1 Exposé

1.1 Motivation

Since the beginning of times there exists a notion of distributing the workload between different stakeholders, who were best fit to collect, hunt, cook the food, or build the accommodation. The method "separation of concerns"[1] increased the efficiency of work of individuals and, consequently, of the whole undertaking, as we see it in construction industry. In the early stages of economics emerging as a science Smith[2], Ricardo[3], Marx[4], and other authors reported of their empiric and scientific research results. Computer era changed little on this principle: on the contrary, this old approach, proven by generations, has been confirmed anew and taken over as modus operandi. In the software industry new professions emerged: software developer and product owner, which could be compared to the established construction professions as stonemason and engineer. The difference lies in the working material - the stone blocks are the software modules. System construction principles have been partially transferred by scientists and practitioners into the software industry: a differentiation between programming and modelling has been made.[5,6,7]

Similarly, both in construction and software engineering there exist **customers**, who wish a certain product to be developed for their needs, let it be a building for company headquarters or a full-scale enterprise resource planning software for company information and communication management support. Those customers need their products consistently well built and as soon as possible, which constitutes the time-to-market conflict. To resolve this seemingly eternal confrontation between customers and manufacturers, many approaches have been elaborated in software industry, e.g. software engineering process models as waterfall, spiral and incremental models, agile programming, V-model, and others.[8]

Whilst construction sites grow larger, issuing such great anthropomorphic achievements as Shanghai Tower[9] or Burj Khalifa in Dubai[10], so does the **complexity** of planning management and information delivery. Nobody has the perception that everybody would know everything about the project under development – after a certain degree of throughput humans wouldn't

---

[1]see Vinay Kulkarni et al. (2003). 'Separation of concerns in model-driven development'. In: *IEEE Software* 20.5, pp. 64–69. issn: 0740-7459.

[2]see Adam Smith (2012). *The Wealth of Nations*. Ed. by Tom Griffith. Hertfordshire: Wordsworth Editions, p. 974. isbn: 978 1 84022 688 1.

[3]see David Ricardo (1821). *On the Principles of Political Economy and Taxation*. url: http://econpapers.repec.org/RePEc:hay:hetboo:ricardo1821 (visited on 26th Aug. 2019).

[4]see Karl Marx (1873). *Das Kapital*. Hamburg. url: www.archive.org/download/KarlMarxDasKapitalpdf/KAPITAL1.pdf (visited on 26th Aug. 2019).

[5]see Klaus Müller (1996). *Allgemeine Systemtheorie: Geschichte, Methodologie und sozialwissenschaftliche Heuristik eines Wissenschaftsprogramms*. Vol. 164. Westdeutscher Verlag.

[6]see Mohammad Jamshidi (2011). *System of systems engineering: innovations for the twenty-first century*. Vol. 58. John Wiley & Sons, Inc.

[7]see Charles Keating et al. (2011). 'Systems of systems engineering: prospects and challenges for the emerging field'. In: *International Journal of System of Systems Engineering* 2.2-3, pp. 234–256.

[8]see Helmut Balzert (2008). *Lehrbuch der Softwaretechnik: Softwaremanagement (German Edition)*. Spektrum Akademischer Verlag. isbn: 3827411610.

[9]see Jian Jiang et al. (2015). 'Fire safety assessment of super tall buildings: A case study on Shanghai Tower'. In: *Case Studies in Fire Safety* 4, pp. 28–38.

[10]see Construction - Burj Khalifa: https://www.burjkhalifa.ae/en/the-tower/design-construction/ (visited on 26th Aug. 2019)

be able to process the whole of the information flow and still have time to react accordingly. The same is valid for the software industry, hence a notion of separation of concerns in both of the branches comes in handy: give every individual the required information at the proper time, at the proper place, and in the proper amount, hereby you will conquer the complexity in a team. In software industry, many concepts have been proposed for that purpose: roles[11,12], viewpoints[13], workflows[14,15], and so on.

Another aspect that is present in both branches is the concept of **reverse engineering**, which is a process of reconstructing the models of an existing system. Imagine, for starters in the construction industry, an existing building, for which the architectural plans and documentation got lost or simply grew so old that very few specialists could maintain it. This constitutes the legacy systems notion in system engineering. To aid the situation in construction, newly adjusted plans and sketches should be created, either using older documentation or even starting from scratch. In software industry, again, many methods have been created to give legacy systems life anew: data mining[16,17], process mining[18], executable models[19], networking interoperability[20] and other approaches.

In addition to all listed obstacles on the way to successful goal achieving the customers throw in their two pennies worth: their **requirements** may also change. Depending on the construction stage we could imagine there would be no choice as to rip the whole building down trying to comply with a completely new set of demands. Some engineers may also miscalculate or not take enough requirements into account for one or another reason, which has indeed taken place in 19th century in Angers, France during a suspension bridge construction in 1836-1839. Not covering all the needed requirements resulted in a disaster in 1850: while a soldiers' regiment was crossing the Maine river in a thunderstorm, the upstream anchoring cable on the right bank broke and lead the complete bridge to collapse, afterwards with 226 people dead in total.[21] In order to be able to handle such situations in software industry, the following concepts arose: traceability for managing and solving changing requirements[22], impact analysis for assessing the resulting changes on different levels of abstraction[23], data evolution for applying the life-

---

[11]see Ian Sommerville (2006). *Software Engineering*. Vol. 8. Addison-Wesley. isbn: 0-321-31379-8.

[12]see Markus Völter et al. (2013). *Model-driven software development: technology, engineering, management*. John Wiley & Sons, Inc.

