram 2

Communication between web interface and computational module

Communication between these two modules is an important part of the system, the whole software structure and efficiency depends on the form of communication.

Two architectural alternatives for modules' communication have been developed:

1) Web interface and computational modules can be placed inside of one software project, so that the division in interface and computation is only a logical notion. In this case the interface and computational parameters can be saved in the same database. The computation itself can be made, for example, with JavaScript language. In the case of JavaScript, the computation will proceed efficiently as no integration of external R and GAMS modules is needed. The communication in this case is trivial and proceeds within one application.

2) In the other case, R and GAMS modules can be stored in a separate application, if the computation needs these modules because of its complexity, as it allows to bring a modular structure to the software. In addition, the exchange of or changes in R or GAMS models are made easier, because they do not influence the execution of the web interface in a negative way. Thus, the two components are not only logically, but also physically separated from each other. The communication between prototype tool and the application where model computation takes place proceeds with HTTP-messages, containing input-output parameters for computation and information about models in JSON format.

Figure 2 illustrates, how this kind of architectural style can be implemented:

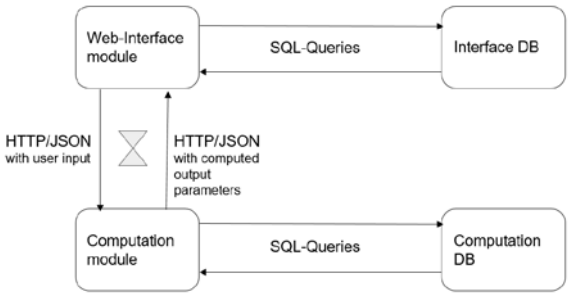

Figure 2

In the system the both ways of communication will be used, depending on the complexity of a model.

2.4 Advantages of the system

The described playground system has a number of advantages:

- The system is scalable and extendable, as the underlying web information system is dynamic and is built accordingly to the database contents. The expandable database allows the insertion of new visual elements and models for the simulation.
- The first architectural style for communication allows the implementation of a run-time reciprocative input-output system.
- The second architectural style for communication contributes to system's modularity and can be approached from two different perspectives: web interface based and computation based perspective. Thus, two scientists can work simultaneously on the two components. Any changes in one of the components would not cause error or stoppage of the execution in the other component. After the adaption of communicational modules, the changes can be accepted by both components.
- The tool supports expert, model and interactive learning, moreover the learning from collective decision is implementable.
- Description of use-cases supports user-friendliness.

3 Conclusion

Described Policy-Lab tool facilitates political decision making by presenting an interactive playground system, that simulates a large opportunity space for policy

decisions and computes possible effects of the model simulation with the decisions made.

As a result, Policy-Lab tool for policy decision enables political practitioners to relate potential policy decisions to corresponding outcomes.

The described tool should be flexible, efficient and user-friendly, in order to be able to simulate the full complexity of the models and to assist in successful decision making.

Related work

There exist other systems, which work with interactive user input-output and use a large number of possible input parameters and calculations, beside the Policy-Lab tool prototype, the precursor of the current simulation tool, mentioned above.

Examples of agricultural frameworks are:
FAPDA Web-based Tool [4] provides a decision making framework for food and agricultural policy decisions.

Another decision making GIS-based tool is ReSAKSS [13], it contains data on agricultural, socio-economic and bio-physical areas. This tool assists policy makers in developing agricultural policies.

Examples of other frameworks are:
Today one can find modeling tools which accept a wide range of parameters and simulate some complex process in order to understand the influence of these parameters on the system in medicine.

The Lives Saved Tool for Maternal and Child Health (LiST) [15], [11] is a modeling framework developed by the Institute for International Programs at Johns Hopkins Bloomberg School of Public Health with intention to estimate the effect of health coverage on maternal and child health. LiST models the status of health coverage under the influence of various factors (e.g. increasing of health care services and usage of nutrition interventions). In this tool users can estimate the impact of different kinds of heath interventions in order to plan the strategies for the improvement of medical methods in maternal, newborn, and child health. The tool contains the data about the effect of some kinds of interventions on peoples' health. Further, the data about maternal and newborn mortality rates, health coverage and interventions of a particular country or region, is collected. Thus, a user can simulate the usage of specific health care methods in a particular region and see the influence of this usage as graphical output.

The Multi-Criteria Analysis Decision framework is a modeling framework for decision making and priority setting, which elaborates on possibilities to create „an equitable, efficient, and sustainable health care system" [15]. All possible health interventions are ranked and compared during a multi-criterion analysis. A specific web-based framework to implement this approach was developed by the EVIDEM Collaboration [3]. The EVIDEM tool is used to provide the participants of the health care process with information and to support decision making during this process. The tool simulates different factors influencing patients' health and produces a graphical output measuring the importance of these factors or the degree of their positive or negative impact.

References

[1] Duke R. D., Origin and Evolution of Policy Simulation: A Personal Journey. Simulation & Gaming. XX(X) I – 17. SAGE Publications (2011)

[2] Durning D. Participatory Policy Analysis in a Social Service Agency: A Case Study Journal of Policy Analysis and Management, JSTOR, 12, 297 (1993)

[3] EVIDEM. https://www.evidem.org/wp/wp-content/uploads/2017/09/EVIDEM-10th-Edition-Tutorial.pdf

[4] FAPDA Web-based Tool. http://www.fao.org/in-action/fapda/web-based-tool/en/

[5] Geurts J. L.A., Joldersma C. Methodology for participatory policy analysis. Department of Policy and Organization Science, Tilburg University, P.O. Box 90153, 5000 LE Tilburg, Netherland. European Journal of Operational Research 128 300-310 (2001)

[6] Habermas, J. Theory and Practice. Beacon Hall, Boston (1974)

[7] Hayek, F. A. The Counter-Revolution of Science. Liberty Classics, Indianapolis (1979)

[8] Henning C., Badiane O., Krampe E. Development Policies and Policy Processes in Africa. Springer International Publishing (2018). DOI: 10.1007/978-3-319-60714-6. ISBN 978-3-319-60713-9. ISBN 978-3-319-60714-6 (eBook).

[9] Joldersma, C. Participatory Policy Making: Balancing between Divergence and Convergence European Journal of Work and Organizational Psychology, Informa UK Limited, 6, 207-218 (1997)

[10] Lasswell H.D. The structure and function of communication in society. http://www.irfanerdogan.com/dergiweb2008/24/12.pdf

[11] Lives Saved Tool. http://www.livessavedtool.org//how-list-works#Interventions

[12] Offe, C. The Divergent Rationalities of Administrative Action. In Disorganized Capitalism, C. Offe. MIT Press, Cambridge (1985)

[13] ReSAKSS. http://eatlas.resakss.org/

[14] Stiglitz J. E. Whither Socialism? MIT Press, (1996) ISBN 0262691825, 9780262691826, 338

[15] The National Academics of Sciences Engineering Medicine. Country-Level Decision Making for Control of Chronic Diseases: Workshop Summary (2012). https://www.nap.edu/read/13337/chapter/5

Models and Their Functions

Hannu JAAKKOLA [a,1] and Bernhard THALHEIM [b,2]
[a] *Tampere University, P.O.Box 300, FI-28101 Pori, Finland*
[b] *Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany*

Abstract. Models are one of the main vehicles in everyday and scientific communication, understanding, learning, explanation, exploration, comprehension, representation, starting points for investigation, pattern detection and exploration, system development, problem solution, hypothetical reasoning, and theory development. Models are mediators, explainers, shortcuts, etc. Models are used as instruments in these scenarios. Their function varies and thus their properties.

This paper investigates the functions of models in dependence on their scenarios. We concentrate the investigation on engineering and construction scenarios which are the main model use in Computer Science and Computer Engineering. The problem solving scenarios, the science scenarios, and the social scenarios are considered as well in a brief form.

Keywords. models as utility, functions of models, instruments, scenarios

1. Introduction

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the solution computation of problems, accurate at least within the scope, and must function within an application scenario.

1.1. Models in Software and Information Systems Engineering

Models in Information Systems (IS) development have three roles:

1. Acting as an abstraction of the real world;
2. Acting as a knowledge base;
3. Acting as a communication tool.

[1] hannu.jaakkola@tuni.fi
[2] thalheim@is.informatik.uni-kiel.de

Proc. 29'th EJC, pages 150-169, LUT, Finland, 2019.

The purpose of Information Systems (IS) is to support tasks of the real world (business) processes; IS modelling creates an *abstraction* of that (Figure 1). The *modelling process* itself creates an evolution chain of models from requirements to design and further to implementation and maintenance. This evolution chain can be seen as an IS related *knowledge base* transferring (*communicating*) IS related understanding (static and dynamic proberties) between the development phases and among the development teams representing a variety of interest groups of the IS.

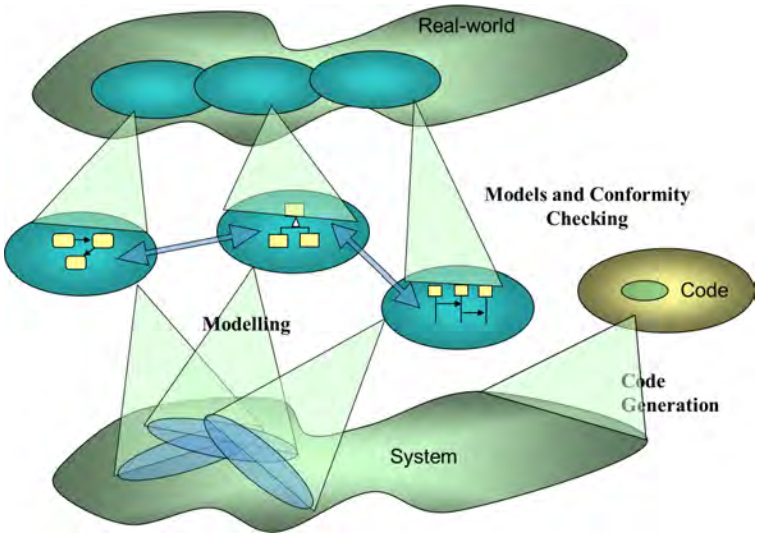

Figure 1. The Role of Abstractions [25](modified by the authors)

In Figure 1 IS models ("system world", lower part) are the *abstractions* of the real-world (business) processes. Abstractions are focused on the essential characteristics; because of that there are detail gaps that provide means for misunderstanding and interpretations. For a single set of characteristics we need several individual models describing the real-world from different points of view (a *viewpoint* represents a *view* to the system under development), in which a single model provides a single view to certain system characteristics. In Figure 1 these viewpoints are presented as overlapping ovals. Overlapping binds the different viewpoints to the whole and provide means for conformity and consistency checking. Such real-world properties that are not included in the IS are represented by the external connections or excluded (based on abstraction, not seen important and essential elements of the whole).

This simplified description of IS modelling rises up several questions:

1. What should be modelled and what can be left out?
2. How many views we have to take to the system?
3. How many individual models (modelling languages) we need?
4. When something is left out (which means that the model is not a complete 1-1 represenatation of the real world phenomenon) how the gaps are filled?

There is no clear right answer to these questions especially if we want to keep the focus in the key issues. However, usually the problems in Information Systems relate more to

the features that are not modelled than to those that are included in the models. Models make things visible.

Views are used in ordering the individual models. There are several approaches in this issue; one of the most referred is the Kruchtens 4+1 view model [26,47]. Kruchten specifies a scenario view as a central point for the other views; it represents the visible external behavior of the system in the form of use cases and other interaction models. The four other views serve different needs: A *logical view* represents end-user functionality and is necessary information for a variety of interest groups, a *development view* is targeted for the software developers and software management, a *physical view* covers aspects important for the system engineers transferring, and the *process view* to the variety of roles responsible for the final software implementation.

Individual models are implemented by *modelling languages*. Every view covers several viewpoints, which means that different *modelling* tools (*languages*) are needed. In practice most commonly used modelling languages are semi-formal ones having formal syntax and semi-formal semantics. Semi-formal modelling languages provide sufficient exactness combined with reasonable easy understability. These languages are located in the middle area of the continuum having easy-to-understand (natural) at the one end and formal exactness combined with difficulty in understanding by non-professionals. There are hundreds of modelling languages for different purposes. The effort of OMG (Object Management Group) has continued the work initiated by Grady Booch, James Rumbaugh and Ivar Jacobson in 1990ies and standardized Unified Modelling Language (UML). UML has gradually become a commonly used set of modelling languages, which have also unified principles of IS development processes (e.g. Rational Unified Process - RUP). UML has 14 diagrams divided in two groups - behaviour and structure diagrams. See the details e.g. in [48].

The problem with the modelling gaps has two sides. On the one side is the exactness and on the other side are the problems caused by the non-modelled details of the reality. The more exact the model is compared to the real world phenomenon the more complex is the model and the more effort is needed to develop it. The non-modelled gaps cause misinterpretations and misunderstandings having final manifestation in system quality - unfortunately so. However, these can be avoided by well organized quality control activities and by keeping the usage related interest groups close to the developers (Agile development). One detail not discussed above is the role of non-functional (quality) properties, assumptions and limitations. Without going to the details, we state that they are changing along the development work to functionality, system architecture, a part of the development process, or stay as they are to be verified and validated in qualitative manner.

1.2. The Notion of Model

Let us first briefly repeat our approach to the notion of model:

A **model** *is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.* [8,39,43]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*.

Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics [38] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

1.3. The Storyline of the Paper

The approach to model functions and its explicit treatment is novel. The maturity of model development generalises SPICE approaches to software development. Section 2 clarifies what are model functions, what is the journey of a model in applications, and what kind of maturity is necessary. We observe that model utilisation can be categorized in four general dimensions. Dimensions are not orthogonal. A model may be used for problem solving and engineering at the same time. This paper orients on the engineering dimension of model deployment. Section 3 elaborates this dimension in detail. We elaborate the role of models and derive which function a model has to play in order to be properly used as an instrument in Section 4.

2. Functions of Models

The function of a model is often taken for granted and seems not to be an issue for investigation. Many modelling problems are caused by the unawareness of functions that models play, of the scenarios in which models are used as instruments, of the specific maturity that is necessary for an instrument, and of the journey of a model within these scenarios. Since models become somehow more universal after their development they are used in additional scenarios and in additional functions. This section now aims at a clarification of functions of models and their necessary level of maturity for dependency and adequacy of a model.

2.1. Models are Instruments

Models are used in some application areas in order to achieve something. They are, thus, aids in accomplishing tasks in those scenarios. They become then useful, usable and used task utensils which have their capacity and potential [2].

Models are, thus, instruments[3] that are adapted to facilitate a definite kind or stage of operation in these scenarios, i.e. the model serves as an instrument or tool in a number of *roles*. In a scenario, the use of an instrument may vary, i.e. the model can be used in some variants of a play.

We observe in science that each science has developed its specific set of approaches. Mathematics, for instance, uses the 'mathematical method'. This method is a specific *mould*, i.e. reveals clearly as having a certain character, forms the flow of processing and thus application of a model, and determines a distinctive nature, character, or type of the model to be used. Engineering and especially information system development have their molds that became common practice. A similar observation can be made for almost all sciences and problem solving tasks.

As tools instruments give practical effects to and ensure of actual fulfillment by concrete goals. They combine the necessary components for this fulfillment what rules the potential structure and the potential behaviour of models in these scenarios. They are not lacking or faulty in any particular according to the purpose.

The role as an instrument in an scenario dictates which *quality characteristics* are essential, i.e. in which case the model is sufficient and what are the evaluation and assessment approaches. Sufficiency implies (1) the soundness and the excellence of every model component, suggests (2) a completeness or perfection characteristics that can be sought or regained by a model, implies (3) perfection deriving from integrity, soundness, or completeness, and implies (4) retention of perfection of a model in its natural or original state.

2.2. What is a Function of a Model

The function of something is determined by a characterisation what it is used for. The function justifies a something's existence. Functioning in a scenario means that models obtain a role which clarifies the actions and activities assigned to or required or expected of a model. The utility and usefulness of a model in some scenario defines the quality of being of practical use. Quality is characterised by essential and distinguishing characterisations of something. The capacity and the potential determines specified functions. These general characterisations provide now a means for consideration of a function of a model.

We distinguish the notion of goal, purpose, and function of a model similar to [9]. These notions are often considered as synonyms. The *goal* of a model is in general the association between a current state and the target state that is accepted by stakeholders or – more general – by members of a community. The *purpose* enhances the goal by means that allow to reach the target state, e.g. methods for model development and utilisation. The *function* extends the purpose by practices or – more systematically – by scenarios in which the model is used. A typical scenario is the modelling method and its specific forms. The purpose is characterised by actions for a model is specially fitted or used or for which a model exists. Actions might be complex and structured. We thus might consider any of a group of related actions contributing to a larger action. The goal is then something set up as an object or end to be attained. The function is performed as expected when applied.

[3] An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool [32].

A model function provides a characterisation (1) as being sufficient; (2) being adequate (either in quality or quantity); (3) as satisfying, fulfilling, meeting the requirements or expectations within a given scenario; (4) as being fit; (4) as conform to, meeting, and fulfilling the wants or needs or condition or restriction; and (5) provide and supply what is desired or needed.

Function, purpose and goal are interrelated. The profile of a model combines the three properties. A function implies a definite end or purpose that the one in question serves or a particular kind of scenario it is intended to perform. A purpose suggests a settled determination. A goal suggests something attained only by prolonged effort and hardship. Other relation notions are intention, intent, design, aim, end, and objective. The objective or goal means what one intends to accomplish or attain. The intention implies little more than what one has in mind to do or bring about. The intent suggests clearer formulation or greater deliberateness. The design implies a more carefully calculated model. The aim adds to these implications of effort directed toward attaining or accomplishing The end stresses the intended effect of model utilisation often in distinction or contrast to the action or means as such. The object may equal end but more often applies to a more individually determined wish or need within a community of practice. The objective implies something tangible and immediately attainable. In the sequel of the paper we will consider mainly the function of a model. A similar investigation can be performed for the other notions.

Models are used as instruments in certain utilisation scenarios such as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. They have to fulfil a number of specific functions in these scenarios. Typical functions of models as instruments in scenarios are

(a) cognition,
(b) explanation and demonstration,
(c) indication,
(d) variation and optimisation,
(e) projection and construction,
(f) control,
(g) substitution, and
(h) experimentation [45].

2.3. Functions of Models in Scenarios

In general, a *scenario* an outline or synopsis of an application story where a sequence of steps is performed by some members of the community of practice. Models may function as instruments in these steps. We notice that each of the functions below require a specific form of model. For instance, typical functions in information system development and maintenance scenarios (e.g. those in [1,30,37]) are typically:

Communication and negotiation scenario: The model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested

parties with different interests find a way to reconcile or compromise to come up with an agreement.

The model has several functions in this scenario: (personal/public/group) *recorder of settled or arranged issues*, *transmitter of information*, *dialogue service*, and *pre-binding*.

Conceptualisation scenario: Models may be used for conceptualisation of information system terms. Conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

The function of a model within these scenario is *semantification* or *meaning association* by means of concepts or conceptions. The model becomes enhanced what allows to regard the meaning in the concept.

Description scenario: In a description scenario, the model provides a specification how the part of the reality that is of interest is perceived and in which way augmentations of current reality are targeted. The model says what the structure of an envisioned information system is and what it will be.

The function of models in these scenarios the *representation of essential properties and qualities* in an accurate or precise form, i.e. delineation.

System construction scenario: Models in system construction scenarios are model suites, i.e. requirement models, informative models, description models, prescription model, and code models.

The functions in this scenario are inherited from those scenarios for models in the model suite.

Prescription scenario: The model functions as a blueprint for or prescription of a information system application, especially for prescribing the structures and constraints in such applications.

Typical function of such blueprint models in these scenarios are: *instruction, direction*, and *guideline*. Often diagrams such as UML diagrams are used in an *inspiration* function. This might, however, too limited.

Documentation scenario: Models are used for providing various concepts that have been used for structuring and functionality development of a system. They specify what will be is the system, how the system can be used, by what means, in what way, what are supporting means, and wherewith facilities of the supporting software systems. They typically describe the structure, purpose, operation, restrictions, and other requirements in a documentation scenario.

Functions of models are similar to functions of manuals, i.e. *support for use, documentary validation*, and *presentation of documentary evidence*.

Explanation and discovery scenario for applications: In early stages of database development, the developer seeks an explanation and understanding of how, when, and when which entities are of interest and should be taken under consideration.

Models serve then for presentation in a form that *makes the origin intelligible* (comprehensibility of relevant ideas or objectives and understanding in an application area) or *support for hypothetical reasoning*. The first function is typical for domain-situation models [41]. The second one for mental models, especially perception models.

Explanation and discovery scenario for systems: In later stages and reorganisation of a system application or 'brownfield' modernisation, the modeler rediscovers which constructs have been the basis for which part of the database, which associations occur among these constructs, which general forms are behind them, and the boundaries within which associations.

Within these scenarios model serve for *information extraction*, *providing awareness*, or *making the origin intelligible*.

Knowledge discovery and experience propagation scenario: Models tacitly integrate knowledge and culture of design, of well-forming and well-underpinning of such models and of experience gained so far, e.g. meta-artifacts, pattern and reference models. This experience and knowledge is continuously enhanced during development and after evaluation of constructs.

Models are functioning for *elaboration, exploration, detection*, and *acquisition* of tacit knowledge behind the origins which might be products, theories, or engineering activities. They allow to understand what is behind drawn curtain.

These scenarios are typically bundled into *use spectra*. Information system development is mainly based on description, conceptualisation, and construction scenarios. The reengineering and system maintenance use spectrum is based on combination of documentation scenarios with an explanation and discovery scenario from one side and communication and negotiation scenario from the other side. Models are also used for documentation scenarios, explanation and discovery scenarios for applications or systems, and for knowledge experience scenario. We concentrate here on the four scenarios.

Furthermore, we cannot handle all functions for these four scenarios. The treatment and the properties of these functions can be exemplarily explained for one of them. Since the theory and techniques for *informative models* have already been developed in detail in [44], we can develop in detail the function that an informative model has to serve in Section 4. The function can be characterised by verbal expressions. Informative models are characteristic for the first phase of system development and for the documentation phase. Informative models are typically used as leaflet or instructions for use.

In general, models can be considered to be the third facet of science[4] beside situation and theories. They are an essential element in problem solving and engineering. They are widely used in daily life without calling them models. We observe that model functions vary a lot. Models can be characterised by the 'logos'[5]. The logos provides a separation of scenarios into perception/utilisation (see 'word'), concordance/acceptance (see 'judge'), intellectual absorbtion and comprehension (see 'mind'), understanding and sense-making (see 'power'), application (see 'deed'), and reseoning-backed application (see 'reason') [40]. We are going to restructure the separation of concern in [40] by developing four main dimensions of model utilisation. Model development can be investigated in a similar form but is left out in this paper.

The four main scenario dimensions of model use in Figure 2 are:

[4]The title of the book [3] has inspired this observation.

[5]We refer to J.W. Goethe poem "Faust" poem where Faust reasons in the study room scene on the meaning of the word 'logos' $\lambda\acute{o}\gamma o\varsigma$. This word has at least 6 meanings where Faust used only four of them: word, *concept, judgement,* mind, power, deed, and *reason*. The first and sophisticated introduction of the logos can be traced by to pre-socratic philosophy and especially the work by Heraclitus [28].

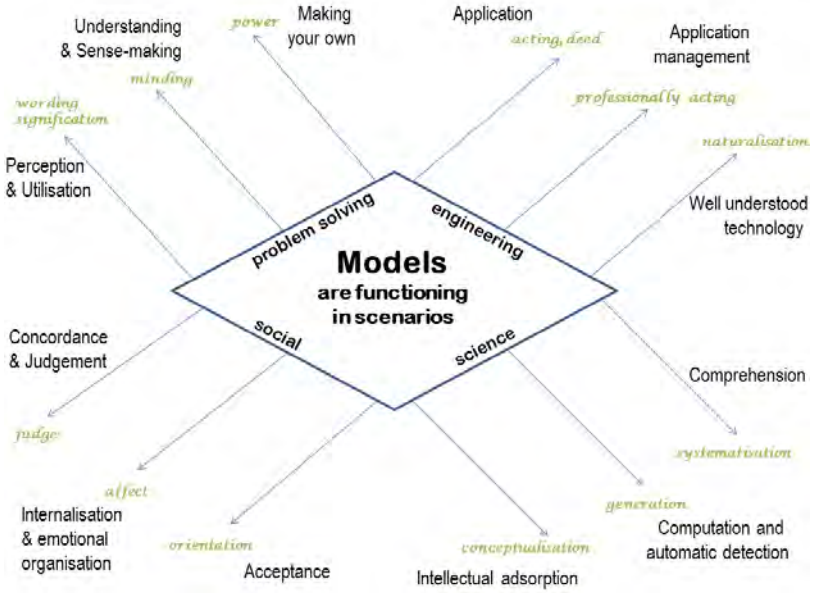

Figure 2. The Kiviat graph for Functions of Models in Scenarios

Problem solving scenarios: Problem solving is a well investigated and well organised scenario (see, for instance, [2,11,13,31,46]). It is based on (1) a problem space that allows to specify some problem in an application in an *invariant* form and (2) a solution space that *faithfully* allows to back-propgate the solution to the application. We first describe a problem, then specify the requirements for its solutions, focus on a context, describe the community of practices and more specifically the skills needed for the collaborative solution of the problem, and scope on those origins that must be considered. Next we develop a model and use this model as an instrument in the problem solving process. This instrument provides a utility for the solution of the problem. The solution developed within the model setting is then used for derivation of a solution for the given problem in the origin setting.
We may distinguish three specific scenarios:

Perception & utilisation: A problem becomes understood and can be investigated in a proper form.

Understanding & sense-making: A problem is elaborated in such a way that its specifics can be critically investigated and a solution can be derived.

Making your own: The problem is profoundly understood, properly formalised, and a number of faithful solutions can be derived.

Engineering scenarios: Models are widely used in engineering. They are also one of the main instruments in software and information systems development, especially for system construction scenario. Engineering is often technology-driven problem solving by practitioners where a problem got a different shape and a solution is some material product within a given infrastructure. Engineering also considers additionally robustness, failure management, safety, human factors, regulations,

practicality, and cost [6]. So, the scenarios add to some perspectives that are used in science scenarios techniques, methods or processes used in production of any kind of goods, e.g. services (see, for instance, [6]). Engineering is a goal-governed process of designing and making tools beyond what is already available. Typical models in engineering are trial-and-error models, inspiration or enlightenment models, and product (consideration or documentation) models.

We may distinguish three specific scenarios depending on the level of sophistication:

Application: Handicraft, apprenticeship, and engineering is an art of creating and manufacturing a technics-based solution. It does not have to be scientifically grounded. They main success criterion is that the solution suffices.

Application management: The art of solution development might be properly organised and this organisation allows to repeatedly produce a product for similar requirements and tasks.

Well-understood technology: Technology of solution development supports professional outcome-oriented (e.g. indicator-based) engineering and manufacturing including mastering the whole process of development of tools, processes, machines and equipment with an integrated view on facilities and systems.

Science scenarios: Sciences have developed a number the distinctive form in which a scenario is organised. Sciences make wide use of mathematical modelling. The methodology of often based on specific moulds that are commonly accepted in the disciplinary community of practice, e.g. [2]. Model development is based on four phases: description, formulation, ramification, and validation. In the description phase, individual perception and situation models involved into the modelling situation, are isolated and the corresponding primary properties are identified and represented, e.g. [12]. In the formulation phase, properties are interrelated, integrated and combined into a preliminary, initial model. This model is analysed in a ramification phase in order to check whether the model is a proper solution and to interpret and to consider its implications. Finally, the model and its capability and capacity are assessed in a validation phase.

We may distinguish three specific scenarios:

Comprehension: Most sciences and also empirical sciences use a systematisation scenario (see, for instance, [22,23]) where individual tasks are considered first, later combined, then considered together, next potentially based on concepts and theories, and finally based on theories.

Computation and automatic detection: Hypothetical reasoning and data-driven discovery scenarios are essentially search scenarios (see, for instance, [17]). Data science, knowledge discovery, deep learning, and business intelligence applications typically use this scenario.

Intellectual adsorption: Advanced mathematics and natural sciences use deep theories that have been developed and conceptualised over several centuries. Their scenarios are based on intertwined theoretical concept systems, on conceptualisation, and on meta-reasoning.

[6] Engineering is the art of building with completely different success criteria (see [33]: "Scientists look at things that are and ask 'why'; engineers dream of things that never were and ask 'why not'." (Theodore von Karman))

Social scenarios: Social scenarios are less investigated although cognitive linguistics (e.g. [19,21]), visualisation approaches (e.g. [34]), and communication research (e.g. [10]) have contributed a lot, e.g. by the notion of mental models and perception models. Social models might be used for the development of an understanding of the environment, for agreement on behavioural and cultural pattern, for consensus development, and for social education.
We may distinguish three specific scenarios:

Acceptance: Models support orientation. People might believe, trust or credit what is given with the model. Schools of thought widely use models for agreeing on approaches, on common understanding and thought, and on some kind of exclusivity.

Internalisation & emotional organisation: Models may also affect behaviour, understanding, communication, and social life. They are appraised in a community and might be the basis for a system of values.

Concordance & judgement: Models are often used in discourses. They allow to collaborate to certain extend, to coact, and to integrate within a society.

In Section 3 we will now consider these specific models and their functions in engineering scenarios. In most cases, a model is in reality a model suite that consists of associated models.

2.4. Maturity of Models

A Maturity Model (MM) represents a path towards increasingly organized and systematic way of doing something. The maturity is defined by capabilities essential to fill the goals of the target system. The Capability Model is a modular description of the capabilities something has and needed to execute its tasks, described in the terms of the desired outcomes. Each component of the capability model has its maturity defined by its attributes. The maturity of the whole is the maturity profile of its components.

The most commonly used and the best known maturity models are CMMI and ISO 33004:2015 [4,15]. CMMI has its roots in late 1980ies, when Watts Humphrey joined to Software Engineering Institute (SEI) of Carnegie Mellon University and published his first version CMM [14] for improving the process quality of software developing organisations. In his maturity model Humhrey applied Quality Management Grid developed by Philip Crosby [5]. Since that CMM was further developed and applied in a big variety of versions for different branches of business. The current version - CMMI v2.0 (Capability Maturity Model Integration) is published by CMMI Institute is from 2018 [4]. ISO 33000 series of standards [15] has its roots in 1990ies, when International Standardization Organization started project called SPICE (Software Process Improvement and Capability dEtermination) as an international collaborative effort to develop a Standard for Software Process Assessment under the International Committee on Software Engineering standard, ISO/IEC JTC1/SC7/WG10. Official standards developed in this project were published as a series of standards ISO/IEC 15500 having the standard 15504 as a definition of their maturity model. The series number was changed to 33000 having the standard 33004:2015 [ISO/IEC 2015] including the maturity model definition.ISO standard included in two models - organizational level maturity model (continuous) and process based staged model; in the current version these are coincided.

The CMMI maturity model is based on five levels, which are (1) Initial (unpredictable, poorly controlled and reactive), (2) Managed (characterized for projects and is often reactive), (3) Defined (characterized for the whole organization and is proacticve), (4) quantitatively managed (measured and controlled) and (5) Optimizing (focus on improvement).
The ISO model has six levels, which are 0. Incomplete, 1. Performed (process performance), 2. Managed (performance and work product management), 3. Established (definition and deployment), 4. Predictable (measurement and control) and 5. Optimized (innovation and optimization). The capability of processes is measured using nine process attributes specific according to the levels.

As seen, both models, CMMI and ISO/IEC, follow the same basic principles. We have adapted the six level model that is applies these principles to characteristics and capabilities in our maturity engineering model.

3. Models in Engineering and Computer Engineering

3.1. The Maturity Level of Model Utilisation in Engineering

The three main scenarios in Figure 3 we investigate are application, application management, and industrial development based on well-understood technology. Software engineering and information system development is currently mainly based on the first two scenarios. Models function then according to system construction. Recent development on component-oriented development, pattern, and product lines can be considered as early stages of the third scenario. Web information systems [35] used the third scenario in the most advanced way.

Application: Engineering is using different approaches to development than science and daily social life. The compilation of engineering knowledge (see, for instance, the six volumes [24]). The acting and deed scenarios are typical for all handicraft and apprenticeship processes. An artisan has developed a number of habits for production. Software engineering and information systems engineering are often limited to this handicraft approach. The level of maturity may be high or ad-hoc. The process of developing a product is somehow organised but not systematic. The quality cannot be properly guaranteed however since the management of the whole process is a prerequisite.
We may distinguish the 6 levels of maturity similar to the taxonomy levels for physical or handicraft work in [22,23]:

1. *Be inspired, guided, imitate*: There are already similar solutions that have been mastered in the past, e.g. code that has been developed for similar application problems. These solutions can be modified and used for the current problem.
2. *Deliberately apply and manipulate*: The current solution collection has led to some understanding of the issues that have been developed, e.g. the code for search algorithms the first volume of [24]. New solutions can be found knowing the properties of these collections.
3. *Deliberately and precisely practise and manage*: The problem area may be properly organised and the algorithms can be organised within flow of work, e.g. the systematic organisation of the data mining process [20].

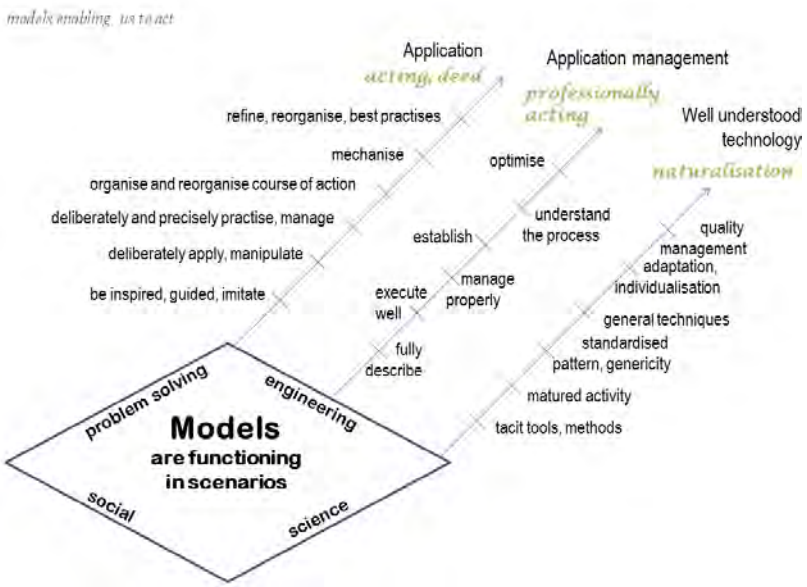

Figure 3. The Kiviat graph for Engineering Scenarios and their Maturity

4. *Organise and reorganise course of action*: Engineering can be reorganised whenever it is necessary, e.g. agile SCRUM-based programming inherits the goodies of classical software engineering.
5. *Mechanise*: The engineering framework has found some systematic treatment that allows to mechanically derive the solution to problems, e.g. algorithmics with the 10 classes of algorithms
6. *Refine, reorganise, best practices*: The problem area is so well understood that solutions in this area can be developed based on best practices and inheriting good solutions, e.g. on the basis of reference models valid for the entire application, refinement, assessment, and getting the most of it.

These scenarios can be based on normal models [18,42]. The underpinning deep model which incorporates the disciplinary modelling matrix (paradigms, postulates, assumptions, ...) and the practised flow of action (mould) is inherited and taken for granted.

Application management: ISO/IEC 33004 and CMMI [4,15] have brought an understanding of the level of maturity for development processes. Engineering as well as software and information system development have developed approaches within such scenarios but did not yet reach the highest levels of maturity. We may distinguish the 6 levels of maturity:

1. *Fully describe*: Each step of the scenario and the corresponding utilisation of a model is well specified. The definitions are given and settled.
2. *Execute well*: The scenario can be executed and documented according to the description and the model can be used as envisioned.

3. *Manage properly*: The scenario can be managed in a form that allows to assess its level of completion, its lacking points, and its deficiencies.
4. *Establish*: The scenario has already practised in a number of applications and the experience gained can be used for new projects by adaptation to minor differences.
5. *Understand the entire process*: The scenario is well understood. A corresponding theory for reasoning about the scenario has been developed and ready for application.
6. *Optimise the process*: The scenario and the corresponding models can be organised according to a number of indicators and properties.

The deep model is partially known at least for the basis part of the background [18,45]. This part and the normal modelling approach can be revised, reorganised and optimised to certain extend.

Well-understood technology: Software and information systems engineering is now developing approaches to become a technology of analysis, design, and development. Civil engineering and production management are the blueprint for such *naturalisation*. Industrial production uses tools, provides facilities for individualisation, and applies measures for integrated quality management. The scenarios and the models become 'naturalised', i.e. industrialised, manufacturable, adaptable to other application cases, more natural, and lifelike.

1. *Existence of tacit tools and ready-to-apply methods*: The scenarios and models are supported by tools and methods which can be used to generate them.
2. *Matured activity in a engineering scenario*: The technology has becoming a standard and is used for manufacturing.
3. *Standardised application that use pattern and are based on genericity concepts*: A number of general and easy to adapt approaches have been developed and allow instantiation, context enhancement and refinement on demand.
4. *Ready for deployment general techniques*: The manufacturing itself has led to machine tool design that can be brought in as off-the-shelf and can be adapted to the current circumstances.
5. *Processes that provide facilities for adaptation and individualisation*: The scenarios and the models can be individualised according to the very specific circumstances.
6. *Processes that integrate quality management*: Quality characteristics are incorporated into the entire process and can be properly maintained.

The technological understanding leads to properly manageable, adaptable and optimised processes that can easily adapted to new circumstance, e.g. to changes in the indicator system. The deep and the normal models are well understood and their effect can be predicted. These scenario are based on meta-models, meta-scenario, and meta reasoning.

3.2. The Engineering Dimension and the Model Notion

It seems that software and information system engineering is far from the maturity levels. We claim, however, that these maturity levels can be successfully accomplished if models are developed according to the model notion. Maturity can be characterised by

capability attributes [4,15]. We base our approach on general quality management [16]. We define capability attributes directly by properties of adequacy and dependability Capability attributes for models are based on questions we can ask for a model or a model suite. The most critical are the following ones:

- What is the function of the model *in which scenario*? What are *consequential purposes* and goals? What are *anti-goals* and anti-purposes?
- *Which origins* are going to be deputed/represented? Which are not considered?
- Does the model contain *all typical, relevant and important features* of the origins under consideration and only those?
- Rhetoric frame: *who* says *what, when, where, why, in what way, by what means* that can be extended to the W*H frame [7].
- Is the instrument adequate and dependable *within the utilisation scenario*? What are the parameters for adequacy and dependability?
- How *purpose-invariance* and *solution-faithfulness* is going to be defined?
- What *kind of reasoning* is supported? What not? What are the *limitations*? Which pitfalls should be avoided?
- Do you want to have a universal model that contains all and anything? Would it be better to use a *model suite* where each of the models depute/represent some specific aspects? What about the *non-essential aspects*?

