Technical Report

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Efficient Monitoring of Multi-Disciplinary Engineering Constraints with Semantic Data Integration in the Multi-Model Dashboard Process

Stefan Biffl[1], Dietmar Winkler[1], Richard Mordinyi[1],
Stefan Scheiber[1], Gerald Holl[2]

[1]Christian Doppler Laboratory CDL-Flex, Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstrasse 9-11/188, AT 1040 Vienna, Austria
{stefan.biffl, dietmar.winkler, richard.mordinyi, stefan.scheiber}@tuwien.ac.at

[2]Johannes Kepler University, CDL Automated Software Engineering, Linz, Austria
Gerald.Holl@jku.at

Technical Report No. IFS-CDL 14-09
Issued: April 2014

Efficient Monitoring of Multi-Disciplinary Engineering Constraints with Semantic Data Integration in the Multi-Model Dashboard Process

Stefan Biffl[1] Dietmar Winkler[1] Richard Mordinyi[1] Stefan Scheiber[1] Gerald Holl[2]

[1]Vienna University of Technology, Institute of Software Technology and Interactive Systems, CDL-Flex*
Vienna, Austria
<firstname.lastname>@tuwien.ac.at

[2]Johannes Kepler University
CDL Automated Software Engineering
Linz, Austria
Gerald.Holl@jku.at

***Abstract*— In a multi-disciplinary engineering project, such as the parallel engineering of industrial production plants, domain experts want to efficiently monitor project-level constraints that depend on technical parameter values in local engineering models. However, the heterogeneous representations of constraint parameters in these engineering models make the automation of constraint monitoring difficult. In this paper, we introduce the multi-model dashboard (MMD) process providing semantically integrated values of parameters and of constraints to domain experts, as parameter values in various local models change during the project. The tool-supported MMD process guides the definition and monitoring of MMD parameters and constraints. We evaluate the effectiveness and efficiency of the MMD process in a feasibility study with requirements and data from real-world use cases at industry partners. Major results are that the MMD process was effective and efficient in eliciting relevant project constraints and model dependencies and in providing data for change impact analysis.**

Keywords—Software Engineering Methods in Automation, Semantic Data Integration, Change Impact Analysis, Engineering Process Observation.

I. INTRODUCTION

Large-scale software-intensive production systems often consist of multiple heterogeneous and loosely coupled systems, which work together as a system of systems (SoS) [13]. In the context of a multi-disciplinary engineering project, like the parallel engineering of industrial production plants, domain experts develop comprehensive engineering models in a variety of tools and data formats.

To cope with the complexity of current production systems, domain experts use sophisticated system performance measures in their engineering models. Key performance indicators (KPIs) provide feedback to decision makers to improve the system performance, make strategic decisions, or report on processes [24]. Depending on the application context, KPIs can include performance measurements with focus on project and quality management issues (e.g., project schedule, maturity of artifacts, or budget), technical constraints (e.g., power consumption limits), or process-related goals (e.g., throughput of production items). While the KPI calculation is routinely conducted during system operation, system design decisions significantly affect the performance of the development process. Technical, economic, and ecological limits can be observed by accumulating independent local engineering model parameters, e.g., the limit of a building foundation can be compared to the sum of masses of independently designed production equipment units [24]. However, this task can become difficult, if the domain experts involved use heterogeneous data models [20].

The domain experts, who engineer the production system, independently take local design decisions on critical parameters, such as KPIs, during parallel work, but need to ensure a valid solution on project level, defined by constraints depending on the parameter values in local models [24]. Some constraints represent significant risks for plant engineering and project management, if a constraint violation manifests late in the project, as the constraint violation may incur significant design changes and extra work to deliver satisfactory overall results [1].

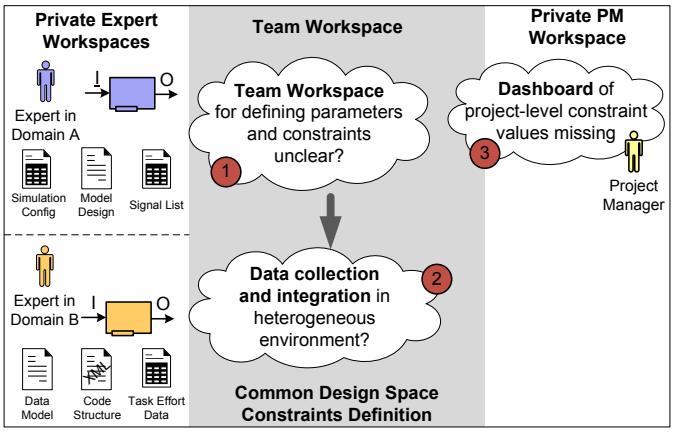


Fig. 1. Challenges and Needs in SoS Engineering Environments.

Figure 1 illustrates challenges in a SoS engineering environment with a heterogeneous landscape of engineering models and tools we found in projects at industry partners in the construction of hydro power plants and steel mills. Figure 1 (left hand side) shows domain experts, who define local models in their private workspaces, e.g., their local file systems, with a multitude of documents that contain a variety of data formats and parameters, which may be relevant for other project partic-

*CDL-Flex: "Christian Doppler Laboratory for "Software Engineering Integration for Flexible Automation Systems". http://cdl.ifs.tuwien.ac.at.

ipants. An example is the project manager (on the right hand side), who gets reports on the progress of the project on the engineering level only infrequently, which makes it hard to address technical and project management risks early.

Risk management on project level can benefit from a central control platform, i.e., an engineering dashboard, which efficiently and quickly provides the values of parameters and of constraints, as parameter values in various local models change. However, the heterogeneous representations of constraint parameters in these engineering models make the effective and efficient automation of constraint monitoring difficult. There are tool chain solutions that propagate in the engineering process all results in bulk at the end of an activity. However, this kind of change propagation is usually not well suited to propagate updates of selected model parameters frequently during the engineering activity due to the propagation overhead and the volatility of data extraction from complex and heterogeneous local models. In the context of distributed configuration of multi-product lines [14] for industrial production systems the Decision Board, a light-weight monitoring approach for monitoring configuration parameters based on human reporting has been found effective for eliciting dependencies between product line models [13], which represent major risks to achieving KPI goals in case of inconsistencies.

