INCONSISTENCY MANAGEMENT IN HETEROGENEOUS MODELS - AN APPROACH FOR THE IDENTIFICATION OF MODEL DEPENDENCIES AND POTENTIAL INCONSISTENCIES

Kattner, Niklas; Bauer, Harald; Basirati, Mohammad R.; Zou, Minjie; Brandl, Felix; Vogel-Heuser, Birgit; Böhm, Markus; Krcmar, Helmut; Reinhart, Gunther; Lindemann, Udo

Technical University of Munich

ABSTRACT

In today's engineering projects, interdisciplinary work leads to an increase in interfaces between different departments and domains. As each stakeholder pursues different goals and tasks, a heterogeneous model landscape is required. In each domain, a variety of different model and software implementations provide the essential basis for efficient work. On the interfaces, the risk of model inconsistencies increases. To handle occurring inconsistencies, various approaches have been presented. For model-based systems engineering projects, rule-based methods are considered as the most suitable technique. However, said approaches require a high manual effort in identifying model dependencies and establishing consistency rules. Unfortunately, in particular these steps are not well described and supported. Therefore, this paper presents an easily applicable approach for the identification of model dependencies in interdisciplinary projects. The method is supported by a software implementation and is directly integrated in engineering workflows. A first industrial case study has shown positive effects of the approach and revealed further research goals.

Keywords: Inconsistency Management, Product-Service Systems (PSS), Case study, Design learning

Contact:
Kattner, Niklas
Technical University of Munich
Mechanical Engineering
Germany
niklas.kattner@tum.de

Cite this article: Kattner, N., Bauer, H., Basirati, M.R., Zou, M., Brandl, F., Vogel-Heuser, B., Böhm, M., Krcmar, H., Reinhart, G., Lindemann, U. (2019) 'Inconsistency Management in Heterogeneous Models - An Approach for the Identification of Model Dependencies and Potential Inconsistencies', in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.373

1 INTRODUCTION

Nowadays systems' components are increasing in number and heterogeneity as a result of new paradigms such as Product-Service-Systems (PSS). This leads to interdisciplinary projects, which involve various domains and a high number of dependencies among the disciplines. In order to manage the uprising complexity, engineers apply model-based systems engineering (MBSE) and create a high number of dependent heterogeneous models (Reift *et al.*, 2017). The more models collaborate, the higher gets the risk of model inconsistencies, meaning that the information in different models cannot be true at the same time (Spanoudakis and Zisman, 2001; Feldmann *et al.*, 2015).
As undetected and unsolved inconsistencies result in project delays and failures, inconsistency management becomes crucial (Feldmann *et al.*, 2015). Many approaches for inconsistency management exist, but in particular the task of identifying model interdependencies is not well described and supported. Therefore, this paper presents an approach for a continuous documentation of model interdependencies during the ongoing workflow. The method is supported by an assistant tool implementation that guides the documentation and automatically creates reports on potential inconsistencies. This provides the basis for further automated inconsistency processing within inconsistency management. Additionally, it can be used as constant work support for inconsistency prevention in particular in the field of change management. The approach was applied in an industrial case study in the development process of an automotive controller. The case study showed the general applicability of the methodology in one area, however, due to the higher amount of domains and interdependencies, the maximum benefit is expected within PSS design (Shani *et al.*, 2017).
The remainder of this paper is structured as follows: chapter 2 introduces inconsistencies in heterogeneous models based on an exemplary PSS use case. Chapter 3 presents existing approaches for inconsistency management. Consequently, shortcomings of the state of the art are described in chapter 4. Chapter 5 outlines the approach for the identification and documentation of model dependencies and inconsistencies. Subsequently, the software implementation and its application are explained in chapter 6. Finally, the paper concludes with the further research goals in chapter 7.

2 CHALLENGES AND ILLUSTRATING EXAMPLE: A PSS USE CASE

To illustrate the problem, the case study of a bike supplier, the PSSycle AG, which offers E-bike-sharing system solutions for the city of Munich, is introduced in this section. This supplier changed its business model from providing recreational bikes in city parks into providing intra-city electric scooters. Correspondingly, the required range of the E-bike is increased from 100 km to 150 km.
In the development of the E-bike system, four departments work collaboratively: Product Design (PD), Mechatronic Design (MD), Manufacturing Planning (MP) and Software Development (SD). In these departments, engineers from different disciplines are involved to reach discipline-specific goals, which requires the usage of different models (Figure 1). Oriented by the changed business model in the use case, a product-service-management model is employed in PD to specify the desired characteristics of the new E-bike system in different levels. In MD, mechatronic models named SysML4Mechatronics are applied to refine the mechatronic characteristics of the battery system. A manufacturing model is used in MP to plan the process and resources for manufacturing battery system, whereas a sequence diagram comes into usage in SD to specify IT requirements and solutions. Though these models are heterogeneous both in syntax and semantics, they are also associated by semantic overlaps. First of all, the requirements in PD should be met in all models, e.g. the energy storage of the battery system should be large enough to reach the required range, the packaging machine should be reconfigured in manufacturing for a lager capacity, and IT functions should also be updated. In case, not all necessary changes are identified and implemented, inconsistencies arise. They are defined as conflicting information in different models (Feldmann *et al.*, 2015) or more specifically, contradiction between two facts or two presentations of facts expressed in formal models as well as in informal artefacts such as requirements written in natural language (Basirati *et al.*, 2018). In the PSSycle AG for example, a semantical inconsistency of the relation type "refinement" would occur, if the documented battery capacity in MD would not satisfy the required battery range of the PD model.