4. Functions of Models Defined by Verbal Expressions

Now we face the description problem for functions of models. Models can be classified according to their instrument properties in scenarios. Let us consider only three specific

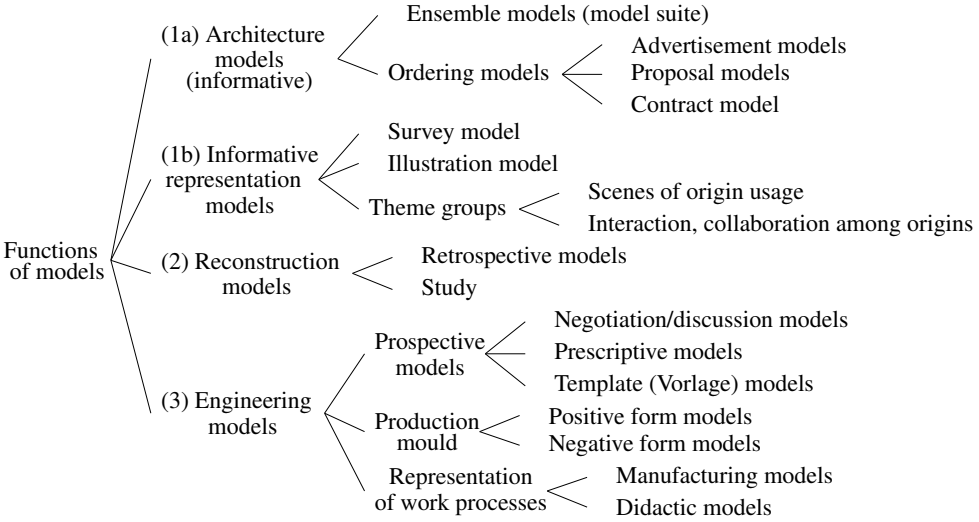

Figure 4. Separation of Functions of Models

functions: inform, reconstruct, and engineer. Moreover, these kind have also sub-kinds.

We arrive at a separation of models in Figure 4. This kind of separation can be combined with the categories presented in [44,45].

These kinds lead to three different kinds of models with different capabilities, i.e. three different styles (or stereotypes) of adequacy and dependability. The properties of adequacy (analoguous, focussed, purposeful) and dependability (justified (corroborated, coherent/conform, validated, stable/plastic) and sufficient (quality characteristics, evaluation)) are specific for each kind. The properties can be considered as parametric characterisation of the model capability.

4.1. Functions of Models Explicitly Given by Verb Fields

We still need, however, a means to characterise the functions of models. We propose to use word fields (or semantic fields of words [27,36]) for characterisation. A word field has typically several words which are related to each other by similar meanings or through a more abstract relation. The bundle of words bears the meaning. The words in a word field all relate to the same subject or concept. A word in a word field can be considered as more general one than another in the same word field.We may thus draw tree like the one in Figure 5. For instance, the word 'believe' is used in two semantic fields: (1) (verb form) accept as true; taken to be true with a number of synonyms (trust, buy, infer, bank, swear, believe in, rely, swallow, accept, understand); (2) (noun form) as make-believe, the enactment of a pretense with other synonyms (feigning, pretense, pretending, simulation, pretence, pretend). It seems that the hypernym (describing a word more broad) and hyponym (more specialised and specific) association (for verbs additionally the troponym) provide a specialisation and generalisation structure for word characterisations of models. Since models are instruments we may reduce consideration

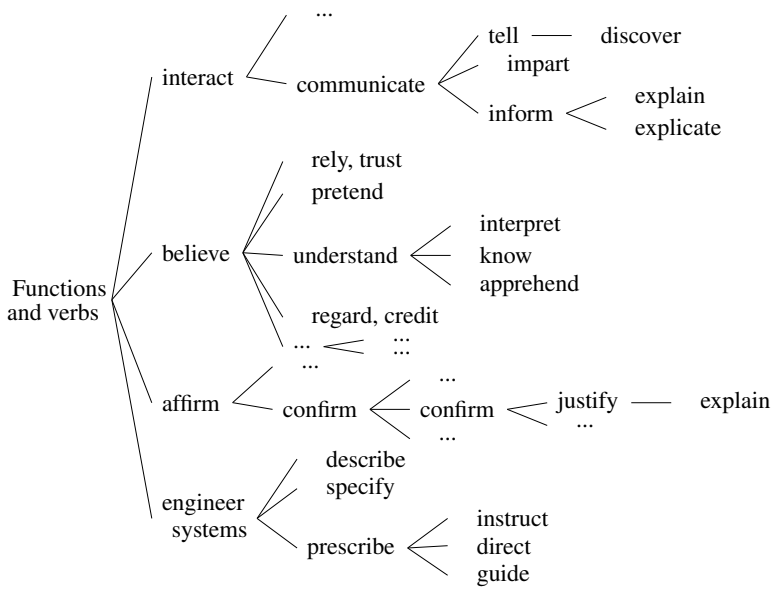

Figure 5. Characterisation of Models by Verbs

to verbs and verbal expressions. For instance, engineering systems may be based on three activities that are related to the three words describe, specify, and prescribe. “Prescribe” itself is related to “instruct”, “direct”, and “guide”. “Instruct” with the meaning to “give instructions or directions for some task” can be linked to either (1) “give instructions or directions for some task” or (2) (less relevant for system engineering) “impart skills or knowledge to”. The third meaning of “make aware of” can be neglected.

If we restrict the word fields to some specific ones then a functions becomes properly shaped and the capabilities which are necessary can be drawn. The corresponding representation can be given as a category network. Each word field thus describes thus the *mission* and the *determination* we have in the mindset as habitual or characteristic mental attitude that determines how you will interpret and respond to situations. The mission separates the task into important ones and less important ones where the first enable to perform properly this task in the given scenario as an assignment. The determination decides and controls the model’s outcome or nature. The determination is an explanatory statement extended by *how* the model is used, what is the *main background* behind this instrument, and *why* to use this model. It provides basic ideas, features, particularities, and the utilisation pattern for the model.

4.2. Model Function Extended to the Cargo

We may now derive a general specification form for the function of a model:

> *Word field* describing the mission and the determination of a model;
> *When, how in general, what for, by whom, for whom* the model is used;
> *Capability and incapability attributes* as adjective characterisation of a model;
> *In what way, with what requirements* the model has been developed.

The specification can be narrative, sign-based, or implicit. We advocate the first form and an explicit statement about the function of a model. This explicit statement allows to reason about when this artefact is *not a model* but simply an object, which *anti-profile* has a model, and which *specific considerations* must be taken into account. The *journey of a model* is given by an association to one or several scenarios. For instance, a model suite in a waterfall approach consists of an inspiration model that is used before requirements are agreed, of a declaration model describing the objectives, of a prescription model for coding, of a documentation model for the code developed, and of an educational model used for elaboration of experience gained during development.

For instance, an entity-relationship diagram may be combined with a number of view schemata and realisation templates. In this case, it functions as a *prescription model* in an information system greenfield development scenario if it is *definitely guiding the codification as a design of structuring and derivable viewpoints*. View schemata are used for representation of user viewpoints and viewpoints for system’s operating how the data can be used. Greenfield scenarios start coding from scratch. Brownfield scenario use another kind of model.

The word field “guide” has two aspects: to act as a guide to and to direct in a codification; to direct, to supervise, to influence, to superintend the codification. Partial synonyms in this word field are “lead”, “steer”, “pilot”, and “engineer”. The background comes with the kind of supporting technology and the deep model that is governed by IS technology. Adequacy requires in this case mapping properties to the origins and to

the code. The focus is given by the origins that are exemplarily considered. The purpose is related to construction. Corroboration, validation, and plasticity are determined by the origins. Conformity is determined by the standards in the information systems area. Quality in use is based on the visual representation requirements and the transformation to code. External and internal quality are those that are commonsense in the area. The tight binding provides also the evaluation of the model. The community of practice is the modeller, the technologist, and the programmer. The binding to origins determines the requirements to the model.

The *cargo* [29] of a model consists of the model functions and additionally of the *abstract declaration* of the meaning, and of the narrative explanation of the *identity*. The abstract declaration of the meaning mainly contains main semantic and pragmatic statements about the model. The description of value of the model is determined by the functions in the utilisation scenarios and its importance within the given settings. The identity of a model is given by the actual and communicated identity. The three other kinds of an identity (accepted, ideal, and desired identity) are often neglected. The cargo can be considered as an abstract that describes the model. The cargo describes why a model should be accepted by some community of practice and in which scenarios the model has which functions, which context and background is (implicitly).

Any instrument that is used as a model has a cargo that determines its utility value and its present utilisation value. The function is the central ingredient of this cargo.

5. Conclusion

In our paper we have introduced a framework that clarifies the essence of modelling. In practice, the development of models in a variety of application scopes is not systematically guided and managed. We have started with a case study focusing in information systems (IS) modelling the area of modelling that is best known by the authors. We pointed out aspects that make IS modelling difficult and cause problems in model quality. Based on these findings we have developed stepwise our general framework explaining the essence of different models and different modelling practices. We have wanted to show the heterogeneity of models, but also the fact that model types have same basic properties, functions of models.

In our framework models are seen as as instruments that facilitate functions. The model has a goal (having a target state) that is expected to reach by modelling, purpose that enhances the goal by means that allow to reach the target, and function that extends the purpose by executable practices represented as scenarios. The paper handles the idea of functions of models in a wide manner covering different (classified) types of models.

The first step in our framework introduces the scenario approach - we see functions as scenarios. A wide variety of scenario types are discussed, but for detailed handling we have selected four scenarios - engineering, science, social and problem solving. This four scenario approach divides each main scenario into three sub-scenarios describing the typical characteristics of them. This approach provides us means to analyse the maturity of models functions in the scale of six. This analysis approach is derived from the principles of CMMI and ISO 33004 maturity models. The detailed handling is focused in engineering scenarios. We have developed similar analysis for the three other scenarios - social, problem solving, science but because of the limited space these were left out of the paper.

The second step of our framework we face the description problem for functions of models. Models are classified in this step according to their instrument properties in scenarios. For that purpose we have constructed three specific functions - inform, reconstruct, and engineer. Two alternatives to specify the functions of models are introduced - functions of models explicitly given by verb fields and model function as the main ingredient of a cargo.

As a summary - we have introduced a framework that can be used in analysis of modelling and developing modelling methods. It provides means for analyzing the maturity and quality of models. The framework is useful both for modelling practitioners and for those who are focused in developing modelling methods.

References

[1] C. Batini, S. Ceri, and S. Navathe. *Conceptual database design (an entity-relationship approach)*. Benjamin/Cummings, Redwood City, 1992.

[2] R. Berghammer and B. Thalheim. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Methodenbasierte mathematische Modellierung mit Relationenalgebren, pages 67–106. De Gryuter, Boston, 2015.

[3] S. Chadarevian and N. Hopwood, editors. *Models - The third dimension of science*. Stanford University Press, Stanford, California, 2004.

[4] CMMI. Capability maturity model integration, version 2.0 CMMI. CMMI Institute, 2018.

[5] P.B. Crosby. *Quality is Free*. New American Library, New York, 1979.

[6] A. Dahanayake and B. Thalheim. A conceptual model for IT service systems. *Journal of Universal Computer Science*, 18(17):2452–2473, 2012.

[7] A. Dahanayake and B. Thalheim. W*H: The conceptual model for services. In *Correct Software in Web Applications and Web Services*, Texts & Monographs in Symbolic Computation, pages 145–176, Wien, 2015. Springer.

[8] D. Embley and B. Thalheim, editors. *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*. Springer, 2011.

[9] F. Engels. *Dialectics of Nature*. Wellred, 2012.

[10] H. Fuks et. al. The 3C collaboration model. In *Encyclopedia of E-Collaboration*, pages 637–644. IGI Global, 2008.

[11] G. Greefrath, G. Kaiser, W. Blum, and R. Borromeo Ferri. *Mathematisches Modellieren für Schule und Hochschule*, chapter Mathematisches Modellieren - Eine Einführung in theoretische und didaktische Hintergründe, pages 11–37. Springer, 2013.

[12] I.A. Halloun. *Modeling Theory in Science Education*. Springer, Berlin, 2006.

[13] H. Hertz. *Die Prinzipien der Mechanik in neuem Zusammenhange dargestellt*. Akad. Verl.-Ges. Geest und Portig, Leipzig, 2. aufl., nachdr. der ausg. edition, 1984.

[14] W.S. Humphrey. Characterizing the software process: A maturity framework. *IEEE Software*, 5(2):73–79, 1988.

[15] ISO/IEC. Process assessment – requirements for process reference, process assessment and maturity. ISO/IEC 33004:2015, 2015.

[16] H. Jaakkola and B. Thalheim. A framework for high quality software design and development: A systematic approach. *IET Software*, pages 105–118, April 2010.

[17] H. Jaakkola and B. Thalheim. Supporting culture-aware information search. In *Information Modelling and Knowledge Bases XXVIII*, Frontiers in Artificial Intelligence and Applications, 280, pages 161–181. IOS Press, 2017.

[18] H. Jaakkola and B. Thalheim. Modelling cultures. In *Information Modelling and Knowledge Bases XXX*, pages 61–80. IOS Press, 2019.

[19] R. Jackendoff. *Foundations of languages: Brain, meaning, grammar, evolution*. Oxford University Press, 2004.

[20] K. Jannaschk. *Infrastruktur für ein Data Mining Design Framework*. PhD thesis, Christian-Albrechts University, Kiel, 2017.

[21] P.N. Johnson-Laird. *Mental models: Towards a cognitive science of language, inference, and consciousness.* Cambridge University Press, London, 1983.

[22] H.-C. Jongebloed and B. Kralemann. Die Bestimmung der Schwierigkeitsgrades von Aufgaben unabhängig von zielgruppenspezifischen Verteilungen der Lösungswahrscheinlichkeiten. MBW Schleswig-Holstein, Abschlußbericht, 2016.

[23] H.-C. Jongebloed and M. Twardy. Lernzielformulierung und -präzisierung. In *Kompendium Fachdidaktik Wirtschaftswissenschaften*, pages 591–729, Düsseldorf, 1993.

[24] D.E. Knuth. *The art of programming I-VI.* Addison-Wesley, Reading, 1968-2015.

[25] K. Koskimies. *Oliokirja*. Talentum, Helsinki, 2000.

[26] P. Kruchten. Architectural blueprints - the "4+1" view model of software architecture. *IEEE Software*, 12(6):42–50, September 1995.

[27] J. Kunze. Generating verb fields. In *Proc. KONVENS*, Informatik Aktuell, pages 268–277. Springer, 1992. in German.

[28] A.V. Lebedev. *The Logos Heraclitus - A reconstruction of thoughts and words; full commented texts of fragments (in Russian).* Nauka, 2014.

[29] B. Mahr. Cargo. Zum Verhältnis vo n Bild und Modell. In I. Reichle, S. Siegel, and A. Spelten, editors, *Die Wirklichkeit visueller Modelle*, pages 17–40. Wilhelm Fink Verlag, München, 2008.

[30] H. Mannila and K.-J. Räihä. *The design of relational databases*. Addison-Wesley, Wokingham, England, 1992.

[31] G. Polya. *How to solve it: A new aspect of mathematical method.* Princeton University Press, Princeton, 1945.

[32] J.E. Safra, I. Yeshua, and et. al. *Encyclopædia Britannica*. Merriam-Webster, 2003.

[33] A. Samuel and J. Weir. *Introduction to Engineering: Modelling, Synthesis and Problem Solving Strategies*. Elsevier, Amsterdam, 2000.

[34] S. Sasaki, Y. Takahashi, and Y. Kiyoki. The 4D world map system with semantic and spatio-temporal analyzers. In *Information Modelling and Knowledge Bases XXI*, volume 206 of *Frontiers in Artificial Intelligence and Applications*, pages 1 – 18. IOS Press, 2010.

[35] K.-D. Schewe and B. Thalheim. *Design and development of web information systems*. Springer, Chur, 2019.

[36] H. Schumacher, editor. *Verben in Feldern*. DeGruyter, 1986.

[37] B. Thalheim. *Entity-relationship modeling – Foundations of database technology*. Springer, Berlin, 2000.

[38] B. Thalheim. Towards a theory of conceptual modelling. *Journal of Universal Computer Science*, 16(20):3102–3137, 2010. `http://www.jucs.org/jucs_16_20/towards_a_theory_of`.

[39] B. Thalheim. The conceptual model $\equiv$ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[40] B. Thalheim. Model-based engineering for database system development. In *Conceptual Modeling Perspectives*, pages 137–153, Cham, 2017. Springer.

[41] B. Thalheim. Conceptual model notions - a matter of controversy; conceptual modelling and its lacunas. *EMISA International Journal on Conceptual Modeling*, February:9–27, 2018.

[42] B. Thalheim. Normal models and their modelling matrix. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 44–72. College Publications, 2018.

[43] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. Springer US, 2019.

[44] B. Thalheim and A. Dahanayake. Comprehending a service by informative models. *T. Large-Scale Data- and Knowledge-Centered Systems*, 30:87–108, 2016.

[45] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

[46] C. von Dresky, I. Gasser, C. P. Ortlieb, and S Günzel. *Mathematische Modellierung: Eine Einführung in zwölf Fallstudien*. Vieweg, 2009.

[47] Wikipedia. 4+1 view architectural model. https://en.wikipedia.org/wiki/4%2B1_architectural_view_model, 2019. Assessed February 5, 2019.

[48] Wikipedia. Unified modelling language. https://en.wikipedia.org/wiki/Unified_Modeling_Language, 2019. Assessed February 5, 2019.

Models as Programs: The Envisioned and Principal Key to True Fifth Generation Programming

Bernhard THALHEIM [a,1] and Hannu JAAKKOLA [b,2]
[a] *Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany*
[b] *Tampere University, P.O.Box 300, FI-28101 Pori, Finland*

Abstract. Programming became more and more comfortable with development of third and fourth generation programming languages. Although the fifth generation project did not achieve its goals, the necessity for more comfortability is still challenging. This paper delineates the path towards *true fifth generation programming*. based on *literate modelling* with model suites that generalises model-driven development and conceptual-model programming. A *model suite* consists of a coherent collection of explicitly associated models. A model in the model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used *as a program of next generation* and will be mapped to programs in host languages of fourth or third generation. So, we claim that *models will become programs of true fifth generation programming*.

Keywords. model-based programming, models are programs

1. Introduction

Programming has become a common cultural technique, esp. for *non-computer specialists*, *engineers*, and *laymen* where the latter already start with simple tools like MIT Scratch programming or LEGO. Programs and computers have become an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Despite the detailed research knowledge gained so far, the *quality of programs* decreases, for instance, due to the wide variety of program applications, due to the large variety of program libraries and their constant evolution, due to the numerous languages and toolboxes, due to integration and impedance problems among already existing programs, due to application of different programming cultures, and due to missing provenance and documentation support. Programming is not the most natural kind of communication for many programmers. They reason in a different language and at different abstraction levels. They often have difficulties in understanding their own programs later on or programs developed by others. At the same time,

[1] thalheim@is.informatik.uni-kiel.de
[2] hannu.jaakkola@tuni.fi

Proc. 29'th EJC, pages 170-189, LUT, Finland, 2019.

systems become more complex and thus less comprehensible. We are thus approaching a *software crisis 2.0* [Ama16,Far16,VM11,WVH17]. Programs are still developed on the basis of the third and fourth generation although the underlying mental concepts are not expressible in these languages. Moreover, critical software components for everyday life systems, for infrastructure, for management and control are developed by *non-computer-specialists* who are not familiar with matters of maintainability, risk avoidance, error tolerance, precision, completeness, integration, migration, and component coherence.

However, *programmers have already initially and intrinsically an idea and models how to solve their problems and how to solve them*. This idea is the rationale which underlies programming, i.e. it is a mental model of the solution that is going to be developed. As long as the models behind are only intrinsic and hidden, the solution background and the program ideas become tacit secrets of programmers. It seems far better if an appropriate support for modelling, gradual improvement, and refinement of the models is provided. If this support becomes sophisticated and code can be generated from models, the need for program development is reduced to the real problematic cases which are resolved by professionals. In this case, some models in a model suite [Tha10] become programs at a higher level of programming. They are compiled to classical programming languages. *Therefore, models become programs at a higher and more comprehensible level. They are more efficiently and correctly developed.*

1.1. The Path Towards True Fifth Generation Programming

Our approach fundamentally revises, combines, and corrects two already existing approaches: (1) Model-driven development approaches (MDD) (or engineering or architecture) are the latest developments (e.g. [SV05]). Users start with requirements and continue with platform-independent models which can be specialised and refined to platform-specific ones. The platform-specific models are translated to code. Yet, the mental model behind the requirements is not explicitly considered. The approach does also not consider the intrinsic details of all the models. (2) Conceptual-model programming [ELP11] asserts that programming activities can be carried out via models. Models are complete and holistic, are conceptual but precise, and are executable. These models can be refined at any level of abstraction. However, the underlying foundations remain incomplete thus hindering full realisation. Both approaches have so far failed to fully generate deployable systems. The path towards model-based programming has however already been tested for web information systems. A third approach, which is mathematically precise, is based on abstract state machines [BR18] that offers a semantically well-defined, pseudo-code language for specification at various abstraction levels. These models provide an accurate high-level description, support quality assessment, and can be mapped to third generation programs. By combining the first two approaches with the mathematically precise description, ***Modelling-as-Programming*** (MaP) will be a *springboard for next generation programming*. Next generation programming will be the first step towards true fifth generation programming.

1.2. The Storyline of This Paper

The paper develops a programme for true fifth generation programming that starts with models and uses models as a program specification. It is similar to second and third gen-

eration programming where programmers are writing programs in a high-level language and rely on a compiler that translates these programs to machine code. We propose to use models instead of programs and envision that models can be translated to machine code in a similar way. This paper presents the first starting vision to such novel kind of programming. The completion and full establishment of this programme may take a decade. Information system modelling is, however, already a positive proof of this kind of programming by models. Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, *changeable* and *adaptable* at different layers, *validatable* and *verifiable*, and *maintainable*. Therefore, we start first with a discussion what models-as-programs means. Next we discuss literate modelling as high quality modelling with model suites. Section 4 describes our envisioned realisation path.

2. Modelling - The Next Generation Programming

Model research has a long, more than 2000 years old history in sciences, engineering, and daily life (e.g. [Mül16,TN15]). Computer science and engineering uses models as the main vehicle for system construction, description of problems and systems, explanation, and system quality investigation. Computer science has developed a very large number of model notions. As investigated in [TN15], these notions mainly differ according to the model purpose, the attention of the community, the background, and especially the language setting.

2.1. Modelling is Often Only Normal Modelling

The main difference to classical programming, model-driven development, and conceptual-model programming is the explicit orientation on the extrinsic surface model called *normal model* (yellow color in Figure 1). By contrast, the *deep model* (green color in Figure 1) consists of the background, the context, the intentions behind the model, the commonly accepted practice in the community of practice (CoP), and the setup of the model. The deep model and the normal model should however be considered as a whole.

We use the model notion from [TN15,Tha18][3] that is depicted in Figure 1. An essential result of the interdisciplinary brainstorming seminars of the modelling commu-

[3] A model is a *well-formed*, *adequate*, and *dependable instrument* that *represents origins* and that *functions in utilisation scenarios*. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its CoP within some context and correspond to the functions that a model fulfils in utilisation scenarios. The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artefact a model comes with its background that is often given only in an implicit and hidden form and not explicitly explained. The *background* consists of an undisputable *grounding* from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable *basis* from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, common sense). A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity

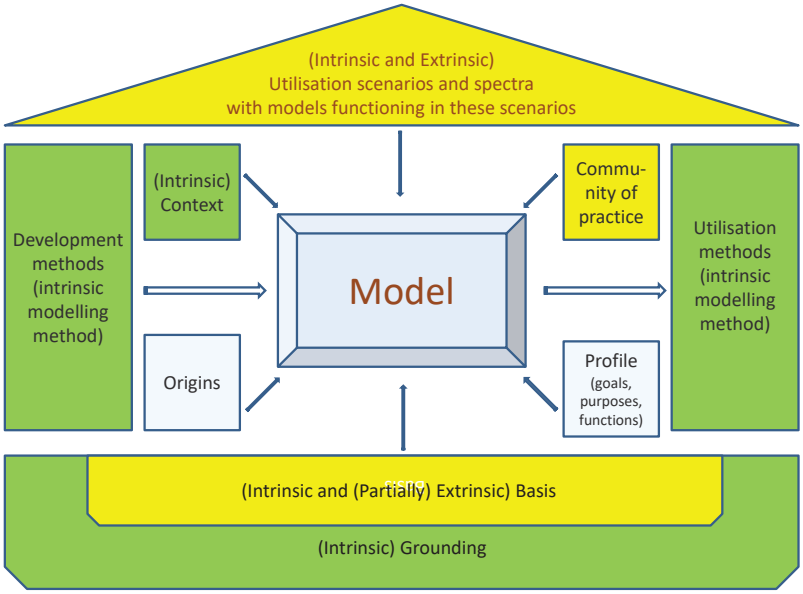

Figure 1. The conception and notion of a model with extrinsic elements of the normal model (yellow color) and intrinsic elements of the deep model (green color)

nity at Kiel University since 2009 has been the explication of the intrinsic enthymeme-like deep model within all models used in science and technology [TN15]. Modelling is currently mainly modelling of the surface-like normal model without explicit description of the background. The normal model is bound to its deep model. It is thus not entirely understandable to anybody, e.g. outside its context (e.g. discipline) and its CoP. The concentration on normal modelling is one of the main reasons why model-driven development and conceptual-model program have not succeeded as expected. If the deep model is not known and not understood then translation or mapping to platform-specific models becomes infeasible. This situation is similar to specifying LaTeX text without the corresponding strategic setup, e.g. by .cls, .clo, .def, .bst, .sty etc. files and libraries. The separation into the intrinsic and extrinsic parts of models is also depicted in Fig. 1 where the light blue part the normal model, the yellow part represents the mixed extrinsic and intrinsic part, the green part the part that is mainly build from the deep model. The explicit description of a deep model reveals the secrets within models.

The deep model has not been considered for model-driven development (MDE, MDA, MDD) and conceptual-model programming. This non-consideration is the main source for impedance mismatches between source and host languages, crucial translation problems, and the failure of these approaches. The *explicit treatment of deep models* and of high-quality source models (e.g. standardised generic and reference models) is

formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is *sufficient* by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called dependable if it is sufficient and it is justified for some of the justification properties and for some of the sufficiency characteristics.

THE *essential* difference of our approach. Complete knowledge about the model is THE *guarantee for modelling as programming*.

2.2. Models as Model Suites

Researchers and engineers often collaborate in interdisciplinary and interacting communities. A model suite [Tha10] can also incorporate viewpoints and sub-models that support interaction and exchange with collaborating partners on the basis of sub-models. I envision that model-backed collaboration is far more effectively support collaborative work and problem solving in communities.

A model may combine several facets at the same time and may thus have its structure where some facets support specific purposes and functions. A model suite is a coherent collection of well-associated models at a variety of abstraction levels, foci, and scopes. The associations are explicitly stated, enhanced to explicit maintenance schemata, and supported by tracers for the establishment of coherence. Coherence describes a fixed relationship between the models in a model suite.

Model suites support holistic and consistent description of models at numerous levels of detail, precision, completeness, foci, and scopes depending on context, function of the model, community of practice, and origins that are really of interest. They close thus the gap among ideas and intentions, requirements, conceptualisations, and realisations. Models in a model suite support various functions such as communication support, mediator for system construction, basis for problem solving, facilitator for contracting and negotiation, documentation, analysis and quality assessment, support for integration, and warrant for migration and modernisations. Representation and informative models are typical models in a model suite. The latter can be generated. Models in a model suite can also be generated from others, e.g. in order to represent viewpoints [Tha10].

Model suite development is an intellectually challenging task if we aim at a complete model suite. For this reason, MaP also incorporates toolbox support.

2.3. Models are New Generation Programs

Models are currently used as a prescription or blueprint for programs of the third or fourth generation. We envision that models themselves can be considered to be the source code, i.e. models and model suites are essentially the program source. The independence concepts (hardware, operation system, physical, and logical independence) will be extended by programming language, platform, environment, and system independence since models can be transformed to different kinds of programming languages. The translation requires sophisticated compilers including optimisation facilities. Models in a model suite can be translated to code while other models in the model suite serve as communication and collaboration means in the CoP.

MaP proposes now new programming paradigms, develops novel solutions to problem solving, integrates model-based and model-backed work into current approaches, and intents to *incubat*e true fifth generation programming.

3. Literate Modelling as Literate Programming

3.1. Towards Literate Modelling

A holistic approach entirely based on models will thus provide a better support. The model support must include multi-model treatment with coherent model ensembles at different levels of abstraction, with explicit and maintainable associations among these models, with supported intellectual management of the complexity, with explicit knowledge of details of the models, and with sophisticated quality management. Models have also their anti-profile [Tha17] that limits applicability of model-backed development. Such a holistic and general approach would be too ambitious and unrealistic. Therefore, MaP focusses its scope on selected areas of Computer Science and Engineering. We, thus, start with four application areas of model-backed development and then use the experiences gained to extend our scope.

Already literate programming [Kn84] considered a central program together with satellite programs, especially for interfacing and documenting. We generalise and extend this new paradigm of programming with GibHub, 'holon' programming, and schemata of cognitive semantics. Projects like the Axiom project or the mathematical problem solver [Pod01] have already shown the real potential of literate programming. The association among models must become manageable and be supported by computational features. The design and development of model suites has realigned the model ensemble approach to meet this challenge. One reason that literate programming has not become a mainstream paradigm is that implicit and intrinsic components remain largely unknown. Another reason is the missing representation of models behind the code and the missing representation of thoughts of people. A third reason is the hidden representation of the intention and the application task that has been the reason for developing a program. A fourth reason is the implicit usage of experience and of generic models behind the program solution. Our approach will reveal intentions, strategic and tactical issues (see Figure 2).

Model-backed development of systems will not be a universal solution to all computational problems. It is however a solution for those application cases for which users have an idea that can be expressed as a mental model. These models can be understood as interfacing or communicated models. In this case, the mental model can be enhanced by models characterising the problem space according to the needs, interest, and intentions of users. Users have their own understanding of the problem space, their educational and work environment, and their culture as 'programming of their mind' [Hof01]. Different users might use different models for the same application case. That means, we support modelling as *literate modelling*. It frees the modeller from the inherent and implicit parts of a model as modelling is understood at present and imposed by modelling languages and means that the modeller can develop models in the order of the flow of their thoughts. A model suite also explains the model and its intrinsic components in a natural language and is interspersed with snippets of representation and realisation models. This means that models are very easy to understand, to justify, and to share, as all its details are well explained. Literate modelling is a change of the mindset by making the story of the model suite explicit. It reduces bugs, misconceptions, and flaws in a model. Models are communicated to both people and machines.

Models can be transferred to programs if all details within the model are known and the models themselves are well-structured based on a sophisticated model language, e.g.

extended entity-relationship models with stereotyped and refinable profiles and directives for realisation (among stereotypes we may select the default one) [KT16]. A general model language would be the basis for a universal solution and thus cannot exist. We can however use modern engineering approaches. Engineers already develop systems based on standardised components. They use composition pattern and some kind of quality and failure management. Components and compositions can be coherently specialised in machine tool building. They are based on standards in this case. Database and workflow models can also be built in this form [MNS+13]. Standards are in these cases generic models or reference models. We restrict our approach to this kind of models. This standard-backed approach can also be applied to model suites. All models that are not directly derived from mental models are standard-backed models. Mental models are going to be enhanced and generalised in such a way that they become the source for a generic or reference model. This harmonised treatment then supports model-backed development of programs. Thus model suites become the source for programs.

3.2. Towards Next Generation Programming as Starting Point for True Fifth Generation Programming

The rationale behind the initial fifth generation program language project was very ambitious [Mo82]: development of a general-purpose, multilingual environment and general-purpose problem solver that also supports natural language communication and multimedia processing; support for general knowledge representation, storage, processing, and retrieval; automatic problem-solving after accurate and abstract problem specification; closing the mental and the language gap between users and computers; development of sophisticated logical and functional machines for backend computation; developing an advanced architecture for support of these features; development of sophisticated theories and technology for support; development of a distribution and collaboration architecture. However, the initial fifth generation programming languages project was never completed. It failed despite its great technological and social advances since it was too early for the hardware progress, it was highly dependent on AI technology, it did not achieve an integration of AI and human-computer interface techniques, it was oriented on one programming paradigm and on mathematical logics, it tried to provide a universal solution for any kind of programming, it routed granularity to basic program blocks, and it was oriented on one final solution instead of coherent reasoning of coherent variants of final solutions depending on the focus and scope of a user[4].

MaP now aims at *true fifth generation programming* where *models are essentially programs of next generation* and models are translated to code in various third or fourth generation languages. Programs of next generation programming must at the same time be understandable by all parties involved, support abstraction, be as accurate and precise for the problem space and the issues to be solved, transferable and distributable to other parties, commonly deployable by all parties, and support quality management and reasoning.

Due to its user orientation, next generation programming cannot rely on single language paradigms. Instead, models as programs must become *language independent*. Languages of third and fourth generation of programming languages became already hard-

[4]Our approach has been inspired by H. Aiso [Ais88] who chaired the architecture sub-committee in the Japanese fifth generation computer project [Mo82].

ware, storage, operating system, firmware, and optimiser independent. Language and platform independence will support non-specialists to program based on their models without forcing programmers to certain style of thinking and programming.

Our approach is based on stereotypes for deep models and on generic and reference models as a starting point for normal models. Model-driven development, engineering and architecture (MDD, MDE, MDA) taught some valuable lessons reported about model-driven approaches, e.g. those in the list in [Web95]. Two main problems limited the applicability of MDA and MDD: the intrinsic and not explicitly stated deep model and the restrictions in layering development.

Models are a more natural kind of human reasoning than programs could be. Programs are often oriented on algorithmic thinking. Most programming languages use simple variable spaces. Object-orientation has added essentially user defined object-relational structures. Network diagrams are one of the reasons why the entity-relationship structuring became quickly popular. All these structures are, however, one-facetted, cognitively simplistic, and without multiple viewpoint representation. *Human communication* is partner-oriented, is ambiguous in structure and meaning, uses partially semantics, is culture-dependent, is oriented on parsimony instead of completeness or preciseness, uses previous communication histories, considers principles of communication such as politeness, uses background information, and incorporates ellipses and context-dependent abbreviations. Models are represented according to the communication flow and the communication partner. Models thus must have a number of faces (or contrast classes and relevance classes [Fra80]) that can be used interchangeably. *Human thinking* does not separate syntax, semantics, and pragmatics but treats them as a coherent and larger whole. It is rather based on mental models such as collections of interrelated personal perception models or environment- and culture-oriented domain-situation models. Moreover, it is complex, multi-facetted, highly adaptable to different viewpoints and opinion, multi-viewpoint oriented, and network-connected.

Non-professional programmers are confronted with problems of transferring their understanding and their models to algorithmic and computer-oriented environments. The transformation process from thoughts to programs is error-prone, is oriented on the normal case without consideration of the entire picture, requires one central representation, and some understanding of computing technology.

Therefore, it is far better to support *non-professionals by model-backed programming* instead of forcing them to learn and to fully understand programming in third or fourth generation languages. True fifth generation programming is a better model-based representation of problems. In this case, the model must be understood both in their extrinsic, directly represented components and their intrinsic background. The second part has not taken into consideration in model-driven development and conceptual-model programming. This second part is, however, a central necessity for model-backed next generation programming. The explicit treatment of this part will become a 'silver bullets' for the new programming.

4. The Programme and Its Realisation Path

4.1. The Layered Model Development Framework

The layered approach has already often and successfully been used in Computer Engineering. Most program language realisations follow this approach since COBOL and ALGOL 60 development (e.g. infrastructure definition; variable space; program space; interpreted or compiled code) and application development (e.g. application case; infrastructure; design; specialisation & tuning; Deliver). Layering has also been the guiding paradigm behind text processing, e.g. behind the TeX and LaTeX realisations [Knu86,Lam94] with a general setup layer, the content layer, the adaptable device-independent layer, and the delivery layer. We assume that this approach is the universal basis for treatment of models as programs at least for programming by non-specialists. The approach for professional programmers is different. It can, however, also be supported in this manner how the success of programming environments such as Eclipse has already been demonstrating. These toolboxes have become accepted for their ease of use.

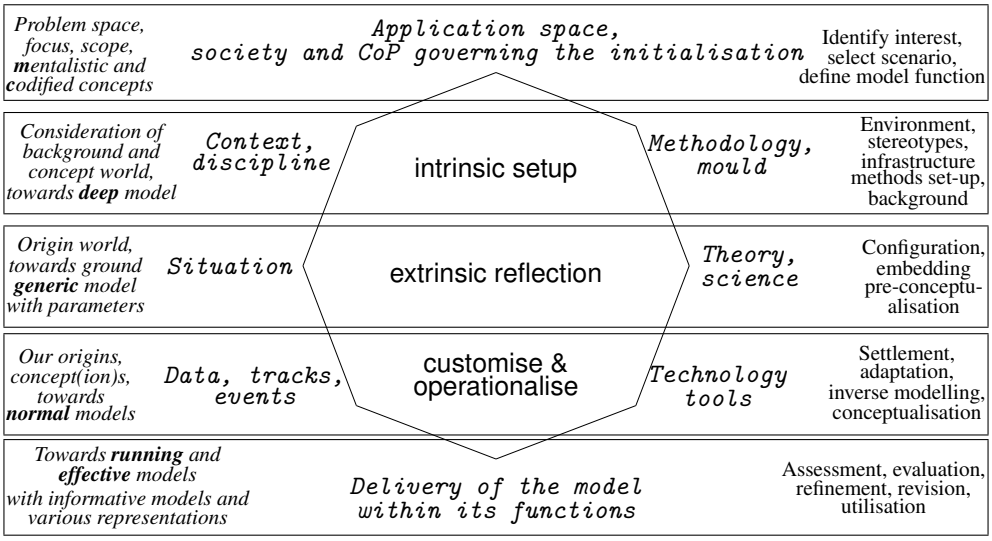

Figure 2. The layered approach to model suite development and program generation

The *model suite* will be layered in Figure 2 into models for initialisation, for strategic setup, for tactic definition, for operational adaptation, and for model delivery. At the left side the issues for the model suite are represented. The right side displays the activities and methods for the development. The corners of the octagon represent the starting and final stages as well as sources and enablers of the intermediate stages. We restrict the picture to the layered model development process. The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built. Currently, one model is considered to be the final product. Model suite development results in a number of models: deep, generic, specific, and normal models. Since any model has its deep elements, we start with the development of this deep model. In many cases, we use reference models or generic models (or tactical model frames like those used in

data mining and analysis). Models have their own background that is typically not stated explicitly but intrinsically. Methods for developing and utilising models are considered to be given. The intrinsic part of a model and these methods form the deep sub-model [Tha18]. The deep model is coupled with methodologies and with moulds that govern how to develop and to utilise a model. The deep as well as the general model are starting points for developing the extrinsic or "normal" part of a model. Consideration of modelling is often only restricted to normal models similar to normal science [Kuh70]. Classical modelling often intentionally presupposes the initialisation and intrinsic layers and assumes that these layers cannot be reconsidered and specifically changed according to the functions. The developer thus loses the understanding of the model and why the model is dependent without an understanding of these layers. Model suites, however, integrate these models over all layers. Another main obstacle why model-driven development and conceptual-model programming has not yet succeeded is the non-consideration of modelling moulds.

Intention modelling extends rational-based software engineering [BCM+08]. The W*H specification pattern [DT15] can be applied to *model initialisation* as well as includes then the following set of statements: (1) a plan, function, and purpose dimension (model as a conception: 'wherefore', 'why', 'to what place or end', 'for when', 'for which reason') within a scenario in which the model is going to be used as an instrument; (2) a user or CoP dimension ('who', 'by whom', 'to whom', 'whichever') that describes the task portfolio in the CoP and profile of users including beliefs, desires and intentions; (3) an application and a problem dimension ('in what particular or respect' , 'from which', 'for what', 'where', 'whence'); the added value dimension (evaluation). The initialisation layer may also be enhanced by a contrast space for user-related separation of a model and a relevance space that is dependent on the user [Fra80]. The contrast and relevance spaces as a form of mind-setting also define what is not of interest.