[13]see Gerald Kotonya et al. (1992). 'Viewpoints for Requirements Definition'. In: *Software Engineering Journal* 7.6, pp. 375–387.

[14]see Nenad Medvidovic et al. (2000). 'A classification and comparison framework for software architecture description languages'. In: *IEEE Transactions on Software Engineering* 26.1, pp. 70–93. issn: 0098-5589.

[15]see Michalis Miatidis et al. (2005). 'Towards an Integrated Modelling Framework for Engineering Design Processes'. In: *EMISA, Proceedings of the Workshop in Klagenfurt October 24-25, 2005*. Ed. by Jörg Desel et al. Vol. P-75. Lecture Notes in Informatics. Gesellschaft für Informatik, Bonn, pp. 36–49.

[16]see Pang-Ning Tan et al. (2006). *Introduction to data mining*. Vol. 1. Pearson Addison Wesley Boston.

[17]see Xavier Amatriain et al. (2011). 'Data mining methods for recommender systems'. In: *Recommender Systems Handbook*. Springer, pp. 39–71.

[18]see Wil van der Aalst (2012). 'Process Mining: Overview and Opportunities'. In: *ACM Trans. Manage. Inf. Syst.* 3.2, 7:1–7:17. issn: 2158-656X.

[19]see Gergely Dévai et al. (2015). 'UML Model Execution via Code Generation'. In: *Proceedings of the 1st International Workshop on Executable Modeling co-located with {ACM/IEEE} 18th International Conference on Model Driven Engineering Languages and Systems {(MODELS} 2015), Ottawa, Canada, September 27, 2015.* Pp. 9–15.

[20]see Michael Karrenbauer et al. (2019). 'Future industrial networking: from use cases to wireless technologies to a flexible system architecture'. In: *at-Automatisierungstechnik* 67.7, pp. 526–544.

[21]see Bridgemeister `http://www.bridgemeister.com/bridge.php?bid=993` (visited on 26th Aug. 2019)

[22]see Jane Cleland-Huang et al. (2005). 'Goal-centric traceability for managing non-functional requirements'. In: *Proceedings of the 27th international conference on Software engineering*. ICSE '05. New York, NY, USA: ACM, pp. 362–371. isbn: 1-58113-963-2.

[23]see Robert Arnold et al. (1993). 'Impact Analysis-Towards a Framework for Comparison.' In: *ICSM*. vol. 93, pp. 292–301.

cycle models on growing enterprise data warehouses[24], etc. These approaches help keeping the constantly changing requirements and the system specifications aligned and up-to-date.

Harnessing the complexity of information system (IS) development during the design time is one of the essential tasks for the IS architects creating blue-prints for enterprises. Having instruments at hand for filing in the **requirements** and being able to trace their possible changes down to the last piece of software code in the functioning enterprise resource planning system would be a perfect set for every IS solution architect, but it is rather far from reality. Building a system according to the defined roles for every kind of industry and being able to define new **viewpoints** might be a helpful feature. This way, stakeholders would efficiently perceive the information system under construction, which is sadly rather lost in the sea of existing standards and numerous unaligned definitions. Should there exist qualitatively precise methods for estimating the **impact analysis of model changes** in the functioning enterprise information systems out of the previously defined models, this would be a perfect decision support for managing the introduction of changes into the existing IS design. Tooling support for **every stage of IS design** and connection to the development teams and stakeholders on the way to implementing the complete enterprise system would bring a lot of clarity into the day-to-day work in the IT world.

1.2 Problem Statement

During both construction and software engineering, numerous errors, defects, incidents, and misconceptions could take place. There are different means for their thorough identification, containment and correction in both industries, although obviously notorious gaps in structuring the expert knowledge and planning the technical system still remain. In the elicitation of the problem areas below, we follow the notion of comparison of similarities between construction and software engineering industries in the motivation section 1.1, notwithstanding that their obvious differences in the analysis and production fields are taken into account, but are deliberately not discussed here.

For a certain building to gain form, customer stakeholders shall present a notion of what they intend to see after production is complete. The lack of clearly stated requirements in form of, e.g., an architectural blue-print might lead to additional efforts during the running projects, causing considerable costs and shifted deadlines. Similar situations might occur during the design of an enterprise system in software engineering, should there be no traceability between the stated requirements and according models on lower abstraction levels in the model-driven architectures.[25] Thus, the first problem area is **incomplete methodology** for *requirements in model-driven architectures*: during requirements elicitation in the phase of systems analysis and specification, circumstances dimming the need for further requirements or stating them incompletely could occur.[26] Technological difficulties during system specification or even deployment may also influence the need for adjusting or even refactoring the requirements stated in the beginning. The following points are relevant for consideration during requirements specification:

- notion of modelling the requirements within the Computation Independent Model (CIM)

[24]see Patrick Maeder et al. (2006). 'Traceability for Managing Evolutionary Change - A Roadmap'. In: *SEDE*, pp. 1–8.

[25]see Dmitri Valeri Panfilenko, Roman Litvinov et al. (2011). 'Traceability and Viewpoints in Enterprise Architectures'. In: *ICEIS 2011 - Proceedings of the 13th International Conference on Enterprise Information Systems*. Vol. Vol. 3. International Conference on Enterprise Information Systems 13th. Beijing, China: SciTePress, Pages 150—156.

[26]see Bernhard Schatz (2011). '10 Years Model-Driven – What Did We Achieve?' In: *Proceedings of the 2nd Eastern European Regional Conference (EERC) on the Engineering of Computer Based Systems (ECBS)*. IEEE. Bratislava, Slovakia.

of Model Driven Architecture®(MDA®);[27]

- need for traceability of the requirements down to the lower abstraction levels;[28]
- bijective relation and influence between requirements and their instances on the adjacent levels of abstraction;[29]
- stakeholder requirements assignment handling.