In this paper we introduce the Multi-Model Dashboard (MMD) approach, which extends the Decision Board approach [14] by adding the concept of constraints, formally defined using shared model parameters, and by automating the data extraction of parameter values from heterogeneous data sources with semantic data integration. The tool-supported MMD process guides the systematic definition, design, monitoring, and evaluation of MMD parameters and constraints, visualized on the MMD. A dashboard provides the semantically integrated values of parameters and of constraints to the domain experts, as parameter values in various local models change during the project. We evaluated the effectiveness and efficiency of the MMD process steps and the prototype tool support in a feasibility study with requirements and data from real-world use cases at industry partners. Major results of the evaluation are: the MMD process was effective and efficient in eliciting relevant project constraints and interdisciplinary model dependencies as well as providing data for change impact analysis in the evaluation context. As a result, the project participants can be notified on changes that are likely to have important impact on the success of their work.

The remainder of this paper is structured as follows: Section II summarizes related work on risk management, constraint awareness in multi-model industrial plant engineering, and data integration in heterogeneous engineering projects. Section III motivates the research issues and approach. Section IV presents the MMD process and tool support. We present the results of the feasibility study in Section V and provide a discussion in VI. Finally, Section VII concludes and suggests future research work.

II. Related Work

This section summarizes related work in risk management, the Decision Board approach, and semantic data integration that provide foundations for the MMD approach.

A. Risk Management in Heterogeneous Engineering Environments

Successful engineering projects require the collaboration of engineers and the synchronization of planning documents and data between heterogeneous engineering models [17]. Distributed and heterogeneous data models hinder efficient collaboration and data exchange and intensify risks, which typically arise in case of changes in planning documents [1], and defects that occur if data models are not linked seamlessly.

Risks and defects come from common concepts (i.e., data sets used for linking data models in heterogeneous Systems-of-Systems (SoS) environments) on project level that do not map to local representations (i.e., data represented in local tool solutions and data models) [13][14]. Further, loosely coupled tools and data models hinder efficient change management and change monitoring and may become inefficient, error prone, and costly due to the need for manual synchronization by domain experts [17]. In addition, the awareness of potential change impacts in a SoS environment can become fuzzy because of a lack of visibility and traceability related to changes in local representations [12].

Risk management is a crucial part of engineering projects, to identify and mitigate risks [1]. Several risk management approaches have been published and are used in practice. For instance, the RiskIT approach provides a process for risk management in software engineering [16]. Methods, e.g., the Failure Mode and Effect Analysis (FMEA) [5] or the Defect Causal Analysis (DCA) [17] are used to identify risks and root causes early in the development process. However, process, method, and tool support can facilitate risk management but strongly depend on engineers, experts, and teams. As the stakeholders are experts in their own domain, but not in related domains, it is not possible for them to track the changes in the engineering and planning documents of their partners. Even the monitoring of their own set of complex set of engineering models is tiresome and prone to errors as the relevant engineering parameters are often dispersed over several model parts.

In addition, inconsistencies between models are necessary and can be tolerated in some project phases, but are risks or defects in other project phases. If important design parameters change in a way that breaks design constraints, which were agreed with a project partner, the timely notification of the partners helps mitigating the risk of costly design rework later in the project. To support the timely awareness of engineering project stakeholders, an approach is required, which supports in a SoS context both, loose coupling and sufficient awareness based on explicit dependencies between multi-model design constraints for traceability and decision support for identifying and analyzing the impact of changes.

Therefore, we see a strong need to support engineers in better identifying and mitigating risk by providing an approach that efficiently enables engineers (a) to become aware of de-

pendencies between engineering models that are not obvious, (b) to selectively observe critical project parameters (e.g., electrical power consumption), and (c) get knowledge on the source of a violation of critical constraints, caused by an individual discipline.

B. Awareness of Constraints in Multi-Model Industrial Plant Engineering Environments

Industrial plants are often systems of systems (SoS) comprising highly-complex heterogeneous systems. As the individual systems are interconnected, SoS constraints frequently affect multiple systems and need to be analyzed carefully [13]. As domain experts usually focus on their individual subsystem, they lack awareness of constraints and dependencies to other systems within the SoS. Ignoring these dependencies can result in broken interfaces and invalidly configured systems due to unsatisfied and/or violated constraints.

The *Decision Board* approach [14] supports engineers of a SoS in uncovering implicit dependencies between systems within a SoS. The individual systems are represented as dedicated product-line variability models and have well-defined interfaces for configuration and setting as well as getting parameters, e.g., the number of strands in a steel mill. The Decision Board approach increases the awareness of the engineers as it enables them to browse through the parameters of other systems part of a SoS. Due to the awareness of configuration parameters of dependent systems, an engineer can recognize a dependency between a specific configuration parameter of another system and a parameter of his system. As such a dependency can be seen as a simple consistency constraint; it can be modeled explicitly and thus can be monitored continuously to immediately detect the violation of such a consistency constraint due to a change on one side that was not propagated. While the Decision Board does not consider advanced constraints or the integration of data from heterogeneous sources, it can provide the foundation for enabling multi-model management capabilities.

C. Data and Tool Integration in Engineering Environments

Providing integrated data efficiently is a success-critical capability to manage a project in a distributed and heterogeneous engineering environment. The interchange standardization approach [26] was found effective [1] to provide semantic data integration capabilities following the steps: (1) *identify* a data model of common concepts, which allow answering relevant queries on project level; (2) *elicit* the data models of heterogeneous local data sources; and (3) *define* mappings between the common data model and the local data model elements in an ontology; and (4) *apply* the data transformation capabilities of semantic web technologies [11] to provide the integrated data.