The *enabling intrinsic setup layer* defines the opportunity space and the infrastructure for the model. The results will be on the one hand a deep model and from the other hand a modelling framework or modelling mould that guides and govern next activities. In future, the developer will define the *context* and the most of the *background* (the grounding (paradigms, postulates, restrictions, theories, culture, foundations) and the basis (assumptions, concept world, practices, language as carrier, thought community and thought style, methodology, pattern, routines, common sense)) of the model. The context, extrinsic, and strategic dimension answers question like 'at or towards which', 'where about', 'to what place or situation', and 'when'. Additionally, developers decide which methodology and environment seem to be the most effective and purposeful. The development and deployment dimension ('how', 'whence', 'what in', 'what out', 'where') defines the modelling methodology, i.e. the modelling mould.

Deep model elements will be separated from elements of the normal model at the extrinsic source reflection layer. According to the model function, the normal model represents extrinsic elements of potential origins based on their content and thus answers questions such as 'what', 'with which', and 'by means of which'. It reflects the extrinsic theory essentials that are necessarily to be represented, e.g. conceptions or pre-conceptions from the theory that is underpinning the application. The normal model can be built from scratch ('greenfield' modelling). It is usually based on experience gained. The latter case thus starts with a generic or reference model that might incorporate parameters. The extrinsic source reflection layer can be understood as a tactical layer.

Generic or general normal models are adjusted to those that a best fitted to those origins that are considered for the application in the *operational customisation layer*. This layer is sometimes holistically handled with extrinsic reflection. Inverse modelling uses this layer for adaptation of the model to the observational data (e.g. data adaption in astrophysics or parameter instantiation in most data mining processes). In some cases, this layer seems to be trivial. It is not trivial in the general case however. It instantiates parameters, adapts the normal model to those origins (or data sources) that are really under consideration, prepares the model for the special use and to the special - most appropriate - solution, and integrates the deep model with the normal model. The normal model is typically pruned in order to become simpler based on Solomonoff and Occam principled deviation and error-prone. The (normal) model could be enhanced by concepts and thus become a *conceptual model*.

The final result of the modelling process is a model suite that is adequate for origins, properly justified, and sufficient at the *delivery and product layer*. We cannot expect that one single model is the best instrument for all members of the community of practice. A sophisticated model that integrates deep and specific normal models is delivered to some members. An informative model that is derived from this model can be better for other CoP members. Models delivered in the finalisation stage are often enhanced by additional annotations, e.g. relating the model to the demands for members of the CoP by answering the 'with', 'by which', 'by whom', 'to whom', 'whichever', 'what in', and 'what out' questions. At the delivery and product layer we, thus, generate a number of associated models.

Models delivered in this approach become more reliable and - in the general sense - dependable on the explication of the deep model and of the initialisation. Dependability can now be considered according to dependability that is already given by deep models from one side and by generic and reference models from the other side. For both kinds of models we use stereotypes and pattern similar to the usage of class and setup libraries in LaTeX and the special document templates that provide a specific parameterised structure for content development for LaTeX content input (e.g. .tex and .bib files). Skipping the operational layer can be an option if a single model is delivered as a program collection. The typical case, however, is adaptation, fitting, pruning, specialisation, operationalisation, and exemplification at the operational layer.

Transformation is based on standardised combinable components (not only basic elements) as pattern and templates. *Generic and deep models* are going to be developed on the basis of standardized stereotypes and pattern. The specialisation and combination of models is supported by a model algebra that generalises the ER algebra [MNS+13]. Each specialisation can be enhanced by directives similar to pragmas in C++.

4.2. A Path towards Modelling-as-Programming

Computer Science and Engineering has resulted in many tools for support of programming. However, we observe that most of these tools have been oriented on bottom-up representation of program language elements and constructors for programs. Some of the tools provide some kind of abstraction. Very few tools also allow introduction of *components* and support modularity with refinement. Many tools are based on their own language variant and own interpretation, e.g. [Kru04]. Tools should however be based on *standard components* that can be refined for specific purposes. This path of *componenti-*

sation is essentially implemented with programming languages of the second, third, and fourth generation - at least for bottom-up elements and block concepts. Many tools tend to be unnecessarily complete in order to provide the full flexibility. All tools consider syntax on its own, define semantics of elements and construction on top of syntax, and do not consider personalised pragmatics. Natural languages have however collocations for words, holistic syntactical-semantical constructions, and their special interpretation in dependence on the context and the community of practice.

We are going to partially represent generic normal models in three frameworks:

- ADOxx [KMM16] is a *configurable meta-modelling development and configuration platform* that supports specification in a larger variety of graphical modelling languages. It follows the MOF (meta-object-facility) approach by OMG [PM07] based on a separation of abstraction layers of specification languages: M1 as the layer of model creation and description; M2 as the layer model language specification (considered as meta-model); M3 as the layer of frames for language specification (considered as meta-meta-model). The compiler approach can be integrated into ADOxx.
- Ptolemy II [Pto18] focuses on *actor-oriented modelling of complex systems*. The application of Ptolemy II in our approach must, however, consider a number of specific problems and must develop solutions to them. Ptolemy is oriented on bottom-up level of components. Abstraction in specifications is still an issue. There is a high freedom for specification and thus the approach struggles still with standardisation. Generic normal model can be used for standardisation. Intrinsic strategic and tactic parts of models are not yet considered. The model suite concept fits well into Ptolemy II.
- KIELER [Kie18] provides an *eclipse-based framework for diagrammatic model specification*. It aims at improving comprehensibility of diagrams, in decreasing development and maintenance time, and in providing facilities for analysis of dynamic behaviour of diagrammatically represented processes. Semantics is based on sequentially constructive sequence charts. Normal models can be represented as long as they are given in diagrammatic form and as long as their semantics if based on sequence charts.

Model-based development and architecture as well as conceptual-model programming have also been bound to imperfect tools. Moreover, they fail since the *deep model* is not taken into consideration. They meet thus all the classical impedance mismatches. A *proper transformation* can only be developed if either the source and target share their deep models or the deep models are transformed as well. Above all, programs are developed by people who have their culture, esp. programming culture and who are biased and framed by their way of programming and working.

This approach is based on a number of *new assumptions*: models consist of specialised and refined components that are combined via construction expressions; model components can be stereotyped and refined based on a specialisation approach; interdependence of refinement can be handled by attribute grammar constructions; construction expressions can also be stereotyped; stereotypes form the strategic layer of description; stereotypes can be specialised to pattern at the tactical layer and to templates at the operational layer; stereotypes, pattern and templates form semiotic units with their own specific syntax and with their fully integrated semantics.

Each sub-discipline in Computer Science has developed its specific style of modelling. This style is based on specific languages which have their specific grammar. Following the Eugenia [PKP14] framework, attribute grammars can be developed for these languages. In this case compiler-compiler approaches become applicable [BL74].

The Kiel team has been participating in tool development for database design, database engineering, and database performance management. Starting in the 1980s with the RADD (Rapid Application and Database Design) workbench (e.g. [AAB+95]), we systematically extended the domain of structure specification by database programming (finally with the VisualSQL tool (e.g. [JT03])), by performance management and tuning (e.g. [TT11]), by integration of workflow specification (e.g. [BR18]), by integrating web information systems design (e.g. [ST04]), and by codesign (e.g. [Tha04]). These specification methods have been extended to a methodological framework (e.g. [JMTV05]) that finally reached maturity level 3 in SPICE in 2005 in one of our collaboration projects. We have also developed the translation tools for transformation of (conceptual) models to code.

4.3. A Proposal for the Realisation Approach

The implementation approach to MaP is inspired by four projects.

(a) *Transformators and compilers for conceptual database models*: The RADD toolbox (Rapid Database Development) is based on the conceptual entity-relationship modelling language. This graph-based language supports conceptual development, documentation, reasoning, and requirements engineering for database analysis, design, and development. The graph-oriented approach has been compiled on the basis of graph grammars [AAB+95,Run94,Tha00] combined with attribute grammar approaches. The conceptual schema formulated in this language or enhancements of this language can be used for derivation and compilation of realisation schemata, especially for object-relational and XML platforms. It is enhanced by view suites, visual query systems (VisualSQL), and by performance optimisers. Currently, the development is transferred to ADOxx [KMM16] from OMIlab. The compilation approach is presented in [KMM16].

(b) *DEPOT-MS (DrEsdner PrOgrammTransformation)* [BL74]: DEPOT system is a compiler-compiler for domain-specific languages (DSL) (historically: little languages, application-domain languages (Fachsprache)) that has been used to compile specific language programs to programs in the mediator language (first BESM6/ALGOL, later PASCAL, finally PL/1 [GHL+85]) which can then be translated to executable code. The approach integrates the multi-language approach [Ers81], attribute grammars [GRW77,RF87], and theory of grammars [Hut86,Tha75]. The system is similar to the MetaCASE toolbox [Dah97] or development environments, e.g. Ptolemy II.

(c) *LaTeX and TeX* [Knu86,Lam94]: The TeX and LaTeX approach is based on a strict onion or layered approach with (1) an internal layer for formatting and general initialisation (e.g. .fmt, .tfm, .fd, .def, .ltx, .dat, .afm, .cfg. .clo files), (2) a structure-style-language layer (e.g. .cls, .sty, .ldf, .bst files) that includes many additional library packages, (3) the input document suite (mainly .tex, .bib, .ist files), (4) the internal supporting and generated layer (e.g. .aux, .log, .lof, .bbl, .ind, .toc, .lof, .log files) that also support related applications, (5) a generic intermediate output layer (especially .dvi files), and (6) a delivery layer (e.g. .pdf, .ps, .html, specific printer files) for multiple output variants.

(d) *Libraries of reference models* (e.g. [BKV17,FL07,KMM16]): Libraries and toolboxes of solutions and programs are widely used in science and engineering. Specific

reference models are universal models [MJ04, Sil01] as well as generic models [Tro16]. Universal and enhanced models may be algebraically combined [MNS+13] and refined [dRE98] based on a model calculus. Models may also be enhanced by metadata descriptions and by informative models [DT12,Kra18]. Models and model suites may be evaluated based on their potential and capacity [BT15].

The integration of these four technologies is very ambitious. Generation of programs from models extends the models@runtime initiative [BFCA14] by direct compilation of programs from given models instead of enhancing runtime environments by models and abstractions. The proposed layering might however provide a solution for this integration and the necessary harmonisation. The variety of application-domain languages is as large (an estimation stated about 2.500 such languages in 1985) as the one of DSL [Fra11] or multi-level languages. Our layered architecture for models is going to be combined with the abstraction/refinement approach [Bör07,dRE98]. The layered architecture became a common culture in Computer Science. Modern systems have been built on thus kind of layering for system development (e.g. by layering into application case - infrastructure - design - specialisation & tuning - delivering), for problem solving frameworks (e.g. by task ordering ((1) problem case, (2) setting, (3) incubation, (4) enlightening, (5) finalising), for data analysis (e.g. by workflow pattern ((1) define & identify, (2) select solution class, (3) select solution pattern, (4) derive parameter values, (4) fit & prune, (5) finalize), and for engineering (based on general approaches ((1) know it, (2) understand it, (3) construct it, (4) configure it, (5) use it)).

The first three inspiring projects are based on compiler technology, attribute and graph grammars, pattern and stereotype architectures [ANT14], and principles of programming languages (starting with early thoughts [Lan73] to more advanced ones [GGZ04,SGM02,Wir96]). We oriented model transformation on macro-level, component-oriented, and refinable translation [FG11] instead of meso- or micro-level transformations used for most syntax-oriented translators. Models typically consist of associated and bundled components that have their inner specialisation.

The translater is also used for generation of warnings and error messages for systematic improvement of models. Since we start with development of generic normal models and deep models, we concentrate on quality assessment and improvement for normal models. Normal models should be as adequate and dependable for the given application. Later on, quality establishment is extended to the strategic and tactical layers.

5. Conclusion

MaP aims at true fifth generation programming as a new programming paradigm where models are essentially programs of next generation and models are translated to code in various third or fourth generation languages. Users program by model development and rely on the compilation of these models into the most appropriate environment.

5.1. The Intended Outcome of Our Approach

The main outcome of MaP is a *proposal for true fifth generation programming as programming by models*. The capacity and the potential of MaP are evaluated. The strengths, weaknesses, opportunities, and the threats are demonstrated in the four application areas in such a way that they can be used for an extrapolation to other application areas.

An essential outcome of MaP is the *layered architecture* and a realisation of modelling as programming. Model suite are used as a foundation for literate programming. Quality and literate models are understandable, transferable, distributable, and commonly usable. MaP users may design their own modelling styles, templates, stereotypes, and configuration. They also may concentrate on development of normal models while inheriting the initialisation and configuration, the deep models, the methodology, and the techniques for model realisation and model representation.

The MaP approach supports *programming by everybody at any time*. Models become the main means for collaboration among partners. Models may evolve and therefore evolution and modernisation are less painful tasks. For non-specialists in programming, models are typically of higher quality than the programs. There-fore, the generated programs are of higher quality. Models can also be used for communication and exchange of experience. Modelling as programming thus support sustainability of developed solutions. The model is then the code. The compiler assures that program execution corresponds to the conceptual specification thus making the model directly executable. At the same time, model suites treat the model in an explicit, complete and holistic way without any intrinsic and hidden details. Elements of a model suite are conceptualisations of the thoughts and understanding of developers. They are precisely defined and commonly agreed with the concepts in the application space. I will thus attain a good level of parsimony for model and therefore program developers. Furthermore, application evolution is going to occur at the level of the model suite. Modernisation, migration, and evolution occurs at the level of the model suite and does not require consideration of lower level details.

MaP supports model-based reasoning as a natural kind of reasoning. Solutions can be developed in a large variety of reasoning styles including hypothetical, abductive, inductive, deductive, and other advanced reasoning methods. Models can be refined. MaP thus also supports inverse modelling.

5.2. Are You Still Programming or Are You Already Modelling as Programming?

Programming has become a central technique in science and engineering. Software systems are often developed by non-specialists in programming without a detailed knowledge and skills, without an insight into the culture of computer science, and without plans for systematic development. These systems and programs often have a poor structure and architecture, little documentation, and lost their insight and knowledge of specific solutions.

Programs of the future must be understandable by all parties involved, must be accurate and precise enough for the task they support, and must support reasoning and controlled realisation and evolution at all levels of abstraction. Programming languages are currently languages of third or fourth generation. Those generations have so far provided hardware independence, linker independence, operating system independence, and execution code independence. Programs are nowadays compiled or at least interpreted and do not require system knowledge by the ordinary programmer.

In the past, the fifth generation computing project sought to develop systems and programs that are closer to people in their communication and knowledge processing capabilities. It should have been a shift to a new paradigm of human-oriented computing in the sense of T. Kuhn [Kuh70]. This world-wide project failed despite its great technolog-

ical and social changes because it was too early, it was highly dependent on AI technology, it did not achieve an integration of AI and human-computer interface techniques, it was oriented on one programming paradigm and on mathematical logics, it routed granularity to basic program blocks, and it was oriented on one final solution instead of coherent reasoning of coherent variants of final solutions depending on the focus and scope of a user.

5.3. Envisioned Deliverables of MaP

This paper develops the general approach to true fifth generation programming. The realisability of the approach has already been demonstrated for database development [KT16]. Database specification follows the *global-as-design* approach. BPMN specification follows the *local-as-design* approach. This approach requires view schema specification for data support of the workflow diagrams. The co-design approach [Tha04] is the basis for integration of workflow specification to database specification.

The general proof of concept is however a task of the future. Our programme can be based on development of the following deliverables:

Model suite description language: The model language consists of an associated bundle of languages for model elements that users may modify depending on their needs or simply reuse them as already established sub-cultures. These elements are different from ordinary programs because they are essentially declarative rather than imperative. Similar to UML stereotypes, MaP model class and model style languages are ready for application, can be extended, combined, and adapted. Users don't have to work on the details of the models as programs. The system takes over the integration and composition work as it deduces the consequences of the model. It also provides a new discipline of modelling according to which principles of a particular modelling language design can be stated precisely. The underlying intelligence does not remain the secret of the modellers. It is spelled out in the style language and based on the model class language. Thus, coherence and consistency can readily be obtained where they are desirable. New model elements can readily be extended to new elements that are compatible with the existing ones. The model suite description language is developed as a collection of grammars, grammar-aware theories and software, and techniques for implementation.

Technologies, techniques, methodologies, and modelling moulds: The development of models in a model suite is based on a model library with models that can be used as an inherited or initial model for systematic composition of the model suite. The approach to model construction is canonised on the basis of methodologies and modelling moulds which systematically combine known and novel techniques and technologies for model development. Modelling moulds enable the modeller to reuse systematics and theories that have been successfully deployed in the past. They enable us to start with application space models, with deep models, with generic models, and with reference models without explicit reinvestigation of these models. The explicit agreement on a given mould eases, enables, and supports an economic development process.

Environment for an extension towards modelware as next generation literate modelling: Our approach aims at development of a general infrastructure for treatment of models as programs. This infrastructure includes also standardised solutions that can be reused in other applications. These solutions are based on application space models and deep models that are typically less volatile than normal models. Therefore, the library

allows quick and well-based modelling by non-specialists which may concentrate on development of normal models instead of developing the entire holistic model. They may accept the library models as a basis and then use generic and reference models as a starting point for normal model development. The model suite is also transformed to programs in programming languages of third or fourth generations. This approach disentangles modellers from programming and allows them to concentrate on the model development. The model is then the product. The transformed program is of a higher quality and more liable.

Compiler-compiler approach to model realisation for models as programs: The modelling infra-structure is an essential element for realisation of model suites and for treatment of models as programs of next generation programming. The metamodel represented in Fig. 1 is a model of a model. Its components and its associations are expressed as attribute grammar rules. The compiler-compiler approach is enhanced by the layered handling of models according to Fig. 2. All components of a model in the model suite are explicit and become thus transformable to representation models and to programs of third or fourth generation of programming. This generation is the basis for language and platform independence of the models themselves. Modellers thus become programmers of next generation programming languages. The quality of the generated programs is therefore higher than a non-specialised programmer could achieve. Compilation allows the integration of standards. Model suite libraries become then the kernel for modelware. Models become directly executable.

Tested, verified, and validated approaches for MaP: The MaP approach is gradually developed in four application areas for which I and my collaboration partners have sufficient experience. It will be assessed, evaluated, analysed, questioned, scrutinised and generalised in such a way that I will open the path to an extension of the approach to other application space, to other CoP with different interest and intentions, to other problem spaces, and to other concept space. This extension will be developed in our scientific network.

References

References

[Ais88] H. Aiso. The fifth generation computer systems project. Future Generation Comp. Syst., 4(3), pp. 159-175, 1988.

[AAB+95] M. Albrecht, M. Altus, E. Buchholz, A. Düsterhöft, and B. Thalheim. The rapid application and database development workbench - A comfortable database design tool. Proc. CAiSE'95, LNCS 932, pp. 327-340. Springer, 1995.

[Ama16] S. Amarasinghe. Third software crisis - the multicore problem.
http://habitatchronicles.com/2016/10/software- crisis-the-next-generation/, 2016.

[ANT14] B. AlBdaiwi, R. Noack, and B. Thalheim. Pattern-based conceptual data modelling. Information Modelling and Knowledge Bases, vol. XXVI of Frontiers in Artificial Intelligence and Applications, 272, pp. 1-20. IOS Press, 2014.

[BCM+08] J. E. Burge, J. M. Carroll, R. McCall, and I. Mistrik. Rationale-based software engineering. Springer, 2008.

[BFCA14] N. Bencomo, R. B. France, B. H.C. Cheng, and U. Aßmann. Models@run.time: foundations, applications, and roadmaps. LNCS, vol. 8378. Springer, 2014.

[BKV17] D. Bork, D. Karagiannis, and J. Vanthienen, editors. Proc. PrOse 2017, vol. 1999 of CEUR Workshop Proceedings. CEUR-WS.org, 2017.

[BL74] J. Bormann and J. Lötzsch. Definition und Realisierung von Fachsprachen mit DEPOT. PhD and Advanced PhD thesis, Technische Universität Dresden, Sektion Mathematik, 1974.

[Bör07] E. Börger. Modeling workflow patterns from first principles. Proc. ER 2007, LNCS 4801, pp. 1 -20. Springer, 2007.

[BR18] E. Börger and A. Raschke. Modeling Companion for Software Practitioners. Springer, 2018.

[BT15] R. Berghammer and B. Thalheim. Methodenbasierte mathematische Modellierung mit Relationenalgebren. Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung, pp. 67 -106. De Gryuter, Boston, 2015.

[Dah97] A. Dahanayake. An environment to support flexible information modelling. PhD thesis, Delft University of Technology, 1997.

[dRE98] W. P. de Roever and K. Engelhardt. Data Refinement: Model-oriented Proof Theories and their Comparison, vol. 46 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1998.

[DT12] A. Dahanayake and B. Thalheim. A conceptual model for IT service systems. Journal of Universal Computer Science, 18(17):2452 -2473, 2012.

[DT15] A. Dahanayake and B. Thalheim. W*H: The conceptual model for services. Correct Software in Web Applications and Web Services, Texts & Monographs in Symbolic Computation, pp. 145 -176, Wien, 2015. Springer.

[ELP11] D.W. Embley, S.W. Liddle, and O. Pastor. Conceptual-model programming: A manifesto. In Handbook of Conceptual Modeling - Theory, Practice, and Research Challenges, pp. 3-16. Springer, 2011.

[Ers81] A. P. Ershov. The transformational machine: Theme and variations. MFCS 1981, LNCS 118, pp. 16-32. Springer, 1981.

[Far16] F.R. Farmer. Software crisis: The next generation. http://habitatchronicles.com/2016/10/software-crisis-the-next-generation/, Oct. 14 2016.

[FG11] R. Farahbod and U. Glässer. The CoreASM modeling framework. Softw., Pract. Exper., 41(2) pp. 167-178, 2011.

[FL07] P. Fettke and P. Loos, editors. Reference Modeling for Business Systems Analysis. Hershey, 2007.

[Fra11] U. Frank. Some guidelines for the conception of domain-specific modelling languages. Proc. EMISA 2011, vol. 190 of LNI, pp. 93-106. GI, 2011.

[Fra80] B. Van Fraassen. The scientific image. Clarendon Press, Oxford, 1980.

[GEPG18] F.D. Giraldo, S. Espana, O. Pastor, and W. J. Giraldo. Considerations about quality in model-driven engineering - current state and challenges. Software Quality Journal, 26(2):685-750, 2018.

[GGZ04] S. Glesner, G. Goos, and W. Zimmermann. Verifix: Konstruktion und Architektur verifizierender Übersetzer (Verifix: Construction and Architecture of Verifying Compilers). it - Information Technology, 46(5), pp. 265-276, 2004.

[GHL+85] R. Grossmann, J. Hutschenreiter, J. Lampe, J. Lötzsch, and K.Mager. DEPOT 2a Metasystem für die Analyse und Verarbeitung verbundener Fachsprachen. Technical Report 85, Studientexte des WBZ MKR/Informations- verarbeitung der TU Dresden, Dresden, 1985.

[GRW77] H. Ganzinger, K. Ripken, and R. Wilhelm. Automatic generation of optimizing multipass compilers. In IFIP Congress, pp.535-540, 1977.

[Hei04] H.D. Heilige, editor. Geschichten der Informatik: Visionen, Paradigmen, Leitmotive. Springer, 2004.

[Hof01] G. Hofstede. Culture's Consequences, Comparing Values, Behaviors, Institutions and Organizations across Cultures. Sage Publisher, Thousand Oaks, 2001.

[Hut86] J. Hutschenreiter. Zur Pragmatik von Fachsprachen. PhD thesis, Technische Universität Dresden, Sektion Mathematik, 1986.

[JMTV05] H. Jaakkola, T. Mäkinen, B. Thalheim, and T. Varkoi. Evolving the database co-design framework by SPICE. Informaton Modelling and Knowledge Bases Vol. XVII, Series Frontiers in Artificial Intelligence, vol. 136, pp. 268-279. IOS Press, May 2006.

[JT03] H. Jaakkola and B. Thalheim.Visual SQL - high-quality ER-based query treatment. Proc. IWCMQ'2003, LNCS 2814, pp.129-139. Springer, 2003.

[JT10] H. Jaakkola and B. Thalheim. A framework for high quality software design and development: A systematic approach. IET Software, pp. 105-118, April 2010.

[KBF+05] T. Kruscha, B. Briel, G. Fiedler, K. Jannaschk, T. Raak, and B. Thalheim. Integratives HMI-Warehouse für einen durchgängigen HMI-Entwicklungsprozess. Elektronik im Kraftfahrzeug 2005. 12. Internationaler Kongress Electronic Systems for Vehicles, no. 1907 in VDI-Berichte. VDI-Verlag, 2005.

[KD17] H. K önig and Z. Diskin. Consistency checking of interrelated models: long version. Fachhochschule für die Wirtschaft Hannover, 2017.

[Kie18] Website Kieler. Kiel Integrated Environment for Layout Eclipse Rich Client.
https://www.rtsys.informatik.uni-kiel.de/en/research/kieler, 2018. Accessed July 29, 2018.

[KMM16] D. Karagiannis, H. C. Mayr, and J. Mylopoulos, editors. Domain-Specific Conceptual Modeling, Concepts, Methods and Tools. Springer, 2016.

[Knu84] D. E. Knuth. Literate programming. Comput. J., 27(2) pp. 97-111, 1984.

[Knu86] D.E. Knuth. The METAFONTbook. Addison-Wesley, 1986.

[Kra18] F.F. Kramer. Ein allgemeiner Ansatz zur Metadaten-Verwaltung. PhD thesis, Christian-Albrechts University of Kiel, Technical Faculty, Kiel, 2018.

[KRT+16] S. Karg, A. Raschke, M. Tichy, and G. Liebel. Model-driven software engineering in the open project: project experiences and lessons learned.Proc. Model Driven Engineering Languages and Systems 2016, pp.238-248. ACM, 2016.

[Kru04] P. Kruchten. The Rational Unified Process - An Introduction, 3rd Edition. Addison Wesley object technology series. Addison-Wesley, 2004.

[KT08] S. Kelly and J. - P. Tolvanen. Domain-Specific Modeling - Enabling Full Code Generation. Wiley, 2008.

[KT16] F. Kramer and B. Thalheim. Holistic conceptual and logical database structure modelling with ADOxx. Domain-Specific Conceptual Modeling, Concepts, Methods and Tools, pp.269-290, Springer, 2016.

[KT18] Y. Kropp and B. Thalheim.Viewpoint-oriented data management in collaborating research projects. Models: Concepts, Theory, Logic, Reasoning, and Semantics, Tributes, pp.146-174. College Publications, 2018.

[Kuh70] T. Kuhn. The Structure of Scientific Revolutions. University of Chicago Press, Chicago, Illinois, 2nd, enlarged, with postscript edition, 1970.

[KWB06] A. Kleppe, J. Warmer, and W. Bast. MDA Explained: The Model Driven Architecture - Practice and Promise. Addison Wesley, 2006.

[Lam94] L. Lamport. LaTeX: A document preparation system. Addison-Wesley, 1994.

[Lan73] H. Langmaack. Übersetzerkonstruktion. Internes Skriptum zu Vorlesungen WS 72/73, SS 73 an der Universität des Saarlandes, FB Angewandte Mathematik und Informatik, 1973.

[Lev12] N.G. Leveson. Engineering in a safer world: System thinking applied to safety engineering systems. MIT Press, 2012.

[LSS11] M.S. Lund, B. Solhaug, and K.Stolen. Model-Driven Risk Analysis - The CORAS Approach. Springer, 2011.

[MGS+13] P. Mohagheghi, W. Gilani, A. Stefanescu, M. A. Fernandez, B. Nordmoen, and M. Fritzsche. Where does model-driven engineering help? experiences from three industrial cases. Software & Systems Modeling, 12(3):619-639, 2013.

[MJ04] D. Marco and M. Jennings. Universal meta data models. Wiley Publ. Inc., 2004.

[MMR+17] H.C. Mayr, J. Michael, S. Ranasinghe, V.A. Shekhotsov, and C. Steinberger. Model centered architecture. Conceptual Modeling Perspectives, pp.85-104, Springer, 2017.

[MNS+13] Hui Ma, R. Noack, K.-D. Schewe, B. Thalheim, and Q. Wang. Complete conceptual schema algebras. Fundamenta Informaticae, 123:1-26, 2013.

[Mo82] T. Moto-oka, editor. Fifth generation computer systems. North-Holland, Amsterdam, 1982.

[Mül16] R. Müller. Geschichte des Systemdenkens und des Systembegriffs.
http://www.muellerscience.com/ SPEZIALITAETEN/System/systemgesch.htm, 2016.

[OMG17] OMG: Meta-object facility (MOF), core. Object Management Group TR, 2016.
www.omg.org/spec/MOF/.

[PKP14] R. F. Paige, D. S. Kolovos, and F. A. C. Polack. A tutorial on metamodelling for grammar researchers. Sci. Comput. Program., 96:396-416, 2014.

[PM07] O. Pastor and J. C. Molina. Model-driven architecture in practice - a software production environment based on conceptual modeling. Springer, 2007.

[Pod01] A.S. Podkolsin. Computer-based modelling of solution processes for mathematical tasks. ZPI at Mech-Mat MGU, Moscov, 2001. (In Russian).

[Pto18] Website PtolemyII. Ptolemy project: heterogeneous modelling and design.

http://ptolemy.berkeley.edu/ptolemyII/, 2018.

[RF87] G. Riedewald and P. Forbrig. Software specification methods and attribute grammars. Acta Cybern., 8(1):89–117, 1987.

[Run94] N. Runge. Scheme transformations on the basis of optimizing combinations of partially applicable elementary transformation methods. PhD thesis, Karlsruhe University, Computer Science Dept., 1994.

[Rus13] L. Rüschendorf. Mathematical risk analysis. Springer Ser. Oper. Res. Financ. Eng. Springer, Heidelberg, 2013.

[SGM02] C. A. Szyperski, D. Gruntz, and S. Murer. Component software - beyond object-oriented programming, 2nd Edition. Addison-Wesley component software series. Addison-Wesley, 2002.

[Sil01] L. Silverston. The data model resource book. Revised edition, vol. 2, Wiley, 2001.

[ST04] K.-D. Schewe and B. Thalheim. Structural media types in the development of data-intensive web information systems. Web Information Systems, pp.34-70. IDEA Group, 2004.

[Ste80] W. Steinmüller. Rationalisation and modellification: Two complementary implications of information technologies. IFIP Congress, pp. 853-861, 1980.

[SV05] T. Stahl and M. Völter. Model-driven software architectures. dPunkt, Heidelberg, 2005. (in German).

[Tha75] B. Thalheim. Theorie deterministischer kontextfreier Grammatiken. Diplomarbeit, Technische Universität Dresden, Sektion Mathematik, 1975 (In German).

[Tha00] B. Thalheim. Entity-relationship modeling - Foundations of database technology. Springer, Berlin, 2000.

[Tha04] B. Thalheim. Codesign of structuring, functionality, distribution, and interactivity. Australian Computer Science Comm., 31(6):3-12, 2004. Proc. APCCM'2004.

[Tha10] B. Thalheim. Model suites for multi-layered database modelling. Information Modelling and Knowledge Bases XXI, volume 206 of Frontiers in Artificial Intelligence and Applications, pp. 116–134. IOS Press, 2010.

[Tha17] B. Thalheim. General and specific model notions. Proc. ADBIS'17, LNCS 10509, pp.13-27, Cham, 2017, Springer.

[Tha18] B. Thalheim. Normal models and their modelling matrix. Models: Concepts, Theory, Logic, Reasoning, and Semantics, Tributes, pp. 44-72. College Publications, 2018.

[TN15] B. Thalheim and I. Nissen, editors. Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung. De Gruyter, Boston, 2015.

[Tro16] M. Tropmann-Frick. Genericity in Process-Aware Information Systems. PhD thesis, Christian-Albrechts University of Kiel, Technical Faculty, Kiel, 2016.

[TT11] M. Tropmann and B. Thalheim. Performance forecasting for performance critical huge databases. Information Modelling and Knowledge Bases, vol. XXII, pp.206-225. IOS Press, 2011.

[VM11] H. Vandierendonck and T. Mens. Averting the next software crisis. Computer, 44(4):88-90, 2011.

[Web95] B.F. Webster. Pitfalls of object-oriented development: a guide for the wary and entusiastic. M & T books, New York, 1995.

[Wir96] N. Wirth. Compiler construction. International computer science series. Addison-Wesley,1996.

[WVH17] H. Werthner and F. Van Harmelen editors. Informatics in the Future: Proceedings of the 11th European Computer Science Summit (ECSS 2015), Vienna, October 2015. Springer, 2017.

Conceptual Models and Their Foundations

Bernhard Thalheim[0000−0002−7909−7786]

Christian-Albrechts University at Kiel, Dept. of Computer Science, D-24098 Kiel
thalheim@is.informatik.uni-kiel.de
http://www.is.informatik.uni-kiel.de/∼thalheim

Abstract. There is no common agreement which artifact should (not) be considered to be a conceptual model although the term 'conceptual model' is used for more than for five decades in computer science and for more than one century in science and engineering. A team from all faculties at our university has been able to develop a notion of model that covers all model notions known in the disciplines of this team. We now introduce three notions of conceptual model in this paper: light, slim, and concise versions of the notion of conceptual model.
The paper answers the following questions: Are all models also conceptual models? What is a conceptual model? Is there a formal notion of a conceptual model? What is not yet a conceptual model? What will never be a conceptual model? What is a concept? Which philosophical and scientific foundations we should consider while modelling? Is the existence of an ontology a necessary prerequisite for the being as conceptual model?

Keywords: conceptual model · concept · model theory.

1 The Model

Humans have learned to use instruments for handling their issues, tasks, and problems in daily life. Sciences and engineering also widely use instruments. Human evolution, sciences, and engineering have been enabled by wide instrument utilisation. The language is one of these instruments – often seen as one of the main. Models are another main instrument in modern computer science and computer engineering (CS&CE). They are often material artifacts. They might, however, also be immaterial or virtual.

It is surprising that models and modelling (and its variants such as conceptual models) have not yet properly founded. This paper contributes to close this lacuna.

1.1 Models are Main Artifacts and Universal Instruments

Models have become one of the main artifacts in CS&CE. This wide usage has not led to a common agreement about the notion of a model. The same observation can be made for other scientific disciplines, for engineering, and for daily life. In our area models became as artifacts the main instrument for system and software construction.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

Models might be combined with other artifacts[1]. Concept and conception development might be integrated into models. In this case, models might be considered as conceptual models.

A Notion of Model

What is a model? According to [7, 27, 29] we define the model notion as follows:

"*A* **model** *is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.*

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios."

Well-formedness is often considered as a specific modelling language requirement. The criteria for adequacy are analogy (as a generalisation of the mapping property that forms a tight kind of analogy), being focused (as a generalisation of truncation or abstraction), and satisfying the purpose (as a generalisation of classical pragmatics properties). The generalisation of [10, 15, 17, 24] is necessary for consideration of model-being.

The model has another constituents that are often taken for granted. The model is based on a background, represents origins, is accepted by a community of practice, and follows the accepted context. The model thus becomes dependable, i.e. it is justified or viable and has a sufficient quality. Justification includes empirical corroboration, rational coherence, falsifiability (in our area often treated as validation or verification), and relative stability. In our area the quality characteristics can be based on software quality characteristics and procedures for evaluating these characteristics.

Scenarios determine functions of models as instruments. A model is utilised. This utilisation is bound to scenarios in which a model functions. Typical scenarios are system construction (with description, prescription, and coding subscenarios), communication, negotiation, conceptualisation, documentation, and learning. The model might have several functions in complex scenarios. For instance, a model functions as a blueprint for realisation in a prescription scenario. Other Other typical functioning are the usage as an informative means, as a companion, as a guide for development. The quality of a model must be sufficient for this usage. Therefore, models used for description and models used for prescription might be different.

The main qualification of models is the potential utilisation as an instrument. This utilisation is based on methods which are developed in the discipline.

[1] Due to the utilisation of artifacts as instrument we will concentrate on the instrument being of artifacts. This approach allows us to additionally consider virtual 'artifacts' such as mental models. An artifact is "something created by humans, usually for practical purpose. It is a product of artificial character due to extraneous (as human) agency". [3]. Furthermore, models can be real artifacts as well as thoughts. An additional difficulty is the negative usage of "artifact" in engineering as artificially introduced change (e.g. in presentation, miss or imperfection).

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

Models are used in Sciences, Engineering, and Daily Life

Models and model suites. There is no CS&CE subdiscipline that does not use models. Since models are abstractions and more generally are focused they are far better to use for investigation and system development. They are used in problem solving, in social, in engineering, and in science scenarios in a wide variety of forms. Often, models either consist of sub-models or form a model suite what is a well associated ensemble of sub-models. The models in a model suite [5] coexist, co-evolve, and support solutions of subtasks.

Models are one of the first instruments before languages. [9] Daily life utilisation of models is often unconscious, subconscious or preconscious. One of the first models that is learned by everybody is the 'model of mother'. It is used before we spell the word 'mother'. It has is variety of interpretations depending on the kind of behaviour of the mother. Models might be perception models that allow to summarise observations.

Matrices and deep models. Models typically consist of a relative stable part and of a part that is dependent on the actual circumstances. Typical modelling languages in CS&CE are predefined, use a limited vocabulary, have a relatively fixed – at least lexical – semantics, and allow to express certain aspects. They use their own techniques in some stereotype way, i.e. their utilisation follows some mould. Origins of models are often mental models such as perception or domain-situation models. The model background forms the deep sub-model. The current model is then the 'rest', i.e. a normal model. The utilisation and the mould form the matrix of the model.

Memes as basic and deep models. Humans reason, memorise, and express their thoughts based on memory chunks. Some of the chunks are relatively persistent and become memes [2, 25] which are then units of cultural evolution and selection. These memes are combined with some identification facilities. They represent a number of properties. They may be combined with other memes. They may be activated and deactivated. They can be grouped. They become reasoning instruments. Memes are thus already models, in most cases primitive or basic mental ones.

1.2 Why there is no Commonly Accepted Notion of a Conceptual Model: 1001 Notions and 101 Scenarios

Why the large variety of notions of model? Already [28] discusses 60 different notions of conceptual model. The variety of notions of model in CS&CE is far larger. Each of these notions concentrates on some aspects and implicitly assumes other properties. One reason for the disagreement on a common notion is the concentration on one utilisation scenario.

The implicit and hidden usage of deep models and the corresponding matrices of one – if not the main – cause for the manifold of model notions in CS&CE.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

May we develop a common understanding of the notion of model? The two main sources for the variety of notions allow a systematic harmonisation of model notions. The definition given above is a result of a discussion on models in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports at Kiel university that continues now for almost 10 years. The discussion is summarised in the compendium [30]. We got a shared understanding of the notion of model, of model activities and of modelling. So, we can envision that a common understanding and a coherent collection of notions of model can be developed. The collection supports a coherent notion that allows to concentrate on the specific utilisation scenario and the specific functions that a model has to fulfill.

1.3 The Storyline of the Paper and Our Agenda

Tasks and foundations of a theory of conceptual models. This paper bases the notion of conceptual models on four observations: (I) Conceptual models integrate concept(ion)s from a conceptualisation into a models. A notion of conceptual model might be a slim, light, or concise one depending on the level of detail we need in model utilisation. (II) A conceptualisation is based on a collection of concepts. (III) Origins of conceptual models are perception models and domain-situation models. (IV) These origins are formed by our understanding of the world, i.e. our observations and our compilations of these observations. We shall answer questions 2, 3, 6-8 from the abstract in the sections and use these answers for answering questions 1, 4, and 5 at the end.

We start with the last observation that leads us back to Ancient Greece. Next we develop an enhanced theory of concepts for an understanding of a conceptualisation. We may now define what are the components of perception and domain-situation models. Finally we arrive with three notions of conceptual model. We thus head forward to a science and culture of modelling in Figure 1.