Another problem area is delivering the required information to the right people at the right time at the right place, thus supporting the construction problem and pro-actively avoiding failures. Suppose the architects would clearly state the blue-print and document it fully, but the construction engineer would not get any notion of it whatsoever or would get a part of the blue-print not corresponding to the current construction state. Analogously, the software engineers are in need of access to the traceable requirements during their work and would otherwise produce erroneous software modules. Thus, the second issue is stated as **lacking qualitative methods** for *thorough definition and usage of viewpoints*. There is a number of viewpoint definitions and notions, which are not defined explicitly, but act in a similar way.[30] Albeit, neither there is a unity in how to define the viewpoints conceptually, nor in how to use them technically, nor what information should be provided. Based on this consent, clarity could be introduced on what are the best practices for information systems viewpoints design and implementation. For that, the following aspects are of utter importance:

- definition of the viewpoints based on the existing approaches;[31]
- technical proposal for handling viewpoints;[32]
- creation of viewpoints out of the existing ones;[33]
- approach for viewpoints assignment depending on the context and stakeholders.

The previously sketched problems of changing requirements and their management with the support of traceability techniques have been mentioned. Changing an architectural blue-print means losing not only the immediate changes influencing the neighbouring construction blocks and requirements, but also a need for recalculation of the whole statics measurements, as the interdependencies with more distant construction parts might be affected, as well. The same is valid for software engineering, especially for model-driven architectures, as changing a model class not only affects the neighboured classes on the same abstraction level, but also impacts the adjacent levels and class-descendants. Thus, the third problem is defined as **lacking qualitative methods** for *model changes and transformation impact analysis evaluation*. During information system elaboration using abstraction layers, models residing on each of them should be connected to

[27]see Object Management Group (2003b). *MDA Guide Version 1.0.1.* url: `http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf` (visited on 26th Aug. 2019).

[28]see Object Management Group (2012a). *Compliance and Requirement Traceability for SysML v.1.0a.* url: `http://www.sysml.org/docs/specs/Compliance-RTM-SysMLv1.0a.pdf` (visited on 26th Aug. 2019).

[29]see Dmitri Valeri Panfilenko, Christian Seel, Keith Phalp et al. (2011). 'Enriching the Model-Driven Architecture with Weakly Structured Information'. In: ed. by J Rech et al. Emerging Technologies for the Evolution and Maintenance of Software Models. IGI Global, Pages 121—145.

[30]see International Organization for Standardization (2011a). *ISO/IEC/IEEE 42010:2011 - Systems and software engineering - Architecture description.* url: `http://www.iso.org/iso/catalogue%7B%5C_%7Ddetail.htm?csnumber=50508` (visited on 26th Aug. 2019).

[31]see Mehrdad Sabetzadeh et al. (2010). 'Viewpoints'. In: *Encyclopedia of Software Engineering.* Ed. by P Laplante. New York: Taylor and Francis, pp. 1318–1329.

[32]see Clive Finkelstein (2006). *Enterprise Architecture for Integration: Rapid Delivery Methods and Technologies.* Artech House Publishers. isbn: 1580537138.

[33]see Dmitri Valeri Panfilenko, Christian Seel, Andreas Martin and Peter Loos (2010). 'The VCLL: A MultiView Computation Independent Modeling Language for MDA-Based Software Development'. In: *Pre-ICIS 2010 Workshop on Enterprise Systems Research. International Conference on Information Systems (ICIS-10), Information Technology: Gateway to the Future, December 12-15, St. Louis„ Missouri, USA.* International Conference on Information Systems. St. Louis, Missouri, USA: Information Technology: Gateway to the Future.

the according ones on adjacent levels. Failure to provide the needed traceability and mechanisms for tracking the relations between models on different abstraction layers may result in unpredictable behaviour, possibly even inconsistency due to formalisation of specifications.[34] There are several prerequisites for providing the proper techniques, which are yet to be organised:

- definition of the domain-specific languages (DSL) for model descriptions;
- model transformation language specification and usage guidelines;
- approaches for tracing the changes in all kinds of models;[35]
- appropriate recommendations for supporting model developers.[36]

In order to be able to correctly execute constructing activities and afterwards be in a position to trace possible incidents and failures back to the defects and errors during design phase, an extensive tooling support is required. This constitutes the problem area, where the tools are not only used for construction execution, but also for documentation and backwards tracing. This is an important aspect in both industries, as (software) architects and engineers need to understand the language of the expert requirements on the one side and the technical requirements on the other side likewise. Thus, the following rather tooling-specific problem area is specified as follows - **missing support** for *full information systems management cycle*: despite numerous attempts, until now there is no tool on the market supporting the complete information systems management cycle.[37] IS management should be starting in the early stages of requirements analysis, going through sequential abstraction layers towards generation and deployment of the system source code, and following the tracks back from the system at the run-time to the highest abstraction level. There are several reasons that may be taken into account:

- complicated deployment to different platforms;[38]
- no need for reverse part of the cycle;
- too vast complexity in different branches for forward engineering;[39]
- general lack of interest in or rather disbelief in the community in such an effort.

1.3 Research Objectives and Questions

Previous sections gave an overview of the general areas under investigation and the specific problems appearing in those areas. Hence, the main research direction of this thesis is **the elaboration of a viewpoint-based model-driven information system architecture engineering method**. More fine-grained, the *objectives of this thesis*, leading to the research questions, shall envision the following

- **Objective I**: Investigate, elicitate, elaborate and classify the requirements for the viewpoint-based model-driven information system architecture engineering methods.