Since data can be stored in various data formats, including relational databases, text files, XML files, spreadsheets, and a variety of proprietary storage formats, each with their own indexing and data access methods, a uniform way of accessing the content is important for efficient processing of data in heterogeneous tool environments. Technologies like *Service Data Objects* [25] or *Web Data Objects* help to unify data programming for handling heterogeneous data sources of a tree structure type in SOA[1] environments.

The integration of tools is a prerequisite to build tool chains in an engineering environment [3]. Widely used tool integration designs use message-based patterns [15] to connect technically heterogeneous and distributed systems. The communication between these systems can be based on events and/or on request-response strategies. "Message-oriented Middleware" systems and the "Enterprise Service Bus" (ESB) concept [6] can provide the infrastructure for physically and logically connecting heterogeneous systems. Technical integration features, such as message processing and a service registry [1], represent the foundation for engineering process services on project or domain level [23] and enable change monitoring and risk management in a distributed and heterogeneous engineering environment [4].

III. Research Issues

Key requirements for the efficient monitoring of agreed project-level constraints in a multi-disciplinary engineering project across several engineering models and disciplines early in the project include the capabilities to *define*, *extract*, and *accumulate* the technical parameters as building blocks for constraints. However, due to the distribution of the parameters on heterogeneous data sources, the definition of parameters, dependencies, constraints, and the collection of parameter values to evaluate constraints can take significant time and effort without appropriate process and tool support.

The Decision Board approach [14] has been found effective for eliciting parameters and propagating parameter changes among multi-product-line models, but does not consider the definition of advanced constraints and relies on human-based reporting. Therefore, we introduce and investigate the Multi-Model Dashboard (MMD) approach as an extension of the Decision Board approach, supporting in a system-of-systems context both loose coupling and sufficient awareness for explicit traceability of constraint values. The name of the MMD comes from the dashboard capability to provide an integrated view on constraint and parameter values coming from several heterogeneous engineering models as data sources. From the key requirements in industry projects we derive the following research issues (RIs) for the MMD process and tool support.

RI-1: Multi-Model Dashboard Process. The main question is how to design a process to support stakeholders in effectively and efficiently defining, extracting, accumulating, and observing critical project and process parameters in heterogeneous engineering environments. To address this issue, we propose the Multi-Model Dashboard (MMD) process for (a) defining design constraints based on parameters from heterogeneous engineering models and (b) for efficient monitoring of parameter changes and the impact of these changes based on semantic data integration. We evaluate the process steps in a feasibility study with requirements and data from real-world use cases in the context of engineering industrial production facilities such as power plants and steel mills.

[1] SOA: Service-Oriented Architecture.

RI-2: Tool Support for the MMD Process. The second research issue focuses on investigating mechanisms to automate the MMD process, i.e., how to provide mechanisms to support the process. From the initial MMD process need analysis (see Figure 1) we derive the following solution requirements for effective and efficient tool support.

1. *Team workspace for defining parameters and constraints.* The team workspace is to provide a simple mechanism to allow the definition and negotiation of parameters and constraints for defining change awareness based on common concepts shared in the team on project level.
2. *Data collection and integration.* To collect the parameter values from the local heterogeneous engineering models, there is a need for (a) specifying the location and format of each parameter in the local model; (b) a mechanism for collecting versions of local model updates; (c) a mechanism for data extraction from the local model; (d) a function for the semantic transformation [11] of local parameter values into the common concept format on project level; and (e) an evaluation mechanism to calculate the value of defined project level constraints based on the parameter values.
3. *Dashboard of project level constraint values.* To inform all project stakeholders on the project level status of selected parameters and constraints that are relevant to them, there is a need for (a) a project level dashboard holding all analysis results and (b) a tailored dashboard for each role with selected analysis results to assess the impact of the distributed changes on the project level.

We build on the EKB approach [4][22] and evaluate the MMD tool support with respect to these requirements regarding the effectiveness to support an overall useful and usable change impact monitoring process.

IV. The MMD Process and Tool Support

This section describes the MMD process and tool support to address three core requirements derived from industry partner needs: (a) parameter and constraints definition; (b) correct and efficient data collection and interpretation; and (c) appropriate notification of stakeholders.

A. The Multi-Model Dashboard Process

The Multi-Model Dashboard (MMD) process adapts the Decision Board approach by Holl *et al.* [13][14] in the scope of a multi-disciplinary engineering project environment. The MMD process specifies needs for support mechanisms and is independent of a concrete technology.

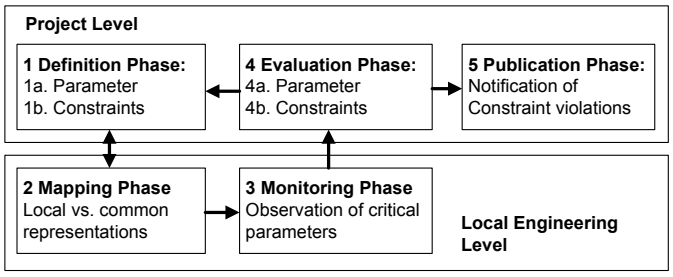


Fig. 2. MMD Process Approach.

The MMD process consists of five basic steps (see Figure 2), their inputs and outputs, stakeholder roles, and required mechanisms. Steps 1, 4, and 5 concern the project level, and steps 2 and 3 focus mostly on local engineering data sources.

1a. Parameter definition. All involved project stakeholders may define parameters that seem important to evaluate project-level constraints. The parameter definitions are based on common concepts [4] that are shared in the project team, e.g., signals in the context of hydro power plant engineering projects. Stakeholders can publish parameters that they can provide from their local models or can request from other stakeholders to provide parameters. Basically, parameters can be used and observed as they are provided or can be aggregated to project-level parameters (e.g., accumulating the maximum weight or power consumption). Advanced parameters can be calculated from available parameter values. The elicitation of parameters can be done informally or can follow a structured approach, such as the *EasyWinWin* method [10]. Output of this step is a list of parameters that stakeholders are willing to publish, so others can subscribe to these parameters based on a request-publish-subscribe interaction design pattern [8] (see also Figure 3 for an example). Note that this step can be repeated, if engineers need to define additional parameters or constraints later in the project. A required mechanism for this step is a team workspace that allows the definition and negotiation of parameters on project level.