Towards a science and culture of models, modelling activities, and modelling. Modelling is currently a creative art, i.e. a skill acquired by experience and observation, and may potentially be enhanced by study. The art extends daily life intelligence that is a part which lays the foundations for modelling, conditions, and socialises. The first level of modelling is based on daily life intelligence between humans or of humans with their environment. Humans become introduced to the deep model and especially the background – in most cases at some preconscious level.

Model science is based on a system of knowledge that is concerned with modelling art and that entails unbiased observations and systematic experimentation. The foundation we envision orients on fundamental laws. We understand, establish, deliberately apply the knowledge, and formalise it. Modelling culture is shared in a community of practice, is based on well-developed principles and methods as well as on established guidelines and practices. Wisdom requires the sapience of schools and matured modelling.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

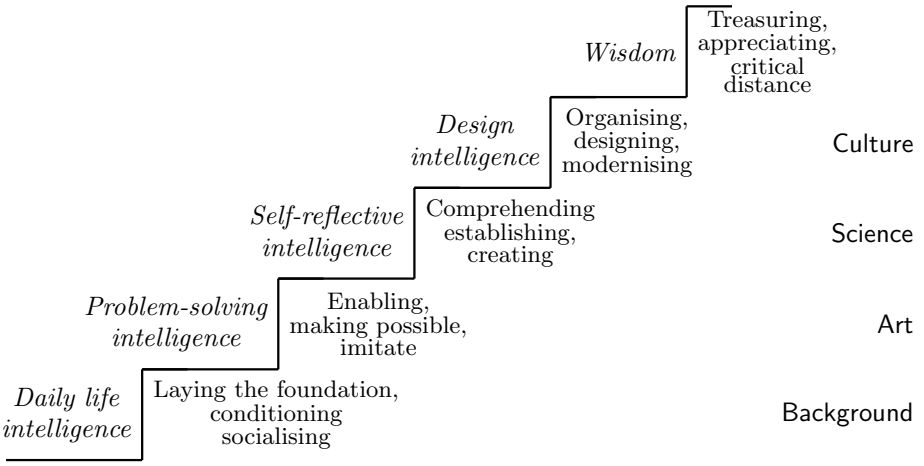

Fig. 1. The five levels of modelling as art, science, and culture

2 Model Theory and its Philosophical Foundations

The earliest source of systematic model consideration we know is Heraclitus [14] with his concept of $\lambda\acute{o}\gamma o\varsigma$ (logos). Explicit model development and model deployment is almost as old as the mankind, however. For instance, Ancient Egyptians already made sophisticated use of models [6]. So, essentially, it is an old subdiscipline of most natural sciences and engineering with a history of more than 5000 years [18]. The notion of model has not explicitly used at that historic time. It was, however, the basis of understanding, manipulation, and engineering.

2.1 Plato's three Analogies

Plato's Republica (for a survey on Politeia see [1] or Y. Lafrance) uses in the sixth book three analogies which can essentially be understood as the underpinning of the concept of model. We follow here the Lattmann's [13] investigation that led to a deep revision of the interpretation by Aristoteles.

The three analogies provide a general understanding of the model-being, of models, and of modelling. We may only observe phenomena of reality, form then trusts in beliefs and observations, next develop conjectures, might next judge and hypothise, and finally provide explanations.

The analogy of the sun distinguishes the 'good' visible world and the intelligible world. The sun stands for the visible things (i.e. 'empirical', 'physical', and 'material' world) and gives the light. The opposite worlds is the world of reasoning, of beliefs, conjectures, ideas, and explanations.

The analogy of the divided line distinguishes the visible world and the reasoning about this world as the thinkable (called intelligible world). The visible world can be separated into the physical things themselves (beliefs (pistis) about physical

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

things) and the reflections and observations about them (called shadows) (eikasia as the illusion of human experience). The intelligible world consists of (mathematical) reasoning and thought (dianoia) and of deep understanding (moesis).

Plato represented these four dimensions by a line (reflections(AB) - physical_things (BC) - thoughts(CD) - understanding(DE)). We shall see in the sequel that a four plane representation allows deeper understanding.
The analogy of the cave explains why humans can only interpret the world based on their observations, i.e. shadows that we can see. The reality cannot be observed.

2.2 Revisiting the Analogies for Understanding the Model World

The four segment presentation in Plato's analogies can be transferred to a four plane meta-model with

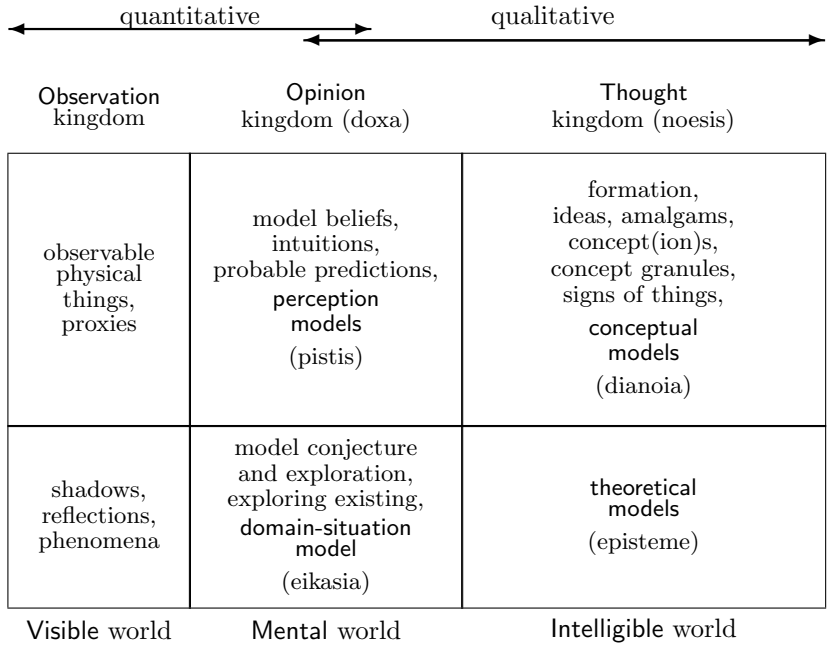

Fig. 2. The model world with the separation of concern into visible, mental, and intelligible worlds

- a separation into a quantitative area and a qualitative area for meaning and opinion (doxa) and a qualitative area for thoughts (noesis) from one dimension and
- a separation into a perception and pre-image area and an area for conceptualisation from the other dimension.

The intelligible world thus consists of a mental world and the real intelligible world. This observation allows us to reconsider the model-being in the approach depicted in Figure 2.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

The visible model world mainly reflects the observations and their perception as phenomena. So, we can consider models of the visible world as models of the first generation.

The mental model world consist already of compilations to perception models that reflect someone's understanding or domain-situation models that represent a commonly accepted understanding of a state of affairs within some application domain.

The intelligible model world includes conceptualisations and theory development.

3 Concepts and Conceptualisations

The separation of model worlds in Figure 2 provides a means to distinguish clearly between models and conceptual models. With this distinction we may now neglect the hypothesis [20] that any model is a conceptual model. We thus solved the demarkation problem for distinction of models into perception, domain-situation, conceptual, and theoretical models.

3.1 Conceptualisation

Conceptualisation is a reflection and understanding of the world on the basis of concepts from some commonly accepted concept spaces. Similar to ontologies, a conceptualisation is never unique. There is no conceptualisation such that every other one can be transformed from it. Conceptualisation means to find adequate concepts and conceptions for representation of a visible and mental world. It aims at the development of knowledge about these worlds. It is based on derivation of abstract concepts and experience, of (scientific) understanding and perception that can be applied in similar worlds, of (pragmatical) experience for modelling, and of reference models for model-driven development (MDD) approaches.

Weakening the Rigidity of Classical Concept Theory

The word 'conceptual' is linked to concepts and conceptions. Conceptual means that a thing, e.g., artifact is characterised by concepts or their conceptions. The word 'conceptional' associates a thing as being or of the nature of a notion or concept. Therefore, we distinguish the 'conceptual model' from 'conceptional modelling'. Classical concept theory and concept systems in mathematical logics are based on a Galois relationship between extensions and intentions of concepts, i.e. a concept is defined as a pair of an intention and of an extension where the intention is fully characterised by the extension and the extension is fully described by the intention. Each definition of a concept is a logical equation consisting of a definiendum and a definiens. We will use here an extension of the classical theory of concepts (e.g. [19]) by R. Kauppi's theory of concept properties [11, 26]).

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

Brentano, Bolzano and Twardowski (e.g. [4, 16]) distinguish three kinds of mental phenomena and inner consciousness: ideas, judgements, and volitions. Concepts and conceptions might be based on prototypes that allow to partially characterise the current understanding of the intention but do not provide a complete characterisation. Mental phenomena, beliefs, intuitions have their prototype view and a representation through best (counter-)examples that a person has been observing. Moreover, they can be represented by an exemplary view that characterises exemplars through similarity relation with measures, weights, and stimuli for their acceptance.

Conceptions are systems or networks of explanation. R.T. White [32] has already observed that concepts are not the same as conceptions. Concepts can be used in the meaning of classification and as an abstraction of a set of knowledge a person associates with the concept's name. Conceptions are however systems or networks of explanation. Conceptions are thus far more complex and difficult to define than the either meanings of the concept.

Conceptional modelling is modelling with associations to concepts. A conceptual model incorporates concepts into the model. Conceptual structures include conceptions (concepts, theoretical statements (axioms, laws, theorems, definitions), models, theories, and tools). Concepts are linked together in a complex multi-dimensional network (is-a-kind-of, is-a-part-of, ...). The links are of variable strength.

3.2 Concepts for Conceptualisation

An advanced concept notion must allow to define a concept in a variety of ways. Some definitions might be preferred over others. They can be application and time dependent, might have different level of rigidity, have their validity area, and can only be used with a number of restrictions. We combine R. Kauppi's theory of concept features with the concept treatment by G.L. Murphy [19].

The definition frame for concepts [23]: Concepts are given by tree-structured structural expression of the following form

ConceptTree(StructuralTreeExpression (Feature, Modality(Sufficiency, Necessity),
Fuzziness, Importance, Rigidity, GraduationWithinExpression, Category))) .

Features are elements of a concept with some modality, some Fuzziness, importance, rigidity, some graduation and some category. A feature is either a basic feature or is a concept.

Concepts are typically hierarchically ordered and can thus be layered. We assume that this ordering is strictly hierarchical and that the concept space can be depicted by a set of concept trees.

A concept might be given by several definitions. A concept is also dependent on the community that prefers this concept. Consider, for instance, the mathematical concept of a set by an enumeration of its elements, by inductive definition of its elements, by an algorithm for the construction of the set, or by explicit description of the properties of the set. Which of the definitions is more appropriate depends on the application domain.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

Interleaved meta-hypergraphs form hyper-networks of concepts: Our definition frame has the advantage that concepts which share features can be decomposed into the shared feature collection and the rest. Therefore, we may base our concept collection on a number of basic concepts.

The network of concepts is a meta-hypergraph [21]

$$MG = (MG^V, MG^{MV}, MG^E, F^{MG}, \Sigma^{MG}) \tag{1}$$

with a set of meta-hypergraph vertices MG^V, a set of meta-hypergraph meta-vertices MG^{MV} which are subsets of meta-hypergraph vertices, a set of meta-hypergraph edges MG^E connecting vertices. A vertex and an edge is described by a set of features F^{MG}. The semantic restrictions are given by Σ^{MG} . An example of a meta-hypergraph is displayed in Figure 3[2].

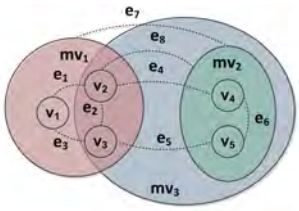

Fig. 3. A meta-hypergraph with vertices $v_1, ...v_5$, meta-vertices mv_1, mv_2, mv_3, and edges $e_1, ..., e_7$ without explicit features.

Conceptions can now be defined as a layered ensemble of meta-hypergraphs. We start with a primary network at layer 0 and associate next layer networks by embedding mappings to a hyper-simplex from networks at lower layer. A simple example is displayed in Figure 4.

3.3 Concept Granules as Basic Constructs of Conceptualisations

Concept granules are collections of concepts and/or conceptions given as meta-hypergraphs and ensembles with specified typicality of features (typical, moderately typical, atypical, borderline), with specified relevance of concept features, and with assigned importance of concept features.

Conceptualisation enhancements of a given model consist of
(1) a context given for various aspects in dependence on the matrix,
(2) a concept granule with several interrelated expressions as alternatives (competing, ...), with abstracts, with extensions (motivation, explanation, ...), and
(3) witnesses as collections of illustrating best (counter-)examples (potentially with several concept trees) mainly based on images/observations on origins.

[2] We acknowledge the communication with J. E. Gapanyuk from Bauman Moscow State Technical University (10.10.2018) who proposed this illustrations in Figures 3 and 4.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

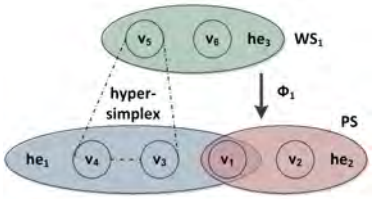

Fig. 4. A meta-hypergraph ensemble associating a simple primary network simplex PS and a first-order network simplex that associates via Φ_1 the vertex v_5 with a hyper-simplex of vertices v_4, v_3.

4 Conceptual Models

Mental models and their elements may be associated to concepts. The elements of a model are interpreted by concepts and conceptions. This interpretation is based on a judgement by somebody that conceive model elements as concept(ion)s within a certain scenario. If the scenario changes then the association to concepts changes as well. [28] categorises more than 50 notions of conceptual model depending on the function that a conceptual model has in a given scenario. We use this categorisation and develop now three integrated notions of conceptual model. Which one is used depends on the complexity of consideration.

4.1 Perception and Domain-Situation Models as Origins

Perception and domain-situation models [28] in Figure 2 are specific mental models either of one member or of the community of practice within one application area. It is not the real world or the reality what is represented in a perception model. It is the common consensus, world view and perception what is represented. Perception models are dependent on the observations, imaginations, and comprehension a human has made. Domain-situation models describe the understanding, observation, and perception of an application domain. The description is commonly accepted within a community of practice.

4.2 The Notion of Conceptual Model

The large variety of notions of conceptual model is caused by the scope of modelling, by the application case under consideration, by the main scenario in which the model functions, by the variety of origins that are represented by the conceptual model, by modelling languages, by the stand-alone orientation instead of integration into a model suite, and by the focus on normal models without mentioning the underpinning by a deep model. It is now our goal to consolidate three versions in such a way that they form a view depending on the level of detail and abstraction. The notions can be refined to an application domain,

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

e.g. to database modelling: "*A conceptual database model is a conceptual model that represents the structure and the integrity constraints of a database within a given database system environment.*" [29]

The Slim, Light, and Concise Notion of Conceptual Model

Slim version: Conceptual Model $\sqsupseteq$ *Model* $\uplus$ *Concept(ion)s* [29]: A conceptual model incorporates concepts into the model.

That means that models are enhanced by concepts from a number of concept(ion) spaces.

Light version: Conceptual Model $\sqsupseteq$ *Model* $\bigoplus$ *Concept(ion)s* [28]: A conceptual model is a concise and function-oriented model of a (or a number of) perception and/or domain-situation model(s) that uses a concept(ion) space.

This notion generalises and enhances a notion that is used in simulation research [22]: *"A conceptual model is a concise and precise consolidation of all goal-relevant structural and behavioural features of a system under investigation presented in a predefined format."*

Concise version: Conceptual Model $\sqsupseteq$ *(Model* $\bigoplus$ *Concept(ion)s)* $\bowtie$ *Enabler* [8]: A conceptual model is a model that is enhanced by concept(ion)s from a concept(ion) space, is formulated in a language that allows well-structured formulations, is based on mental/perception/situation models with their embedded concept(ion)s, and is oriented on a matrix that is commonly accepted.

The conceptual model of an information system consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema [31]. Conceptual modelling is either the activity of developing a conceptual model or the systematic and coherent collection of approaches to model, to utilise models, etc.

Literate programming [12] considers a central program together with it satellite programs, esp. for interfacing and documenting. This paradigm has become the basis for GitHub and model suites. Conceptual modelling is typically explicit modelling by a model suite. Association of conceptual and other models models in a model suite might follow the layered approach to model coherence maintenance and to co-evolution of models.

Descriptive and Prescriptive Conceptual Models.

A model functions in a number of scenarios. For instance, the conceptual model is used in documentation, negotiation, learning, communication, explanation, discovery, inspiration, modernisation, reflection, and experience propagation scenarios. We may categorise and enhance the notion of conceptual model depending on given scenarios. The system construction scenario integrates description and prescription scenarios beside specification and coding scenarios.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

One main scenario for conceptual database models is the description scenario. A conceptual model as a descriptive conceptual model is a deliverable of an understandable ((may be, ready to apply or to practise) and formalised (or well-formed) [concept-based], unconditionally acceptable conceptualisation of perception and domain-situation models for interaction and discourses.

For database applications it is thus a model suite consisting of a conceptual database model (or schema), of a collections of conceptual views for support of business users, and of a collection of commonly accepted domain-situation models with explicit associations to views (see [31]).

The second main scenario for conceptual database models is the prescription scenario. A conceptual model as a prescriptive conceptual model is a coding supporter as an analysed or synthesised, ready-to-apply blueprint because it can be deployed, it is unconditionally accepted, and appraised in a deliberately and precise practice as a tacit tool which provides notion explanations [from descriptive conceptual models].

For database applications it is thus a model suite consisting of a conceptual database model (or schema), of a collection of views for both support of business users and system operating, and of realisation templates (see [31]).

4.3 Models, Languages, and Ontologies

The major goal of an ontology [16] is to determine what exists and what not. It is independent of humans to conceive it and what kinds of existing things there are. It is independent of perception models although it can be shared among humans. Languages might be textual, visual or audio ones. The classical modelling approach often assumes artificial or partially formal languages.

Languages as enablers for conceptual models: Most models are language based. The language is an instrument similar to models. Moreover, the first models that a human develops are preconscious or subconscious, e.g. the model of a 'mother'. Languages are however enablers since the words in languages can be used for denoting concepts. Many conceptual languages integrate several languages, e.g. ER modelling uses the vocabulary from a domain and a graphical language for schema representation.

Conceptual models must not be based on an ontology: The notion of ontology is overloaded similar to the notion of model. Ontologies are considered as shared and commonly agreed vocabularies.

A controlled and thus matured ontology must combine a controlled vocabulary, a thesaurus, a dictionary, and a glossary. There is not real need for associating such ontologies with models.

Languages are not necessary preconditions for conceptual models: Social models are often used for teaching human behaviour. They are based on concepts which might also be not explicit or integrated into the deep model. They are thus conceptual models. We observe however that in most cases conceptual normal models use some language.

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

5 Conclusion

We developed an approach to conceptual modelling with an explicit integration of concepts into the model. This explicit integration is based on a theory of concepts, conceptions, and conceptualisation. Concepts are developed for our understanding of the world we observe. Therefore, perception and domain-situation models become the origins of our conceptual models.

There are models that are not conceptual models: Sciences and engineering use models without explicit integration of concepts. It is often also difficult to use concepts within the model. A model performs a function in a scenario. Explicit conceptualisation would make the model more complex and thus less useful.

What is not yet a conceptual model: Middle-range theories are essentially mediator models. They are used for mediation between qualitative theories (e.g. their conceptualisations) and quantitative observations. For instance, sciences such as archeology make use of modern or medieval concepts without having yet an appropriate concept for prehistoric time, e.g. the concept of settlement or a village. Another typical model that might be enhanced by concepts is the graph model for the Königsberg bridge problem that uses pathes within a graph for solving this problem. The topographical model for the bridge problem uses the concepts of islands and bridges and thus allows to explain the solution.

What will never be a conceptual model: Most life situations do not need conscious models since we can live with what we have learned. Preconscious, unconscious, and subconscious models guide life, emotions, and intuitions. Conscious models require efforts and thus must have an explicit need. Concept(ion)s must not be explicated since there might be no necessity in that.

References

1. Annas, J.: An Introduction to Plato's Republic. Oxford (1981)
2. Blackmore, S.: The Meme Machine. Oxford University Press, Oxford (1999)
3. Bosco, S., Braucher, L., Wiechec, M.: Encyclopedia Britannica, Ultimate Reference Suite. Merriam-Webster (2015)
4. Brentano, F.: Psychologie vom empirischen Standpunkte. Leipzig: Dunker & Humblot (1874)
5. Dahanayake, A., Thalheim, B.: Co-evolution of (information) system models. In: EMMSAD 2010. LNBIP, vol. 50, pp. 314–326. Springer (2010)
6. Deicher, S.: KunstModell in Ancient Egypt. BMBF Project description, University of Applied Sciences, Wismar (2018)
7. Embley, D., Thalheim, B. (eds.): The Handbook of Conceptual Modeling: Its Usage and Its Challenges. Springer (2011)
8. Jaakkola, H., Thalheim, B.: Cultures in information systems development. In: Information Modelling and Knowledge Bases XXX. pp. 61–80. IOS Press (2019)
9. Kangassalo, M.: Changes in children's conceptual models and the development of children's exploration strategies in the PICCO environment. In: Information Modelling and Knowledge Bases XI:. Frontiers in Artificial Intelligence and Applications, vol. 61, pp. 251–255. IOS Press (2000)

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

10. Kaschek, R.: Konzeptionelle Modellierung. Ph.D. thesis, University Klagenfurt (2003), Habilitationsschrift
11. Kauppi, R.: Einführung in die Theorie der Begriffssysteme. Acta Universitatis Tamperensis, Ser. A, Vol. 15, Tampereen yliopisto, Tampere (1967)
12. Knuth, D.E.: Literate programming. Comput. J. 27(2), 97–111 (1984)
13. Lattmann, C.: Vom Dreieck zu Pyramiden - Mathematische Modellierung bei Platon zwischen Thales und Euklid. Habilitation thesis, Kiel University, Kiel (2017)
14. Lebedev, A.: The Logos Heraclitus - A reconstruction of thoughts and words; full commented texts of fragments (In Russian). Nauka, Moskva (2014)
15. Mahr, B.: Information science and the logic of models. Software and System Modeling 8(3), 365–383 (2009)
16. Mahr, B.: Intentionality and modeling of conception. In: Judgements and Propositions - Logical, Linguistic and Cognitive Issues, pp. 61–87. Berlin (2010)
17. Mahr, B.: Modelle und ihre Befragbarkeit - Grundlagen einer allgemeinen Modelltheorie. Erwägen-Wissen-Ethik (EWE) Vol. 26, Issue 3, 329–342 (2015)
18. Müller, R.: Model history is culture history. From early man to cyberspace. http://www.muellerscience.com/ENGLISH/model.htm (2016), assessed Oct. 29,2017
19. Murphy, G.L.: The big book of concepts. MIT Press (2001)
20. Pastor, O.: Conceptual modeling of life: Beyond the homo sapiens. http://er2016.cs.titech.ac.jp/assets/slides/ER2016-keynote2-slides.pdf (2016), keynote given at ER'2016 (Nov. 15)
21. Popkov, G., Popkov, V.: A system of distributed data processing (In Russian). Vestnik Buryatskogo Gosudarstvennogo Universiteta (9), 174–181 (2013)
22. Robinson, S., Arbez, G., Birta, L., Tolk, A., Wagner, G.: Conceptual modeling: Defintion, purpose and benefits. In: Proc. of thew 2015 Winter Simulation School. pp. 2812–2826. IEEE (2015)
23. Schewe, K.D., Thalheim, B.: Semantics in data and knowledge bases. In: SDKB 2008. pp. 1–25. LNCS 4925, Springer, Berlin (2008)
24. Stachowiak, H.: Allgemeine Modelltheorie. Springer (1973)
25. Tanaka, Y.: Meme media and meme market architectures: Knowledge media for editing, distributing, and managing intellectual resources. J. Wiley, Hoboken (2003)
26. Thalheim, B.: The conceptual framework to user-oriented content management. In: Information Modelling and Knowledge Bases. Frontiers in Artificial Intelligence and Applications, vol. XVIII. IOS Press (2007)
27. Thalheim, B.: The conceptual model ≡ an adequate and dependable artifact enhanced by concepts. In: Information Modelling and Knowledge Bases. Frontiers in Artificial Intelligence and Applications, 260, vol. XXV, pp. 241–254. IOS Press (2014)
28. Thalheim, B.: Conceptual model notions - a matter of controversy; conceptual modelling and its lacunas. EMISA International Journal on Conceptual Modeling February, 9–27 (2018)
29. Thalheim, B.: Conceptual modeling foundations: The notion of a model in conceptual modeling. In: Encyclopedia of Database Systems. Springer US (2019)
30. Thalheim, B., Nissen, I. (eds.): Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung. De Gruyter, Boston (2015)
31. Thalheim, B., Tropmann-Frick, M.: The conception of the conceptual database model. In: ER 2015. pp. 603–611. LNCS 9381, Springer, Berlin (2015)
32. White, R.: Commentary: Conceptual and conceptional change. Learning and instruction 4, 117–121 (1994)

Proc. MEDI2019, LNCS 11815, pages 123--139. Springer, 2019.

Usage Models Mapped to Programs

András J. Molnár[1,2][0000−0002−2194−0320] and
Bernhard Thalheim[1][0000−0002−7909−7786]

[1] Christian-Albrechts-University Kiel, Computer Science Institute,
D-24098 Kiel, Germany
{ajm,thalheim}@is.informatik.uni-kiel.de
[2] MTA-SZTAKI Institute for Computer Science and Control,
Hungarian Academy of Sciences, H-1111 Budapest, Hungary
modras@ilab.sztaki.hu

Abstract. Model-based programming can replace classical programming based on compilation and systematic development of models as well on explicit consideration of all model components without hiding intrinsic details and assumptions. A key element of model-based programming is the proper definition and management of model suites, by which multiple, interrelated models can be transformed from one another and their consistency is ensured after modifications. A usage model is based on the specification of user roles and types, together with an interaction space described in a form of a storyboard, showing which activities are supported, in which order, by which actors. A workflow model is an extended, well-formed declaration of how specific processes should be carried out. It can directly be translated to program code, using a proper workflow or process engine. A novel way of programming is being opened up by usage modeling, which is being investigated in this paper: given a storyboard with supported usage scenarios, it is possible to derive a workflow model from it. We present our two translation methods using a working example, identifying guidelines as requirements for model refinement and normalization, rules for model translation, and propose considerations towards improved methods and model specifications.

Keywords: Model-centered programming · Model to program · Model suite · Model transformation · Storyboard · Process model.

1 Introduction

1.1 Programming by Modeling

Programming is nowadays a socio-technical practice in most disciplines of science and engineering. Software systems are often developed by non-programmers or non-computer scientists, without background knowledge and skills, or insight into the culture of computer science, without plans for systematic development. Maintenance, extension, porting, integration, evolution, migration, and modernisation become an obstacle and are already causing problems similar to the software crisis 1.0, since such systems often have a poor structure, architecture,

Proc. M2P -- New Trends in Database and Information Systems, Bled, CCIS 1064, pages 163-175, Springer, 2019.

documentation, with a lost insight of specific solutions. Programs of the future must be understandable by all involved parties and must support reasoning and controlled realisation and evolution at all levels of abstraction.

Our envisioned *true fifth generation programming* [13] is a new programming paradigm where models are essentially programs of next generation and models are translated to code in various third or fourth generation languages. Programming is done by model development, relying on the compilation of these models into the most appropriate environment.

Application engineers and scientists are going to develop and use models instead of old-style programming, supported by templates from their application area. They can thus concentrate on how to find a correct solution to their problems, managing the complexity of software intensive systems. The process will be supported by model-backed reasoning techniques, as developers will appreciate and properly evaluate the model suite at the desired level of abstraction.

1.2 Usage Models and Workflow Models

In our study we are considering the case of web information system development.

A *usage model* of a web-is consists of specification of user roles and types, their associated goals and tasks, and an interaction space. The latter can be expressed as a graph, called a *storyboard*, describing what activities are supported and in which possible order, by which actors [9]. Supported interaction playouts can be formulated as *scenarios* (exact graph paths), *story algebra expressions* (path scemata), or more generally, subgraphs of the storyboard, including actor-specific views. The usage model is developed by a global-as-design approach.

A *workflow model* is an extended, well-formed declaration of how specific processes should be carried out, in a notation that is readily understandable by all stakeholders, including business analysts, technical developers and people who manage and monitor those processes [7]. A de facto standard is *BPMN* [7], but it is possible to use another workflow description language. The workflow model can directly be translated to software process components, using a proper workflow or process engine (e.g. [3]). This opens up a novel way of programming by usage modeling, via intermediate translation to a workflow model.

1.3 Related Work

Our current contribution can be related – amongst others – to the following previous works. Notions of *models* are discussed in [12]. [11] introduces *model suites* consisting of multiple, explicitly associated models, where the association uses maintenance modes, similar to integrity support in databases [15]. Amongst others, MetaCASE tools [1] were developed to support the definition of metamodel packages and the creation and customization of CASE tools based on them. The *models as programs – true fifth generation programming* agenda is proposed in [13]. For data structuring, translation of entity-relationship models to relational database schemata is well-known [4]. We are proposing a similar approach for the dynamics of functionality, motivated by compilers [8]: phases of preprocessing, parsing and syntax checking is followed by semantic analysis resulting an intermediate structure, and finally a possible optimization phase of the resulting,

translated model. [10] discusses *proceses-driven applications* and *model-driven execution* in terms of BPMN [7] diagrams. [2] elaborates a generative approach to the functionality of interactive information systems. [14] introduces dynamically combinable *mini-stories* to handle workflow cases with large flexibility. Although these latter works consider steps and ideas we can apply here, our currently addressed problem of usage model translation to workflow model is not explicitly discussed in any of the publications known to us.

1.4 Goal and Outline of the Paper

Our general vision is to generate running program code based on a usage model specification. We investigate on a particular sub-case in this paper: given a usage model as a storyboard with supported scenarios [9], is there a formalizable method to derive a workflow model in BPMN [7] from it. We present our proposed path and general framework for modeling as next generation programming in Section 2, based on [13]. Section 3 introduces our target case of workflow model elicitation from a usage model, illustrated by a working example, with general guidelines for model refinement and enhancement, rules and two different methods for translation. We conclude and close with future issues in Section 4.

2 Modeling and Programming Based on Model Suites and Layering

Models are universal instruments for communication and other human activities. Ideas and thought chunks can be presented to those who share a similar culture and understanding without the pressure to be scientifically grounded. A model is an adequate (i.e. analogous, focused, purposeful) and dependable (i.e. justified, sufficient in quality) instrument that represents origins and performs functions in some deployment scenario [12]. As an instrument, the model has its own background (i.e. grounding, basis) and should be well-formed. Models are more abstract than programs, but can be as precise and appropriate as programs. They support understanding, construction of system components, communication, reflection, analysis, quality management, exploration, explanation, etc. Models can be translated to programs to a certain extent, therefore, models can be used as higher-level, abstract, and effective programs. They are, however, independent of programming languages and environments. Models encapsulate, represent and formulate ideas both as of something comprehended and as a plan. Models declare what exactly to build and can be understandable by all stakeholders involved in software system development. They become general and accurate enough, and can be calibrated to the degree of precision that is necessary for high quality [13].

A *model suite* [11] consists of a coherent collection of explicitly associated models. A model in the model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used as a program of next generation and will be mapped to programs in host languages of fourth or

Proc. M2P -- New Trends in Database and Information Systems, Bled, CCIS 1064, pages 163-175, Springer, 2019.

third generation. Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Consistency can be ensured similarly to relational databases [15]. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, changeable and adaptable at different layers, validatable and verifiable, and maintainable.

Similarly to database modeling, *layering* has already often and successfully been used, including most program language realisations and application development methodologies. We assume a general layered approach as the universal basis for treatment of models as programs [13]. Layering has also been the guiding paradigm of the TeX and LaTeX text processing realisations [5, 6].

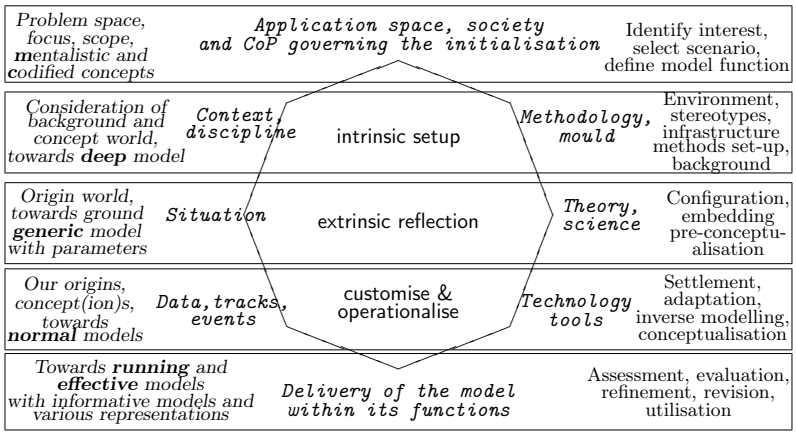

Fig. 1. The layered approach to model suite development and program generation

Model suite development and deployment will be based on separation of concern into *extrinsic* and *intrinsic* parts of models. Models typically consist on the one side of a *normal model* that displays all obviously relevant and important aspects of a model and on the other side of a *deep model* that intrinsically reflects commonly accepted intentions, the accepted understanding, the context, the background that is commonly accepted, and restrictions for the model. The model suite will be layered into models as shown in Fig. 1. Taking it as basis, we can formulate our proposed agenda for usage and workflow models.

The *initialisation layer* is given by the application and the scenarios in which models are used, by the problem characterisation, by background elements of the CoP and especially commonly accepted concepts in this community, and additionally by interest, intensions, and the value. In our case, it consists of a declaration that a website is needed for a specific application, analogously to selecting a *documentclass* in LaTeX. It determines the possible syntax and semantics of the underlying layers.

The *enabling strategic setup layer* defines the opportunity space and especially the hidden background for the model. Its main result is the deep model

that is typically assumed to be given (normal models are not entirely developed from scratch). In our case, it will correspond to what a website means, what are the side conditions and underlying infrastructure of it and the selected application domain. It gives an opportunity space and can impose requirements or proposals for the way of system development.

The *tactic definition layer* starts with some generalisation, i.e. select a ground generic model that will be customised and adapted to become the normal model. It can be, for example, a generalization of a previous storyboard development, or a configurable storyboard composed of best-practice patterns. Decision of the modeling framework or language (here, the use of storyboarding, with or without story algebra usage, in which format) must have been taken. Generic modeling must be supported by meta-models assumed to be available as (re)usable packages. Further model contents are interpreted based on the selected packages.

The *operational customisation layer* fits, calibrates and prunes the model suite to the problem space. This is where the actual design is made, forming a normal model (here: the generic storyboard is customized as needed or allowed by the generic model: missing parameters are set up, defaults can be overridden). Requirements for an acceptable normal model must have been given in the generic model or the metamodel, in order to ensure well-formedness and consistency, and to allow proper model transformations possible on the delivery layer. The normal model(s) must be validated according to these requirements.

Finally, the *model is delivered* in various variants depending on the interest and the viewpoints of the CoP members. It is elicitated from the normal model using a model translation, extraction or enhancement method. The target model language (here, BPMN) must be given with the selection and customization of the available translation methods. Interrelations and consistency management between the normal and the delivered model can be further declared.

The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built [13].

3 Elicitation of Workflow Models from Usage Models

One of the main challenges for model translation is the existence of different intentions behind the two modeling languages. BPMN is stricter than storyboarding, while a proper translation needs to make use of the inherent flexibility of the storyboard. Therefore, besides a direct and full translation option, we are proposing a way for constructing BPMN workflows by formulating story path schemata, based on selected parts of the storyboard.

3.1 An Application Case and Its Usage Model

We are assuming the development an information system for a touristic and recreational trail network, providing guidance for visitors, as well as facility management of the trails and related field assets. A map interface is being provided with planning and navigation features along the designated trails, connected to an issue tracking system for reporting and managing trail and asset defects.

The high-level usage model is given as a storyboard on Fig. 2. Abstract usage locations represented by graph nodes are called *scenes*. Users can navigate between these scenes, by performing actions associated to *directed transition links*. Some indicative *action names* are given for the reflexive links (which are in fact, denote multiple links, one by named action). The *entry point* is marked with a filled black circle. An *end point* may also be added as a double circle (by default, each scene is assumed to be a potential end point).

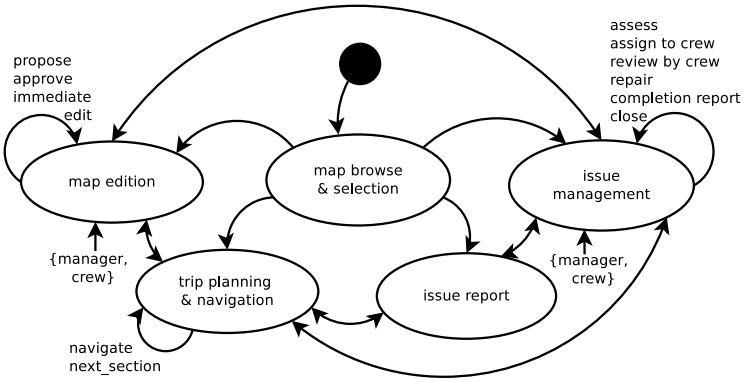

Fig. 2. High-level storyboard graph for a sample trail management system

We declare three *actor roles*: *visitor*, trail *manager* and trail *crew*. A set of authorized actors are pointed to the bottom of restricted scecnes by vertical arrows [9, p. 410]. By default, all actors are allowed to enter a scene.

The storyboard is about to represent supported *normal* scenarios, as specific means the users can accomplish given tasks. Context-loosing random navigations (e.g. back to the main page at any time) can be treated as breaking or canceling the started scenario, and starting a new scenario with a new context. These moves are not explicitly modeled so the focus can be kept on meaningful issues.

We are limiting our current discourse for one-session, one-actor scenarios.

The storyboard can be enhanced with input-output content specifications for each scene. We use the notation of [9, p. 410] so that input and output content for a scene is displayed using a short horizontal arrow on the left and the right side, respectively. Input-output content is named and an output content is assumed to be delivered as an input content to the next scene along each transition link, where the content names are equal. Square brackets denote optional input or output. Content names can be prefixed by generic database operation names.

3.2 Refinement and Normalization of the Storyboard

The top-level storyboard (Fig. 2) has to be refined and enhanced, so that actual scenarios as paths in the graph will be self-descriptive and consistent, and the graph is formally sound and contains enough details for a working and meaningful translation into workflow model(s). We state the following semantical

considerations and guidelines for developing the refined usage model. If all these criteria are met, and guidelines have considered, we call the storyboard *normalized*. This is only partially verifiable formally – for the items marked with (*) – and refers to a quality and stage of model development:

- Complex scenes must be decomposed into atomic sub-scenes, each having a single, well-defined action, task or activity which is fully authorized by a given set of actor roles. The interaction paths must be modeled by directed links between the sub-scenes and directly connected to outside (sub)scenes.
- Each transition link with active actor participation (action) must be replaced by a link-scene-link combination, where the action or activity is performed at the scene and the new links are only for navigation. This new scene can be handled and parametrized together with other scenes in a unified way.
- No parallel links between two scenes are allowed (*). They must either be translated using separate scenes (see above), or merged into one link, or their source or target scenes must be decomposed to separate sub-scenes.
- The routing decision (which link to follow after a scene) is assumed to be taken as part of the activity inside a scene, by default. If it is not intended, then only one outgoing link is allowed and an extra routing decision scene must be explicitly introduced after the original scene as necessary (this may be later optimized out).
- Unique names are assumed for all scenes and links (except that two or more links pointing to the same target scene can have the same name) (*).
- There must be a unique start node (entry point) with a single link to an initial scene and either a unique end node or a default rule declaring which scenes can be places for story completion (*).
- Each scene must be enhanced with a set of authorized actor roles. Without that, a default rule must be supplied. There must be no (normal) links between scenes without at least one common authorized actor role. (*)
- Input and output content is to be specified by symbolic names for each scene wherever applicable. Content names will be matched along the links (*): For each input content of a scene s there must be an output content with the same name provided by the source scene of each link directed to s. Optional content is written in square brackets.
- Input and output content names can be prefixed by database operations: SELECT is allowed for input, while INSERT, UPDATE, and DELETE are allowed for output content. Without detailed semantics of these operations, a single central application database is assumed by default.