[34] see Valeri Pavlovic Panfilenko (1993). 'Consistency of formalized specifications in multilevel programming'. In: *Cybernetics and Systems Analysis* 29.2, pp. 210–219.

[35] see Ming Hao et al. (2006). 'Business process impact visualization and anomaly detection'. In: *Information Visualization* 5.1, pp. 15–27.

[36] see Dmitri Valeri Panfilenko, Andreas Emrich et al. (2014). 'Recommendations for Impact Analysis of Model Transformations'. In: *ICEIS 2014 - Proceedings of the 16th International Conference on Enterprise Information Systems*. International Conference on Enterprise Information Systems 16th. Lisbon, Portugal: INSTICC, pp. 428–434.

[37] compare OMG - MDA Tools: `http://www.omg.org/mda/committed-products.htm` (visited on 26th Aug. 2019)

[38] see Hector Florez et al. (2012). 'Coevolution Assistance for Enterprise Architecture Models'. In: *Proceedings of the 6th International Workshop on Models and Evolution*. ME '12. New York, NY, USA: ACM, pp. 27–32. isbn: 978-1-4503-1798-6.

[39] see Thomas Vogel et al. (2014). 'Model-Driven Engineering of Self-Adaptive Software with EUREMA'. in: *ACM Trans. Auton. Adapt. Syst.* 8.4, 18:1–18:33.

For that, a feature-based overview of the existing methods and a comparison to the method proposed in the thesis should be performed.

- **Objective II**: Based on the artefact from the previous objective, propose an adaptive viewpoint-based information system architecture (ISA) engineering method for bidirectionally closing the gap between domain expert and technical specification levels, which includes viewpoints recommendation support for model transformations.
  In order to prove that the method works efficiently, a certain degree of improvement in development speed on different stages should be verifiable.
- **Objective III**: Conceive, design, implement and evaluate a viewpoints-based model-driven information system architecture engineering method, which provides support for business process life-cycle.
  This solution has to be evaluated through real-world application cases showing the applicability and improvements of the system design.

The following **research questions** arise from those elaborations on problem statements and research objectives that this thesis is addressing. In the first place, the essential terms for flexible information systems architectures shall be named, defined, and discussed in chapter 2, whereas the general information on this kind of ISA and its position in business informatics is discussed in chapter 3. Thus, the first research question reads as follows:

Research Question I

What are the constituent terms and aspects of flexible information system architectures? How can those architectures be defined, delimited and classified for efficient and flexible usage in business administration and management?

Along with the essential viewpoint terms, especially the flexibility analysis approaches and flexibility term delimitation, FISA definition and classification will have to be completed in order to answer to the first part of research question I. Business and IT alignment notion in relation to enterprise architecture management and governance, as well as viewpoint definition and viewpoint-based method classification criteria shall illustrate the second part of research question I. Further, what are the existing methods for constructing multi-viewpoint enterprise information systems and how do they differ from each other? What are the different approaches to defining model transformations and analysing impact induced by the model changes onto adjacent abstraction levels? The related work referred to in these questions is listed and analysed in chapter 4, shaping the following research question:

Research Question II

What are the classification strategies for viewpoint definitions and engineering methods for viewpoint-based enterprise architectures? What are classification aspects of model to model transformations and impact analysis strategies?

To answer research question II, a selection of viewpoint definitions and viewpoint-based engineering methods shall be surveyed and classified according to the criteria synthesised during answering research question I. Moreover, model-to-model and impact analysis classification and assessment of usage feasibility for viewpoint-based engineering shall be provided. The knowledge of different aspects of enterprise systems architectures and their usage in engineering processes leads to feature extraction and their comparison, thus shaping a set of requirements for such viewpoint-based architecture definition. These requirements are elicitated in chapter 5 and their fulfilment is evaluated in chapter 7, thus yielding the following research question:

Research Question III

What are the constituent features for supporting the architecture engineering methods? How do the derived requirements shape the viewpoint-based model-driven information system architecture engineering method and tooling support?

Survey of the open source and proprietary viewpoint supporting solutions shall help defining the constituent features for viewpoint-based engineering method support. Synthesis of requirements both for a viewpoint-based method and tooling shall constitute the answer of research question III. The derived requirements for method elaboration make the concept aspects elicitation organised in an efficient way. What are the models and metamodels provided for the definition of viewpoints, how are model transformations defined, what kinds of impact analyses are conceivable in the course of a viewpoint-based information system architecture elaboration? The concept for closing the gap between domain expert knowledge and the technical level, including model transformation and evaluation features, is elaborated throughout chapter 6, thus shaping the following research question:

Research Question IV

What is a generally efficient procedure model to bidirectionally close the gap between domain expert and technical specification levels through viewpoints provision? What are the strategies for recommendations on model transformations based on impact analysis approaches?

Constructing a reference procedure model and term definitions for viewpoint-based methods along with metamodels and their instantiation shall answer the first part of research question IV. A viewpoint-based engineering method including application reference models and viewpoint maintenance in form of set extension or shrinking is providing the detailed elaborations for this first part. Describing the viewpoint-based method techniques on model transformations, impact analysis and viewpoint recommendation engineering shall cover the second part of research question IV. Having conceived and designed any approach, it is advisable to apply the concept on real-world scenarios, evaluate their suitability for the envisioned objectives, and assess further research and development demand. Tooling design and evaluation in chapter 7 lead to the following research question:

Research Question V

How can one, from the tooling point of view, best support the impact analysis and model transformations for model changes with the aid of viewpoints? How can one most efficiently provide support for changing user requirements by thorough viewpoint definition and assignment in application cases from steel production legacy software and automotive safety systems?