1b. Constraint definition. Based on defined parameters derived from step 1a, stakeholders can define constraints as formulas [7], which allow determining a *Boolean* value. If constraints need parameters, which are not yet published, the stakeholders can go back to step 1a. Output of this step is a list of constraints that stakeholders can subscribe to (see also Figure 3). The mechanism requirements are similar to step 1a.

2. Linking parameters to local representations. Each stakeholder, who published a parameter, has to provide a specification in his local data source, on how to find and measure this parameter, e.g., a specific position in an engineering artifact or document and the semantic meaning of the parameter value. In addition, he has to specify whether/how a parameter has to be transformed between the local data model and the shared (common) concept on project level, e.g., type conversion, splitting of a string, or an algorithmic transformation. Output of this step is for each parameter an access and a transformation specification, which can be implemented. These specifications should be robust to the change of the local engineering model, e.g., named spreadsheet cells are more robust to change than hard-coded cell coordinates. Required mechanisms are languages for the specification and transformation of the location and format of each parameter in the local engineering model.

3. Change monitoring in local engineering models. Project stakeholders have to agree on mechanisms to detect relevant changes in local engineering models. An example for a simple mechanism is to save a copy of a changed model in a monitored folder in the team workspace. More elaborated mechanisms include a project repository with version management or a change monitor in the private engineering environment, which automatically publishes changed models, which the stakeholder wants to share, to the team workspace. Output of

this step is a collection of changed local models in the team workspace.

4a. Parameter evaluation. Based on the specification of the parameter location in the local engineering model, an appropriate data sensor can extract the parameter value from the engineering model. Based on the specification of the transformation between local and common concepts, the local parameter value can be transformed into the value of the shared concept on project level. For example, the value of the mass of an equipment unit can be extracted from a spreadsheet, can be transformed as necessary, and can be published to the MMD. Output of this step is a list of published parameters, their values, and an indication when a parameter value has changed the last time. Note, that the history of values is available to enable backtracking to specific plan versions. Required mechanism is an infrastructure for running the data sensors and for holding the states of parameters and their changes.

4b. Constraint evaluation. Based on the list of changed parameter values, the constraints that use these parameters have to be re-calculated to detect violations (see also [12]). Output of this step is a list of published constraints, their values, and an indication when a constraint value has changed or been violated the last time. Required mechanism is an infrastructure to calculate constraints running the constraint specification language.

5. Publication of constraint/parameter values. The evaluation results of constraints and parameters can be published on project level on a Multi-Model Dashboard (see Figure 3 for a conceptual overview and Figures 5-7 for sample snapshots of the prototype implementation). In addition, project stakeholders can have their private role-specific view on the values of the constraints and parameters they subscribed to. A stakeholder can also specify how she wants to be informed [9], in case of constraint violations, e.g., immediately or just in a digest report at the end of a period. Output of this step is a set of dashboards with selected constraints/parameters and their values. Required mechanism is an infrastructure for publication and subscriber notifications.

B. Description of Evaluation Use Cases

In multi-disciplinary engineering projects, such as production or automation systems design, several expert roles – coming from various disciplines – have to collaborate. For instance, the engineering of an industrial production system requires at least the collaboration of and information exchange between mechanical engineers (MEs), electrical engineers (EEs), software engineers (SEs), and building engineers (BEs). During the engineering process, complex and heterogeneous local models are used by discipline-specific experts, which are hard to understand by collaborating domain experts.

In a typical project, domain experts set up a *"collaboration contract"* defining a common design space based on shared parameters and related constraints to enable the concurrent design of the solution and the observation of critical project parameters. Individual domain experts can work within their private workspaces, e.g., BEs define construction plans, SEs implement PLC code, EEs design wiring concepts, and MEs plan the layout of machines and tools of the planned system. To enable the observation of critical project parameters across individual engineering disciplines, collaboration contracts also include constraints that focus on typical settings of key parameters regarding technical, economic, and ecological limits. Constraints may be defined by parameters like length (m), size (m^2 or m^3), mass (t), heat radiation (°C/°K/kW) or power consumption (kW), maximal noise level (db), available resources, project phase, and time. For instance, while the building foundation limit depends on the accumulated mass (t) of equipment units needed in the production system, the BE has to consider cooling capacities (kW) that limit the capacity to address the energy radiation (kW) of deployed devices.

Some constraints represent significant risks for plant engineering and project management, if a constraint violation manifests late in the project, as the constraint violation may incur significant design changes and extra, which exceeds the planned project budget or schedule. A single change in a discipline may trigger a chain of adaptations in other disciplines, which finally test constraint limits. While parameter values are defined by independent contributors from several disciplines, the implication among disciplines is not always clear, leading to avoidable rework in later project phases. Therefore, project stakeholders would like to evaluate defined constraints as early as possible for each relevant change of a local model.

Based on observations and discussions with our industry partners, we identified a set of four illustrative use cases (UCs) and scenarios that can benefit from MMD process applications because of early identification of constraint violations.

Automated process monitoring of a production system simulation (UC-SI). Because the capacity of conveyors is limited a process engineer has to observe the workload of conveyor belts for manufacturing process planning, derived from process constraints (e.g., throughput or cycle time).

Manufacturing plant design and construction (UC-DE). Because of building foundation restrictions (e.g., maximum mass of a base plate or dimensional constraints) the production hall is capable to hold only a limited number of equipment units, defined by independent contributors. System architects need to collect data from contributors and check whether or not the defined physical constraints are violated. In addition, the cooling power limit has to be considered, e.g., impact of the process design on heat radiation.