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

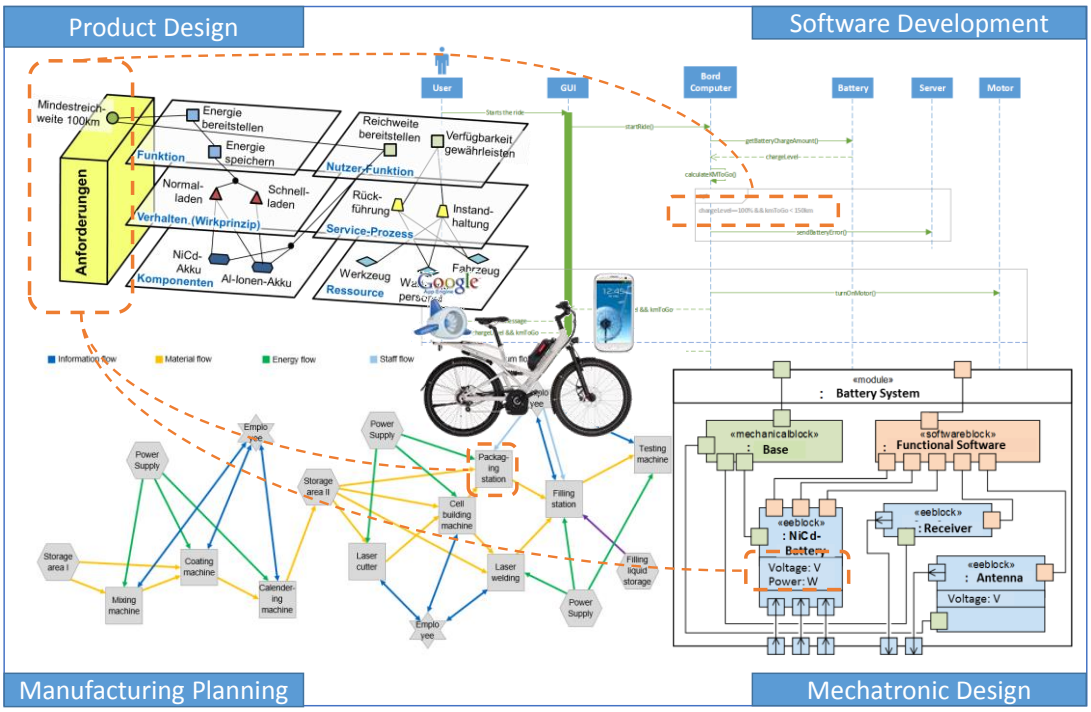

Figure 1: E-bike-sharing system (in the middle) and interdisciplinary models employed in the E-bike-sharing system (marked in corners). Dependencies among models are marked with doted lines when the business model is changed.

3 INCONSISTENCY MANAGEMENT IN HETEROGENOUS MODELS

Goal of inconsistency management is an adequate prevention and reaction in order to minimize negative impacts caused by inconsistencies on project success. While in certain cases, inconsistencies are planned or can be tolerated, other inconsistencies need to be eliminated or, if possible, prevented (Nuseibeh *et al.*, 2000). However, potential or existing inconsistencies must be detected in a first step.

In general, inconsistency management is divided in the steps of monitoring, diagnosis – locating, identifying, and classifying –, and handling – resolve or tolerate – (Nuseibeh *et al.*, 2000), which is applied in approaches of all domains. For the area of MBSE, Feldmann *et al.* (2016) add the visualization of inconsistencies as an essential supporting step. Regarding their basic methods, inconsistency management approaches are classified in proof theory-based, rule-based, and synchronization-based procedures (Feldmann *et al.*, 2015). For MBSE, rule-based approaches were identified as most suitable (Feldmann *et al.*, 2016) and thus, will be the focus of this paper.

Within these methods, inconsistencies can be detected by ongoing monitoring of an established rule set (Nuseibeh *et al.*, 2000). Consistency rules describe a correct state of the models (Hehenberger *et al.*, 2010). The subsequent inconsistency locating entails the determination of elements that broke an inconsistency rule. As basis for suitable handling strategies, the cause for the inconsistency is identified and the inconsistency is classified. Various strategies are available to handle inconsistencies (Zou *et al.*, 2017). In the rule-based inconsistency management, a set of rules between the different models needs to be established as criteria. Therefore, Feldmann *et al.* (2016) suggest creating model links on a metamodel level and deriving consistency rules between linked elements.