3.3 View Generation by Actor Roles

Given a selected actor role, a specific storyboard view can be generated for it as a basis of role-specific workflow models, by removing unauthorized scenes for a selected role with their links, resulting a cut-out of the storyboard, with reachable scenes by actors of the chosen role. An enhanced, normalized version of the visitors' storyboard view is shown on Fig. 3, with multiple sub-scenes. Links are denoted by italic numbers. An explicit end node is placed additionally, reachable from chosen scenes.

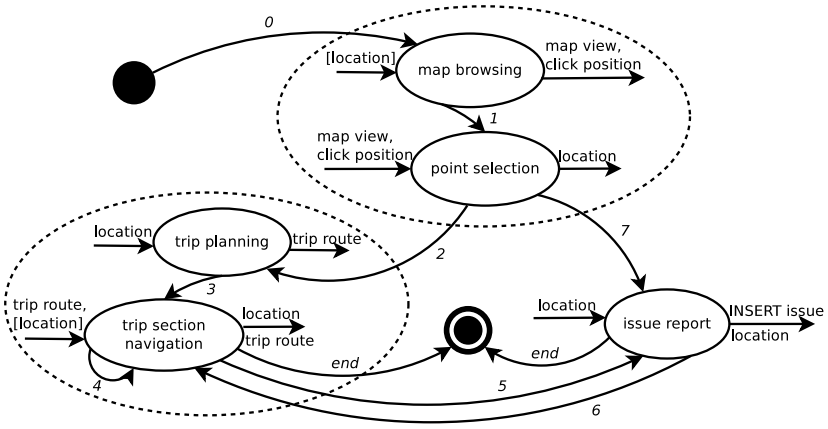

Fig. 3. Visitors' view of the storyboard after refinement and normalization

3.4 Graph-Based, Direct Translation Method

At this point, a default translation algorithm we have developed, can be applied to generate a BPMN process flow diagram, based on the graph connectivity of the storyboard. Details of the algorithm are omitted due to space limitations, but the result of the translation of Fig.3 is shown on Fig. 4 as a demonstrative example. The translation process can continue with enhancements of Section 3.8.

3.5 Modeling Supported Scenarios by Story Algebra Expressions

Alternatively to the previous method, a more sophisticated and targeted method is developed, if specific scenarios, which are intended to be supported by the system, are collected and expressed as patterns in a story algebra.

A particular playout of system usage becomes a path in the storyboard and is called a *scenario*. A set of possible scenarios can be modeled as using the *story algebra SiteLang* [9, p. 76], similar to regular expressions. Such a *scenario schema* can be a pattern for generating a workflow model. The original notation uses link names for description. We found that using scene names in the story algebra more naturally supports the translation to workflow models.

For example, a scenario schema of a visitor can be modeled out of the following variations: a visitor looks at the map, selects a destination point. The scenario may continue by reporting an issue for the selected point, or by planning a trip, navigating along it, and maybe at certain points, reporting an issue on-site. Each of these variants correspond to different scenarios the system should support and can be summarized as one or more scenario schemata.

Using abbreviated scene names (by first letters of words, e.g. *mb* stands for *map browse*), the above mentioned visitor scenarios can be modeled by the following story algebra expression (semicolon is used for denoting sequential steps, plus sign for at-least-once iteration, square brackets for optionality and box for expressing alternatives):

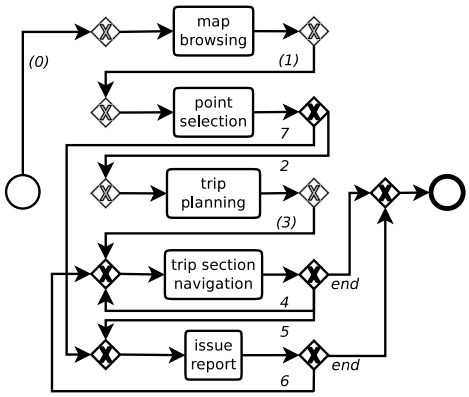

Fig. 4. Direct translation of visitor usage to BPMN, based on the storyboard graph. The full connectivity of the usage model is represented as possible process flow paths. Scenes become tasks. Numbers denote choices based on scene transition edges. Bracketed numbers are only for information, referring to original transition edges without alternatives. The grey-colored gates can be removed by merging their connections. Further refinements and optimizations are possible.

$$mb; ps; (ir \square (tp; (tsn; [ir])^{+})) \tag{1}$$

Given a storyboard (view) specification, a scenario schema must be *compatible* with the given scene transitions, which means the following: Atoms of the story algebra expression must match to authorized scenes of the storyboard. The defined scenarios must correspond to valid directed paths within the storyboard (view). The defined scenarios must start with the marked initial scene and finish at the defined (or default) end scene(s).

Expression (1) is compatible with the visitors' storyboard view (Fig. 3). Consistency of the input-output content declarations can also be checked along the possible playouts. Note the link *6* will not be available if the visitor is coming from link *7* (there is no navigated route to go back to).

3.6 Decomposition Into Mini-Stories

A scenario or story schema might contain semantically meaningful, reusable patterns of scene transition playouts, which can be combined with each other flexibly. Story algebra expressions, however, may be too complex and hard to handle by human modelers, and such semantical information remains hidden. A possible solution is to take the union of the relevant scenarios and decompose them into *mini stories* [14] (or, at least, extract some mini-stories from it).

A *mini-story* is a semantically meaningful, self-contained unit, which can be used flexibly in different scenarios, sometimes by possibly different actors.It can be modeled explicitly and translated as a reusable subprocess in the workflow model. Syntactic hints or heuristics can reveal possible mini-story candidates, but at the end the modeler has to explicitly define or verify them.

In our case, given the story algebra expression (1), candidate mini-stories can be recognized by maximal, non-atomic subexpressions with none of its non-trivial parts appearing elsewhere. Based on modeler decision taking into account semantics as well, we define the following two mini-stories, and substitute them in the story algebra expression (in a real case, with more scenarios, their reusability could be better verified): 1. *Select location from map*: $Slfm ::= mb; ps$ and 2. *Navigate along trip (with reporting issues)*: $Nat ::= (tsn; [ir])^+$. We keep referring to scenes *ir* and *tp* as atomic mini-stories. The resulting story algebra expression with the above mini-story substitutions of (1) becomes:

$$Slfm; (ir \square (tp; Nat)) \tag{2}$$

3.7 The Story-Based Translation Method

After the storyboard (viewed by an actor role, refined and normalized) and the desired story schemata (story algebra expresions) are given as above, with the mini-stories modeled, the workflow model in BPMN for each story schema can be elicitated the following, inductive way:

- Translate atomic mini-stories,
- Translate compound mini-stories based on their story algebra expressions (which are not translated yet),
- Compose the complex workflow based on the full story algebra expression.

Translation can be hierarchically carried over using structural recursion along the story algebra atoms and connectives, as displayed on Fig. 5. A choice for rule alternatives is proposed, with given defaults. The modeler can either leave the defaults as they are, or utilize the alternatives by applying pragma-like declarations to the usage model, or stereotypes associated to story algebra elements or subexpressions. For example, compund mini-stories can be translated as subprocesses, or connected using the link event notation. Conditionals for process flow gates match the names of correspoinding storyboard edges (based on user choice) or their associated conditions or triggers (if such conditions are given for links of the storyboard).

3.8 Enhancement of the Translated Model

Transition link names (here, numbers) can be added to the workflow model, as well as input-output content as data objects and database connections associated to the workflow tasks and subprocesses (see the additional rules of Fig. 5).

Fig. 6 displays a result of the refined, normalized visitors' storyboard view (Fig. 3) being translated to BPMN, based on story algebra expression (2) and mini-stories of Section 3.6, using rules of Fig. 5.

Model translation may be guided by additional information in forms of scene or link stereotypes.BPMN provides a variety of assets and some of them could be directly elicitated.Stereotypes offer more semantic information such as data or user-driven navigation, cancellation or rollback of started transactions, etc., to be mapped to native BPMN constructs.

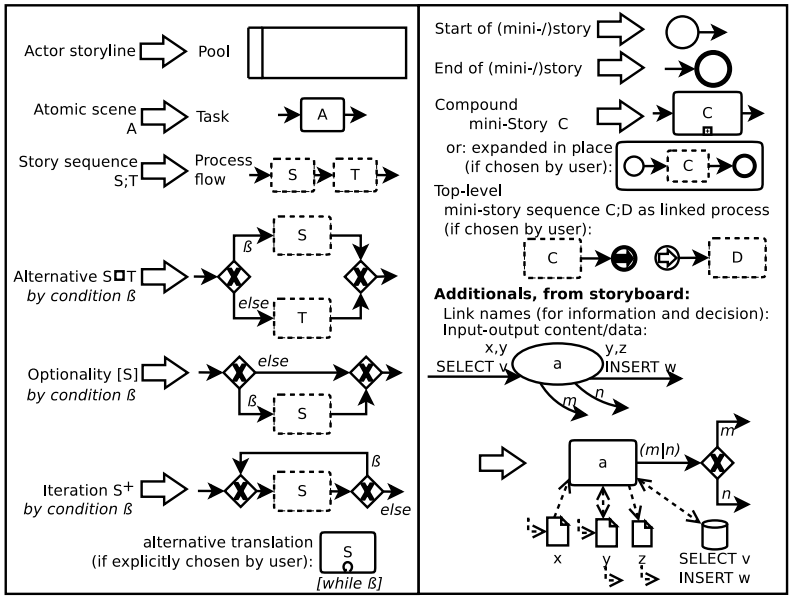

Fig. 5. Translation rules for story algebra expressions and additional assets based on storyboard. Dotted-lined rectangles denote arbitrary workflow model parts already translated from story subexpressions. There is a default translation for each construct, with possible alternatives that can explicitly be chosen by the modeler.

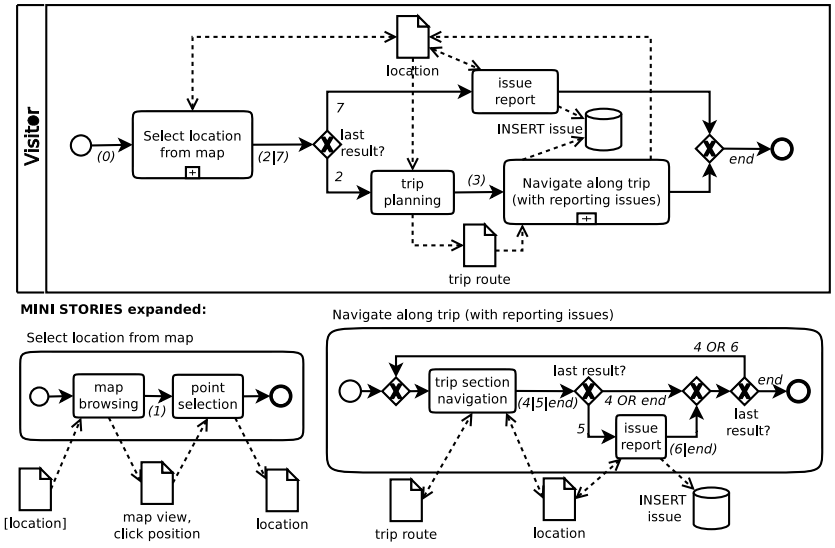

Fig. 6. BPMN workflow translation of visitors' view usage model, based on story algebra expression (2)

The modeling process is based on laying out default values for model formats, start/end scenes, authorized actor roles, context objects containing scenario history, handling of exceptions and invalid routing, stereotypes and other semantical or transformative guidance (e.g. how to connect mini-stories together, how to translate iteracted sub-processes). Defaults should work for conventional modeling cases. For customized, more sophisticated modeling, defaults can be overwritten. A possible post-translation optimization phase can improve the workflow model in each case. Most of these issues are left for future investigation.

4 Conclusion and Future Work

Model-based programming can be the true fifth generation programming, supported by sound foundation and appropriate tools, based on model suites of explicitly interrelated models. Models have their specific functions, viewpoints and can be given in various levels of details. Ensuring coherence, consistence and translatability among them is a crucial issue. In this paper, we have presented a general, layered modeling framework as a basis, and showed its feasibility by giving methods and guidelines for model development and translation between two specific types of models: the usage model (expressed by storyboard graphs and story algebra expressions) and the workflow model (expressed by BPMN).

The workflow model is claimed to be directly translatable to program code [10, 3]. We have introduced the concept of user view and the normalization of the storyboard, providing guidelines to the modeler to refine an initial, top-level usage model. We gave two methods for translating the refined usage model to workflow models, and successfully applied the mini-story concept for semantically structured and flexible workflow elicitation. Translation is based on default rules, while alternatives can be chosen explicitly by the modeler.

The method is ready to be tested with more examples or prototype implementations. Future issues include actor collaboration modeling, defining stereotypes and pragmas determining model semantics and translations. The metamodeler has to implement packages of generic models and add-ons, enrich generic models with pre-defined patterns and templates. The actual application modeler can choose among them or let the modeling system decide on which defaults it uses for which cases. It points towards a generic model-suite framework, which is, in our view, essential for truly working general model-based programming.

References

1. Alderson, A.: Meta-case technology. In: European Symposium on Software Development Environments. pp. 81–91. Springer (1991)
2. Bienemann, A.: A generative approach to functionality of interactive information systems. Ph.D. thesis, CAU Kiel, Dept. of Computer Science (2008)
3. Camunda: The *Camunda BPM* manual. https://docs.camunda.org/manual/7.10/, accessed: 2019-05-17
4. Chen, P.: Entity-relationship modeling: Historical events, future trends, and lessons learned. In: Software pioneers. pp. 296–310. Springer (2002)

5. Knuth, D.E.: The METAFONTbook. Addison-Wesley (1986)
6. Lamport, L.: LaTeX: A document preparation system. Addison-Wesley (1994)
7. OMG: Business process model and notation (*BPMN*) version 2.0 (2010)
8. Pittman, T., Peters, J.: The Art of Compiler Design: Theory and Practice. Prentice Hall, Upper Saddle River (1992)
9. Schewe, K., Thalheim, B.: Design and Development of Web Information Systems. Springer (2019)
10. Stiehl, V.: Process-Driven Applications with BPMN. Springer (2014)
11. Thalheim, B.: Model suites for multi-layered database modelling. In: Information Modelling and Knowledge Bases XXI, volume 206 of Frontiers in Artificial Intelligence and Applications. pp. 116–134. IOS Press (2010)
12. Thalheim, B.: Normal models and their modelling matrix. In: Models: Concepts, Theory, Logic, Reasoning, and Semantics, Tributes. pp. 44–72. College Publications (2018)
13. Thalheim, B., Jaakkola, H.: Models as programs: The envisioned and principal key to true fifth generation programming. In: 29th International Conference on Information Modelling and Knowledge Bases. IOS Press (2019)
14. Tropmann, M., Thalheim, B.: Mini story composition for generic workflows in support of disaster management. In: DEXA 2013. pp. 36–40. IEEE Computer Society (2013)
15. Türker, C., Gertz, M.: Semantic integrity support in *SQL:1999* and commercial (object-)relational database management systems. The VLDB Journal **10**(4), 241–269 (2001)

Models: The Main Tool of True Fifth Generation Programming

Igor Fiodorov[1] [ORCID number ???], Alexander Sotnikov[2][ORCID number???], and Bernhard Thalheim [3] [0000-0002-7909-7786]

[1] Plekhanov Russian University of Economics, Russia
[2] Joint Supercomputer Center, Russian Academy of Sciences, Russia
[3] Christian-Albrechts-University Kiel, Computer Science Department, Germany

Igor.Fiodorov@mail.ru,ASotnikov@jscc.ru,
thalheim@is.informatik.uni-kiel.de

Abstract. Models are one of the main and most commonly used instruments in Computer Science and Computer Engineering. They have reached a maturity for deployment as the main tool for description, prescription, and system specification. They can be directly translated to code what enables us to consider models as the main tool for modern software development. Models are the power unit towards new programming paradigms such as true fifth generation programming. This paper introduces model-centered programming as one of the main ingredients and main tool of true fifth generation programming.

Keywords: models, true fifth generation programming, model-centered programming.

1 Introduction

1.1 Towards New Programming Paradigms

Programming has become a technique for everybody, especially for non-computer scientists. Programs became an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Solution development for real life complex systems becomes however an obstacle course due to the huge variety of languages and frameworks used, due to impedance mismatches among libraries and environments, due to vanishing programming expert knowledge, due to novel and partially understood paradigms such as componentization and app programming, due to the inherent tremendous complexity, due to programming-in-the-large and programming-in-the-web, and due to legacy and integration problems.

Programming languages have evolved since early 1950's. This evolution has resulted in a thousand of different languages being invented and used in the industry. First generation languages – although low-level and machine-oriented at micro-code

Proceedings of the XXII International Conference
«Enterprise Engineering and Knowledge Management" (EE & KM'2019),
April 25-26, 2019, Moscow, Russia

level - are still used for instruction-based programming. Second generation languages are assembly languages that can be translated to machine language by an assembler. Third-generation languages provide abstractions and features such as modules, variables, flow constructs, error handling, support packages, many different kinds of statements etc. Fourth generation languages are more user friendly, are portable and independent of operating systems, are usable by non-programmers, and have intelligent default. The fifth generation project has been oriented on logic programming and did not result in a wide acceptance and usage. The main supporting feature for programming is however that programs written in these languages are translated by compilers to programs in low-level machine languages.

Programs became an infrastructure of the modern society. At the same time, we face a lot of problem for such infrastructure. Its maintenance, extension, porting, integration, evolution, migration, and modernization become an obstacle and are already causing problems similar to the software crisis 1.0. Programs are developed in a variety of infrastructures and languages that are partially incompatible, in teams with members who do not entirely share paradigms and background knowledge, at a longer period of time without considering legacy problems at a later point of time, without development strategies and tactics, and with a focus on currently urgent issues. A crucial point is the development of critical software by non-professionals. Programming has already changed to programming-in-the-large beyond programming-in-the-small and is going to change now to programming-in-the-mind. Moreover, systems become more complex and less and less understandable by team members. The software crisis 2.0 (e.g. [15]) is also be exacerbated by understandability, communication, comprehension, complexity, and provenance problems.

We thus need better and more abstract techniques for development of our programs. We envision that ***true fifth generation programming*** can be based on ***models and model suites*** which can be automatically transformed to corresponding programs without additional programming.

1.2 Models as Programs

Our notion and understanding of models and model suites is based on the compendium on models in sciences and engineering [11].

A *model* is a well-formed, adequate and dependable instrument that effectively and successfully functions in utilization scenarios. It is *adequate* if it is analogous to the origins to be represented according to some analogy criterion, if is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and if it sufficiently satisfies its purpose. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is sufficient by its quality characterization (internal quality, external quality and quality in) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed

instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics. A model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form.

A model reflects only some focus and scope. We thus use *model suites* that consists of a set of models, an explicit association or collaboration schema among the models, controllers that maintain consistency or coherence of the model suite, application schemata for explicit maintenance and evolution of the model suite, and tracers for the establishment of the coherence.

A typical model suite is used for co-design of information systems that is based on models for structuring, models for functionality, models for interactivity, and models for distribution. This model suite uses the structure model as the lead model for functionality specification. Views are based on both models. They are one kernel element for interactivity specification. Distribution models are additionally based on collaboration models.

Model-centered development is used in many branches of modern computer science and computer engineering. Model-as-Programs approaches will become the traction machine characterized by slow beginning at present and a progressive increase in speed. In the sequel we discuss this change of paradigms for database development. A similar approach has already been practiced for editing systems such as literate programming and as the LaTeX environment or such as compiler-compiler approaches for domain-specific languages.

1.3 The Storyline of this Paper

Models and model suites became easy-to-use and easy-to-develop instruments that are used by everybody and therefore also by non-programmers. We envision that modern programming could be based on model suites that are translated to programs. As discussed in Section 2, this vision is already real for users that use advanced database development techniques. However, model-based database programming is used only in a less sophisticated and rather implicit form. Investigating the more advanced approach, we develop a path towards true fifth generation programming that is based on model suite development in Section 3. The entire framework is inspired by and can be considered as a generalization model-centric database development and modern specification approaches.

2 Case Study: Model Suites Direct Database Specifications

2.1 Data Specification for Database Applications with Conceptual Models

Conceptual schemata and models are widely used for database structure specification and as a means for derivation of user viewpoints [10,13]. These models are used for concept-backed description of the application domain or of thoughts, for prescription of the realization and thus system construction, for negotiation and iterative develop-

ment of the model decisions, and for documentation and explanation of the decisions made in the modelling process.

Viewpoints can be represented by view schemata that are defined on the main database schema by expressions given in an advanced algebra. Interaction models for business users are the third kind of models that are used in a database model suite. Collaboration models can be specified in a similar form and are based on viewpoints.

The usage of a conceptual model as a description model of thoughts and understanding in an application area is commonsense today. The usage for system realization must be based on specific properties of the database management platform and requires thus a lot of additional information. We thus enhance conceptual modelling by additional information. Pragmas and directives are essential elements that we use for enhancement of conceptual models for system realization. Pragmas have originally introduced for C and C++. Directives have been used as additional control units for compilation.

2.2 Transformation of Conceptual Models to Logical and Physical Models

Conceptual models and schemata are often taken as an initial structure for logical and physical schemata. The transformation is still often based on some brute-force interpreter approach that requires corrective specification for integrity maintenance and for performance management at a later stage by experienced database operators. The transformation of integrity constraints is not yet automatically enhanced by enforcement mechanisms and control techniques. Procedural enhancement on the basis of triggers and stored procedures is still a challenge for database programmers. Performance support includes at the first step CRUD supporting indexing. Support for querying can be based on hints.

The transformation approach can however be based on rule-based compilation. Essentials of rule-based transformation are in a nutshell: syntactical and semantic analysis of the models and schemata according quality characteristics within the platform setting; preprocessing of the models and schemata to intermediate normalized models and schemata; extension of the models by support models for performance support; derivation of integrity maintenance and other support schemes; derivation of association schemata for models in the model suite; derivation of tracers for coherence maintenance; rule-based transformation of models; optimization after transformation.

It does not surprise that this approach follows classical four-layer compiler technologies (lexical and syntactical analysis, derivation of intermediate models, preparation for optimization, translation, performance management) [14]. It is enhanced by a compiler configuration pragmas according to the profile of the DBMS. The models must be complete for performance consideration. Therefore, a number of directives have to be added to all models in the model suite: treatment of hierarchies, redundancy control, constraint treatment, realization conventions, and quantity matrices for all larger classes. Directives and pragmas must not be fully described. Instead we may use templates, defaults and stereotypes, e.g. realization style and tactics, default configuration parameters (coding, services, policies, handlers), generic operations, hints

for realization of the database, strategies for matching performance expectations, constraint enforcement policies, and support features for the system realization.

This transformation approach is already state-of-the-art for challenging applications. Advanced database programming is based on such techniques. Web information systems development uses such transformations [10]. However, it is currently the professional secret of database operators and administrators.

2.3 Generalizing the Approach for Database Programming

The specification and transformation approach has already becoming common practice for database structuring development. The Higher-Order Entity-Relationship Modelling (HERM) language [13] is the basis for development of conceptual database schemata and for specification of derivable view schemata. The latter are used for support of viewpoints for a given database system user community. Derivation is based on the HERM algebra that allows specification of user schemata. The specification of a database schema follows the disciplinary matrix of database development, e.g. approaches such as global-as-design and viewpoint support as derivable structures. This structuring may be enhanced by generic or reference models which are essentially package for modelling. The foundation and the models background is supported by the HERM theory. The entire schemata development is based on tools, e.g. ADOxx [4,6] as a specification environment. An essential element of this environment is a compiler for compilation of the specifications to logical schemata. The database developer specifies the schemata within this environment as HERM schemata, view schemata, and pragmas and directives as an additional description for database performance.

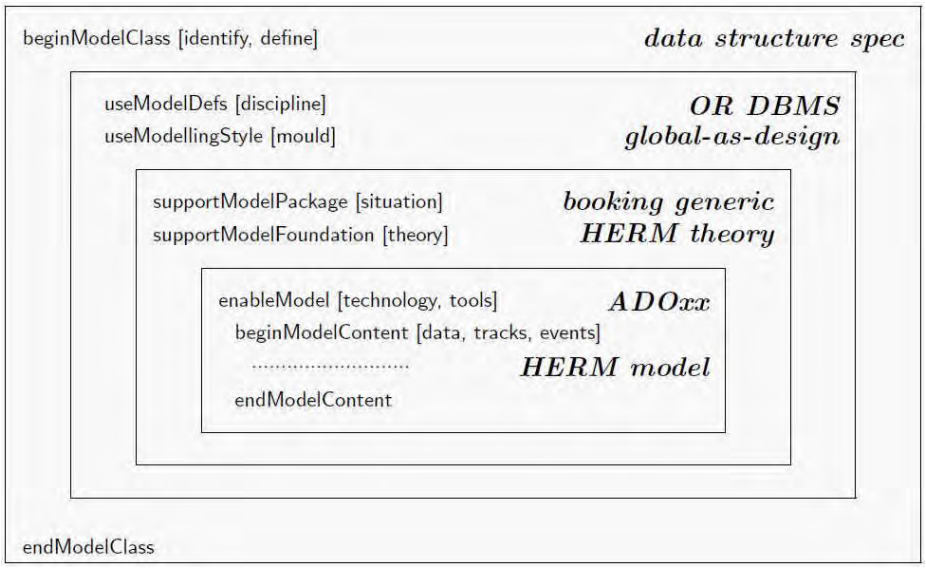

Fig. 1. The model-centric database structure development with automatic mapping of conceptual models to logical models for HERM models

Fig. 1 displays this approach. The database developer uses the development environment within the environment of the ADOxx workbench, the model definitions provided for HERM, the packages for schemata (e.g. a generic reference model for booking applications), the foundations provided by the HERM theory, and the transformation features embedded into the ADOxx generator [6].

3 Model Suites Used As Programs

3.1 Towards New Programming Paradigms

Modern programming languages provide as much as possible comfort to programmers.

Models are a universal instrument for communication and other human activities. Thought chunks can be presented to those who share a similar culture and understanding without the pressure to be scientifically grounded. Models encapsulate, represent and formulate ideas both as of something comprehended and as a plan. They are more abstract than programs. They can be as precise and appropriate as computer programs. They support understanding, construction of system components, communication, reflection, analysis, quality management, exploration, explanation, etc. From the other side, models can be translated to programs to a certain extent. So, models can be used as higher-level, abstract, and effective programs. Models are however independent of concrete programming languages and environments, i.e. programming language and environment independence is achieved. Models declare what exactly to build. They can be developed to be understandable by all main parties involved in system development. They become general enough and accurate enough. They can be calibrated to the degree of precision that is necessary for high quality.

Model-based programming can then replace classical programming based on compilation and systematic development of models as well on explicit consideration of all model components without hiding intrinsic details and assumptions. Our approach to true fifth generation programming will be extendable to all areas of computer science and engineering beside the chosen four exemplary ones (information system models; horizontally and vertical layered models; adaptable and evolving models; service line models). This paper develops a general framework to true fifth generation programming for everybody.

The framework is based on model suites since the user interface models and the collaboration models must be an integral part of modelling. Interface and collaboration treatment generalizes literate programming [5] to literate modelling as ‘holon’ programming that is combined with schemes of cognitive reasoning. Model suites enable the programmer of the future to develop their programs in a multi-facetted way. They can reason in a coherent and holistic way at the same time on representation models as the new interfaces, on computing and supporting models, on infrastructure models, on mediating models for integration with other systems, etc.

Application engineers and scientists are going to develop and to use models instead of programming in the old style. They will be supported by templates from their application area, can thus concentrate on how to find a correct solution to their prob-

lems, can manage the complexity of software intensive systems, will be supported by model-backed reasoning techniques, and will appreciate and properly evaluate the model suite at their level of abstraction. Literate modelling with model suites supports all members of a community of practice (CoP) by reflecting their needs and demands in a given situation and scenario by an appropriate model in the model suite. It becomes thus an effective and efficient means of communication and interaction for users depending on their beliefs, desires, needs, and intentions.

The generalization of the database approach is depicted in Fig. 2. We use a similar form as the experienced one that is displayed in Fig. 1.

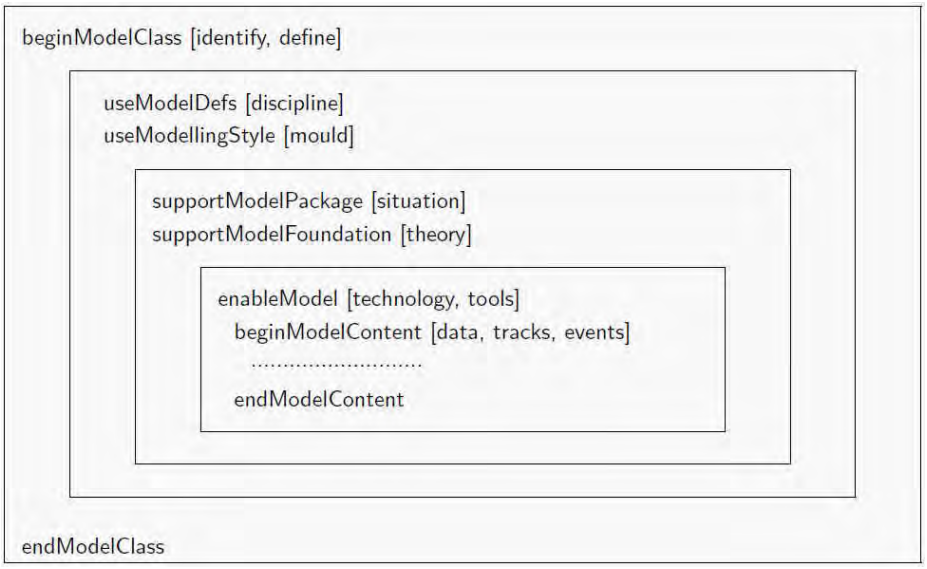

Fig. 2. The general approach to true fifth generation development based on models

Our approach proposes new programming paradigms, develops novel solutions to problem solving, integrates model-based and model-backed work into current approaches, and intents to incubate true fifth generation programming. This new kind of programming enhances human capabilities and could become the kernel of new industrial developments. Models are thus programs of the next generation.

3.2 The Layered Approach to Modelling

Our approach is based on model suites as the source, on systematic development of model suites in a layered approach, on compilers for transformation to programs in third or fourth generation, and on quality assurance for the model as a program. The notion of the model suite is based on [11]. Model suites generalize approaches developed for model-driven development from one side and conceptual-model programming from the other side. Model suite development and deployment will be based on separation of concern into intrinsic and extrinsic parts of models. Models typically

consist on the one side of a normal model that displays all obviously relevant and important aspects of a model and on the other side of a deep model that intrinsically reflects commonly accepted intentions, the accepted understanding, the context, the background that is commonly accepted, and restrictions for the model. The model suite will be layered into models for initialization, for strategic setup, for tactic definition, for operational adaptation, and for model delivery (see Fig. 3).
Model development can be layered in a form that is similar to the onion structure in Figures 1 and 2. We use essentially five layers for true fifth generation programming as shown in Figure 3:

(1) an internal layer for general initialization,
(2) an application definition language layer that includes many additional library packages,
(3) the internal supporting and generated layer with its generic and reference libraries,
(4) the input model suite that reflects the application and which is essentially the main task for an application engineer, and
(5) the generic intermediate output layer, and its delivery layer for multiple output variants depending on the target programming language.

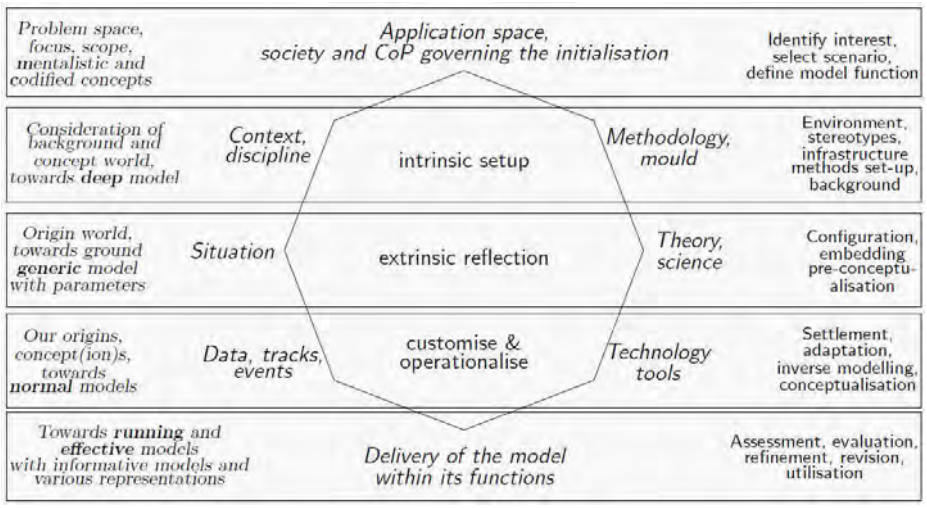

Fig. 3. The five layers to model development (initialize, setup, reflection, customize, delivery)

The model suite will be layered into models for initialization, strategic setup as an intrinsic setup, tactic definition as an extrinsic reflection, customization and operationalization as the main program development layer and as operational adaptation, and for model delivery. The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built. Currently, a model is considered to be the final product. Models have their own background that is typically not given explicitly but intrinsically. Currently, methods for developing and utilizing

models are accepted as to be given. The intrinsic part of a model and these methods form is called deep sub-model. The deep model is coupled with methodologies and with moulds that govern how to develop and to utilize a model. The deep as well as the general model are starting points for developing the extrinsic or "normal'' part of a model. Consideration of modelling is often only restricted to normal models similar to normal science. Model suites integrate however these model kinds. The main obstacle why model-driven development and of conceptual-model programming has not yet succeed is the non-consideration of the deep model and of modelling moulds.

4 Conclusion

We envision that true fifth generation programming can be based on development of high-level program descriptions that can be mapped to third-generation or fourth-generation programs. These programs may then be directly executed within the corresponding environment. This approach has already been the essential idea and its generalization behind a system for translation of domain-specific languages in the 80ies. The DEPOT-MS (DrEsdner PrOgrammTransformation) [7] was a compiler-compiler for domain-specific languages (historically: little languages, application-domain languages (Fachsprache)) that has been used to compile specific language programs to executable programs in the mediator language (first BESM6/ALGOL, later PASCAL, finally PL/1 [2]). The approach integrates the multi-language approach [1], the theory of attribute grammars [9], and theory of grammars [3,12].

A second source for true fifth generation programming is literate programming [5] that considered a central program together with satellite programs, especially for interfacing and documenting. This approach can be generalized and extended by new paradigms of programming (e.g. GibHub, 'holon' programming, schemata of cognitive semantics, and projects like the Axiom project or the mathematical problem solver [8] have already shown the real potential of literate programming. Our approach extends literate programming to model suites which are sets of models with well-specified and maintainable associations.

The developed framework, its theoretical underpinning and the realization approach is novel, targets at new programming styles, supports programmers from applications without requiring from them a deep program language knowledge and skills, and is going to overcome current limitations of programming. Layering is one of the great success stories in computer engineering. Already early languages such as COBOL used layered programs (division-section-paragraph-sentence-statement-command; ICCO: initialize-configuration-content_enhancement-operationalisation; environment-declaration-program). Our approach continues and generalizes this approach and will be thus the basis for true fifth generation programming.

A model in the model suite is used for different purposes such as communication, documentation, conceptualization, construction, analysis, design, explanation, and modernization. The model suite can be used as a program of next generation and will be mapped to programs in host languages of fourth or third generation. Models will become programs of true fifth generation programming.

Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, changeable and adaptable at different layers, validateable and verifiable, and maintainable.

References

1. Ershov, A.P.: The transformational machine: Theme and variations. In Proc. *MFCS 1981*, LNCS 118, pp. 16-32. Springer (1981).
2. Grossmann, R., Hutschenreiter, J., Lampe, J., Lötzsch, J., and Mager, K.: DEPOT 2a Metasystem für die Analyse und Verarbeitung verbundener Fachsprachen. Technical Report 85, *Studientexte des WBZ MKR/Informationsverarbeitung der TU Dresden*, Dresden (In German) (1985).
3. Hutschenreiter J.: *Zur Pragmatik von Fachsprachen*. PhD thesis, Technische Universität Dresden, Sektion Mathematik (In German) (1986).
4. Karagiannis, D., Mayr, H.C., and Mylopoulos, J., editors: Domain-Specific Conceptual Modeling, Concepts, Methods and Tools. Springer (2016).
5. Knuth, D.E.: Literate programming. *Comput. J.*, 27(2) pp. 97-111 (1984).
6. Kramer, F.F.: *Ein allgemeiner Ansatz zur Metadaten-Verwaltung*. PhD thesis, Christian-Albrechts University of Kiel, Technical Faculty, Kiel (In German) (2018).
7. Lötzsch, J.: *Metasprachlich gestützte Verarbeitung ebener Fachsprachen*. Advanced PhD thesis (habilitation), Dresden University of Technology, Germany (In German) (1982).
8. Podkolsin, A.S.: *Computer-based modelling of solution processes for mathematical tasks*. ZPI at Mech-Mat MGU, Moscov (In Russian) (2001).
9. Riedewald, G. and Forbrig, P.: Software specification methods and attribute grammars. *Acta Cybern.*, 8(1):89—117 (1987).
10. Thalheim, B., Schewe, K.-D., Prinz, A., and Buchberger, B. editors: *Correct Software in Web Applications and Web Services*. Texts & Monographs in Symbolic Computation. Springer, Wien (2015).
11. Thalheim, B. and Nissen, I. editors: Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung. De Gruyter, Boston (In German) (2015).
12. Thalheim, B.: Theorie deterministischer kontextfreier Grammatiken. Diplomarbeit, Technische Universität Dresden, Sektion Mathematik (In German) (1975).
13. Thalheim, B.: Entity-relationship modeling - Foundations of database technology. Springer, Berlin (2000).
14. Wirth, N.: *Compiler construction*. International computer science series. Addison-Wesley (1996).
15. Werthner, H. and Van Harmelen, F., editors: Informatics in the Future: Proceedings of the 11th European Computer Science Summit (ECSS 2015), Vienna, October 2015. Springer (2017).

Towards a Great Design of Conceptual Modelling

Yasushi KIYOKI[a,1], Bernhard THALHEIM[b], Marie DUŽÍ[c,] Hannu JAAKKOLA[d,]
Petchporn CHAWAKITCHAREON[e] and Anneli HEIMBÜRGER[d,f]
[a]*Graduate School of Media and Governance, Keio University SFC, Japan*
[b]*Christian-Albrechts-University Kiel, Germany*
[c]*Technical University of Ostrava, Czech Rebublic*
[d]*Tampere University, Pori, Finland*
[e]*Environmental Engineering Department, Faculty of Engineering, Chulalongkorn University, Thailand*
[f]*University of Jyväskylä, Faculty of Information Technology, Finland*

Abstract. Humankind faces a most crucial mission; we must endeavour, on a global scale, to restore and improve our natural and social environments. This is a big challenge for global information systems development and for their modelling. In this paper, we discuss on different aspects of conceptual modelling in global environmental context. The paper is the summary of the panel session "The Future of Conceptual Modelling" in the 29th International Conference on Information Modelling and Knowledge Bases.