Electrical systems design (UC-EL). Various constraints, e.g., power consumption and physical characteristics of wiring, have to be observed by electrical engineers and configuration managers.

Project effort and cost monitoring (UC-PM). In distributed engineering projects involving different organizations, the overall project effort has to be tracked. Thus, project managers require aggregated data (derived from project participants) for project planning and control.

Common to all use cases and scenarios – on a more generic level – is the need for observing critical project parameters on project level, derived from contributors coming from various disciplines. The manual synchronization of these data typically requires additional effort and tends to be error prone and induces avoidable project risks.

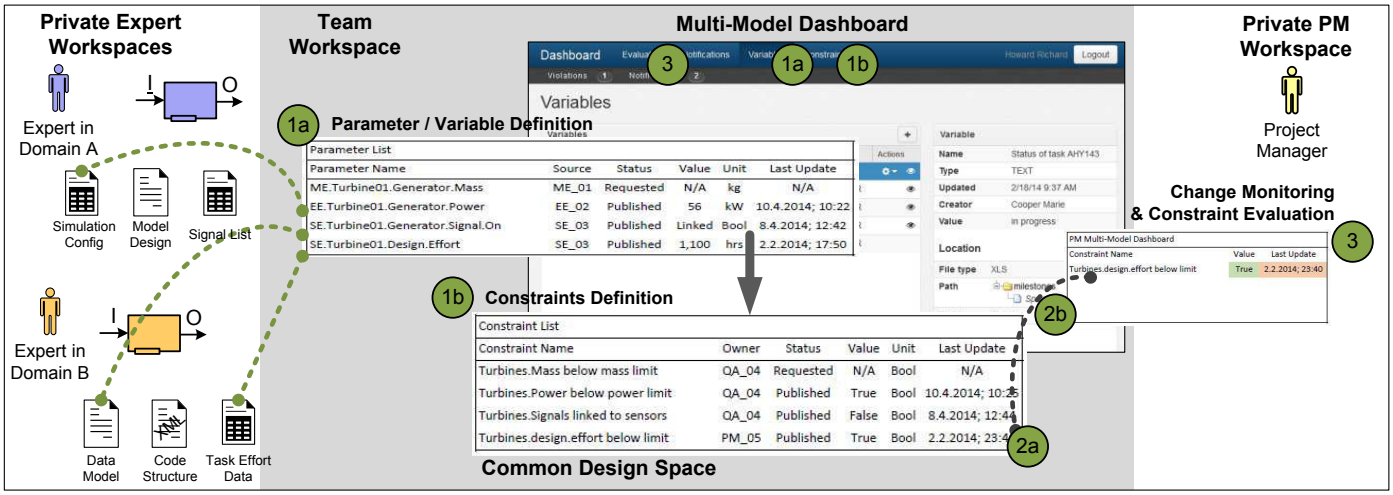


Fig. 3. Multi-Model Dashboard Process – UC Scenario with Contributions.

Based on Figure 1, presenting challenges motivating the MMD approach, Figure 3 illustrates an example use case scenario in the area of project management in more detail (UC-PM): On the left hand side several domain experts (engineers) use a multitude of heterogeneous and complex models in their private workspaces. On the right hand side the Project Manager (PM) needs to be aware on critical project constraints, e.g., project effort. In the central part, i.e., the team workspace, represented by the Multi-Model-Dashboard, all related project stakeholders can define, request, and publish project parameters (see the green numbered circle 1a in Figure 3) that are relevant for their individual needs or that are used to identify risks if projects constraints are violated. In this example, the PM needs to track the overall project effort in a distributed environment involving different organizations. Thus, the PM can define a constraint that aggregates individual working effort to a common project effort (e.g., "turbine.design.effort below limit") (1b). As soon as a stakeholder submits an update of a model containing the working effort to the team work space (2a), the observed parameter gets extracted and evaluated automatically (2b). Subscribers of parameters and constraints get notified automatically, if there is a change of a specific value (3). Note that the individual steps of the MMD process are represented in the MMD dashboard, supporting all features required by the process description. In addition, note that the concept of the MMD process also applicable to a variety of use case scenarios, e.g., UC-SI, UC-DE, and UC-EL.

C. *Tool Support for the MMD process*

This section describes the main architectural components of the prototypically implemented mechanisms required by MMD process steps, which fulfill the requirements of the presented use case, described in Section IV-B. Consequently, the tool support is likely to vary depending on the application context.

Private Expert Workspace. Figure 3 illustrates the private workspaces of two domain experts with their local technology for planning and simulating engineering models, e.g., office and application-context-specific tools.

Team Workspace. The team workspace enables the exchange of software artifacts and data values among team members. The functionality of the supporting mechanism can be provided by solutions, which facilitate the versioning of stored artifacts, e.g., *SVN*[2] or *Git*[3].

Specification Language for Parameters. Figure 3 presents a list of parameters (see 1a) in the team workspace linked to local representations of these parameters in domain expert models and documents within the private expert workspace. We used *Java* for the specification of the parameter types and names to enable an efficient description of calculation formulae for constraints. The specification of data sensors to access local parameter representations depends on the technology used, e.g., *XPATH* for accessing XML-based files or *XLS range* for cell areas in spreadsheets solutions. An example for the data extraction of a specific parameter using the local data sensor is the contract for an interface parameter between a mechanical engineer and a software engineer using the IEC61131-3 representation. An XPATH statement[4] could check changes on a specific parameter specification relevant for both roles. An important issue is to keep data sensor specifications robust against changes in the local model; for instance, applying location independent identifiers (e.g., bookmarks) instead of concrete value identifiers (e.g., cell coordinates) in a spreadsheet solution.

Constraint specification language. Figure 3 shows a list of constraints based on the published parameters (see 1b). The constraint specification language should be compatible to the parameter specification language. We used *JavaScript* and *Java* to support the definition of constraints both in web-based and rich-client environments.