Egyed *et al.* (2018) describe a similar procedure. A cloud-based design space enables engineers to manually define traceability links and consistency rules between concrete models.

Dávid *et al.* (2018) propose to link constraints to the engineering process activities and the engineering system and provide a modelling tool, while other authors suggest the use of ontologies for model coupling (Hoppe *et al.*, 2017). Further approaches also emphasize, that a set of rules must be defined by the users in advance and provide examples for consistency rules (Hehenberger *et al.*, 2010; Mens *et al.*, 2006; Hegedus *et al.*, 2011; Herzig and Paredis, 2014; Herzig *et al.*, 2014). Rule-contents and processing techniques are described respectively in these works in detail. In general, a formal representation form of

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

the dependencies is required by these studies. However, they do not investigate the sources of these dependencies, which are the essential basis for rule acquisition in inconsistency management.
It is assumed that the relevant models are already identified and experts can autonomously describe interdisciplinary dependencies. Literature does not provide a procedure to achieve these prerequisites with a modest manual effort and dispersed expert knowledge in various domains.

4 SHORTCOMINGS

Based on the state of the art, four main shortcomings were identified:

Lack of methods for the identification of potentially inconsistent models
As basis for defining rules between different models, it is necessary to identify, which models must be checked for inconsistencies. Therefore, a detailed knowledge of the specific engineering workflows and the model structure of a company is required and must be analyzed before starting with the definition of concrete consistency rules. Only Basirati *et al.* (2018) mentioned it as an activity and that particularly such a method is needed in interdisciplinary projects where a wide range of models and presentations is used. Also the teams work separately, while their models and components are dependent on each other (Song, 2017).

Dependency identification
Having identified relevant models, model interdependencies must be detected before defining the concrete consistency rules. This phase requires high manual effort. However, currently no methodical support exists, even though the quality of inconsistency checking highly depends on the results of this step.

Applicability on heterogeneous models
Most approaches require formal models, as automated inconsistency checking is the goal. In industrial scenarios, important information is often documented informally e.g. in presentations or text documents. Thus, an approach for these scenarios is required, even if a fully automated solution might not me feasible.

Easy to apply and scale
An easily applicable and scalable approach is mandatory for the application in industry, since the acceptance of taking additional effort to handle inconsistencies is often low. A low barrier to apply a method can help to create acceptance. Current approaches require high frontloaded activity in order to provide first results. However, already the documentation of model dependencies – without the definition of consistency rules – helps creating higher transparency in projects and preventing inconsistencies.

5 APPROACH FOR THE IDENTIFICATION OF MODEL DEPENDENCIES IN HETEROGENEOUS MODELS

The method remedies the previously introduced shortcomings by introducing a structured procedure to

- gather information about model usage and interrelations,
- identify arising inconsistencies during actual operative tasks, e.g. product development or production planning, and
- prevent future inconsistencies within a company's context.

In addition, it highly focuses on the applicability within an industrial environment, as it easily scales to the defined application's constraints. Since the user applies the method in parallel to his tasks, the acquisition of model dependency information and inconsistency types is unobtrusively embedded in their work routine. By monitoring their daily tasks and documenting the models they use (and its processed information), the model identification fits to the actual workflows of the users. On one hand, the created database is the basis for further consistency rule definition. On the other hand, selected extracts of model dependencies in standard forms support the daily work in preventing unknown inconsistencies to occur.
In the context of the general inconsistency management framework of Basirati *et al.* (2018), the methodology of this paper addresses the first two steps (see Figure 2).

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

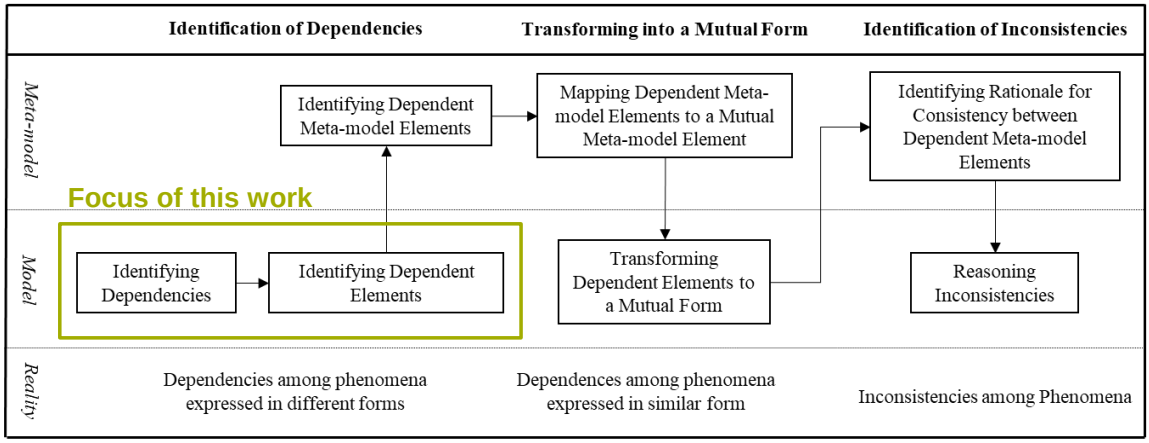

Figure 2: Integration into the inconsistency management process (based on Basirati et al. (2018))