A proof-of-concept evaluation approach along with an architecture engineering tool prototype shall illustrate the first part of research question V. Evaluation of the viewpoint-based method and tooling requirement fulfilment on the basis of application case instantiation are required to deliver the answer to the second part of research question V. The research objectives and questions derived above, also partly the elaboration of the solutions, as well as some of the research results are connected to the research projects SHAPE[40] and ViBaM[41] funded by CORDIS FP7[42] and Eureka Eurostars[43] European research programmes respectively.

1.4 Scientific Method Placement

Elaboration of the problem areas in the motivation and problem statement sections lead to the following considerations. Firstly, the topic under concern is in the field of software systems engineering and thus belongs to the information systems engineering and design.[44,45] Secondly, many methods have been created and elaborated, mitigating the lack of concepts for developing consistent and sustainable information systems.[46,47,48] Thirdly, according to systems engineering, there are many abstraction layers in software engineering.[49] Fourthly, the relations between models on different abstraction layers and their interweaving has always been a problem on the conceptual, as well as on the technical level.[50,51,52,53]

The framework of the research in this thesis, i.e. proof-of-concept through application prototyping, adheres to Hevner Design Science Cycles elaborations[54], specifically the Relevance, the Rigor and the Design Cycle[55]. In addition, another renown design science approach developed by Frank has been found suitable for elaborations in this thesis. This conceptual model of idealised design science approach in information systems research includes purposes listed previously in section 1.2. Design artefact, conceptual framework, prototype and hypotheses underpin the purposes through adequacy and natural language justification.[56] The previous considerations and the notion of design science constitute the following design science artefacts:

- a procedure model for the viewpoint-based information systems architecture engineer-

[40]see SHAPE Consortium (2019). *SHAPE Project - Homepage.* url: https://cordis.europa.eu/project/rcn/85337%7B%5C_%7Den.html (visited on 26th Aug. 2019).

[41]see ViBaM Consortium (2019). *ViBaM Project - Homepage.* url: https://www.eurostars-eureka.eu/project/id/5529 (visited on 26th Aug. 2019).

[42]see EU CORDIS: https://ec.europa.eu/research/fp7/ (visited on 26th Aug. 2019)

[43]see Eurostars Eureka: https://www.eurostars-eureka.eu (visited on 26th Aug. 2019)

[44]see Kees Max van Hee (1994). *Information systems engineering: a formal approach.* Cambridge University Press.

[45]see David Avison et al. (2003). *Information systems development: methodologies, techniques and tools.* McGraw Hill.

[46]see Thomas Allweyer (2007). 'Erzeugung detaillierter und ausführbarer Geschäftsprozessmodelle durch Modell-zu-Modell-Transformationen'. In: *EPK.* vol. 303, pp. 23–38.

[47]see Jörg Becker, Patrick Delfmann et al. (2010). 'Ein automatisiertes Verfahren zur Sicherstellung der konventions-gerechten Bezeichnung von Modellelementen im Rahmen der konzeptionellen Modellierung.' In: *Modellierung*, pp. 49–65.

[48]see Sabine Buckl, Sascha Krell et al. (2010). 'A Formal Approach to Architectural Descriptions - Refining the ISO Standard 42010'. In: *Advances in Enterprise Engineering IV*. ed. by Antonia Albani et al. Vol. 49. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, pp. 77–91. isbn: 978-3-642-13048-9.

[49]see John Zachman (2008). *The Zachman Framework: The Official Concise Definition.* url: https://www.zachman.com/16-zachman/the-zachman-framework/35-the-concise-definition (visited on 26th Aug. 2019).

[50]see Object Management Group (2003b). *MDA Guide Version 1.0.1.* url: http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf (visited on 26th Aug. 2019).

[51]see Clive Finkelstein (2006). *Enterprise Architecture for Integration: Rapid Delivery Methods and Technologies.* Artech House Publishers. isbn: 1580537138.

[52]see Thomas Vogel et al. (2014). 'Model-Driven Engineering of Self-Adaptive Software with EUREMA'. in: *ACM Trans. Auton. Adapt. Syst.* 8.4, 18:1–18:33.

[53]see Marco Brambilla et al. (2009). 'A Transformation Framework to Bridge Domain Specific Languages to MDA'. in: *Models in Software Engineering.* Ed. by Michel Chaudron. Vol. 5421. Lecture Notes in Computer Science. Heidelberg: Springer Berlin / Heidelberg, pp. 167–180. isbn: 978-3-642-01647-9.

[54]see Alan Hevner and Samir Chatterjee (2010). *Design science research in information systems.* Springer.

[55]see Alan Hevner (2007). 'A three cycle view of design science research'. In: *Scandinavian journal of information systems* 19.2, p. 4.

[56]see Ulrich Frank (2006). *Towards a pluralistic conception of research methods in information systems research.* Tech. rep. ICB-research report, pp.46-47.

ing, which shall involve impact analyses and model transformation features aiding the requirements changes prediction and assessment.
- a method for closing the adjacent modelling levels gaps in MDA context, which shall include impact analyses and model transformations features for effective information propagation between the abstraction levels.