Data transformation language. In order to provide a consistent project view to project members mapping local parameters to common concepts on project level has to be performed. The prototype [22] used ontology-based transformation be-

[2] Subversion: http://subversion.tigris.org/
[3] GIT: http://git-scm.com/
[4] Sample XPATH Statement: //ppx:project/ppx:instances/ppx:configurations/ppx:configuration/ppx:resource/ppx:task[@name='CtrlT']

tween *Java* classes that represent common concepts and local parameters [18][21]. An example for a common concept (see Figure 4) is a so-called *signal* that identifies overlapping model areas and thus bridges the gap between disciplines like software, electrical, and process engineering on team level. Local parameters of a PLC programming environment are correlated via the common concept "signal" with variables in engineering tools for electrical planning or hardware configuration.

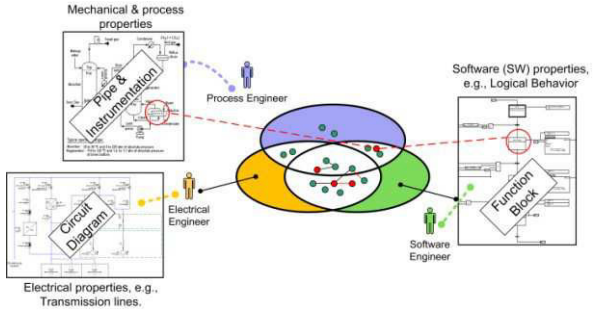


Fig. 4. Local Representations mapped on Common Concepts [22].

Parameter Change Monitoring. Changes in local workspaces need to be made available to other team members to enable efficient collaboration. The prototype used a *Git* repository to version software artifacts and to automatically raise notification events on changes on any file stored in the repository. Built-in mechanisms (i.e., *GitHooks*[5]) enable triggering additional operations based on executed *Git* operations, such as a "*Git commit*". Triggered operations were implemented in *Python* to copy changes in local models from the private expert workspace to the team workspace.

Run-time environment. The components of the prototype (such as data sensors and constraint evaluation) were deployed in the *Automation Service Bus (ASB)* [2][3][19] context, which aims at integrating engineering tools with complex models from heterogeneous engineering disciplines. The ASB is implemented in *Java* and provides a web-based dashboard as end-user interface. Additionally, the ASB facilitates application notification mechanisms (e.g., e-mail and ticketing) to notify relevant project members in case of constraint violations.

V. EVALUATION

This section presents (a) a feasibility study of the MMD process and tool support and (b) an initial cost/benefit consideration of the presented approach in comparison to traditional – manual – project parameter and constraint evaluation. Based on a derived set of use cases and scenarios (see Section IV.B) we implemented the MMD approach and present the individual steps of the process with respect to a selected use case, i.e., the UC-PM, to support project managers in project effort monitoring and control. To evaluate costs and benefits we elicited current needs and estimations from domain experts at the industry partner, who are responsible for engineering and project management of large-scale hydro power plant projects. Traditionally, the observation of critical project parameters and constraints is based on manual data capturing by requesting individual data from involved stakeholders and aggregating them with spreadsheets functions to enable a complete view on the project and the individual states. The application of the MMD process enables project managers to efficiently capture and analyze data from heterogeneous sources.

A. Feasibility Study

The MMD process aims at bridging the gap between heterogeneous, distributed, and loosely coupled tools and the data models on project level. We evaluate the basic concept of the MMD approach by following the steps of the MMD process description (see Section IV.A and Figure 2).

Step 1a. Parameter/Variable Definition. Individual stakeholders define critical project parameters that are relevant for observation. Table 1 presents an overview of selected relevant parameters used in the conceptual evaluation and related data to be observed.

TABLE I. UC OVERVIEW AND SELECTED PARAMETERS.

Use Case	Parameters	Related Data
UC-SI	Throughput, Cycle Time	Items per time interval, duration, number of items
UC-DE	Maximum weight and applied weight Cooling power needs and capacity	Capacity of basement, individual weights of equipment Cooling capacity, heat radiation of machines
UC-EL	Power consumption and needs	Power needed by equipment, overall power available
UC-PM	Time and project plans and effort	Individual milestone planning, working effort per person/artifact

Note that we refer to UC-PM for the evaluation of the MMD process approach in the remainder of this section. Data and parameters are available in various documents (i.e., local representations in private work spaces) such as planning documents in various file formats (e.g., XLS, DOC, PDF, or TXT). To enable the observation and aggregation by other stakeholders, selected data have to be submitted to the team workspace, a *Git* repository in the context of the evaluation. Next, individual parameters/variables that should be available for reuse and observation have to be defined.

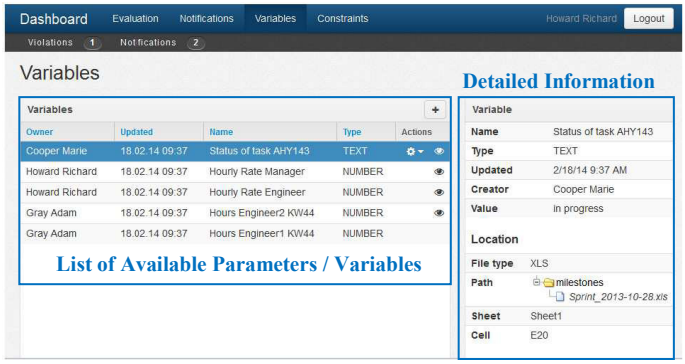


Fig. 5. Parameter / Variable Definition and Subscription.