5.1 Methodology overview

The overall methodology consists of four major steps (see Figure 3) and guides through a procedural approach to identify model dependencies within a company. The initial step comprises the definition of an appropriate scope for the analysis. The actual identification of information flows, which is used for the identification of inter- and intra-model dependencies, is conducted in step 2. The third step includes the mapping of various information pools about the dependencies due to the distributed nature of the information. In parallel to the creation of workflow support, the definition of consistency rules can start based on the model interdependency database.

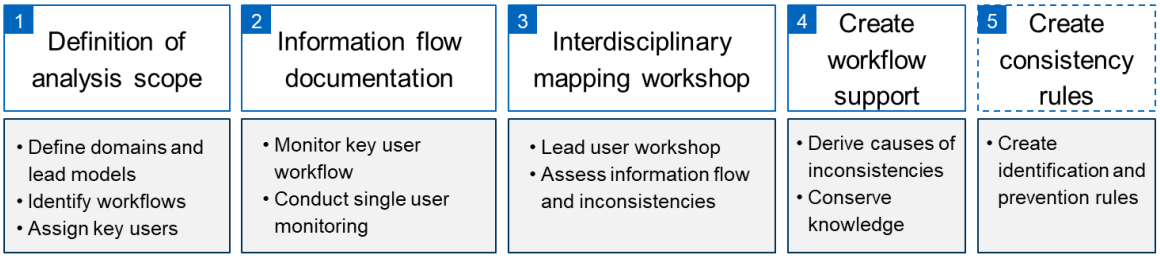

Figure 3: Approach for the identification of model dependencies in heterogeneous models

The following chapters outline the four main steps.

5.2 Definition of analysis scope

The first step addresses the initial definition of the scope of the investigation. To apply the method and derive viable results, it is mandatory to set the boundaries for the application. Criteria for limiting the application's scope could be the organizational structure to investigate the model usage in certain departments or groups. To gain a more profound knowledge about the model linkage, roles of "inconsistency managers" could be defined, which represent several different workflows in a variety of domains (software-/ hardware development, production, maintenance etc.).

5.3 Information flow documentation

The second step addresses the actual identification of model dependencies, inconsistencies and actual information flows. To do so, lead users observe their own tasks and document their usage of information and arising inconsistencies in their daily workflows. The documentation includes the models they use and the models they get information from as well as which information is used. By embedding the gathering of information flows into the daily routine of the user, actual dependencies occurring in the workflow are stored. In addition, a longer timeframe for monitoring the information flow and emerging inconsistencies increases the likelihood to reveal all major issues during the workflow of the user. To generate a holistic understanding of the model dependencies, users must document the following information when creating the model dependency overview during their daily routine:

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

- **Model type:** it is mandatory to identify the origin of the information as well as the target model for which the information is used.
- **Software:** a software is often used when dealing with models within systems engineering. Since there can be differences in handling models between different software versions of the same software type, storing the software and its version is essential to monitor dependencies and inconsistency occurrence. E.g. in the calculation of FE-Models by numerical approaches, calculation results can differ between various software versions.
- **Model Content:** the model content enumerates a possible subsystem of the model. It is necessary to specify the exact part of a model that is required.
- **Information:** finally, it is essential to identify the actual information that is derived from a certain model. In addition, the use of the information – either in the same model or in a different model – must be defined to document the information flow and derive interdependencies between model elements.

The direction of the information flow helps to understand the step-by-step transfer of information through the company. It supports the connection between model dependencies and actual workflows and thus, increases the comprehensibility of the analysis. In case it is not clearly defined, in which order users create the model, or the workflow contains an iterative procedure, bi-directional dependencies are documented.
Furthermore, intra-model dependencies have to be considered as well since a lot of information is in distributed use within the same model. In case only a personal contact is known as information source or target, a placeholder can be documented as basis for the interdisciplinary mapping workshop. The overall approach to collect model dependencies by monitoring and documenting arising inconsistencies is illustrated in Figure 4.