Taking combination of design science approaches by Hevner and Frank, the following principles should be observed for the placement of the thesis' topic into the information systems area:[57,58]

- **Principle 1**: Design science in business information systems research is a **search process**. The method, prototype and case studies have undergone several incremental spiral spins of improvement, which constitutes the search component in the research process.
- **Principle 2**: Clearly **recognisable contribution** to the body of knowledge in business information systems research. Knowledge about the artefact presented in this thesis has been accumulated during the course of the according research projects, published respectively, and thus represents the addendum to the existing knowledge corpus.
- **Principle 3**: Valid design science **results are all artefacts** with a relation to information systems. Here, the results comprise a method, a model and notation for engineering of viewpoint-based enterprise information systems.
- **Principle 4**: Design science research should focus on **relevant problems**. The relevance of the problems is given in the motivation and problem statement section, as well as through research projects carrying out the development and the evaluation.
- **Principle 5**: **Evaluation and publication** of the design artefacts is of utter importance. As the artefacts of these particular research projects, these have been evaluated with the aid of the objective criteria in each of the research projects respectively. There have been manifold publications on the topics of the thesis, which helped gathering feedback and opinions in advance and thus improve the process.
- **Principle 6**: **Accepted research methods** must be applied while conducting design science. The research in this thesis is built upon argumentative-deductive method elaboration, application prototyping and case study evaluation, which are accepted research methods.
- **Principle 7**: **Transparency postulate** in design science suggests making all non-evident suppositions within an argument explicit. This thesis makes all the research goals and questions, the research method and the corresponding artefacts as clear as possible. All the assumptions made during the course of elaborations are either found in text or referenced to, thus making the non-trivial inferences transparent.
- **Principle 8**: **Justification postulate** for design science as support by convincing reason. There is a clear deductive line from the claimed research goals to reviews of current state-of-the-art and state-of-the-technique used for requirements elicitation in the course of concept elaboration, which is assessed in the corresponding application cases, thus providing the required justification by convincing reason.

The main steps of the applied research from the design science point of view imply considering the following:[59]

[57] see Shirley Gregor and Alan Hevner (2013). 'Positioning and presenting design science research for maximum impact.' In: *MIS quarterly* 37.2, pp. 337–355.

[58] see Ulrich Frank (2006). *Towards a pluralistic conception of research methods in information systems research.* Tech. rep. ICB-research report.

[59] reworked from: Ulrich Frank (2006). *Towards a pluralistic conception of research methods in information systems research.* Tech. rep. ICB-research report, p.56.

- **Research Field Analysis**: overview of viewpoint-based software systems engineering, model transformation and impact analysis methods. Information, software, model and viewpoints evolution constitute adjacent theoretical background concepts for the whole research area.
- **Requirements Elicitation and Concept Elaboration**: based on derived requirements for concept and tooling, a method for the definition of the viewpoint-based model-driven flexible information system architectures as a solution for software systems engineering has been elaborated (impact analysis and model transformations – features of the system to achieve practical applicability).
- **Prototype Implementation and Application**: SHAPE project provides an application case in cooperation with Saarstahl for the service-based legacy steel production system re-engineering. ViBaM project provides an application case with Softeam for implementing an existing viewpoints standard TOGAF, as well as with ikv++ for modelling automotive safety assurance process for work products.
- **Concept Evaluation**: based on the SHAPE and ViBaM project results from both the Saarstahl and Softeam application cases, the evaluation of the achieved requirements is conducted. The viewpoint-based software system modelling method is thus being substantially empirically circumstantiated in different industry settings.
- **Proof-of-Concept Considerations**: as in every research attempt, there should be the critical observation of the results for evaluation, which constitutes the implications for future research demand. This demand is an integral part of the critical observation and marks the way for the improvements to make upon the developed concept.

There are several other authors pointing out various aspects of design science approach and providing structured methods for conceiving scientific approaches in business informatics as Becker[60], Heinzl[61], Lange[62], Niehaves[63], Österle[64], Wilde[65], and Winter[66], to name a few. The comparison of their soundness and advantages is out of scope for this thesis, although might be a useful exercise for the future research work and scientific elaborations. The previously claimed combination of design science approaches by Hevner and Frank promises a viable framework for the envisioned applied research.

1.5 Thesis Structure

Exposition sketches the research conducted in the course of research projects in the European context, whereas the thesis is structured as follows (see Figure 1.1):

[60] see Jörg Becker, Roland Holten et al. (2003). *Forschungsmethodische Positionierung in der Wirtschaftsinformatik: epistemologische, ontologische und linguistische Leitfragen*. Tech. rep. Arbeitsberichte des Instituts für Wirtschaftsinformatik, Westfälische Wilhelms-Universität Münster.

[61] see Armin Heinzl (2001). 'Zum Aktivitätsniveau empirischer Forschung in der Wirtschaftsinformatik - Erklärungsansatz und Handlungsoptionen'.

[62] see Carola Lange (2006). *Entwicklung und Stand der Disziplinen Wirtschaftsinformatik und Information Systems. Interpretative Auswertung von Interviews: Teil III-Ergebnisse zur Wirtschaftsinformatik*. Tech. rep. ICB-Research Report.

[63] see Bjoern Niehaves (2007). 'On episemological diversity in design science: New vistas for a design-oriented IS research?' In: *ICIS 2007 Proceedings*, p. 133.

[64] see Hubert Österle et al. (2010). *Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz*. Infowerk.

[65] see Thomas Wilde et al. (2006). *Methodenspektrum der Wirtschaftsinformatik: Überblick und Portfoliobildung*. Tech. rep. Institut für Wirtschaftsinformatik und Neue Medien, Fakultät für Betriebswirtschaft, Ludwig-Maximilians-Universität.

[66] see Robert Winter (2008). 'Design science research in Europe'. In: *European Journal of Information Systems* 17.5, p. 470.