Figure 5 presents a sample snapshot of the tool prototype that shows (a) the list of available parameter (left hand side) and (b) detailed information on the selected parameter (right hand side). Details include variable characteristics and the location within the team work space as well as the data type and

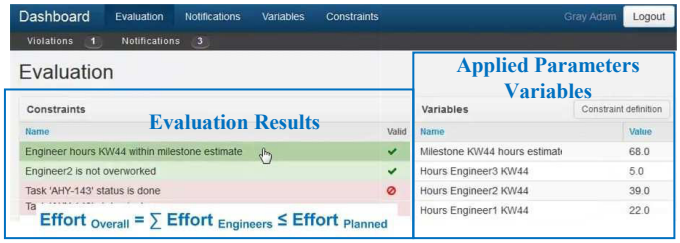
[5] GitHooks: http://git-scm.com/book/en/Customizing-Git-Git-Hooks

the location within the document; e.g., the status of the task is available in a spreadsheet in cell E20. In the traditional manual approach, relevant parameters are implicitly used but not explicitly documented.

Step 1b. Constraint definition refers to defining detailed rules and conditions for evaluation, e.g., the target number of items transported per minute on a conveyor (UC-SI); sum of individual equipment weights that must not exceed the maximum load limit of the basement (UC-DE); overall maximum power consumption as an aggregated value of individual units, that must not exceed the maximum limit including some buffer (UC-EL); or the sum of effort per person and milestone (UC-PM). Individual constraints can be defined by selecting data sensors, aggregating their parameter values and comparing them to pre-defined limits. Typically, planning data can be derived from individual local representations, e.g., a project plan in PDF in the context of UC-PM. Individual local data are represented in TXT, XLS, or XML files. Traditional approaches include implicit constraint definitions without explicit definitions; available data are available in similar representations.

Step 2. Mapping of local representations to common concepts. This step refers to enabling a mapping of individual local data models to common concepts on the team level. We applied an ontology-based transformation between individual *Java* classes to map local representations (in the private expert workspace) to common concepts (in the team workspace) [19]. This transformation enables domain experts to use their own local data models. On team level, common concepts are used to link and aggregate local models and to aggregate parameters on team level. In the traditional approach this mapping step is done by experts by identifying and analyzing the related parameter and variables which requires additional effort.

Observation (Step 3) and evaluation of parameters (Step 4a) and constraints (Step 4b). This central part of the MMD provides an overview on the selected and subscribed parameters and constraints, including the (a) observation of selected parameters, i.e., a list of changes on team-level to indicate what has been updated (and uploaded) by domain experts from local expert work spaces; and (b) Evaluation of Parameters and Constraints.



Parameter Observation and Constraint Evaluation.

Figure 6 presents a snapshot of the MMD tool prototype with focus on constraint evaluation. On the left hand side a list of constraints (including color coding) presents the status of the individual constraint. For instance, a red bar indicates a constraint violation: UC-SI: based on the number of items and time the throughput is too low; UC-DE: the overall weight of the planned machinery in the production hall exceeds the maximum possible overall weight; UC-EL: the aggregated power consumption is to high; or UC-PM: the current working effort of team participants exceeds the estimated effort in the project plan. On the right hand side of Figure 6 details on the affected parameters are presented, in the UC-PM example the current effort of the involved engineers. In the traditional approach the effort for constraint evaluation can be significant in case of complex data models or data models/formats that are difficult to process.

Step 5. Publication of Constraint Violations. The last step of the MMD process includes an active notification of subscribers (i.e., team members, who should be aware of selected critical project constraint violations). The MMD tool prototype provides two notification strategies, (a) an E-Mail notification in case of constraint violations and (b) a personalized overview on related notifications on the subscribed parameters and constraints. Figure 7 provides a snapshot of a role-specific notification overview provided by the MMD.

Fig. 6. UC-PM: Notification Overview.

This approach enables stakeholders to receive immediate feedback on critical process parameters in case of changes to avoid possible project risks. In the traditional approach this information often takes significant communication overhead.

B. Cost/Benefit Considerations

To evaluate costs and benefits of the proposed approach, we interviewed key users at our industry partner to evaluate effectiveness and effort measures based on their expert knowledge. Table II presents an overview of the findings for every step of the MMD process in a selected use case, i.e., UC-PM. We applied a Likert-Scale (++, +, o, -, --), where "++" indicates very positive effects and "--" very negative effects. Positive effects refer to high effectiveness of the individual approaches and low effort for implementation and application.

TABLE II. COMPARISON OF TRADITIONAL AND MMD PROCESSES.

Process Step		Effectiveness		Effort	
		Manual	MMD	Manual	MMD
1a	Parameter definition.	o	++	+	-
1b	Constraint definition	o	++	+	-
2	Linking parameters to local representations	-	+	--	-
3	Change monitoring in local engin. models	-	+	--	++
4a	Parameter evaluation	o	++	o	++
4b	Constraint evaluation	o	++	+	++
5	Publication of parameters / constraint	o	++	-	+
	Overall	**o**	**++**	**o**	**+**

Regarding effectiveness, MMD was found very effective by the interviewed stakeholders because they were able to observe selected critical project parameters effectively and traceable; following in contrast the traditional approach stakeholders had to capture, transform, and aggregate the required data with significant effort and prone to errors. However, the application of the MMD process requires additional effort, especially during the definition and linking phase (i.e., process step 1a, 1b, and 2), which improves the visibility of dependencies between engineering model on the team level. A stable Multi-Model Dashboard enables an immediate feedback on changed parameters and constraints without any additional effort.

VI. DISCUSSION

This section discusses the evaluation results for the MMD process and tool support as well as limitations and threats to validity.

Multi-Model Dashboard (MMD) Process (RI-1). The MMD process guides the definition and monitoring of MMD parameters and constraints in order to support risk management with early awareness on constraint violation. Discussing with domain experts constraints that are linked to relevant risks, the following types of needs were found: (a) monitoring of changes to important parameters, which can be directly measured or derived from measured parameters; and (b) monitoring of constraints for violation. Parameter monitoring is already provided by the Decision Board approach [13][14], while constraint monitoring was found a useful extension.