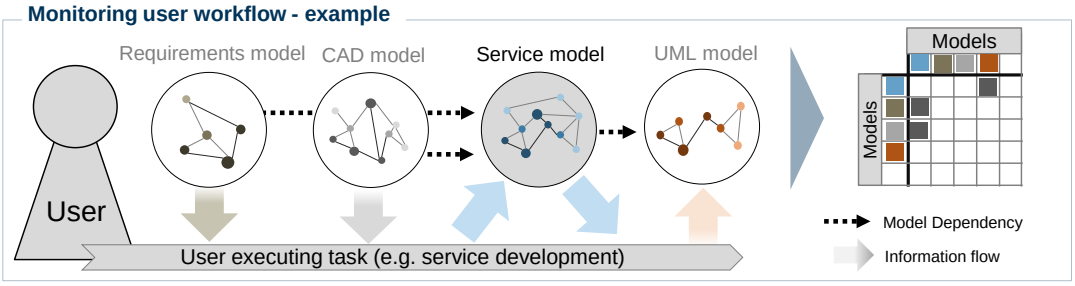

Figure 4: User workflow documentation

5.4 Interdisciplinary mapping workshop

After the users observed their daily routine, i.e. the usage of models and the flow of information, the gathered information about inter-model and intra-model dependencies have to be consolidated. In particular, placeholders in the documentation need to be completed.
To derive a big picture of model interrelations for the previously defined scope of the analysis, a workshop including all stakeholders helps to consolidate the data and create the model dependency and inconsistency map (see Figure 5). Furthermore, a critical reflection of the holistic model dependency map is crucial for the validity of the information. The result of the workshop is an extensive documentation of model interdependencies and identified inconsistencies that occurred during the time of the workflow documentation. However, a continuous monitoring of the user's daily work can extend the documentation. Hence, regular workshops to discuss newly identified dependencies are necessary. Eventually, the holistic documentation can be used twofold: to derive workflow support for later model usage (step 4) or as input for a more sophisticated approach by creating consistency rules based on the gathered information (step 5).

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

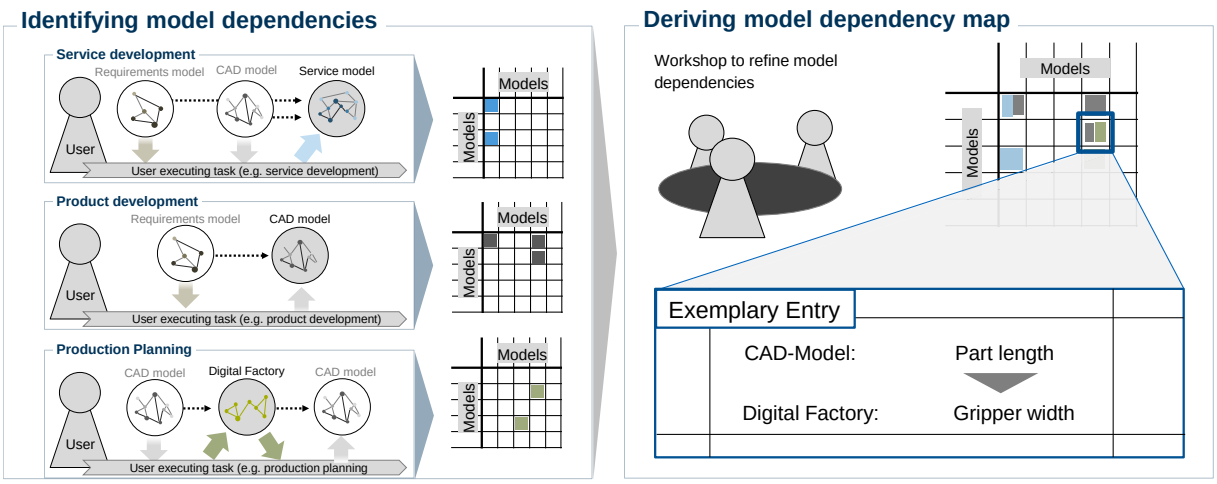

Figure 5: Interdisciplinary mapping workshop

5.5 Create workflow support

Once the workflows were monitored and documented, improvement of future iterations of the same workflow is possible. Since the users of the method already identified model dependencies and possible inconsistencies, dependency and inconsistency reports for certain models within the company can be created. The reports support employees performing the workflows by making situations' possible inconsistencies transparent.

The benefit of using this data to create inconsistency templates for certain models is the derived information from "real world" daily workflows, i.e. actual dependencies and inconsistencies emerged in the user's task. In contradiction, a more theoretical approach to define model dependencies and its inconsistencies can result in information that in theory could occur, but in practice will not be relevant. As a result, people using certain models can individually and tailored to their needs derive inconsistency templates for the models they use. Thereby, they can proactively reduce the likelihood of implementing new inconsistencies during their daily workflow.

5.6 Create consistency rules

Beside the derivation of inconsistency templates for the models a more sophisticated usage of the database is to translate the identified information dependencies into consistency rules. This is the basis for an automated approach to manage inconsistencies as suggested by Feldmann *et al.* (2016).