Keywords. Conceptual modelling, model suites, multi-agent system, artificial intelligence, machine learning, semantic computing, data mining, 5D World Map System, context computing, environmental ICT, globalization.

Introduction

Conceptual modelling has been one of the essential academic subjects in the computer science area and includes highly significant topics not only in academic communities related to information systems, but also in the area of environmental and globalization studies.

Humankind faces the essential and indispensable mission; we must endeavor on a global scale to perpetually restore and improve our natural and social environments. One of the essential research activities in environmental study is conceptual modelling to express, share, analyze and visualize the environmental and social phenomena of various situations. It is essentially significant to create new conceptual modelling for making appropriate and urgent solutions to various environmental changes and social situations in the nature and society.

The nature and society are expecting our activities to cover environmental research areas, towards "Environmental Artificial Intelligence" with sensing-data processing, big-data analysis, machine learning, deep learning, spatio-temporal computing, GIS

[1]Corresponding author.

Information Modelling and Knowledge Bases XXXI, Frontiers in AI and Applications, 312, pages 530-543, IOS, Press, 2020.

(Geographical Information Systems) processing and semantic computing. From the viewpoint of conceptual modelling, much research activity should focus on environmental issues for realizing sustainable nature and society.

To promote discussion on the future trends and challenges of conceptual modelling, we organized a panel session on "The Future of Conceptual Modelling" during the 29th International Conference on Information Modelling and Knowledge Bases (EJC2019). The panelists were Professor Yasushi Kiyoki (panel moderator and chair), Professor Bernhard Thalheim, Professor Marie Duží Professor Hannu Jaakkola and Professor Petchporn Chawakitchareon. In the panel session, we focused on discussions on new conceptual modelling towards Environmental Artificial Intelligence. Open questions to this aim are:

- How to give actual interpretations and understandings to the nature and societies? Nature cannot interpret the meanings of situations/phenomena by itself. Only the human can interpret the meaning of nature's situation by our senses with brain.
- How to give meanings to environmental phenomena in computational processes?

The paper is based on the presentations of the panelists' own viewpoints on the panel session topic. The paper is organized as follows. In Section 1, Professor Thalheim introduces model suites as a maintained collection of associated models. In Section 2, Professor Duží describes communication in a multi-agent world based on the Transparent Intensional Logic (TIL). In Section 3, Professor Jaakkola discusses on artificial intelligent (AI) in modelling landscape and technological changes in conceptual modelling. In Section 4, Professor Kiyoki and Professor Chawakitchareon present the 5D World Map System and its applications to global environmental engineering.

1. Model Suites as a Maintained Collection of Associated Models

Most disciplines simultaneously integrate a variety of models or a society of models. The theory of model suite has been developed in [1, 2, 3]. A model suite is essentially a well-associated and coherent ensemble of models. The models in a model suite coexist, co-evolve, and support solutions of subtasks. A *model suite* [3] consists:

- of set of models which are defined within a common language understanding on the basis of several modelling languages,
- of an association or collaboration schema among the models,
- of controllers that maintain consistency or coherence of the model suite,
- of application schemata for explicit maintenance and evolution of the model suite, and
- of tracers for the establishment of the coherence.

We observe three opportunities of building a model suite:

- *Horizontal model suites*: Horizontal model suites use the same level of abstraction and reflect different but integratable viewpoints or foci or scopes on the basis of different languages. Computer science and engineering modelling is mainly modelling at the same abstraction layer. The model ensemble used in UML (Unified Modelling Language) separates modelling into several concerns such as use case, classes, interaction, packaging, and collaboration. Model-based engineering can be based on a five-level model suite [4]. Business (layer) data models and conceptual (layer) data models are a typical example of a horizontal

model suite since the first one is typically business-oriented and the second one can be considered to be a refinement of the first one. The binding among these models is often implicit. We may however enhance the two models by a mapping that maps the first model to the second one. This mapping combines and harmonizes the different views that are used at the business user layer. A good example is a model suite consisting of a global conceptual model and a rather large number of conceptual viewpoints that reflect the needs of database system users.

- *Vertical model suites*: Vertical model suites combine models that have different abstraction levels, that vary in their level of detail and complexity, and that reflect different time and space abstractions. At the same time they are coupled through some kind of mapping mechanism and within a specific coupling style. Typical well-known vertical model suites are: (a) strategic, tactical and operational models used in business informatics, (b) OLTP-OLAP-Data_Mart decision support systems (OLTP= On-line Transaction Processing, OLAP= On-line Analytical Processing) and (c) database structure pattern. The OSI (Open Systems Interconnection Reference Model) layering model is a good example of a well-associated model suite. Another example is the vertical model suite consisting of models for micro-data, meso-data, and meta-data for instance in decision support systems. Data streaming data and big data applications might become another example of model suite support. These applications can use a data stream profile, a task model that allows to derive the data collection portfolio, a model of analysis-driven data exploration, and a model for data collectors according to the analysis space. A typical example of a sophisticated model suite is the model suite of the human heart. It consists of a 5-layer model of the heart. At the genes layer the networks of genes are given by molecular functions. Proteins form the elementary units, define the chemistry, and their composition. Cell structures are the basis elements for explanation of functions and key organizational unit with biological processes and pathway models. The tissue model describes the structure and function and with cellular components. The human heart as an element of the body is described by a system of myocardial activation.
- *Collection of models at some abstraction layers*: Model suites may also consist of associated models at various abstraction layers. These models are combinations of observations or thought models. A categorization of models at abstraction layers is given in Figure 1.

The third opportunity is less obvious. We thus consider the wide class of mental, semantical, and semantical scientific models as displayed in Figure 1. Mental models can be graphical (iconic, representation) ones. Semantical models are internal ones (perception, cogitative) or external ones (linguistic, meta-, meta-meta-, meta-meta-meta-models). Semantic scientific models are formal scientific, empirical scientific, technological, or praxeological ones [5]. The categorization may be extended to technical models (physico-technical, other technical or engineering models) which are not considered in [5].

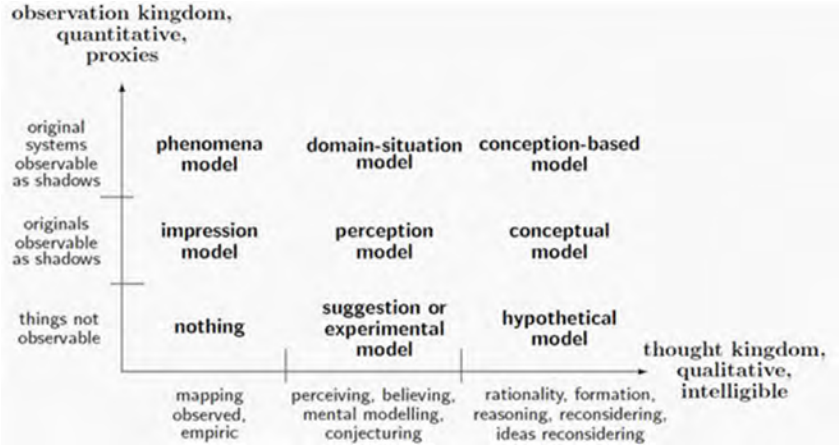

Figure 1. Models at abstraction layers.

The eight kinds of models in Figure 1 show a categorization of model kinds that generalizes the classification in [6, 7]. The classification has been developed after analysis of classical Greek thoughts about models. The main sources are Platon's Politeia [8] and C. Lattmann's [9] and our analysis of the analogies of sun, of the divided line, and of the cave. It allows a categorization into a combination of *observation or empirical quantitative models* and *qualitative intelligible models of thought*. Observations are some kind of reflection, i.e. shadows according to [8]. Intelligible models are based on different modes and qualities of reasoning first of all by one thinker (or a community of thinkers) after perceiving, believing, and digesting observations or beliefs. They are also based on proper and matured thoughts including rationality, formation and reconsideration of ideas. The eight models in Figure 1 are summarized as follows:

- *Impression models*: Humans are observing their environment and summarize their observations on things in models according to their interest, their tasks, their knowledge, and their profile.
- *Phenomena models*: Humans are considering their observations on their system environment, use pragmatic reasoning schemata, and internalize the entities around them. Phenomena models only represent what has been observed and not what is false, in each possibility.
- *Suggestion or experimental models*: Humans might have concluded that some their thoughts so far and develop based on that suggestion models that might be the basis for a falsification process esp. on the basis of experiments.
- *Perception models* are reflecting human understanding on entities observed by a human and are based on the setting of a human, esp. the orientation and the priming. They combine mentalistic concepts that are intuitively formed according to some (empirical) human understanding.
- *Domain-situation models* represent the common worldview on systems that are commonly observed, are governed by shared knowledge and beliefs, and reflect a shared opinion within a community of practice. The modelling method is governed by communication and human interaction.

- *Hypothetical models* mediate between quantitative theories and qualitative theories. They are applied to hypothetical and investigative scenarios, should support causal reasoning as well as network-oriented reasoning, and are developed in an empirical framework.
- *Conceptual models* are models that are enhanced by concepts from concept spaces, are formulated in a language that allows well-structured formulations, are based on mental/perception/situation models with their embedded concept(ion)s, and are oriented on a modelling matrix that is commonly accepted [10].
- *Conception-based models*: Conceptions are consolidated systems of explanation. A model is enhanced by such systems of explanation and provides a generalizing and consolidated viewpoint.

Humans synchronously use a number of models according to their tasks, according to the model functions in task resolving scenarios, according to the collaboration needs, according to their actual context, and according to their partners. Models must not be coherent in the general case. Humans use various models for various purposes. The models might be contradicting and inconsistent as a model society. A model suite however must however be coherent. It is then concise and precise consolidation of all function-relevant structural and behavioral features of systems under investigation. It is represented in a number of predefined formats, e.g. modelling languages such as diagrammatic languages.

2. Communication in a Multi-Agent World

We learn, communicate and think by means of concepts; and regardless of the way in which the meaning of an expression is encoded, the meaning is a concept2. Yet in our background theory, i.e. Transparent Intensional Logic (TIL), we do not define concepts within the classical set-theoretical framework. Instead, we explicate concepts as abstract procedures that can be assigned to expressions as their structured meaning3. In particular, complex meanings, which structurally match complex expressions, are complex procedures whose parts are sub-procedures. The moral suggested here is this. Concepts are not flat sets that cannot be executed and lack a structure; rather, concepts are algorithmically structured abstract procedures. Unlike sets, concepts have constituent parts, i.e. sub-procedures that can be executed in order to arrive at the product the procedure is typed to produce. Not only particular parts of a concept matter, but also the *way of combining* these parts into one whole 'instruction' that can be followed, understood, executed, learnt, etc., matters[4].

Having accepted semantic conception as described above, fundamental questions arise. How to reach those *abstract* procedures? How to examine their structure, how to derive what is entailed by them and not to derive what is not entailed? How to compute

[2] To avoid misunderstanding, we also explicate the meaning of a *sentence* as the *concept* of a proposition denoted by the sentence. For more arguments in favour of structured procedural concepts as the means of our communication, see [11].

[3] For details, see [12] and [13, §2.2].

[4] This had been known already to Bernard Bolzano who criticized the classical Port-Royal school and the law of inverse proportion between the content (intension) and extent (extension) of a concept. The content itself does not determine the concept, the *way of combining* its parts matters; see [14: §120].

their products? There are two possibilities. Either not to do it and instead just specify axioms and rules of using them. Or, *do* it in a systematic way using a language fine-grained enough. In TIL, we vote for the second strategy. To this end, we apply the language of TIL λ-terms that denote these procedures and mirror their structure in an isomorphic way. To give just a hint of our conception, in Figure 2 there is a simple example of a valid argument formalized in TIL.

Tilman is seeking an abominable snowman
Tilman is seeking something abominable

$$\frac{\lambda w \lambda t\,[\text{`}Seek_{wt}\ \text{`}Tilman\ [\text{`}Abominable\ \text{`}Snowman]]}{\lambda w \lambda t\ \exists x\,[\text{`}Seek_{wt}\ \text{`}Tilman\ [\text{`}Abominable\ x]]}$$

Figure 2. An example of a valid argument formalized in TIL

All the entities of TIL ontology receive a type within a ramified hierarchy of types, which makes it possible to distinguish different levels of abstraction. In our example, the variable *x* must not range over individuals; this would turn logic into magic, and we would prove the existence of yetis. Instead, *x* ranges over *properties* of individuals. In addition, TIL is a typed, *hyperintensional* partial λ-calculus. Our procedures (concepts) are structured wholes that can occur in two fundamentally distinct modes, namely executed and displayed (for details, see [15]). When dealing with natural language, we need to operate not only within an extensional or intensional level, but also within *hyperintensional* level where substitution of analytically equivalent terms fails, because the very *meaning procedure* is *displayed* as the object of predication. Hyperintensional context is introduced inter alia by agents' attitudes like knowing, believing, designing, seeking and finding, computing, and many others[5]. Thus we come to the second issue, which is a cogent argumentation in favour of multi-agent systems.

In [17] the authors introduce the system for disaster resilience. Protecting nature, environment and people against disasters is very important and primary goal, of course. Yet, there are critical situations such as an unexpectable air disaster, natural disaster, traffic collapse, and so like, in which we can hardly do any more but to minimalize damages and harms by providing well-organized, professional *disaster relief*. In these situations, multi-agent systems are successfully applied.

Multi-agent systems are dynamic, distributed applications that run on many computers over the network. There can be thousands of agents who are *active* in their perceiving environment and acting in order to achieve their individual as well as collective goals. In general, there is no central dispatcher and the system is driven only by messaging. The agents communicate with their fellow agents by exchanging *messages* and they *learn by experience*. They are resource bounded, yet less-or-more intelligent and rational.

A multi-agent system should be designed in such a way that it is apt for handling *critical situations* where a centralised system is prone to a chaotic behaviour or even collapse. While behaviour of a centralised system heavily depends on the centralised

[5] For a summary on hyperintensionality, see the Introduction to the special issue of Synthese [16].

control so that its fail causes the fall of the whole system, in a multi-agent system there are still some other agents who can act reasonably even in a very critical situation; in the worst case they can at least send a warning message to the public.

The theory formalizing reasoning of agents has to be able to *'talk about'* and *quantify over* the objects of agents' *attitudes*, i.e. *structured meanings* of the embedded clauses, *iterate attitudes* of distinct agents, express *self-referential statements*, respect different *inferential abilities* of resource bounded agents. While this is beyond the capacity of first-order logic systems, we have the theory at hand; it is Transparent Intensional Logic (TIL). Thus, the content of agents' messages is formalized in TIL so that all the above issues are successfully dealt with. Each active agent has its own *ontology* and *knowledge base.* While an agent's knowledge base usually contains dynamic empirical facts, formal ontology is a result of the *conceptualization* of a given domain. It contains definitions (i.e., complex concepts) of the most important entities, forms a conceptual hierarchy together with the most important attributes and relations between entities. Due to TIL ramified hierarchy of types, the agents can reason about concepts themselves, learn new compound concepts via refinement of less complex concepts and exhibit an adequate dynamic behaviour.

In the multi-agent and multi-cultural world procedurally structured *concepts* are central to our communication. We model such concepts as TIL *procedures*, coined *constructions.* Flexible systems that we need to deal with critical situations in our rapidly changing dynamic world are best modelled and implemented as *multi-agent systems* composed of autonomous, resource-bounded yet less or more rational agents who communicate by messaging. In our systems, the content of a message is formalized in terms of concepts, i.e. in the language of TIL constructions.

Acknowledgements. This research has been supported by the Grant Agency of the Czech Republic, project No. GA18-23891S "Hyperintensional Reasoning over Natural Language Texts".

3. AI in Modelling Landscape - Technological Changes in Conceptual Modelling

The role of information modelling as a part of information systems (IS) development is to transfer human knowledge of the requirements of the system to the IS implementation. Along the development path, first step is to create a *conceptual model*. It represents the joint view of the interest groups of the IS's data in a structured way. This structure is manipulated in the evolution path of the IS first to reflect to the needs of *requirements engineering*, further to include the *architectural design* related aspects, then technical aspects coming from *technical design* and finally the elements of *implementation*. During the whole life cycle it is question on the data model of the same IS, but from different points of view. The purpose of the models is to transfer IS related decisions through the life cycle from phase to phase and from interest group to interest group. In addition, it is the key issue for communication in IS development.

Data modelling is always done in its context. The context covers technologies available, tools, development processes – in general, a huge amount of environmental issues of modelling. If we look at the history of computing and IS development as a part of it, it is easy to see dramatical changes in technologies; IS development and data modelling at the principal level have remained the same, the purpose of modelling has remained the same, but technological progress has changed the environment of it. The key enabler in the progress of ICT (Information and Communication Technology) is the

improvement of VLSI technology. According to Moore's Law [18] the packing density of VLSI circuits doubles every 15 months. It reflects directly to the *processing capacity* of computers (doubles in 18 moths), memory capacity (doubles in 15 months), data transmission capacity (doubles in 20 months) and mass memories (capacity doubles in 18 months). If we compare the time of birth of systematic data modelling (from early 1960s) to the situation today, we have currently the computing environment having processing capacity of 2^{40}-, memory size of computers 2^{49}-, mass memory size 2^{40}- and data transmission capacity 2^{36}-times compared with the computing environment of early 1960s. In spite of this progress, it is acceptable to say that data modelling has more or less remained the same over the decades – or has it?

The progress introduced above is handled in more detail in [19] from the point of view of AI (Artificial Intelligence). AI has one of the key roles in the current era of data modelling. In [20] the era of systematic data modelling is divided (by binding it to the progress of database management) in four phases (quoted from the original article):

- *Phase I* (from roughly the 1960s to 1999) included the development of Database Management Systems (DBMS) known as hierarchical, inverted list, network, and during the 1990s, object-oriented Database Management Systems.
- *Phase II* (starting about 1990) relates to relational databases, SQL and SQL products (plus a few nonSQL products).
- *Phase III* (starting also around 1990 simultaneously to the Phase II) supported Online Analytical Processing (OLAP), along with specialized DBMSs.
- *Phase IV* (started 2008) introduced NoSQL and supported the use of Big Data, non-relational data, graphs, etc.

If we simply analyze the progress above, it is easy to notice that data modelling varies over the phases. D*evelopment tools* are an essential part in IS development. In the development process, we aim to look at the system structure "through the glasses" of the tool. Because of that it is easy to agree that also tools create a part of the IS development context and aim to guide the data modelling.

One interesting view to the changes in data modelling can be found in the traditional classification of data related concepts. The DIKW pyramid[6] (the good overview of it is available in [21]) represents structural and/or functional relationships between the concepts data, information, knowledge, and wisdom:

- Data: facts and figures relaying something in a non-organized way.
- Information: Contextualized, categorized, organized data.
- Knowledge: know-how, understanding, experienced, insight, intuition, contextualized information.
- Wisdom: Knowledge applied in action.

The hierarchy, in addition to help understanding about the role of the data in information systems, gives a view to the progress in data modelling. Without hesitance, it is possible to say that during the decades it is easy to see the transfer of the modelling target from lower towards upper levels. Nowadays, in the new era of AI, we should be able to model the connections between items of wisdom, instead of the traditional (data) concepts.

[6] The origin of the DIKW Pyramid is not unanimously specified. The classification is handled in several articles; based on a simple literature review the author reviewed e.g.: http://www.infogineering.net/data-information-knowledge.htm; http://www.knowledge-management-tools.net/knowledge-information-data.html; https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/

What are the current trends in data and information systems modelling? This topic is handled in [22]. The author of the aeticle points out the important role of cloud platforms, coupled with Big Data and IoT technologies powered by AI and Machine Learning (ML), providing professional means for non-professionals - citizen data analysts. The article gives nice scene to the data modelling of 2019. We list some of the most important aspects of it as follows:

- Tools embedding AI and ML change the modelling landscape. Built in intelligence in the tools decrease the amount of human work. Model analytics decreases the opportunity for human errors and increases the quality of models. Automatic model generation decreases the amount of human work
- Gartner's predicts 40 percent automation in data science tasks. In data analytics, the role of citizen data analysts is growing. It indicates the changing role of the experts. Two types of data models are needed: one for data professionals and one for citizen users to be used for quick solutions in a plug and play manner.
- There is transfer from problem specific to problem area specific instruments and towards framework dominance. It indicates higher abstraction of the IS models.
- The increasing role of Robotic Process Automation (RPA; Software Robotics) indicates the growing importance of business process modelling.
- Transfer out of relational databases, especially in new types of applications. New technologies - NoSQL databases, data lakes, algorithmic intelligence, self-describing data formats, standardized data models - initiate new challenges for and take place of data modelling. Cloud dominance affects in data structures.
- The growing role of "on-line" continuous data handled dynamically without knowing its structure in advance.
- Globalization of IS business indicates complexity in the Data Management ecosystem and unknown Data Governance issues. Data landscape becomes distributed, having a wide variety of data sources in applications. The importance of interoperability issues runs towards commonly used standardized data structures and models. Importance of interfaces and interface modelling will be an essential part of data modelling.

As a summary, it is easy to see the growth of modelling complexity, transfer of data (modelling) related tasks from professionals to end-users and AI to support human work. In basic level, data modelling remains as it has always been, but in practice, a lot of new challenges will appear.

4. Applications of AI in Global Environmental Engineering

4.1. Conceptual Modelling with Semantic Computing for Environmental Analysis

Humankind, the dominant species on Earth, faces the most essential and indispensable mission; we must endeavor on a global scale to perpetually restore and improve our natural and social environments. It is essentially significant to apply conceptual modelling and knowledge computing to global environment-analysis for finding out difference and diversity of nature and livings with a large amount of information resources in terms of global environments.

The 5D World Map (5DWM) System has introduced the concept of "SPA (Sensing, Processing and Analytical Actuation Functions)" for realizing a global environmental

system [23, 24, 25, 26]. The 5DWM system has been proposed by Kiyoki and Sasaki in [24, 25], and its architecture has been implemented as a multi-visualized and dynamic knowledge representation system. The 5DWM is a system for visualizing the data resources to the map, which can be analyzed with multi-dimensional axes. Environmental Knowledge Base creation with 5DWorld Map is implemented for sharing, analyzing and visualizing various information resources to the map, which can display and facilitate the comparisons in multidimensional axes.

This system realizes Physical-Cyber integration, as shown in Figure 3, to detect environmental phenomena with real data resources in a physical-space (real space), map them to the cyber-space to make knowledge bases and analytical computing, and actuate the computed results to the real space with visualization for expressing environmental phenomena, causalities and influences.

The 5D World Map System and its applications create new analytical circumstances with the SPA concept (Sensing, Processing and Analytical Actuation) for sharing, analyzing and visualizing natural and social environmental aspects, as shown in Figure 4. This system realizes "environmental analysis and situation-recognition" which will be essential for finding out solutions for global environmental issues. The 5D World Map System collects and facilitates many environmental information resources, which are characteristics of ocean species, disasters, water-quality and deforestation.

As conceptual modelling for making appropriate and urgent solutions to global environment changes in terms of short and long-term changes, "*six functional-pillars*" are essentially important with "*environmental knowledge-base creation*" for sharing, analyzing and visualizing various environmental phenomena and changes in a real world: (1) Cyber & Physical Space Integration, (2) SPA-function, (3) Spatio-Temporal computing, (4) Semantic computing, (5) World map-based visualization, and (6) Warning message propagation

As an actual implementation of the SPA architecture, 5D World Map System Project has presented a new concept of *"Water-quality Analysis Semantic-Space for Ocean-environment*" for realizing global water-environmental analysis [26]. The semantic space and the computing method have been implemented with knowledge-base creation for water-quality-analysis sensors for analyzing and interpreting environmental phenomena and changes occurring in the oceans in the world. We have focused on sea-water quality data, as an experimental study for creating *"Water-quality Analysis Semantic-Space for Ocean-environment*" [26].

4.2. International Collaborative Research Activities with SPA-based 5D World Map System

The 5D World Map System focuses on sharing, analyzing and visualizing various environmental influences and changes caused by natural phenomena and disasters in global environments with "environmental multimedia data resources." As a new meta-level system of international collaborative environment analysis, this system creates a remote, interactive and real-time academic-research exchange in global scopes and areas [24, 25]. Applications of 5DWorld Map for global sharing analysis of environmental situations and changes were studied as case studies in the international collaborative research activities with KEIO University (Japan) and Chulalongkorn University (Thailand).

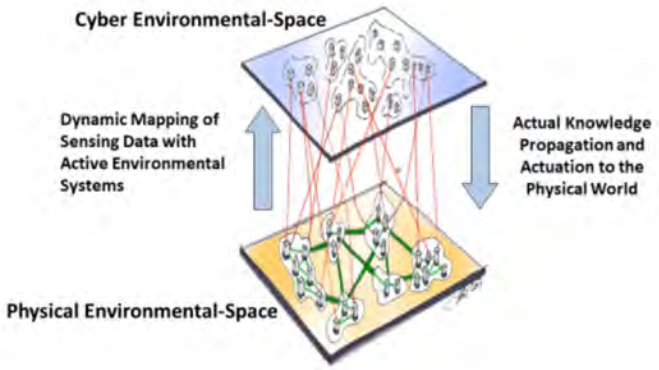

Figure 3. Global & Environmental System with "Cyber & Physical Spaces"**.**

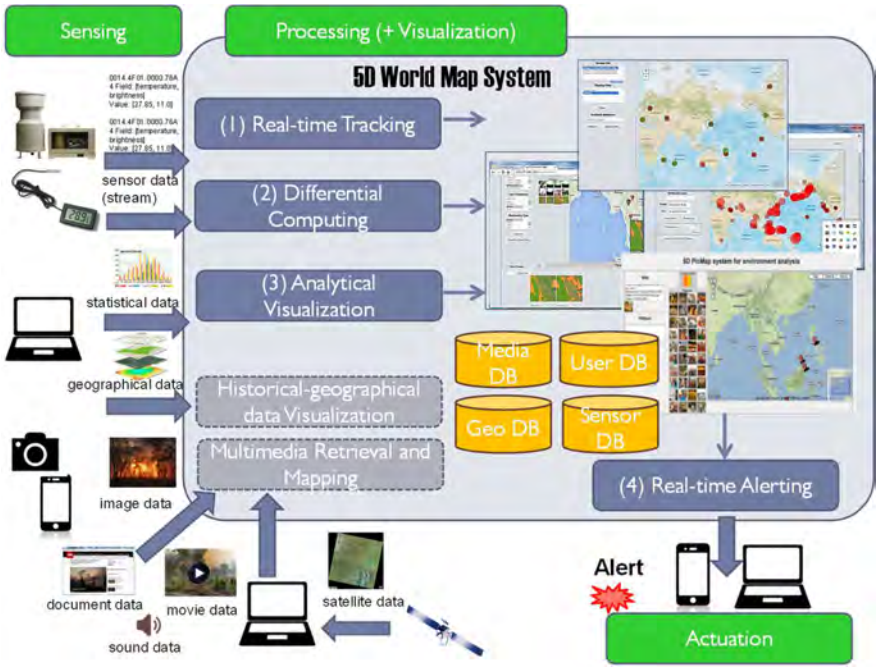

Figure 4. Basic SPA functions in 5D World Map System.

An actual international and collaborative research project on the 5D World Map System started in 2011 with Chulalongkorn University [26, 27, 28]. This project focuses

on the global coral-analysis with multi-visualized knowledge sharing with 5D World Map System, applied to "coral-health-level analysis with images and water-quality data".

This project [27] stared at Sichang Island, Thailand. Those coral information resources and research results have been mapped onto the 5D World Map system [24, 25]. Three species of corals at Sichang Island i.e. *Acropora sp. Goniopora sp.* and *Pavona sp.* were subjected to a stress test with low salinity and normal salinity at concentration 10, 20 and 30 psu, respectively. Under water photographs and eye observation of coral activity were recorded at 12, 24 and 48 hours. The entropy or surface roughness and percent polyp activity were analyzed with comparison to eye observation of coral activity. The experiment was carried out under continuous water temperature and underwater light intensity controlled. The results indicated that "Healthy" entropy values for Acropora sp. are 1.57-1.62 and for *Goniopora sp.* are 4.26-4.46. In contrast, for *Pavona sp.*, short polyp coral, there was no "Healthy" entropy value resulted from any photographic assessment in this study. The "Healthy" value of *Acropora sp*. evaluated from percent active polyp was more than 52.4.

This collaborative project focuses on the effects of temperature and ammonia to coral health levels on *Acropora sp., Turbinaria sp., Porites sp.* by coral health levels evaluation with Coral Health Chart [28]. It was a standardized color reference card, which is a flexible tool that anyone can use for rapid, wide-area assessment of changing coral condition. The acute toxicity of ammonia concentration that affects to bleach coral more than 50% (50% Lethal Concentration: LC_{50}) was calculated by Probit analysis and coral bleaching analysis by polyp image analysis.

In addition to coral-analysis in this place, this project integrated a global sharing analysis and visualization of water quality analysis [29], with 5D World Map system. This integration is a typical and advantageous result to be realized with 5D World Map system, as typical and effective integration between different subjects with high relationships each other. The data resources in this research were collected from Sichang Island, Chonburi province, Thailand during 1990 to 2002. Six input parameters of water quality i.e. chlorophyll a, ammonia, nitrite, nitrate, phosphate and silicate were collected and displayed in 5D World Map system. The total location-sites were 21 stations, which situated around Sichang Island. All data of water quality were added and displayed with 5D World Map system in order to visualize and share the water quality from 1990 to 2002. Our results showed that 5D World Map system integrates environmental analysis with the coral-analysis subject related to the coral health level. We apply the dynamic evaluation and mapping functions of multiple views of temporal-spatial metrics, and integrate the results of semantic evaluation to analyze environmental multimedia information resources**.** 5D World Map System for world-wide viewing of global environmental analysis with coral-analysis and water quality around Sichang Island in Thailand was reported in this study.

To conclude, we have introduced a conceptual modelling methodology for realizing global environmental analysis with ″5D World Map System." This methodology is essential to make appropriate and urgent solutions to global environment changes in terms of short and long-term changes. We have applied this methodology to "international and collaborative environmental-system research and education" as a new platform of environmental computing. This platform realizes remote, interactive and real-time academic research exchanging for international and collaborative research activities, and this system is currently utilized as an international and environmental research platform. This system is also expected to create several new original approaches to global environmental-knowledge sharing, analysis and visualization with "spatio-

temporal & semantic computing." This system concept will be a basic structure to create new international and collaborative research and education for making important solutions and knowledge sharing on global environmental issues in the world-wide scope.

Acknowledgements: We would like to appreciate Dr. Xing Chen, Dr. Shiori Sasaki, Dr. Asako Uraki, Dr. Chalisa Veesommai and Dr. Irene Erlyn Wina Rachmawan, "5D World Map System Project" members and Dr. Sompop Rungsupa for their significant discussions and experimental studies.

References

[1] Dahanayake, A. *An environment to support flexible information modelling*. PhD thesis, Delft University of Technology, 1997.
[2] Dahanayake, A. and Thalheim, B. *Co-evolution of (information) system models*. Proc. EMMSAD 2010, LNBIP vol. 50, 314-326. Springer, 2010.
[3] Thalheim, B. *Model suites*. Selected topics on distributed disaster management: Towards collaborative knowledge clusters, (ed.) Jaakkola, H. pp. 108 - 128. Porin yksikkö, Tampere University Press, 2008.
[4] [Thalheim, B. *Model-based engineering for database system development*. Conceptual Modeling Perspectives, 137-153. Springer, Cham 2017.
[5] Stachowiak, H. *Modell*. Handlexikon zur wissenschaftstheorie, (eds.) H. Seiffert, and G. Radnitzky, pp. 219-222. Deutscher Taschenbuch Verlag GmbH & Co. KG, München, 1992.
[6] Thalheim, B. *Conceptual models and their foundations*. Proc. MEDI2019, published in LNCS, Springer 2019.
[7] Thalheim, B. and Nissen, I. (eds.), Wissenschaft und kunst der modellierung: Modelle, Modellieren, Modellierung. De Gruyter, Boston, 2015.
[8] Platon, R. *Platonis rempublicam recognovit brevique adnotatione critica instruxit*. S.R. Slings, Oxford, 2003.
[9] Lattmann, C. *Vom dreieck zu pyramiden - Mathematische modellierung bei Platon zwischen Thales und Euklid*. Habilitation thesis, Kiel University, Kiel, 2017.
[10] Jaakkola, H. and Thalheim, B. *Cultures in information systems development*. Information Modelling and Knowledge Bases XXX, pp. 61-80. Frontiers in Artificial Intelligence and Applications 312, IOS Press, 2019.
[11] Materna, P. *Conceptual systems.* Logos Verlag Berlin, 2004.
[12] Duži M., Jespersen B. and Materna P. *Procedural semantics for hyper-intensional logic. Foundations and applications of trasnsparent intensional logic*. Berlin: Springer, series Logic, Epistemology, and the Unity of Science, vol. 17, 2010.
[13] Bolzano, B. *Wissenschaftslehre*. Sulzbach: von Seidel, 1837.
[14] Duží, M. Communication in a multi-cultural world. *Organon F*, vol. 21, No. 2, pp. 198-218, 2014.
[15] Duží, M. If structured propositions are logical procedures then how are procedures individuated? *Synthese*, vol. 196, No. 4, pp. 1249-1283, 2019.
[16] Jespersen, B. and Duží, M. (2015): Introduction to the special issue on Hyper-intensionality. *Synthese*, vol. 192, No. 3, pp. 525-534, 2015.
[17] Sasaki, S., Kiyoki, Y., Sarkar-Swaisgood, M., Wijitdechaku, J., Rachmawan, I.E.W., Srivastava, S., Show, R. and Veesommai, Ch. 5D World Map System for Disaster-Resilience Monitoring from Global to Local: Environmental AI System for Piloting SDG 9 and 11. In *Proceedings of the 29th International Conference on Information Modelling and Knowledge Bases - EJC 2019*, Ajantha Dahanayake (ed.), pp. 500-510, 2019.
[18] Moore, G.E. Cramming more components onto integrated circuits, *Electronics* 38, 8 (April 1965), Retrieved February 12th, 2019 from ttp://download.intel.com/museum/Moores_Law/ Articles-Press _Releases/Gordon_Moore_1965_Article.pdf, 1965.
[19] Jaakkola, H., Henno, J., Mäkelä, J., and Thalheim, B. *Artificial intelligence yesterday, today and tomorrow*. In Karolj Skala (Ed.), Proceedings of The 42nd International ICT Convention – MIPRO 2019 (pp. 985-992). Opatia: MIPRO and IEEE, 2019.
[20] Foote, K. D. *A brief history of data modeling*. Retreived August 7th 2019 from https://www.dataversity.net/brief-history-data-modeling/. 2017.
[21] Wikipedia. DIKW pyramid. https://en.wikipedia.org/wiki/DIKW_pyramid, 2019.
[22] Ghosh, P. *Data modeling trends in 2019*. Retriewed August 7th, 2019 from from https://www.dataversity.net/data-modeling-trends-in-2019/, 2019.

[23] Kiyoki, Y., Kitagawa, T. and Hayama, T. A metadatabase system for semantic image search by a mathematical model of meaning, *ACM SIGMOD Record*, vol. 23, no. 4, pp.34-41, 1994.
[24] Kiyoki, Y., Chen, X., Sasaki, S. and Koopipat, C. *Multi-dimensional semantic computing with spatial-temporal and semantic axes for multi-spectrum images in environment nnalysis*, Information Modelling and Knowledge Bases (IOS Press), Vol. XXVI, 20 pages, March 2016.
[25] Kiyoki, Y., Chen, X., Veesommai, C., Sasaki, S., Uraki, A., Koopipat, C., Chawakitchareon, P. and Hansuebsai, A. *An environmental-semantic computing system for coral-analysis in water-quality and multi-spectral image spaces with "multi-dimensional world map*, Information Modelling and Knowledge Bases, Vol. XXVIII, 20 pages, March 2018.
[26] Kiyoki, Y., Uraki, A. and Veesommai, C. *A seawater-quality analysis semantic space in Hawaii-Islands with multi-dimensional World Map System*, 18th International Electronics Symposium (IES2016), Bali, Indonesia, September 29-30, 2016
[27] Rungsupa, S., Chawakitchareon, P., Hansuebsai, A., Sasaki, S. and Kiyoki, Y. *Photographic assessment of coral stress: Effect of low salinity to Acropora sp. Goniopora sp. and Pavona sp. at Sichang Island, Thailand*, Information Modelling and Knowledge Bases, Vol. XXVIII, 20 pages, March 2018.
[28] Udomsap, B., Chawakitchareon, P., Rungsupa, S. and Kiyoki, Y. *An effect-analysis method for species-dependent coral health levels in temperature and ammonia: A case study of Acropora sp., Turbinaria sp., Porites sp. at Sichang Island, Thailand*, Information Modelling and Knowledge Bases (IOS Press), Vol. XXX, March 2019.
[29] Chawakitchareon, P., Ladsavong, K., Kiyoki, Y., Sasaki, S. and Rungsupa, S. *Global sharing analysis and visualization of water quality by 5D World Map: A case study at Sichang Island, Thailand*, Information Modelling and Knowledge Bases (IOS Press), Vol. XXX, March 2019.

Models for Communication, Understanding, Search, and Analysis

Bernhard Thalheim[0000−0002−7909−7786]

Christian-Albrechts University at Kiel, Dept. of Computer Science, D-24098 Kiel
thalheim@is.informatik.uni-kiel.de
http://www.is.informatik.uni-kiel.de/∼thalheim

Abstract. Models are one of the universal instruments of humans. They are equally important as languages. Models often use languages for their representation. Other models are conscious, subconscious or preconscious and have no proper language representation. The wide use in all kinds of human activities allows to distinguish different kinds of models in dependence on their utilisation scenarios. In this keynote we consider only four specific utilisation scenarios for models. We show that these scenarios can be properly supported by a number of model construction conceptions. The development of proper and well-applicable models can be governed by various methodologies in dependence on the specific objectives and aims of model utilisation.

1 Introduction

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the solution computation of problems, accurate at least within the scope, and must function within an application scenario.

The Notion of Model

Let us first briefly repeat our approach to the notion of model:

A **model** *is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.* [6, 24, 25]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

Proc. XXI DAMDID/RCDL, 2019, CEUR-WS, vol. 2523, pages 19-34, Kazan, Russia, 2019

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its *background*, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is *adequate* for a collection of origins if it is *analogous* to the origins to be represented according to some analogy criterion, it is more *focused* (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its *purpose*.

Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics [23] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Model Deployment Scenarios are Multi-Facetted

The model notion can be seen as an initialisation for more concrete notions. We observe that model utilisation follows mainly four different kinds of scenarios (see Figure 1). The four scenarios do not occur in its pure and undiffused form they are interleaved. We can however distinguish between:

Problem solving scenarios: Problem solving is a well investigated and well organised scenario (see, for instance, [1, 8]). It is based on (1) a problem space that allows to specify some problem in an application in an *invariant* form and (2) a solution space that *faithfully* allows to back-propagate the solution to the application. We may distinguish three specific scenarios: perception & utilisation; understanding & sense-making, and making your own.

Engineering scenarios: Models are widely used in engineering. They are also one of the main instruments in software and information systems development, especially for system construction scenario. We may distinguish three specific scenarios depending on the level of sophistication: direct application:, managed application, and application according to well-understood technology.

Science scenarios: Sciences have developed a number the distinctive form in which a scenario is organised. Sciences make wide use of mathematical modelling. The methodology of often based on specific moulds that are commonly accepted in the disciplinary community of practice, e.g. [1]. We may distinguish three specific scenarios: comprehension, computation and automatic detection for instance in data science, and intellectual adsorption.