- Chapter 2 introduces the terms and methods in the field of research conducted in this thesis and answers the first part of research question I.
- Chapter 3 explains the idea behind the research field and economics aspects, as well as process management and answers the second part of research question I.
- Chapter 4 provides a review on the research conducted in the adjacent fields of study and thus delivers the response to research question II.
- Chapter 5 derives the requirements for the methodology of the developed method and hence provides the reply to research question III.
- Chapter 6 defines in detail the procedure model for information system architectures engineering along with viewpoint aspects, model transformations and impact analysis, thus elaborating the answer to research question IV.
- Chapter 7 presents the evaluation of the procedure model presented in the concept on the basis of two industry application cases in steel production and automotive safety, thus concluding research question V.
- Chapter 8 concludes the thesis with a critical review of the method, future research and development demand for flexible information system architectures.

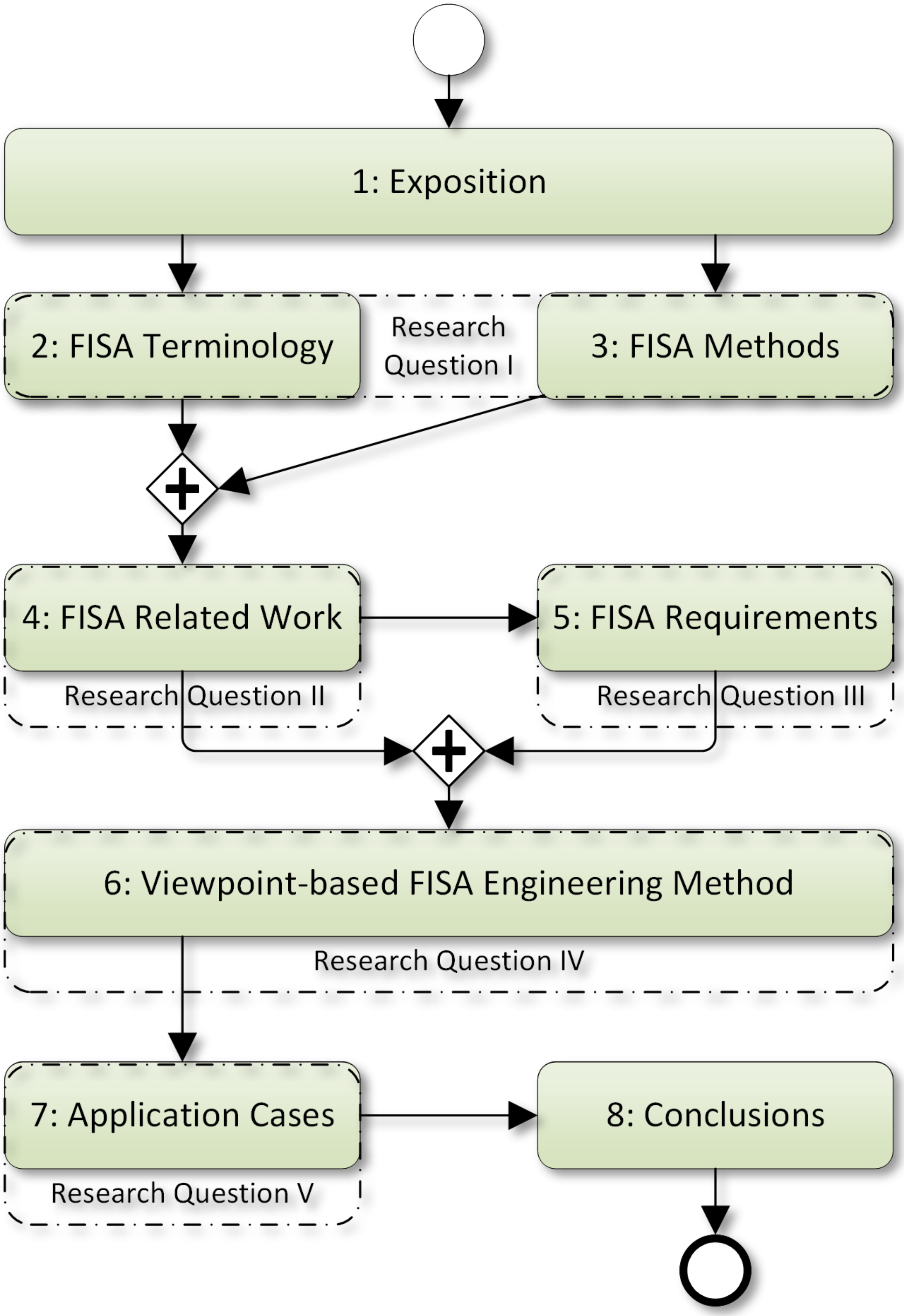

Figure 1.1: Thesis Structure in BPMN Notation

2 Viewpoint-based FISA: Terminology

Following the design science approach proposed in section 1.4, according to principle 1 design science in business information systems, research is a **search process**. In this chapter, using the databases of scientific publications (e.g. an alphabetical selection CiteSeerX[67], computer science bibliography DBLP[68], GoogleScholar[69], IEEE Xplore[70], Research Gate[71], ScienceDirect Elsevier[72], Springer[73], library of Saarland University (SULB)[74]), the search for relevant literature in form of conference and journal articles, books and book chapters, as well as online publications or presentations has been conducted. This search was focused on **relevant topics** stated by principle 4 of the design science approach, in this chapter on terminology for viewpoint definitions and methods, models for model transformations and impact analysis, differences and similarities between information systems and enterprise architectures, as well as for flexibility for this kind of architectures. After identification of the relevant publications by analysing abstracts, introductions and conclusions, where applicable, from the pool of the discovered publications, the step of research field analysis according to the principles 5 (**evaluation and publication**) and 6 (**accepted research methods**) proceeds with the classification findings in order to provide the basis for the following step of requirements elicitation and concept elaboration (see section 1.4). This chapter answers the first part of research question 1 defined in section 1.3 on constituent terms and aspects of flexible information system architectures (see Figure 1.1).