6 SOFTWARE IMPLEMENTATION AND INDUSTRIAL APPLICATION

In order to enable an industrial application, a software support – the model dependency identification tool (MoDIT) – was developed, which replicates the workflow of adding and saving potential inconsistencies, i.e. model dependencies and information flows. The MoDIT provides the following functions that are embedded in the starting sheet (see Figure 6): (1) Documentation of information flows and model dependencies; (2) Creation of workflow support in standard templates; (3) Creation of an overall dependency matrix. The functions are clustered in two major objectives of the prototype: documentation of model dependencies and creation of workflow support. To document model dependencies, a standardized user form is provided (see Figure 6). If available, potential inconsistencies can directly be entered in connection with the information flow. A holistic approach to assess the model dependencies is to investigate all interrelations between these models by a dependency matrix. The overall dependency matrix is created as a model DSM (Design Structure Matrix, see Figure 7). It indicates all model interactions that include at least one dependency in the database. The direction of the dependency is based on the information flow. The table's rows represent the source model, while the columns indicate the target models. The MoDIT was applied and improved during the development of a controller for driving dynamics in the automotive industry. The main task was to improve the controller's behavior in certain driving conditions.

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

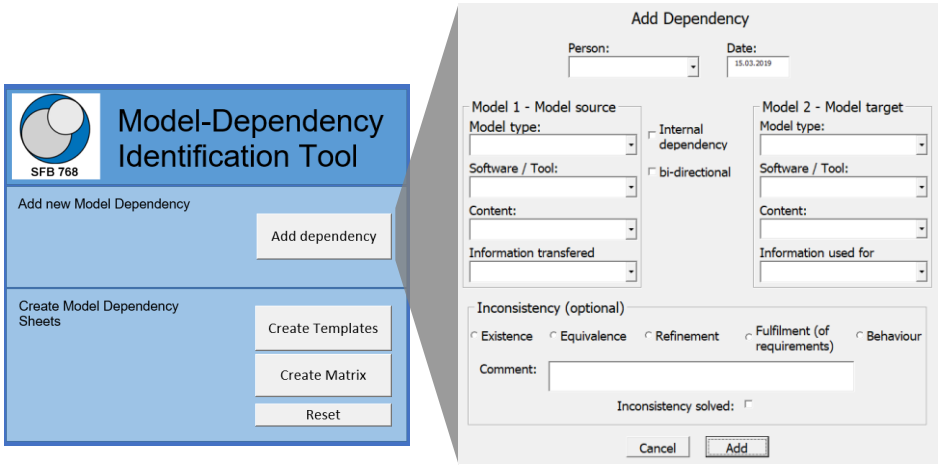

Figure 6: Software implementation - starting window and "add dependency"

Over a period of six months, the user observed the workflow and added all emerging information flows and inconsistencies revealed during the daily workflow in the MoDIT.
An exemplary entry of the database considering the use case is detailed in the following. The example describes an intra-model dependency where an inconsistency occurred during controller development:

Model Source type: Driving dynamics model
Model Source Tool: Matlab/Simulink (+version number)
Model Source Content: {detailed description of the model representation}
Information used: "M-driver-input" (Torque)

Model target type: Driving dynamics model
Model target Tool: Matlab/Simulink (+version number)
Model target Content: {detailed description of the model representation}
Information used for: "MDG-Input" (Torque)

Inconsistency type: syntax
Comment: varying labels for same variable

As a result, for this specific entry, a clear linkage between the two model contents has been identified. Furthermore, the need for similarity in labeling was emphasized by the syntactical inconsistency.
For the creation of a workflow support or as a starting point to create consistency rules, the relevant model is chosen in a first step. Subsequently, all interdependencies of the chosen model are displayed in standardized templates (see Figure 7). For each dependent model, one template is completed.