Social scenarios: Social scenarios are less investigated although cognitive linguistics, visualisation approaches, and communication research have contributed a lot. Social models might be used for the development of an understanding of the environment, for agreement on behavioural and cultural pattern, for consensus development, and for social education.
We may distinguish three specific scenarios: development of social acceptance, internalisation & emotional organisation, and concordance & judgement.

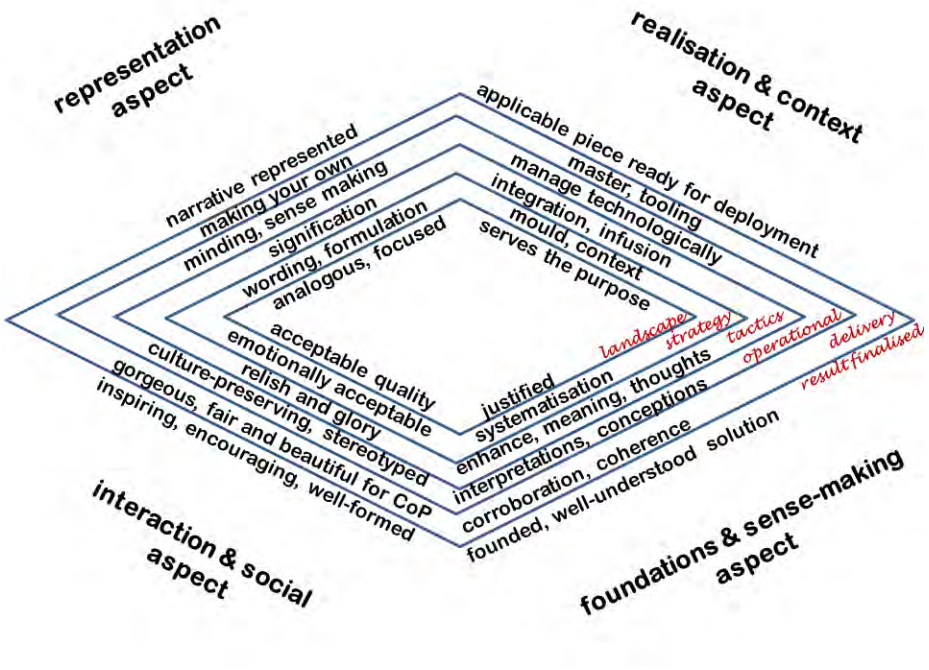

Fig. 1. The four aspects of model scenarios: problem solving, engineering, science, social scenarios. Each of the scenarios combines initialisation by exploring the landscape, strategy, tactics, operational, and delivery layers. Each of these layers adds quality characteristics and specific activities to the previous layer in dependence on the aspects considered.

The notion of model mainly reflects the initialisation or landscape layer. Depending on the needs and demands to model utilisation we may distinguish various layers from initialisation towards delivery. The strategy, tactics, operational, and delivery layers are essentially refinements and extensions of the initialisation. The dependability and especially the sufficiency are based on other criteria while the landscape layer is permanent for all models due to the consideration of the concern, the issue, and the specific adaptation to the community of practice, . The strategy layer is governed by the context (e.g. the discipline) and the mould for model utilisation, and the matrix (including methodologies and commonly

accepted approaches to modelling). The tactics layer depends on the settlement of the strategy and initialisation layers considers the well-acknowledged experience (e.g. generic approaches), the school of thought or more generally the background, and the framing of the modelling. Which origin(al)s are reflected and which are of less importance is determined in the operational layer that orients on the design and on mastering the modelling process. Finally the model is delivered and form for its application in scenarios that are considered. We thus observe various specific quality characteristics for each of these aspects and layers.

Do We Need a Science of Models and Modelling?

Since everybody is using models and has developed a specific approach to models and modelling within the tasks to be solved, it seems that the answer is "no". From the other side we deeply depend on decisions and understandings that are based on models. We thus might ask a number of questions ourselves. Can models be misleading, wrong, or indoctrinating? Astrophysics uses a Standard Model that has not been essentially changed during the last half century. Shall we revise this model? When? What was really wrong with the previous models? Many sciences use modelling languages in religious manner, e.g. think about UML and other language wars. What is the potential and capacity of a modelling languages? What not? What are their restrictions and hidden assumptions? Why climate models have been deeply changing and gave opposite results compared to the previous ones? Why we should limit our research on impacts of substances to a singleton substance? What is the impact of engineering in this case? What has been wrong with the two models on post-evolution of open cool mines after deployment in Germany which led to the decision that revegetation is far better than water flooding? Why was iron manuring a disaster decision for the Humboldt stream ocean engineering? What will be the impact of the IPCC/NGO/EDF/TWAS proposal for Solar Radiation Management (SRM) for substantial stratosphere obscuration for some centuries on the basis of reflection aerosols (on silver, sulfate, photophoretic etc. basis)? Why reasoning on metaphors as annotations to models may mislead? Are "all models wrong"[1]?

Developing a science of models and modelling would allow us to answer questions like the following one: What is a model in which science under which conditions for whom for which usage at a given time frame? What are necessary and sufficient criteria for an artefact to become a model? What is the difference between models and not-yet-models or pre-models? What is not yet a model? How are models definable in sciences, engineering, culture, ...? Under which conditions we can rely on and believe in models? Logical reasoning: which calculus? Similarity, regularity, fruitfulness, simplicity, what else (Carnap)? Treatment, development, deployment of models: is there something general in common? Models should be useful! What does it mean? Is there any handling of usage,

[1] "All models are wrong. ... Obtain a 'correct' one. ... Alert to what is importantly wrong." [2] We claim: *Models might be 'wrong'. But they are* ***useful****.*

usefulness, and utility? What is the difference between an object, a model, and a pre-model? What might be then wrong with mathematical models? What is the problem in digging results through data mining methods?

The Storyline of this Paper

Models are the first reasoning and comprehension instruments of humans. Later other instruments are developed. The main one is language. Models then often become language-based if they have to be used for collaboration. Others will remain to be conscious, preconscious or subconscious. Based on the clarification of the given notion of model and a clarification of the model-being we explore in this paper what are the constituents of models, how models are composed, and what are conceptions for model constructions. Since models are used in scenarios and should function sufficiently well in these scenarios we start with an exploration of specific nature of models in four scenarios. We are not presenting all details for a theory of models[2].

2 Case Study on some Scenarios for Model Utilisation

Models are used in various *utilisation scenarios* such as construction of systems, verification, optimization, explanation, and documentation. *In these scenarios* they *function* as *instruments* and thus satisfy a number of properties [7, 26–28].

Models for Communication

The model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested parties with different interests find a way to reconcile or compromise to come up with an agreement.

The model has several functions in this scenario: (personal/public/group) *recorder of settled or arranged issues*, *transmitter of information*, *dialogue service*, and *pre-binding*. Users act in the speaker, hearer, or digest mode.

The communication act is composed of six sub-activities: derive for communication, transfer, receive, recognise and filter against knowledge and experience, understand, and integrate. We may distinguish two models at the speaker side and six models at the hearer side: speaker's extracted model for transfer, transferred model for both, hearer's received model, hearer's understanding and recognition model, hearer's filtered model, hearer's understood model, and hearer's integration model. These models form some kind of a model ensemble. Some are extensions or detailing ones; others are zooming ones. Communication is based on some common understanding or at least on transformation of one model to another one.

[2] Collections of papers wich are used as background for this paper is downloadable via Research Gate. Notions and definitions we used can be fetched there.

Models for Understanding

Models may be used for understanding the conceptions behind. For instance, conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

The function of a model within these scenario is *semantification* or *meaning association* by means of concepts or conceptions. The model becomes enhanced what allows to regard the meaning in the concept.

Models tacitly integrate knowledge and culture of design, of well-forming and well-underpinning of such models and of experience gained so far, e.g. meta-artifacts, pattern and reference models. This experience and knowledge is continuously enhanced during development and after evaluation of constructs.

Models are functioning for *elaboration, exploration, detection*, and *acquisition* of tacit knowledge behind the origins which might be products, theories, or engineering activities. They allow to understand what is behind drawn curtain.

Models for Search

Users often face the problem that their mental model and their fact space are insufficient to answer more complex questions [12]. Therefore, they seek information in their environment, e.g. from systems that are available. Information is data that have been shaped into a form that is meaningful and useful for human beings. Information consists of data that are represented in form that is useful and significant for a group of humans. This information search is based on their on the *information need*, i.e. a perceived lack of some information that is desirable or useful. The information is used to derive the current *information demand*, i.e. information that is missing, unknown, necessary for task completion, and directly requested. Is is thus related to the task portfolio under consideration and to the intents.

Search is one of the most common facilities in daily life, engineering, and science. It requires to examine the data and information on hand and to carefully look at or through or into the data and the information.

There is a large variety of information search [5] such as:

1. querying data sets (by providing query expressions in the informed search approach),
2. seeking for information on data (by browsing, understanding and compiling),
3. questing data formally (by providing appropriate search terms during stepwise refinement),
4. ferreting out necessary data (by discovering the information requested by searching out or browsing through the data),
5. searching by associations and drilling down (by appropriate refinement of the search terms),
6. casting about and digging into the data (with a transformation of the query and the data to a common form), and
7. zapping through data sets (by jumping through provided data, e.g., by partially uninformed search).

Models for Analysis

Data analysis, data mining or general analysis combines engineering and (systematic) mathematical problem solving [20]. The model development process combines problem specification and setting with formulation of the analysis tasks by means of macro-models, integration of generic models, selection of the analysis strategy and tactics based on methodology models, models for preparation of the analysis space, and model combination approaches for development of the final model society as the analysis result [16, 14]. The typical process model that governs the analysis process is based on a layering approach, e.g. initial setting, strategy, tactics with generic (or general parameterised models), analysis initialisation, puzzling the analysis results, and final compilation. It is similar to experiment planning in Natural Sciences. The analysis puzzling may follow a number of specific scenarios such as pipe scenarios [19].

3 Model Conceptions for These Scenarios

It seems that these scenarios require completely different kinds of models. This is however often not the case. We can develop stereotypes which are going to be refined to pattern and later to templates as the basis for model development. We demonstrate for the four scenarios (communication, understanding, search, analysis) how models can be composed in a specific form and which kind of support we need for model-backed collaboration.

Deep Models

A typical model consists of a normal (or surface) sub-model and of deep (implicit, supplanted) sub-models which represent the disciplinary assumptions, the background, and the context. The deep models are the intrinsic components of the model. Conceptualisation might be four-dimensional: sign, social embedding, context, and meaning spaces. The deep model is relatively stable. In science and engineering it forms the disciplinary background. It is often assumed without mentioning it. For instance, database modelling uses the paradigms, postulates, assumptions, commonsense, restrictions, theories, culture, foundations, practices, and languages as carrier within the given thought community and thought style, methodology, pattern, and routines. This background is assumed as being unquestionable given. The normal model mainly represents those origins that are really of interest.

The deep model combines the unchangeable part of a model and is determined by (i) the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), (ii) the outer directives (context and community of practice), and (iii) the basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. The deep model can be dependent on mould principles such as the conceptualisation principle [9].

A typical set of deep models are (the models and) foundations behind the origins which are inherited by the models of those origins. Also modelling languages have there specific deep parts. As well as methodologies or more generally moulds of model utilisation stories.

Model Capsules

Model capsules follow a global-as-design approach (see Figure 2). A model has a number of sub-models that can be used for exchange in collaboration or communication scenarios. A model capsule consists of a main model and exchange sub-models. Model capsules are stored and managed by their owners. Exchange sub-models are either derived from the main model in dependence on the viewpoint, on foci and scales, on scope, on aspects and on purposes of partners or are sub-models provided by partners and transformed according to the main model. A sub-model might be used as an export sub-model (e.g. $A_{4,E}$) that is delivered to the partner on the basis of the import sub-model (e.g. $B_{4,I}$). The sub-models received are typically transformed. We thus use the E(xtract)T(ransform)L(oad) paradigm where extraction and loading is dependent on the language of the sending or receiving model and where transformation allows adaptation of the export sub-model to the import sub-model.

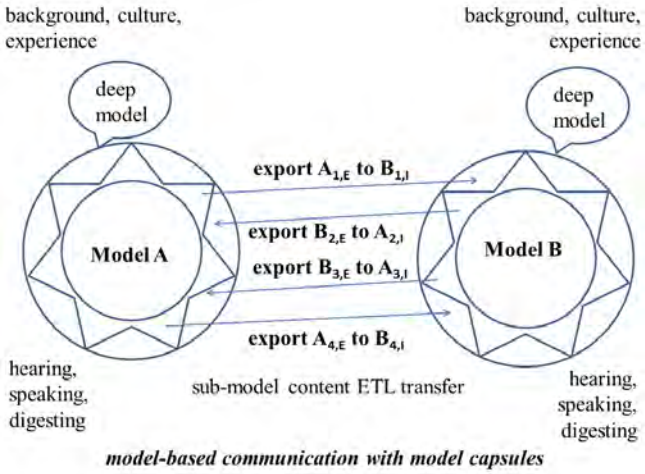

Fig. 2. Exchange on the basis of model capsules with sub-models in model-based ETL-oriented communication scenarios

Model Suites

Most disciplines simultaneously integrate a variety of models or a *society of models*, e.g. [3, 11]. The four aspects in Figure 1 are often given in a separate form as an integrated society of models. Models developed vary in their scopes, aspects and facets they represent and their abstraction.

A typical case are the four aspects that might coexist within a complex model. For instance, models in Egyptology [4] [3] can be considered have four aspects where each of the aspects has its specific model. The entire model is an integrated combination of (1,2) signs in textual representation and an extending it hieroglyph form (both as representation), (3) interpretation pattern (as the foundation and integration into the thoughts), (4) social determination (as the social aspect), and (5) a context or realisation models into which the model is embedded. The co-design framework for information systems development (integrated design of structuring, functionality, interaction, and distribution) uses four different interrelated and interoperating modelling languages. These modelling languages are at the same level of abstraction and may be combined with additional orientation on usage (as a social component, e.g. represented by storyboards [21]). In this case, the foundational aspect is hidden within the modelling language and within the origins of the models, for instance in the conceptualisation. Following the four aspects in Figure 1, we derive now models that consider one, two, three, or all four aspects (Figure 3).

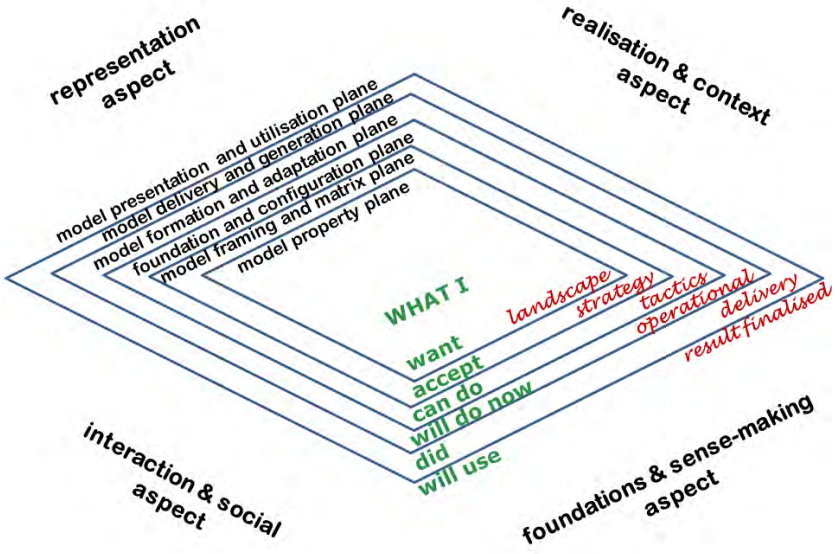

Fig. 3. The four aspect model suite and the corresponding planes for the layers within a model. Activities are governed at each plane by the WHAT I <actually_consider> as main activity.

A *model suite* consists of set of models $\{M_1, ..., M_n\}$, of an association or collaboration schema among the models, of controllers that maintain consistency or coherence of the model suite, of application schemata for explicit maintenance

[3] The rich body of knowledge resulted in [22] or the encyclopedia with [17].

and evolution of the model suite, and of tracers for the establishment of the coherence.

Model suites typically follow a local-as-design paradigm of modelling, i.e. there must not exist a global model which combines all models. In some cases we might however construct the global model as a model that is derived from the models in a model suite. The two approaches to model-based exchange can be combined. A model capsule can be horizontally bound to another capsule within a horizontal model suite or vertically associated to other model capsules. Model capsules are handled locally by members in a team. For instance, model capsules are based on models A and B that use corresponding scientific disciplines and corresponding theories as a part of their background. The models have three derived exchange sub-models that are exported to the other capsule and that are integrated into the model in such a way that the imported sub-model can be reflected by the model of the capsule.

Model Scenes

Model scenes for the development process may be specified in a similar way as storyboarding [21]. A scene is used by members of the community of practice, follows a certain modelling mould, considers a typical ensemble of origin(al)s, inherits certain stereotypes and pattern, is embedded into a context and the tasks, and uses the deep model as the background for model development. These parameters govern and thus control the scene. The developer or modeller is involved into this scene. The input for the scene is the current model, the specific properties of the ensemble of origin(al)s, and especially the experience gained so far. This experience may be collected in a library or generalised to generic models. The output is an enhanced model. We notice that model utilisation scenes can be specified in a similar way.

Figure 4 displays the embedding of a model scene into the model mould or more specifically into the methodology as a macro-model for development.

A model scene is an element of a model story. We imagine that the story can be represented as a graph. A model scene considers an actual or normal model and at the same time the desired embedding into the deep model. The scene is relevant for the community of practice. The model should be accepted by this community. The model scene also embeds the deep model. The scene has its cargo [18], i.e. its mission, determination, meaning, and specific identity. The cargo allows to determine the utility that the model gained so far.

Model Stories

Model development and utilisation can be described as a graph of scenes. Let us consider the model development for search scenarios [12, 13] in Figure 5. This story can be used for derivation of a waterfall-like approach in Figure 6. We start with initialisation of the search landscape. The result is a search guideline (or search activity meta-model). The information demand is transferred to a search question. The search strategy is configured out of the seven kinds of search. The

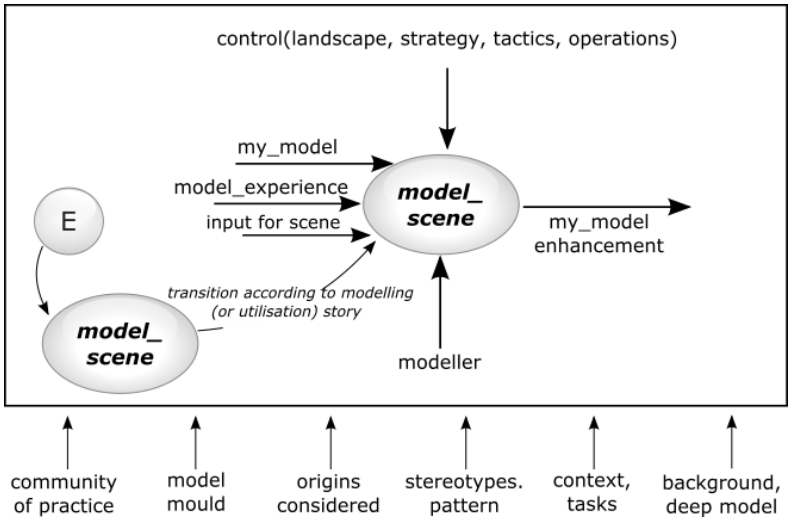

Fig. 4. Model scenes for development (similar to for utilisation)

result is a macro-search model. Selection of the search pattern depends on system information and on the data that is available. The result is a search meso-model, for instance, question-answer forms. Finally we may derive a model on the basis of the data. We might also reconsider the intermediate results and preview or prefetch the potential solutions.

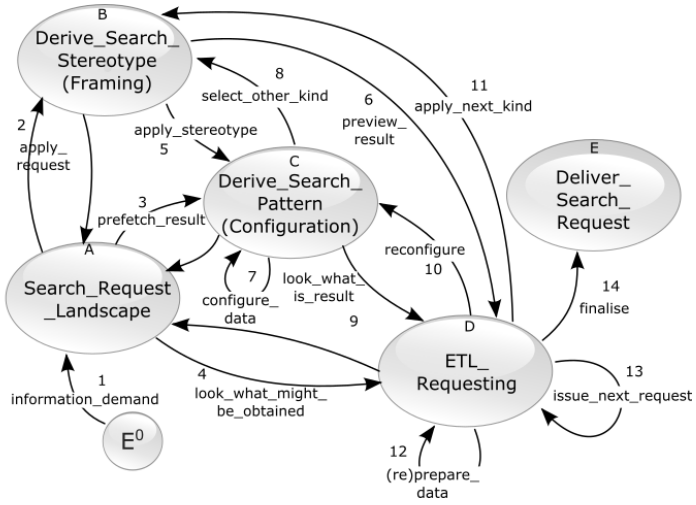

Fig. 5. The layered search story that is used for general search

This story is similar to data mining stories [13, 15]. Data mining uses macro-models as methodological foundation. Frameworks for data mining start with

problem specification and setting, continue with formulation of data mining tasks by means of macro-models, reuse generic models according to required adequacy and dependability, next then select appropriate algorithms according to the capacity and potential of algorithms, prepare furthermore the data mining as a process, and finally apply this process. The data mining mould can be supported by controllers and selectors.

Spaces for Models

Figures 1 and 3 use six planes for detailing models. Each of the planes has it specific quality requirements, support tools, and tasks. At the landscape layer we determine the orientation of the model that should be developed, its problem space, its focus and scope, its integration into the value chain of the application (domain), and its stakeholder from the community of practice with their specific interests and their responsibilities, We rely on mental and codified concepts which are often provided by the world of the origins that a model should properly reflect. The strategic layer adds to this the 'normal way' of development for utilisation of models as methodologies or mould, the embedding into the context and especially the infrastructure, the disciplinary school of thought or more generally the background of the model. The tactics plane embeds the foundations into the modelling process, for instance, by deep model incorporation. It also allows to sketch and to configure the model. The operational plane orients on the formation of the model and the adaptation to the relevant origin(al)s that are going to represented by the model(s). The main issue for the delivery plane is the design of the model(s). The last plane orients utilisation of the model(s) that have been developed. This outer plane might also be structured according to the added value that a model has for the utilisation scenario. Each plane allows to evaluate the model according to quality characteristics used in the sufficiency portfolio.

The model planes have their own workspace and workplaces which are part of the infrastructure for modelling and utilisation.

4 Model Development

Model Development Story

The modelling story consists of the development story and of the utilisation story. The model development story integrates activities like

1. a selection and construction of an appropriate model according to the function of the model and depending on the task and on the properties we are targeting as well as depending on the context of the intended outcome and thus of the language appropriate for the outcome,
2. a workmanship on the model for detection of additional information about the original and of improved model,
3. an analogy conclusion or other derivations on the model and its relationship to the application world, and

4. a preparation of the model for its use in systems, for future evolution, and for change.

Model utilisation additionally uses assured elementary deployment that includes testing and model detailing and improvement. It may be extended to paradigmatic and systematic recapitulation due to deficiencies from rational and empirical perspectives by the way(s) incommensurability to be resolved. Model deployment also orients on the added value in dependence on the model function in given scenarios. A typical model mould is the mathematics approach to modelling based on (1) exploration of the problem situation, (2) development of an adequate and dependable model, (3) transformation of the first model to a mathematical one that is invariant for the problem formulation and is faithful for the solution inverse mapping to the problem domain, (4) mathematical problem solution, (5) mathematical verification of the solution and validation in the problem domain, and (6) evaluation of the solution in the problem domain [1].

Greenfield Development

Although development from scratch is rather seldom in practice nd daily life we will start with the activities for model development. These activities can be organised in an explorative, iterative, or sequential order in the way depicted in Figure 3. We can separate activities into[4]:

(1) Exploration of the origin(al)s what results in a well-understood domain-situation and perception models: The origin(al)s will be disassembled into a collection of units. We ensemble (or monstrate) and manifest the insight gained so far in a domain-situation model and develop nominal or perception models for the community of practice. It is based on a plausible model proposition, on a selection of appropriate language and of theories, on generic models, and on commonsense structuring.

(2) Model amalgamation and adduction is going to result in a plausible model proposition according to the selected aspects of the four aspects. Amalgamation and adduction are based on an appropriate empirical investigation on origin(al)s, on agreed consensus in the school of thought within the community of practice, on hypothetical reasoning, and on investigative design.

(3) Final model formulation results in an adequate and dependable model that will properly function in the given scenarios. We use appropriate depictions for a viable but incomplete model formulation, extend it by corroborated refinements and modifications, and rationally extrapolate the model in dependence on the given ensemble of origin(al)s. In order to guarantee sufficiency of the model, we assess by elementary and prototypical deployment for proper structuring and dependability, within the application domain, within the boundaries of the background, and within the meta-model or mould for model organisation.

[4] As a generalisation, reconsideration of [10]

A number of moulds can be used for refinement of this development meta-model such as agile or experience-backed methodologies Modelling experience knowledge development might be collected in a later rigor cycle (see design science, for instance, [29]). Model development is an engineering activity and thus tolerates insufficiencies and deficiencies outside the quality requirements. A model must not be true. It must only be sufficient and justified. It can be imperfect.

The result of development can also be a model suite or a model capsule. For instance, information system modelling results in a conceptual structure model, a conceptual functionality model, a logical structure and functionality model, and a physical structure and functionality model. It starts with a business data and process viewpoint model.

Model development can be based on a strictly layered approach in Figure 6 that follows the mould in Figure 5 based on planes in Figure 3.

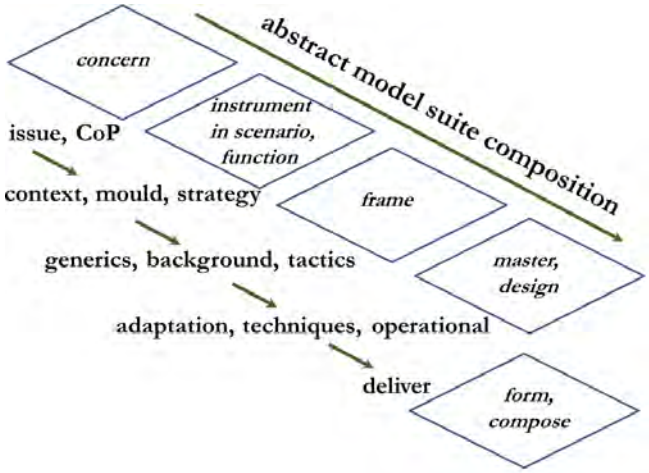

Fig. 6. Model suite development mould for some of the four aspects in Figure 1

Brownfield Development

Modelling by starting from scratch ('greenfield') must be extended by methods for 'brownfield' development that reuses and re-engineers models for legacy systems and within modernisation, evolution, and migration strategies The corresponding model already exists and must be revised. It may also need a revision of its deep sub-model, its basis and grounding, and its ensemble of origin(al)s. All activities used for greenfield development might be reconsidered and revised.

5 Conclusion

Models are widely used and therefore many-facetted, many-functioning, many-dimensional in their deployment, and . Based on a notion of model developed

at Kiel university in a group of more than 40 chairs from almost all faculties, we explore now the ingredients of models. The model-being has at least four dimensions which can be grouped into four aspects: *representation* of origins and their specific properties, providing essential *foundations* and thus *sense-making* of origins, relishing and glorifying models as things for *interaction and social collaboration*, and blueprint for *realisation* and constructions within a *context*. This four-aspect consideration directly governs us during introduction of model suites as a model or model capsules. The utilisation scenario and the function of a given model (suite) determine which of the four aspects are represented by a normal model and which aspects are entirely encapsulated in the deep model .

Models are embedded into their life, disciplinary, and technical environment, and their culture. They reuse intentionally or edified (or enlightened) existing sub-models, pre-model, reference model, or generic models. A model typically combines an intrinsic sub-model and an extrinsic extrinsic sub-model. The first sub-model forms the deep model. For instance, database modelling is based on a good number of hidden postulates, paradigms, and assumptions.

The model-being is thus dependent on the scenarios in which models should function properly. We considered here four central scenarios in which models are widely used: communication, understanding, search, and analysis. These four utilisation scenarios can be supported by specific stereotypes of models which model assembling and construction allows a layered mastering of models. The mastering studio has its workspace and its workplace, i.e. in general space for models.

References

1. Berghammer, R., Thalheim, B.: Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung, chap. Methodenbasierte mathematische Modellierung mit Relationenalgebren, pp. 67–106. De Gryuter, Boston (2015)
2. Box, G.E.P.: Science and statistics. Journal of the American Statistical Association **71**(356), 791–799 (1976)
3. Coleman, A.: Scientific models as works. Cataloging & Classification Quarterly, Special Issue: Works as Entities for Information Retrieval **33, 3-4** (2006)
4. Deicher, S.: The language of objects. BMBF Project KunstModell in Egyptology, https://www.bmbf.de/files/Kurztexte_SdOIII.pdf (2018)
5. Düsterhöft, A., Thalheim, B.: Linguistic based search facilities in snowflake-like database schemes. Data and Knowledge Engineering **48**, 177–198 (2004)
6. Embley, D., Thalheim, B. (eds.): The Handbook of Conceptual Modeling: Its Usage and Its Challenges. Springer (2011)
7. Feyer, T., Thalheim, B.: E/R based scenario modeling for rapid prototyping of web information services. In: Proc.ER '99 Workshops. pp. 253–263. LNCS 1727, Springer (1999)
8. Greefrath, G., Kaiser, G., Blum, W., Ferri, R.B.: Mathematisches Modellieren für Schule und Hochschule, chap. Mathematisches Modellieren - Eine Einführung in theoretische und didaktische Hintergründe, pp. 11–37. Springer (2013)
9. Griethuysen, J.J.V.: The Orange report ISO TR9007 (1982 - 1987) Grandparent of the business rules approach and sbvr part 2 - The seven very fundamental princi-

ples. https://www.brcommunity.com/articles.php?id=b479 (May 2009), accessed Sept. 21, 2017
10. Halloun, I.: Modeling Theory in Science Education. Springer, Berlin (2006)
11. Hunter, P.J., Li, W.W., McCulloch, A.D., Noble, D.: Multiscale modeling: Physiome project standards, tools, and databases. IEEE Computer **39**(11), 48–54 (2006)
12. Jaakkola, H., Thalheim, B.: Supporting culture-aware information search. In: Information Modelling and Knowledge Bases XXVIII. pp. 161–181. Frontiers in Artificial Intelligence and Applications, 280, IOS Press (2017)
13. Jannaschk, K.: Infrastruktur für ein Data Mining Design Framework. Ph.D. thesis, Christian-Albrechts University, Kiel (2017)
14. Kiyoki, Y., Thalheim, B.: Analysis-driven data collection, integration and preparation for visualisation. In: Information Modelling and Knowledge Bases. vol. XXIV, pp. 142–160. IOS Press (2013)
15. Kropp, Y., Thalheim, B.: Data mining design and systematic modelling. In: Proc. DAMDID/RCDL'17. pp. 349–356. FRC CSC RAS, Moscov (2017)
16. Kropp, Y.O., Thalheim, B.: Deep model guided data analysis. In: DAMDID/RCDL 2017, Revised Selected Papers. Communications in Computer and Information Science, vol. 822, pp. 3–18. Springer (2018)
17. Liepsner, T.: Lexikon der Ägyptologie, vol. IV, chap. Modelle, pp. 168–180. Otto Harrassowitz, Wiesbaden (1982)
18. Mahr, B.: Visuelle Modelle, chap. Cargo. Zum Verhltnis von Bild und Modell, pp. 17–40. Wilhelm Fink Verlag, Mnchen (2008)
19. Nissen, I.: Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung, chap. Hydroakustische Modellierung, pp. 391–406. De Gryuter, Boston (2015)
20. Podkolsin, A.: Computer-based modelling of solution processes for mathematical tasks (in Russian). ZPI at Mech-Mat MGU, Moscov (2001)
21. Schewe, K.D., Thalheim, B.: Design and development of web information systems. Springer, Chur (2019)
22. Teeter, E.: Religion and ritual in Ancient Egypt. Cambridge University Press (2011)
23. Thalheim, B.: Towards a theory of conceptual modelling. Journal of Universal Computer Science **16**(20), 3102–3137 (2010), http://www.jucs.org/jucs_16_20/towards_a_theory_of
24. Thalheim, B.: The conceptual model ≡ an adequate and dependable artifact enhanced by concepts. In: Information Modelling and Knowledge Bases. Frontiers in Artificial Intelligence and Applications, 260, vol. XXV, pp. 241–254. IOS Press (2014)
25. Thalheim, B.: Conceptual modeling foundations: The notion of a model in conceptual modeling. In: Encyclopedia of Database Systems. Springer US (2019)
26. Thalheim, B., Jaakkola, H.: Models and their functions. In: Proc. 29'th EJC. p. 150. LUT, Finland, Lappeenranta, Finland (2019)
27. Thalheim, B., Nissen, I. (eds.): Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung. De Gruyter, Boston (2015)
28. Thalheim, B., Tropmann-Frick, M.: Wherefore models are used and accepted? The model functions as a quality instrument in utilisation scenarios. In: Comyn-Wattiau, I., du Mouza, C., Prat, N. (eds.) Ingénierie Management des Systèmes d'Information. pp. 131–143. Cépaduès (2016)
29. Wieringa, R.J.: Design Science Methodology for Information Systems and Software Engineering. Springer (2014)

Between Natural and Human Sciences: On the Role and Character of Theory in Socio-Environmental Archaeology

VPJ Arponen, Sonja Grimm, Lutz Käppel, Konrad Ott, Bernhard Thalheim, Yannic Kropp, Kim Kittig, Johanna Brinkmann, Artur Ribeiro

Abstract

Prominent voices in archaeology have expressed deep skepticism about the role of theory in archaeology while with new, exciting methods at its disposal, archaeological science is occasionally perceived as not needing theory at all. This paper reflects upon the debate about theory in archaeology to arrive at a robust but critical middle range concept of the role and character of theory in socio-environmental archaeology. It is argued that archeology is a data-based science and, consequently, in order for theory to be meaningful in socio-environmental archaeology, theory ought explicitly aim to make its qualitative concepts quantitative to establish a clear relation to data and its interpretation. On the turn side, theory plays an important role critically reflecting upon the use of concepts in archaeological understanding and explanation, as well as their origins in particular paradigms, as examples of which certain debates in scientific archaeology are discussed (aDNA and migration, evolutionism). We argue that such model would serve archaeology far more than a naïve dismissal of theory on the one hand and the continued production of "high" theory in absence of operationalization on the other.

Introduction

Archaeology is a historical discipline between the natural sciences and the humanities. There is a more scientific and positivistic side to archaeology, and there is a more theoretical and speculative side (Sørensen, 2017; Killich 2015; Kristiansen 2014). The division echoes the classic "two cultures" argument made by C.P. Snow (1959). At least since the emergence of postprocessual archaeology, the relationship between archaeological science and archaeological theory has been tense. Our concern in this essay is to attempt to locate the crux of the tension and then attempt an ecumenical but critical account of the role of theory in archaeological science. We write in the context of a larger, interdisciplinary, socio-environmental archaeological research effort the fruits of which this special issue displays.

On the scientific (positivistic, empirical, material) side of archaeology, we find research and data mining at different sites, we find refined dating methods, restoration of fragmentary remnants, inventories, collections, and archives. Remnants are excavated, dated, physically analyzed, and stored. This side represents the actual and solid disciplinary work proceeding according to established as well as innovative scientific methods. It constitutes a growing database. On this what one might call *positivist* side, we see firm and impressive results and we register slow but steady progress over decades, from DNA and isotope analysis to digitalization. One can take this dimension as the hard core of archaeology (Kristiansen, 2014). The history of archaeology, then, can be written as the development "of techniques of recovery and material analysis" (Ion & Barrett, 2016: 133). In this sense, sequencing of ancient DNA, pollen analysis, and isotope analysis would be paradigm examples of scientific progress.

Given the above, it is not obviously wrong to define the epistemic self-understanding of archaeology in a prudent, modest, and enlightened, positivist, research-oriented way. This definition will entail some skepticism of "lofty" or "mere" theoretical speculations.

In works such as John Bintliff's provocative “The Death of Archaeological Theory” (2011, see also Bintliff, 2015), the problematic nature of the relationship of scientific archaeology to theory was discussed. Bintliff argued that

> Published papers increasingly begin with pages of scholastic citation to works of theory, followed by applications to archaeological data which rely more on repeated reference to the favoured approach than providing convincing matching of concepts to recovered material evidence. (Bintliff, 2011: 9)

In other words, theory appears to serve a lofty role detached from the archaeological practice while the real core of the archaeological practice is to be found in rigorous empirical work (see also Johnson, 2006). ~~Bintliff's views were echoed by Matthew H. Johnson who wrote that~~
~~There is, to put it very simply, a disjuncture between what we say we do as ‘archaeological theorists’ and what we actually do as archaeologists […] The case studies offered in support of a particular theoretical position frequently do not match up to the claims made about them in the preceding theoretical excursus. (Johnson, 2006: 118, 119)~~
Arguably, the actual targets of the “death of theory” charge may not so much be theory in the broadest sense —for often, the critics themselves are prolific theorists themselves— but rather some particular instances of (postprocessual) theory.

Nonetheless, ~~In general, still,~~ well-known practitioners of scientifically but also social theoretically informed archaeology continue to be unimpressed by the fruits of the latest theoretical work citing a “a lack of interpretive implementation and progress” (see e.g. Kristiansen, 2017). Elsewhere Kristiansen observed in the literature a wider “critical stance against a previously predominant post-modern/post-processual hegemony, and the reintroduction of a revised modern/processual approach” in archaeology (Kristiansen, 2014: 11). Similarly, in reference to the concept of agency as it came to archaeology from the works of Pierre Boudieu and Anthony Giddens some decades ago, Dobres and Robb (2000: 4) argued that “agency in archaeology is not a theoretically sophisticated paradigm, but rather a lingua franca —an ambiguous platitude meaning everything and nothing”.

A closely related critique of theory is that where the import of a theoretical framework with regard to archaeological interpretation is made explicit, the results regularly fail to convey anything substantially new about the research object at hand. Such a morale arises, for example, from John Barrett's (2014: 68-71) discussion of the inanimate agency thesis in new materialist archaeology, a fairly fresh entrant to the archaeological theoretical scene (Harris & Cipolla 2017; not to be confused with the agency theory mentioned above). In closer scrutiny, says Barrett, either the inanimate agency thesis is vague in its statements as to wherein causality resides in an assemblage of human and non-human things; or it implausibly proposes that causal agent to be able to be a material thing; or, finally, that what the inanimate agency thesis really is proposing is the kind of holism that most would accept anyway: “Archaeologists have long characterized the conditions of the past as the operation of a complex system of relationships between different kinds of component” (Barrett, 2014: 65).

Where, then, does all this leave theory? How, if at all, can it be combined with scientific archaeology? What is the role of theory in archaeology? Despite these difficult questions, a longer standing view is “that everything archaeologists do is infused by theory (much of it, regrettably, still implicit)” (Schiffer, 1988: 461). Similar observations have been made at various junctures over the decades such as Alison Wylie (2002: 7) here:

> observations are theory-laden and richly dependent on extended networks of theoretical claims and assumptions … that include generalizations about observables as well as claims about unobservable dimensions of the reality under study ... These constitute a conceptual framework without which observations have no meaning or evidential import—indeed, without which they cannot be identified as observations.

It is thus not that there is no need for archaeological theory, but we seem to be at loss as to what it can be in the cross-fire of natural and human scientific conceptions.