This terminology explification chapter deals with the definition of the constituent terms and aspects of the flexible information system architectures (FISA): viewpoint definition and construction methods (see section 2.1.1), model transformations and impact analysis (see section 2.2), as well as the clear distinction between information system and enterprise architectures (see section 2.3).

2.1 Viewpoints: Foundation for Efficient FISA Engineering

What are viewpoints? Do we need an extra term for perspectives and views? What are they good for? These and further related questions we pursue to answer in this section. Viewpoint definition is the first cornerstone in this aspect, followed by the methods, which are using these viewpoints in the course of elicitating the functionalities and helpful aspects of the information provision we aim to achieve. Those terms applied in information systems field might be of essence in other research fields, too, still having slight or more prominent differences to our approach, which we try to delimit herewith.

2.1.1 Viewpoint Definitions

This section introduces the rationale on which viewpoint definitions drew our attention, which is general in nature and thus can be applied to common software engineering and not only to

[67] see CiteSeerX: `citeseerx.ist.psu.edu` (visited on 26th Aug. 2019)
[68] see DBLP: `dblp.uni-trier.de` (visited on 26th Aug. 2019)
[69] see GoogleScholar: `scholar.google.com` (visited on 26th Aug. 2019)
[70] see IEEE Xplore: `ieeexplore.ieee.org/Xplore` (visited on 26th Aug. 2019)
[71] see Research Gate: `www.researchgate.net` (visited on 26th Aug. 2019)
[72] see ScienceDirect: `www.sciencedirect.com` (visited on 26th Aug. 2019)
[73] see Springer: `www.springer.com` (visited on 26th Aug. 2019)
[74] see SULB: `www.sulb.uni-saarland.de/en/` (visited on 26th Aug. 2019)

architecture specification. In fact, some of them are taken from a number of viewpoint-related methods, which are themselves described in detail later in section 4.1.2. Therefore, the adoption of each definition by a corresponding method is considered to be one of the features to be evaluated.

According to our observations, most of the definitions we found have something in common: they define the concept of a "viewpoint" as a guideline for constructing views. Thus, the viewpoint can be seen as a pattern that defines a set of views. Another common feature for most of those definitions is that a "viewpoint" explicitly specifies one or more stakeholders, whose perspective on the model under investigation it represents. Furthermore, some definitions explicitly note that a "viewpoint" should be as much self-contained as possible (cp. ACID approach, especially isolation principle in software engineering[75]).

What we are particularly interested in and will be investigating in the next section and later on in chapter 4, is whether viewpoints are defined in terms of a metamodel, as well as whether this metamodel is separate or somehow related to other metamodels. Having relations or guidelines that connect viewpoints will be referred to as centralisation of viewpoints. Another feature we are interested in is whether a viewpoint directly reflects the needs of a particular stakeholder. There are two comparison tables at the end of section 4.1.1, which achieves to answer these questions. The approaches on view and viewpoint definitions had to be categorised into approaches for general software engineering, enterprise systems, and technical systems. In each of these categories the standard proposals for the respective category is used as a starting point.

2.1.2 Viewpoint-based Methods

A method in general can be seen as an approach for achieving a certain goal with certain instruments under given constraints.[76] In this thesis, we are particularly interested in viewpoint-based information system architecture (ISA) construction methods and their application, which are thoroughly listed and analysed in chapter 4, section 4.1.2. At this point, the division of the methods into three categories based on our research suffices: general software, enterprise, and technical systems engineering.

Regarding *general software systems engineering* the standard proposals IEEE Standard 1471-2000[77] and Reference Model of Open Distributed Processing (RM-ODP)[78] are important starting points. Both aim at the design of general purpose software systems. Unified Modeling Language (UML) supports the concept of views but does not directly provide a standard proposal for views or viewpoints, nevertheless there are research efforts driven by industry on the creating links to BPMN[79] and SoaML[80] standardised by Object Management Group (OMG), thus connecting the standards to Model Driven Architecture (MDA) approach, which in turn does not say anything about standards that have to be used on any of levels of abstraction. The most concise proposal

[75]see Helmut Balzert (2008). *Lehrbuch der Softwaretechnik: Softwaremanagement (German Edition).* Spektrum Akademischer Verlag. isbn: 3827411610, pp.907-908.

[76]see Thomas Wilde et al. (2006). *Methodenspektrum der Wirtschaftsinformatik: Überblick und Portfoliobildung*. Tech. rep. Institut für Wirtschaftsinformatik und Neue Medien, Fakultät für Betriebswirtschaft, Ludwig-Maximilians-Universität, p.1.

[77]see IEEE (2007). *IEEE Recommended Practice for Architectural Description of Software-Intensive Systems*. Tech. rep. Software Engineering Standards Committee of the IEEE Computer Society.

[78]see International Organization for Standardization (2011b). *RM-ODP Reference Model.* url: `http://www.joaquin.net/ODP/` (visited on 26th Aug. 2019).

[79]see Object Management Group (2011a). *Business Process Model and Notation (BPMN) Version 2.0.* url: `http://www.omg.org/spec/BPMN/2.0` (visited on 26th Aug. 2019).

[80]see Object Management Group (2012b). *Service oriented architecture Modeling Language (SoaML) Specification, Version 1.0.1.* url: `http://www.omg.org/spec/SoaML/1.0.1` (visited on 26th Aug. 2019).