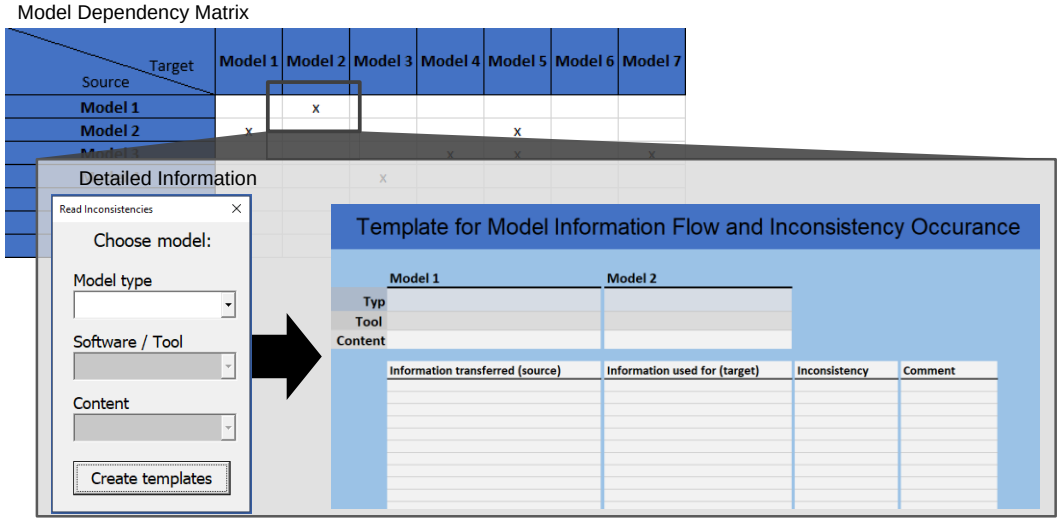

Figure 7: Software implementation – dependency matrix and dependency template

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

These templates can increase the awareness of the user regarding model dependencies and thus, potential inconsistencies during the task. Hence, the MoDIT sets the focus on the prevention of inconsistencies by proactively drawing attention to connected models and previously induced inconsistencies.
Considering all identified inconsistencies and the concerning inter- and intra-model dependencies, the MoDIT allows to create an overview of relevant connections using the model dependency template (see Figure 7). This template can then be used manually by persons working with the concerning models to improve the inconsistency handling during their work.

7 CONCLUSION AND OUTLOOK

The approach to identify model dependencies within companies by monitoring people's daily tasks has advantages in real world scenarios. Since time and money constraints often limit the effort to address problems not directly linked to the operations, this approach creates a foundation for a holistic inconsistency management approach with little effort. Due to its embedded nature in the daily tasks, the approach is affordable and easily applicable. The created database with model interdependencies and emerged inconsistencies can be used either as knowledge base for future work with certain models or as an input for a rule-based automated inconsistency management approach. However, the quality of the database highly depends on the conscientiousness of the tool's users. Hence, a workshop to discuss and evaluate the model dependencies, the informational links, and the inconsistencies can be crucial for a software based inconsistency solver. In future research, the tool will be applied in more diverse development situations and industrial contexts. An automated approach to gather the information flow, for example based on product lifecycle management systems, could help to improve the acceptance of the tool. In addition, a rule-based approach based on the model dependency information gathered with this method is investigated in detail. Furthermore, translating the tool into a web based software environment to support distributed and parallel collection of information would be beneficial for the application in an industrial context. In addition, further use cases for the information database that is created with the presented method will be investigated. In particular, the fields of engineering and manufacturing change management are connected to inconsistency management as changes are a common cause for inconsistencies (Feldmann *et al.*, 2016). Within change management processes, an important step is the identification of change propagation and impacts (Wickel *et al.*, 2015). Therefore, approaches for change impact analysis exist, which depend on a holistic representation of dependencies within the analysed system (Bauer *et al.*, 2017). Even though this exceeds information flows – e.g. material or personnel flows are equally important in manufacturing systems –, the model dependency database provides a valuable basis for further investigations.

REFERENCES

Basirati, M.R., Zou, M., Bauer, H., Kattner, N., Reinhart, G., Lindemann, U., Böhm, M., Krcmar, H. and Vogel-Heuser, B. (2018), "Towards systematic inconsistency identification for product service systems", *Proceedings of the DESIGN 2018 15th International Design Conference, May, 21-24, 2018, Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Croatia*, The Design Society, Glasgow, UK, pp. 2811–2820.

Bauer, H., Schoonmann, A. and Reinhart, G. (2017), "Approach for model-based change impact analysis in factory systems", *2017 IEEE International Symposium on Systems Engineering: ISSE 2017 Vienna, Austria, October 11-13, 2017 2017 symposium proceedings, Vienna, Austria, 10/11/2017 - 10/13/2017*, IEEE, Piscataway, NJ, pp. 1–7.

Dávid, I., Denil, J. and Vangheluwe, H. (2018), "Process-oriented Inconsistency Management in Collaborative Systems Modeling", In: J. Machado, A. Abelha, L. Mendes Gomes and H. Guerra, (Ed.), *16th International Industrial Simulation Conference 2018: ISC '2018 June 6-8, 2018, Ponta Delgada, Portugal*, EUROSIS-ETI, Ostend, Belgium, pp. 54–61.

Egyed, A., Zeman, K., Hehenberger, P. and Demuth, A. (2018), "Maintaining Consistency across Engineering Artifacts", *Computer*, Vol. 51 No. 2, pp. 28–35.

Feldmann, S., Herzig, S.J.I., Kernschmidt, K., Wolfenstetter, T., Kammerl, D., Qamar, A., Lindemann, U., Krcmar, H., Paredis, C.J.J. and Vogel-Heuser, B. (2015), "A comparison of inconsistency management approaches using a mechatronic manufacturing system design case study", In: Q.-S. Jia, (Ed.), *2015 IEEE International Conference on Automation Science and Engineering (CASE): 24 - 28 Aug. 2015, Gothenburg, Sweden, Gothenburg, Sweden, 8/24/2015 - 8/28/2015*, IEEE, Piscataway, NJ, pp. 158–165.