~~John Bintliff argued that "[a] rehearsal of the old antagonisms between New Archaeology and the postprocessual programme is unproductive, if not tiresome, and it does not seem to me to be taking us anywhere" (Bintliff, 2000: 160). We believe there is nonetheless value in trying to make the fault lines of that antagonism as clear as possible and that is our aim in this paper.~~

~~In a nutshell, according to the analysis of the field put forward in this paper,~~Against the foregoing background, this paper argues that archaeology is essentially a discipline orientated to the extraction and analysis of data, and that being the case, theory can be meaningful in archaeology only if theory connects theoretical concepts with data. That is to say, archaeological theory has to say what patterns in our data does a given piece of archaeological theory lead us to look for. To the detriment of not an insignificant portion of 20th and 21st century archaeological theory, the relationship of theoretical concepts to data and interpretation has unfortunately often been felt to be tenuous.

At the same time, we draw attention to a longer history of conceptual problems arising from the extension of emerging natural scientific techniques and ideas being applied to understand (pre)history that were later criticized for having ignored considerable socio-cultural depth to them. Issues from Social Darwinism via the selfish gene to the modern day archaeological debates about aDN and migration are cases in point. This is a history that any third and subsequent scientific revolutions in archaeology and elsewhere ought to keep in mind. With a view on this history, our paper is a plea for reflective Middle Range theory in archaeology.

~~That said, our paper also makes a case for theoretical reflection as an essential scientific skill. Reflection is characterized as the process of raising (self)awareness about the (implicit) role of paradigms and theoretical concepts in archaeological interpretation including the traversing of natural scientific concepts to human scientific exaplanation. We propose a modest concept of Middle Range Theory at the core of reflective, interdisciplinary socio-environmental archaeology.~~

Middle Range Theory

Any introduction to archaeology will characterize the discipline as essentially orientated to the excavation and analysis of the archaeological record. Therefore, if an archaeological concept means anything, it does so because the concept is somehow, even if indirectly, connected with the kinds of stuff the archaeologist find in the ground. In a phrase, in archaeology, quantitative concepts should be associated with qualitative concepts.

This simple observation allows us to pinpoint the nature of Bintliff's and others' critique of archaeological theory. Basically, in their view, ever since the emergence of postprocessualism, archaeological theorists have been struggling to connect theoretical concepts with archaeological data. As a result, postprocessual archaeological theory has been charged with making grand theoretical assertions but failing to connect these with particular data in a way that is unequivocal and/or substantially new.

We would like to suggest that the concepts of Middle Range Theory (MRT) on the one hand and that of High Theory on the other, can be used to conceptualize the situation. We do not wish to go in too much detail to the long debate about MRT in archaeology (Forslund, 2004) but few words on the notion are in order to make explicit the kinds of issues MRT has historically involved.

The term goes back to the American sociologist Robert K. Merton who in his *Social Theory and Social Structure* (1949, 1957, 1968) criticized the tendency in ~~the~~ American sociology of the time to work with grand theoretical systems (Merton, 1968: 39; Geels, 2007: 628). At the same time, Merton also expressed his criticism towards the opposite approach —small-scale empirical propositions informing day-to-day research. Merton (1968: 44) suggested MRT as a middle way between minor empirical hypotheses and grand theory:

> […] middle-range theory enables us to transcend the mock problem of a theoretical conflict between the nomothetic and the idiothetic, between the general and the altogether particular, between generalizing sociological theory and historicism.

One hallmark of MRT for Merton was its ability to "guide empirical inquiry" (Merton, 1968: 39). Merton (1968: 39) wrote:

> Middle-range theory involves abstractions, of course, but they are close enough to observed data to be incorporated in propositions that permit empirical testing.

The concept of MRT entered archaeology with Lewis Binford (1977) who used the term to refer to the theories of the formation processes of the archaeological record. Others followed with a critique introducing concepts of MRT that were taken as more Mertonian than Binfordian in spirit and letter (Schiffer, 1988; Raab & Goodyear, 1984).

As a result of the multifaceted history of the concept, the words "Middle-Range Theory" tend to evoke many associations in archaeology ~~and its use is rarely precise~~. In our modest concept of MRT, it simply denotes archaeological theory that prioritizes the relation of theoretical concepts to empirical data.

Relatedly, MRT signals a certain desire to mediate "between undesirable extremes", as Frank Geels observed in the context of Science and Technology Studies[1] —the appearance of the concept of MRT is "an indication of discontent in a discipline" (Geels, 2007: 627). This arguably is the situation in archaeology in that there exists the perception that the latest breakthroughs in archaeological theory are failing to make an unequivocal interpretative difference. Its content as a concept aside, MRT is, therefore, quite specifically situated in a particular juncture in the history of archaeology.

High Range Theory and Reflectivity

Generally, theories can so to speak "fly at different altitudes". In biology, a theory within population ecology flies at a different altitude than the general theory of evolution. This holds true also for archaeology. As we saw above, aversion against theory in archaeology often stems from the impression that here is too large a distance between the archaeological record and some "satellite" altitude of general theories stemming from the remote stratosphere of social theories.

[1] Science and Technology Studies are a sociological and philosophical branch of history of science studying social and cultural formation processes of scientific knowledge and theories.

Merton, and several commentators on the MRT debate thereafter, distinguish between high and middle level theory, and sometimes low level theory (Raab & Goodyear, 1984; Smith, 2011). High level theory is by definition something that sets off from fairly abstract (philosophical, if you like) debates about the fundamental nature of something —say, of agency, of materiality, to pick some recent examples (Witmore, 2014; ANONYMIZED; Dobres & Robb 2000). The worry raised by Bintliff and others about "high" theory is that it threatens not to have an obvious empirical application for archaeological purposes. In our view, Bintliff and others' worry is essentially justified.

That said, "high" theor~~yetical~~ and *reflectivity* ought to be considered a part of any scientist's toolkit[2]. In the most general sense, the ~~The~~ concept of reflection ~~can be understood as~~ refers~~ring~~ to the awareness of research traditions or paradigms, thei~~r~~ key "high" theoretical concepts and the influence these have on archaeological interpretation. In a more particular sense, reflectivity critically looks at how concepts are (implicitly) defined and used for explanatory purposes within a given paradigm (Kuhn, 1969; Lucas, 2017). A case in point is the on-going debate about how the aDNA techniques are being used to (implicitly) define migration.

~~Gavin Lucas' (2017) recent discussion of the concept of a paradigm (Kuhn, 1996 [1962]) is very interesting pointing out some important nuances in the term —as well as the loose use the concept is regularly put to in archaeology and beyond. Lucas (2017: 265) notes that one can think of paradigms and research matrices in terms of, both, what they *are* and what they *do*. One of the central ways in which a paradigms *does* something, namely divide the field of science in schools or camps, is by conveying their orientation through classic publications, methods, techniques, instruments, and the like as these are disseminated through research training and communication within a field that shares a paradigm. Taking one approach or another will probably cancel out others (if not by meaning, then simply by time and energy used to pursue one or the other pathway), therefore, paradigms are not a trivial matter.~~

~~Every once in a while, archaeologists implicate themselves as guilty of naïve empiricism, a certain "fetishisation of 'data', 'facts' and quantitative methods" (Sørensen, 2017: 1), the way we look at archaeological features, findings or data and the question we ask are often implicitly directed by those theoretical approaches and certain "controlling models" (Clarke, 1972; Wylie, 2002). Therefore, it is of substantial importance to be aware of how theoretical and other (e.g. political, or ethical) conceptions implicitly may shape archaeological thought —a struggle that surely is difficult and never-ending, so to speak, a hermeneutic circle or spiral. The danger of unreflective and unnoticed migration of concepts from one domain to other is perhaps particularly present in socio-environmental archaeology understood as archaeology working with, both, approaches from the cultural and social sciences as well as environmental natural sciences.~~

A~~tn integral~~ issue is the implicit transmission~~immigration~~ of concepts and thought-models from one domain to another~~,~~ as recently debated in the aDNA studies regarding the concept of migration (Heyd, 2017; Ion, 2017; Furholt, 2018). For decades in archaeology, Gustav Kossinna's concept of monolithic archaeological cultures identified on the basis of shared material culture, and the associated concept of migration as geographical movement of such a monolith, has been discussed critically to the point of rejection (cp. a similar paradigm in "New World archaeology", see Clark, 1993). However, with the rise of aDNA, this concept of migrations has seemingly returned, this time migration being equated with the movement and appearance of certain aDNA in different

[2] Reflectivity is perhaps related but still distinct, in particular in its methods, from Reflexive Archaeology as introduced in archaeological field practice by Ian Hodder (1997; Berggren, 2015).

areas. The debate is on-going, but arguably —alongside technical questions about the adequacy of sample sizes and the like— the danger here is the unreflective equation of the ~~appearance~~"movement" of aDNA from one geographic area to the next with the concept of migration the latter of which arguably contains social, cultural, and political dimensions not visible in ~~bare~~mere transmissions of aDNA. In Ion's words, aDNA "is just data in want of a narrative" (Ion, 2017: 186; see also Gramsch, 2015: 343). ~~Surely, there may have been periods for which demic expansion is the appropriate model, yet the complex nature of the record may not have been sufficiently considered (Gramsch, 2015: 343).~~ In other words, ~~E~~explanatory power has been sought in a reduction to the supposed essentials seemingly allowing the archaeologists not to step into the bog of interpretative socio-cultural particularities in the first place.

An underlying issue we wish to draw attention here is that we have been here before. The history of science knows of cases of how a natural scientific discoveries have first seemed to reveal the nature of things only for significant doubts and reversals to surface later. Thus, ~~I~~in a parallel case of a reduction to the essentials, Richard Dawkins' view on the selfish gene were once, in not too distant past (Dawkins, 1976), used to provide a one stop shop accounts of such arguably quite complex and mixed phenomenon as altruism. Later critics would indeed argue that the reduction to the essentials was mistaken as, once again, what was once thought to be the essential causal core of a phenomenon was probably better thought of as at least partly socio-culturally shaped (Gould, 2002; Sterelny, 2007).

The immigration of natural scientific concepts into human scientific interpretation, of course, has a longer history. A classic example is social evolutionism, inspired by Darwin's evolutionary theory, and leading to the imposition of the concept of stages of development upon the variety of human social and cultural life. The concept of evolution extended to human social and cultural development had the analysts project the concept of evolution upon a domain that, however, worked by rather different principles, as prominent critics such as Tylor, Spencer, and Boas would argue.

In archaeology, Shanks and Tilley's classic critique of evolutionism in archaeology is complex, but one of the key statements was that, in this tradition, "societies were viewed as being involved in an endless series of technologically governed environmental adaptations" (Shanks & Tilley, 1988: 152). In other words, in evolutionistic research, "[t]he search is for universal processes underlying different empirical sequences of societal change, and the reason for this change is environmental adaptation" (Shanks & Tilley, 1988: 140). Shanks and Tilley trace this heritage to a range of literature from Childe (1936) to then-recent work in the 1970s and 1980s by Binford (1972), Flannery (1972), Renfrew (1972), Bintliff (1984), and others.

The alternative to evolutionism Shanks and Tilley proposed was put forward with apologies for the general, outline-like quality —one might say "high" theoretical character— of their remarks (Shanks & Tilley, 1988: 185). A central aspect of it was that

> Societies, unlike individual organisms, do not have any clear-cut physical parameters or boundaries, nor do societies have conscious problems of self-maintenance or a need to adapt. Individuals may have these characteristics but they cannot be validly anthropomorphized in terms of entire social totalities … Societies construct their own social reality and the reproduction of societies entails far more than physical, biological reproduction. (Shanks & Tilley, 1988: 155)

The alternative proposed there essentially says that humans are subject to different, socially constructed determinations that can and do supplant and redefine the environmental constraints. The

jury is essentially still out on this question and both sides are able to field important argumentation in support of their position (ANONYMIZED).

In any case, the present point is, *it is only "high" theory that produces this sort of reflection on the fundamentals of our scientific conduct*. Reasoned and reflective archaeology is better than one conducted unaware~~s~~ of major paradigm alternatives (ANONYMIZED).

Furthermore, reflection may help avoid implausible equations of scientifically detected phenomena (such as demic shifts of aDNA) with ~~im~~migration. Interdisciplinary work environments —ones in which representatives of both "cultures" regularly meet, present, and discuss on-going work— constitute an excellent forum for facilitating awareness and debate. In the end, however, as argued above, archaeology is a fundamentally data-based science, and "high" theory really ought to make the extra effort to descend from principled discussions upon the middle-range level, that is, to questions of operationalization of theoretical insights.

Scales of Transformation

Building on the foregoing metaphor of altitude, we see a continuum of theory formation in archaeology. This array can be organized as different layers or levels of theory formation (cp. Schiffer 1988[3]). The layer model gives a static picture of theoretical altitudes, while a dynamic perspective would explain, why and how theories can "reach" specific different altitudes and how layers can come into contact. What is needed, then, would be a fleshed-out meta-theoretical hierarchical layer model of theories within archaeology. We argue that such model would serve archaeology far more than a naïve dismissal of theory on the one hand and the continued production of "high" theory in absence of operationalization on the other. Our ultimate interest is to improve our explanations through providing an explicit epistemological model of the scales of transformation.

To begin to conceptualize these scales, an analytic distinction between climatic and environmental spheres on the one hand, and social and cultural spheres on the other, may be made (Figure 1). in such a scheme, a reductivist explanation can be understood as one that seeks to collapse the two spheres together again by proposing that some one explanatory factor or factors from one sphere would alone account for the phenomenon under investigation. An example of a reductivist account could be the naïve Marxist view that changes in economic structure in the last instance drive other changes. A second example would be the environmental determinist view that the changing biological frame ultimately drives change in (pre)history.

[3] We adopt Schiffer's (1988) classic model of the structure of archaeological theory in the recognition that there is in fact a plurality of theories used in archaeology. In Schiffer's words, this ranges from theories of Reconstruction of the "cultural and natural past" (Schiffer 1988: 469), via Methodological theory concerning "methods and techniques (of recovery, analysis, and inference)" (Schiffer 1988: 474), to Social Theory, the last of which has been our focus in this paper.

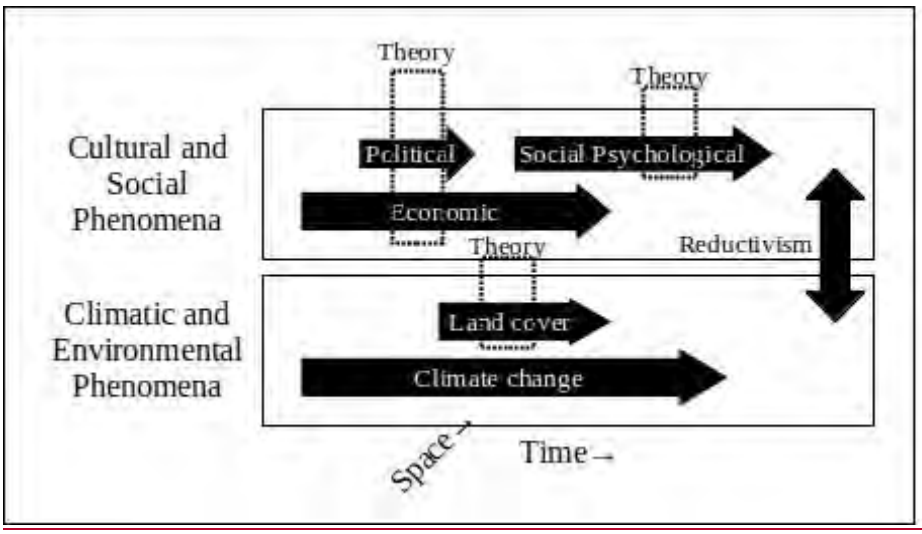

Figure 1. Scales of Transformation and Socio-Environmental Archaeology. The sphere of cultural and social processes is analytically distinguished from the sphere of climatic and environmental processes. Sub-processes and theories of them, within these spheres are characterized by their differing hypothetical temporal and spatial scales (lengths of the arrows).

By contrast, the non-reductivist argument can be made that even if we assume triggers from one sphere —say, changing climate and environment— the logic by which these changes develop in the other sphere would be its own. Thus, for example, differences and changes in the form of governance will crucially affect the way communities deal with environmental challenges (Oliver-Smith 2012, Keyzer 2016).

A second point we wish to argue is, there are differing *spatial* and *temporal* scales at which a given (pre)historic phenomenon could plausibly be said to be occurring which bears some significance to explanation and understanding. To give a simple example, unless we talk about sudden catastrophes (Grattan 2006, Middleton 2017), most climatic and environmental changes occur over a lot longer time spans than most socio-cultural processes do. In modern contexts, this is referred to as the shifting base line: over human generations, changes in the environmental and climate may become imperceptible due to the relative comparative shortness of human memory as well as contextualism pertaining to human perception of the environment. As a rule, phenomena in the socio-cultural sphere can perhaps be said to be occurring in temporally and spatially shorter scales, yet, arguably, there is great variation there in how some economic processes may be global while cultural processes often are more local.

The point is, given the potential differences in the scales in which different phenomena can be plausibly be said to be occurring, socio-environmental archaeology probably ought to be conscious of the differences in the scales. For example, it is not plausible that, say, the temporally long and geographically large scale phenomenon of the neolithisization could be reductivistically understood and explained by appeal to long term climatic-environmental nor by short term ideological changes. In so far as the neolithisization was a spatially and temporally dispersed phenomenon, it's explanation would seem to require as much. The ideal type of socio-environmental archaeology would be one that is conscious of, studies, and provides insights into the different scales of transformation.

References

Barrett ~~AB~~JC (2014) The material constitution of humanness. Archaeological Dialogues 21(1) 65-

74. An integration of integrated information theory with fundamental physics. Front. Psychol. 5: 63.

Ion A & Barrett JC (2016) What kind of archaeology do we want? Introduction. Archaeological Dialogues 23 (2) 131-132.

Barrett AB (2016) A comment on Tononi & Koch (2015) 'Consciousness: here, there and everywhere?' Phil. Trans. R. Soc. B 371: 20140198.

Berggren (2015) Revisiting reflexive archaeology at Çatalhöyük: integrating digital and 3D technologies at the trowel's edge. Antiquity 89: 433–448.

Binford LR (1977) For Theory Building in Archaeology. New York: Academic Press.

Bintliff J (1984) European Social Evolution: Archaeological Perspectives. Bradford: University of Bradford.

Bintliff J (2000) Archaeology and the philosophy of Wittgenstein. *Philosophy and Archaelogical Practice. Perspectives for the 21st Century*: 153 – 172.

Bintliff J (2011) The Death of Archaeological Theory? In: Bintliff J and Pearce M (eds) *The Death of Archaeological Theory?* Oxford: Oxbow Books, pp. 7-22.

Bintliff J (2015) Beyond Theoretical Archaeology: A manifesto for reconstructing interpretation in archaeology. In: Kristiansen K, Smejda L and Turek J (eds) *Paradigm Found. Archaeological Theory, Present, Past And Future. Essays in Honour of Evžen Neustupný*. Oxford: Oxbow Books, pp. 24-35.

Clark GA (1993) Paradigms in Science and Archaeology. *Journal of Archaeological Research* 1(3): 203-234.

Clarke DL (1972) Models and Paradigms in Contemporary Archaeology. In: Clarke DL (eds) *Models in Archaeology*. London: Methuen, pp. 1-60.

Dobres MA, Robb JE (2000) *Ag*ency in Archaeology. London, New York: Psychology Press.

Flannery KV (1972) The Cultural Evolution of Civilizations. Annual Review of Ecology and Systematics 3: 399-426.

Forslund P (2004). MRT Confidential. In: Material Culture and Other Things, Publisher: University of Gothenburg, Department of Archaeology, Editors: Fahlander, Fredrik and Oestigaard, Terje, pp. 213-258.

Geels FW (2007) Feelings of Discontent and the Promise of Middle Range Theory for STS: Examples from Technology Dynamics. Science, Technology, & Human Values 32(6): 627-651.

Giddens A (1979) Central Problems in Social Theory: Action, Structure, and Contradiction in Social Analysis. Berkeley, Los Angeles: University of California Press.

Giddens A (1984) The Constitution of Society: Outline of the Theory of Structuration. Berkeley, Los Angeles: University of California Press.

Gould SJ (2002) The Structure of Evolutionary Theory. New York: Belknap Press.

Gramsch, Alexander (2015) Culture, Change, Identity — Approaches to the Interpretation of Cultural Change. Anthropologie LIII:3: 341-349.

Grattan J (2006). Aspects of Armageddon: An exploration of the role of volcanic eruptions in human history and civilization. Quaternary International 151: 10-18.

Harris OT and Cipolla CN (2017) *Archaeological Theory in the New Millennium: Introducing Current Perspectives*. London: Routledge.

Heyd V (2017) Kossina's smile. *Antiquity* 91(356): 348-359.

Hodder I (1997) 'Always momentary, fluid and flexible': towards a reflexive excavation methodology. *Antiquity* 71: 691–700.

Johnson MH (2006) On the nature of theoretical archaeology and archaeological theory. *Archaeological Dialogues* 13(2): 117–132.

Keyzer, M. D. (2016) 'All we are is dust in the wind: The social causes of a "subculture of coping" in the late medieval coversand belt', *Journal for the History of Environment and Society*. doi: 10.1484/J.JHES.5.110827.

Kuhn TS (1996) The Structure of Scientific Revolutions
. Third Edition. Chicago: University of Chicago Press

Kristiansen K (2014) Towards a New Paradigm? The Third Science Revolution and its Possible Consequences in Archaeology. *Current Swedish Archaeology* 22: 11-34.

Kristiansen K (2017) From deconstruction to interpretation. *Archaeological Dialogues* 24(01): 41-44.

Lucas, Gaving (2017) The Paradigm Concept in Archaeology. *World Archaeology* 49(2): 260-270.

Merton RK (1949) *Social Theory and Social Structure.* New York: Free Press.

Merton RK (1957) *Social Theory and Social Structure. Toward the Codification of Theory and Research*. 2nd ed. New York: Free Press.

Merton RK (1968) *Social Theory and Social Structure*. 3rd ed. Macmillan.

Middleton GD (2017). Understanding Collapse: Ancient History and Modern Myths. London: Cambridge University Press.

Oliver-Smith A (2012). Haiti's 500-Year Earthquake. In Tectonic Shifts: Haiti Since the Earthquake. Schuller, Mark & Morales, Pablo (eds.). Kumarian Press, Sterling, Virginia, USA.

Polanyi K (1944) *The Great Transformation: the Political and Economic Origins of Our Time*.

Boston: Beacon Press.

Raab LM and Goodyear AC (1984) Middle-Range Theory in Archaeology: A Critical Review of Origins and Applications. American Antiquity 49(2): 255.

Renfrew C (1972) The emergence of civilisation. The Cyclades and the Aegean in the third millennium BC. London: Methuen.

von Rüden C (2013) Beyond and East-West Dichotomy in Syrian and Levantine Wall Paintings. In: Brown B and Feldman M (eds) *Critical Approaches to Near Eastern Art*. Berlin: DeGruyter, pp. 55-78.

Schiffer MB (1988) 'The Effect of surface treatment on permeability and evaporative cooling effectiveness of pottery'. In: Hancock RGV and Pavlish LA (eds) Proceedings of the 26th International Archaeometry Symposium. Toronto: University of Ontario, Department of Physics, Archaeometry Laoratory, pp. 23-29.

Shanks M and Tilley C (1988) *Social Theory and Archaeology*. University of New Mexico Press: Albuquerque.

Smith ME (2011) Empirical Urban Theory for Archaeologists. Journal of Archaeological Method and Theory 18: 167-192.

Snow CP (2001) [1959]. *The Two Cultures*. London: Cambridge University Press.

Sørensen TF (2017) The Two Cultures and a World Apart: Archaeology and Science at a New Crossroads, *Norwegian Archaeological Review*.

Sterelny K (2007) Dawkins vs. Gould. New York: Icon Books.

Witmore C (2014) Archaeology and the New Materialisms
. *Journal of Contemporary Archaeology* 1(2) 203-246.

Wylie A (2002) *Thinking from Things. Essays in the Philosophy of Archaeology*. Berkeley:

Conceptual Modelling and Humanities

Yannic Ole Kropp[1] Bernhard Thalheim[2]

Abstract: Humanities are becoming a hyping field of intensive research for computer researchers. It seems that conceptual models may be the basis for development of appropriate solutions of digitalisation problems in social sciences. At the same time, humanities and social sciences can fertilise conceptual modelling. The notion of conceptual model becomes enriched. The approaches to modelling in social sciences thus result in a deeper understanding of modelling. The main aim of this paper is to learn from social sciences for conceptual modelling and to fertilise the field of conceptual modelling.

1 The Value of Conceptual Modelling

1.1 Computer science is IT system-oriented

Computer system development is a complex process and needs abstraction, separation of concern, approaches for handling complexity and mature support for communication within development teams. Models are one of the main artifacts for abstraction and complexity reduction. Computer science uses more than 50 different kinds of modelling languages and modelling approaches. Models have thus been a means for system construction for a long time. Models are widely used as a universal instrument whenever humans are involved and an understanding of computer properties is essential. They are enhanced by commonly accepted concepts and thus become conceptual models. The main deployment scenario for models and conceptual models is still system construction (with description, prescription, and coding sub-scenarios) although other scenarios became popular, e.g. documentation, communication, negotiation, conceptualisation, and learning.

1.2 Learning from Digital Hunanities

Digital humanities become a hyping buzzword nowadays due to digitalisation and due to over-applying computer technology. We have been engaged in a number of projects, e.g. [1, 2, 4, 6, 9]. We step back now and reconsider the challenges to conceptual modelling in

[1] Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, yk@is.informatik.uni-kiel.de
[2] Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, thalheim@is.informatik.uni-kiel.de

these projects and generalize the experience we have gained in these projects. Let us first present a number of observations:

Observation (1): *The concept spaces used in social sciences underpins the conceptual model. Conceptions are systems of concepts. The concept space is typically complex structured. It is used in a multi-viewpoint manifold.*

Observation (2): *Conceptualisation has to be co-considered at various abstraction levels at the same time, e.g. at the micro-, meso-, and macro-level.*

Observation (3): *The mould[3] (and methodology) determines model handling and the utilisation scenarios in which a model functions by playing roles. Models incorporate their function.*

Observation (4): *The model consists of a surface (or normal) sub-model and of deep (implicit, supplanted) sub-models which represent the disciplinary assumptions, the background, and the context. The deep models are the intrinsic components of the model. Conceptualisation might be four-dimensional: sign, social embedding, context, and meaning spaces.*

Observation (5): *Models benefit and suffer from the art of omission. Social and cultural embeddings are considered to be obvious and can thus be omitted.*

Observation (6): *Models may be materialised and then they have a material obstinacy due to the chosen material.*

Observation (7): *Conceptual models have to carry at the same time a manifold of understandings and a manifold of domain-situation models.*

1.3 The storyline

These observations and lessons are useful for conceptual modelling in our area. They are often not explicitly observed in computer science. They are however implicitly used. Think, for instance, on conceptual database models. We often use a conceptual schema that describes the structure of the entire database system and use additionally a number of conceptual views that describe the viewpoints of users of the database. Therefore, we explain now how conceptual modelling can learn from successful approaches in social sciences. The learning process will enhance the added value of conceptual modelling.

2 Learning from Humanities for Conceptual Modelling

According to [5, 10, 13] we define the model notion as follows:

[3] The mould is a hollow form or matrix or simply frame for giving things (such as models) a particular shape. In production, moulds are used as a shaped cavity for forming fluid or plastic things.

> "*A* **model** *is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.*"
>
> *"Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios."*

Well-formedness is often considered as a specific modelling language requirement. The criteria for adequacy are analogy (as a generalisation of the mapping property that forms a tight kind of analogy), being focused (as a generalisation of truncation or abstraction), and satisfying the purpose (as a generalisation of classical pragmatics properties).
The model has another constituents that are often taken for granted. The model is based on a background, represents origins, is accepted by a community of practice, and follows the accepted context. The model thus becomes *dependable*, i.e. it is justified or viable and has a sufficient quality. *Justification* includes empirical corroboration, rational coherence, falsifiability (in our area often treated as validation or verification), and relative stability. The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

2.1 The notion of conceptual model

A notion of conceptual model might be a slim, light, or concise one depending on the level of detail we need in model utilisation. We will use in the sequel one notion, i.e. the concise notion and refer for slim and light versions to [12, 14].

Concise version:
Conceptual Model $\sqsupseteq$ **(Model** $\bigoplus$ **Concept(ion)s)** $\bowtie$ **Enabler** [7]:

> A conceptual model is a model that is enhanced by concept(ion)s from a concept(ion) space, is formulated in a language that allows well-structured formulations, is based on mental/perception/situation models with their embedded concept(ion)s, and is oriented on a mould and on deep models that are commonly accepted.

The mould and the deep models form the matrix of a model [11]. We notice that a conceptual model typically consists of a model suite in social sciences. Each of the models in a model suite reflects some viewpoint or aspect.

2.2 The added value of conceptual modelling

Models do not have to be conceptual models. Conceptual models do not have to be based on an ontology. The main purpose of conception as a system of concept or of a concept(ion) space is the integration of interpretation pattern that ease the communication, understanding, delivery of a model in dependence on the model functions. The concept(ion) space, the mould of model utilisation, and the explicit knowledge of the social determination provide a means for the correct and sufficiently precise interpretation of the model elements.

2.3 The four dimensions of conceptual modelling

The consideration of the strategic, tactical, and operational sides of modelling and of conceptual modelling drives us to consider the four dimensions in Figure 1. These dimensions

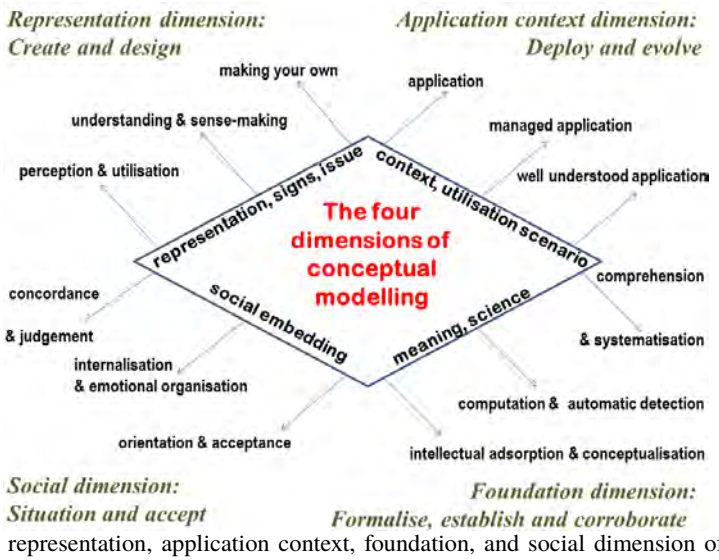

Fig. 1: The representation, application context, foundation, and social dimension of conceptual modelling

cover the application areas in [15] and especially those in humanities. Information systems typically consider the representation dimension and only one of the branches of the foundation dimension. Computer engineering especially considers the application context dimension.

Prescriptive conceptual models that are used as the blueprint for system realisation also consider this dimension. The social embedding is typical for social sciences. The foundation dimension has additional aspects in social sciences since corroboration, comprehension and systematisation are far more complex. Conceptualisation is based on complex concept and conception spaces.

2.4 Handling forgetful mappings to IT and DBMS technology

In is often claimed that conceptual database or data models are mainly descriptive ones. Description is, however, only one of the functions that a conceptual data model has in a system development scenario. Other typical scenarios are documentation, prescription, communication, negotiation, and explanation. These scenarios are also observed for humanities.
In system construction we transform the conceptual data model to corresponding realisation models. This transformation also changes the semantics from rich semantics of conceptual models to lexical semantics which is based on the lexical interpretation of the words used in realisation models according to the meaning in the given application area. It is thus forgetful. The reestablishment of the conceptualisation must thus be handled by a reference to the conceptual model what also means to use a tight bundling of all models in the case of system maintenance (e.g. evolution and migration) and integration. The social dimension and the foundation dimension get also lost during transformation.

2.5 Sophisticated conceptual models are model suites

Based on the observations, we should consider a conceptual model as a model suite, i.e. a coherent collection of explicitly associated models. The associations are explicitly stated, enhanced by explicit maintenance schemata, and supported by tracers for the establishment of coherence [8]. Each model in the model suite has its orientation and its functions in utilisation scenarios. The association schema among the models allows to consider the model suite as a complex but holistic model.
Model suites in most sciences and engineering incorporate some conceptual models. This situation is not different for social sciences. For instance, the CRC 1266 [1, 2] uses as a complex model of transformation a model suite consisting of models for socio-economic formation (cluster B-E), for socio-environmental components of change (cluster F), and for natural science investigation (cluster G). The interplay of these models allows to suppose hypotheses and to draw conclusions. Most models are already conceptual ones. They use, however, different conception spaces. The association among these models is handled by interlinkage groups within the CRC.

2.6 Models as mediating instruments instead of middle-range theories

Middle-range theories [2] are essentially model suites. They are used for an integrating consideration of quantitative sources and theory conceptions. Quantitative sources are used for derivation of quantitative concepts. The theory offer underpins these concepts. Qualitative theory-oriented research uses theoretical concept(ion)s. These concepts are supported by supporting sources which are often generated before and might use the current

quantitative sources. A theory integrates these concepts. We use typically several theories, e.g. for plausibility check, for investigation, explanation, knowledge experience propagation, and discovery scenarios. In a proxy-based research we start with proxy sources that might be underpinned by proxy concepts. This research results in a theory request that can be satisfied by a theory offer.
This approach often results in a gap between qualitative and quantitative research. Models can be used to render the theory offer. At the same time, models may also render a qualitative theory. The rendering procedures are typically different. A model suite can now be constructed by models for theoretical concepts from one side and by models for quantitative concepts from the other side. In this case, we use models for the quantitative theory offers and for the qualitative theories. This approach is depicted in Figure 2.

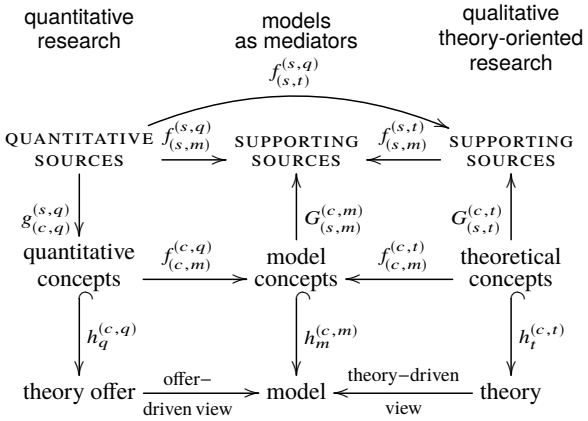

Fig. 2: Models as integrating and mediating instrument for conceptualisation, investigation, explanation, knowledge experience propagation, and discovery

This approach has already used for the investigation in the CRC 1266 [2]. In a similar form we can consider now conceptual models for other application cases.

3 Concluding: Conceptual Modelling Inspired by Humanities

Conceptual modelling is a widely used practice in many science and engineering disciplines. The current practice used for database conceptualisation can be enhanced by a number of insights that we observed in social science research.

- The concept(ion) space is often far more complex structured than finally represented and used for a singleton conceptual model. We should consider conceptual models that orient on different aspects and different levels.
- The context dimension should not be neglected for conceptual models.

- The social dimension and the foundation dimension are equally important as the representation dimension.
- Model and especially conceptual models consist of a number of models and thus form a model suite.

We got now additionally a number of special necessities for conceptual modelling without which conceptual models are of low quality, not justified, and also not adequate.

Deep models: Models consist of normal sub-models and deep sub-models. The first ones are given in an extrinsic and explicit form. The later ones are often concealed.

Model mould: The second element of the matrix of modelling is the mould. We know a number of canonic approaches that guide the modelling process, the modelling outcome, and the capacity of the finally developed model.

Concept-biased modelling: Conceptual models are typically deeply biased by the concepts in a given domain. Concepts such as "village", "settlement" and "center" are essentially representing the same understanding but are used in very different contexts. The same applies to database models, e.g. the concepts of "Person" or "Address" depend on geographic, law etc. assumptions.

Functions of models as the guiding principle: The utilisation scenarios determine the functions that a model has in such scenarios. The model is an instrument in these scenarios. Whether it is a proper and fit-to-use instrument depends on the function the model has (and thus on the purpose and the goal).

Bibliography

[1] CRC 1266. Scales of transformation - Human-environmental interaction in prehistoric and archaic societies. Collaborative Research Centre. http://www.sfb1266.uni-kiel.de/en/, accessed May 13, 2018.

[2] V.P.J. Arponen, S. Grimm, L. Käppel, K. Ott, B. Thalheim, Y. Kropp, K. Kittig, J. Brinkmann, and A. Ribeiro. Between natural and human sciences: On the role and character of theory in socio-environmental archaeology. *The Holocene*, 29(10, Special Issue "Scales of Transformation: Human-Environmental Interaction in Prehistoric and Archaic Societies"):tba, October 2019.

[3] A. Dahanayake and B. Thalheim. Development of conceptual models and the knowledge background provided by the rigor cycle in design science. In *Models: Concepts, Theory, Logic, Reasoning, and Semantics*, Tributes, pages 3–28. College Publications, 2018.

[4] S. Deicher. The language of objects. BMBF Project KunstModell in Egyptology, https://www.bmbf.de/files/Kurztexte_SdOIII.pdf, 2018.

[5] D. Embley and B. Thalheim, editors. *The Handbook of Conceptual Modeling: Its Usage and Its Challenges*. Springer, 2011.

[6] GSHDL. Graduate school at Kiel University Human Development in Landscapes. http://www.gshdl.uni-kiel.de/, accessed July 26, 2019.

[7] H. Jaakkola and B. Thalheim. Cultures in information systems development. In *Information Modelling and Knowledge Bases XXX*, pages 61–80. IOS Press, 2019.

[8] M. Skusa and B. Thalheim. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*, chapter Kohärente Multi-Modell-Entwicklung, pages 431–454. De Gryuter, Boston, 2015.

[9] The DigiCULT team. DigiCULT: a portal of the museum network at Schleswig-Holstein and Hamburg. http://www.digicult-verbund.de/de and http://www.museen-sh.de/, accessed July 26, 2019.

[10] B. Thalheim. The conceptual model ≡ an adequate and dependable artifact enhanced by concepts. In *Information Modelling and Knowledge Bases*, volume XXV of *Frontiers in Artificial Intelligence and Applications, 260*, pages 241–254. IOS Press, 2014.

[11] B. Thalheim. General and specific model notions. In *Proc. ADBIS'17*, LNCS 10509, pages 13–27, Cham, 2017. Springer.

[12] B. Thalheim. Conceptual model notions - a matter of controversy; conceptual modelling and its lacunas. *EMISA International Journal on Conceptual Modeling*, February:9–27, 2018.

[13] B. Thalheim. Conceptual modeling foundations: The notion of a model in conceptual modeling. In *Encyclopedia of Database Systems*. Springer US, 2019.

[14] B. Thalheim. Conceptual models and their foundations. In *Proc. MEDI2019, LNCS 11815*, pages 123–139. Springer, 2019.

[15] B. Thalheim and I. Nissen, editors. *Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung*. De Gruyter, Boston, 2015.

Remark

We thank the reviewers for their remarks, suggestions, and critics. The paper is based on our previous papers on models, e.g. on the generalisation of approaches used in design science research [3]. The compendium [15] presents model notions and modelling used in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports at Kiel university. It is based on a decade of Tuesday-evening-open-end discussions on models and modelling in sciences. We selected only four of our collaboration projects from humanities research and discussed some of the modelling lessons.

Acknowledgement

This research was performed in the framework of the CRC 1266[4] '*Scales of Transformation - Human-Environmental Interaction in Prehistoric and Archaic Societies*' which is funded by the *Deutsche Forschungsgemeinschaft* (DFG, German Research Foundation - Projektnummer 2901391021 - SFB 1266)[5]. We thank both institutions for enabling this work.

[4] http://www.sfb1266.uni-kiel.de/en/
[5] http://www.dfg.de/en/index.jsp