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press

Feldmann, S., Wimmer, M., Kernschmidt, K. and Vogel-Heuser, B. (2016), "A comprehensive approach for managing inter-model inconsistencies in automated production systems engineering", *2016 IEEE International Conference on Automation Science and Engineering (CASE): 21-25 Aug. 2016, Fort Worth, TX, USA, 8/21/2016 - 8/25/2016*, IEEE, Piscataway, NJ, pp. 1120–1127.

Hegedus, A., Horvath, A., Rath, I., Branco, M.C. and Varro, D. (2011), "Quick fix generation for DSMLs", In: G. Costagliola, (Ed.), *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2011: 18 - 22 September 2011, Pittsburgh, Pennsylvania, USA ; proceedings, Pittsburgh, PA, 9/18/2011 - 9/22/2011*, IEEE, Piscataway, NJ, pp. 17–24.

Hehenberger, P., Egyed, A. and Zeman, K. (2010), "Consistency Checking of Mechatronic Design Models", *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference - 2010: Presented at ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, August 15 - 18, 2010, Montreal, Quebec, Canada, Montreal, Quebec, Canada, August 15–18, 2010*, ASME, New York, NY, pp. 1141–1148.

Herzig, S.J.I. and Paredis, C.J.J. (2014), "Bayesian Reasoning Over Models", *CEUR Workshop Proceedings, 11th Workshop on Model-Driven Engineering, Verification and Validation, No. 1235*, pp. 69–78.

Herzig, S.J.I., Qamar, A. and Paredis, C.J.J. (2014), "An Approach to Identifying Inconsistencies in Model-based Systems Engineering", *Procedia Computer Science*, Vol. 28, pp. 354–362.

Hoppe, T., Eisenmann, H., Viehl, A. and Bringmann, O. (2017), "Guided systems engineering by profiled ontologies", *2017 IEEE International Symposium on Systems Engineering: ISSE 2017 Vienna, Austria, October 11-13, 2017 2017 symposium proceedings, Vienna, Austria, 10/11/2017 - 10/13/2017*, IEEE, Piscataway, NJ, pp. 1–6.

Mens, T., van der Straeten, R. and D'Hondt, M. (2006), "Detecting and Resolving Model Inconsistencies Using Transformation Dependency Analysis", In: O. Nierstrasz, (Ed.), *Model driven engineering languages and systems: 9th international conference, MoDELS 2006, Genova, Italy, October 1 - 6, 2006 ; proceedings, Lecture Notes in Computer Science*, Vol. 4199, Springer, Berlin, pp. 200–214.

Nuseibeh, B., Easterbrook, S. and Russo, A. (2000), "Leveraging inconsistency in software development", *Computer*, Vol. 33 No. 4, pp. 24–29.

Reift, J., Koltun, G., Drewlani, T., Zaggl, M., Kattner, N., Dengler, C., Basirati, M., Bauer, H., Krcmar, H., Kugler, K., Brodbeck, F., Lindemann, U., Lohmann, B., Meyer, U., Reinhart, G. and Vogel-Heuser, B. (2017), "Modeling as the basis for innovation cycle management of PSS: Making use of interdisciplinary models", *2017 IEEE International Symposium on Systems Engineering: ISSE 2017 Vienna, Austria, October 11-13, 2017 2017 symposium proceedings, Vienna, Austria, 10/11/2017 - 10/13/2017*, IEEE, Piscataway, NJ, pp. 1–6.

Shani, U., Franke, M., Hribernik, K.A. and Thoben, K.-D. (2017), "Ontology mediation to rule them all: Managing the plurality in product service systems", *11th Annual IEEE International Systems Conference: Montreal, Quebec, Canada, Marriott Chateau Champlain Hotel, Monday-Thursday, April 24-27 2017 proceedings, Montreal, QC, Canada, 4/24/2017 - 4/27/2017*, IEEE, Piscataway, NJ, pp. 1–7.

Song, W. (2017), "Requirement management for product-service systems: Status review and future trends", *Computers in Industry*, Vol. 85, pp. 11–22.

Spanoudakis, G. and Zisman, A. (2001), "Inconsistency Management in Software Engineering: Survey and Open Research Issues", *Handbook of Software Engineering and Knowledge Engineering* No. 1, pp. 329–380.

Wickel, M., Chucholowski, N., Behncke, F., Lindemann, U. and Vajna, S. (2015), "Comparison of Seven Company-Specific Engineering Change Processes", In: M. Schabacker, K. Gericke and N. Szélig, (Ed.), *Modelling and Management of Engineering Processes*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 125–136.

Zou, M. and Vogel-Heuser, B. (2017), "Feature-based systematic approach development for inconsistency resolution in automated production system design", *13th IEEE Conference on Automation Science and Engineering*, IEEE, pp. 687–694.

ACKNOWLEDGMENTS

We wish to thank the German Research Foundation (DFG) for funding this work as part of the collaborative research center SFB 768 "Managing cycles in innovation processes: Integrated development of product-service-systems based on technical products".

https://doi.org/10.1017/dsi.2019.373 Published online by Cambridge University Press