While the individual MMD process steps are straight forward to conduct, the overall MMD process allows the project stakeholders to systematically structure a complex web of dependencies between engineering models. The comparison of the MMD process to the traditional and manual process in Table II shows that the traditional process is easier to set up but takes more effort to conduct regularly than the MMD process with tool support. Therefore, the MMD process seems well suited to projects, which require the frequent observation of parameter changes for early warnings on risks. In the evaluation context the MMD process was effective to fulfill the requirements to define relevant constraints, correct data collection and interpretation, and appropriate notification. In addition, the MMD enables analyzing violated constraints to see affected parameters and stakeholders for effective and efficient conflict discussion and resolution.

Tool support for the MMD process (RI-2). The prototypic implementation of the MMD process revealed that in multi-disciplinary engineering projects there is the need for integration concepts, which were chosen general enough to allow the easy adaptation to other application and technology contexts. Nevertheless, additional costs accrue from initial setting up and maintaining the process in a certain technology environment. The considerable number of different technological components can bring in additional risks, as those tools introduce structural complexity and need to be tested to ensure the quality of the integration. However, once testing and the initial setup of the environment have been done, tool support ensures the effective and efficient enactment of the process, and thus contributes to quality-assurance at the collaboration interfaces of heterogeneous engineering disciplines. Data inconsistencies are detected once changes have been added to the team workspace and thus help avoid expensive and time-consuming refactoring tasks in the plant design in later project phases.

Limitations and Threats to Validity. The purpose of this work was to present the MMD process with tool support and a feasibility study in context of industry partner needs. However, the current research has some limitations that need further investigation.

User acceptance of the MMD process approach. In a practical environment there may be the risk of the MMD process not to be used, even if effective and efficient, if the MMD would report many false positives, e.g., due to broken data sensors from fundamental changes in models and the data collection environment. Therefore, a stepwise and iterative introduction of the process seems appropriate to ensure a positive benefit for the domain experts. Fortunately, even a partial implementation of the MMD process can be useful for the elicitation of dependencies between disciplines, similar to the Decision Board [14], and to replace the manual measurement of selected parameters.

Feasibility study. We evaluated the MMD process approach with a focus on specific use cases in cooperation with industry partners. The evaluation results are based on observations from a limited sample of projects, stakeholder types, and data models. Therefore, a more detailed investigation in a wider variety of domains and application contexts is planned. In addition, a case study with measurement data is planned to model cost and benefit factors of the MMD approach more precisely.

The e*xpressiveness of specification languages*, used in the presented prototype, can be considered as a limitation. The prototype is able to address specific values in local representations, while industrial scenario evaluations showed that value and aggregated ranges have to be expressible in the ideal constraint specification language.

Parameter and constraint changes. Finally, we considered an initial set of parameters and constraints along the project course. In practical environments, the parameters that are relevant to observe may change during the project. The capability of the MMD process and tool support to adapt parameters and constraint changes as needed needs to be investigated.

VII. SUMMARY AND FUTURE WORK

In this paper we introduced and investigated the Multi-Model Dashboard (MMD) process to provide project participants in a System-of-Systems context with a systematic approach to define and efficiently monitor agreed design constraints across heterogeneous engineering models. The MMD process supports both loose coupling and sufficient awareness based on explicit dependencies between multi-model design constraints for traceability and decision support for change impact.

The *MMD process* provides a clear structure to define parameters and constraints and was found effective in eliciting relevant design constraints and parameters. The domain experts found it useful to make interdisciplinary dependencies explicit

in complex Systems-of-Systems engineering environments. The *data model* allows the timely integration of data from heterogeneous data sources. Data model mappings were sufficient to describe the parameters, constraints, and dependencies of systems. We found the integration standards approach useful to provide an integrated data model view. The *automated tool chain* enables the efficient monitoring and analysis of parameter value changes in complex models in the evaluation context. The data sensor network was effective for monitoring multi-model design constraints in a heterogeneous engineering environment for typical model formats, but has limitations regarding proprietary model formats. In the evaluation context the tool-supported process was more efficient than the traditional alternative solution approach. As a result the project participants can be notified early on changes that are likely to have an important impact on the success of their work.

Future work. Future work is to investigate the scalability of the approach in a real-world environment, e.g., to introduce a negotiation process, such as EasyWinWin [10], to allow the efficient negotiation of comprehensive model parameters, derived values, and design constraints with a large number of stakeholders and models. Further, future work will focus on the derivation of test cases to test the MMD infrastructure. Test scenarios show the results of all parameters after test runs, which check the correct working of the mechanisms for each MMD process step. Finally, future work includes the extension and evaluation of the MMD process and tool support, applicable for various domains and application areas.

ACKNOWLEDGEMENT

This work was supported by the Christian Doppler Forschungsgesellschaft, the Federal Ministry of Economy, Family and Youth and the National Foundation for Research, Technology and Development, Austria.

REFERENCES

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, "Web services: concepts, architectures and applications", *Springer*, 2004.

[2] S. Biffl and A. Schatten, „A Platform for Service-Oriented Integration of Software Engineering Environments". In: *Proc. of the 2009 Conf. on New Trends in Software Methodologies, Tools and Techniques (SoMeT)*, pp.75–92, 2009.

[3] S. Biffl, A. Schatten, and A. Zoitl, „Integration of heterogeneous engineering environments for the automation systems lifecycle". In: *Proc of the 7th IEEE Int. Conf. on Industrial Informatics (INDIN)*, pp.576–581, 2009.

[4] S. Biffl, T. Moser, and D. Winkler, „Risk Assessment in Multi-Disciplinary (Software+) Engineering Projects", In: *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, Special Issue on Risk Assessment, Volume 21(2), pp. 211–236, 2010.

[5] C. Carlson, "Effective FMEAs: Achieving Safe, Reliable, and Economical Products and Processes using Failure Mode and Effects Analysis", *John Wiley & Sons*, 2012.

[6] D. Chappell, "Enterprise Service Bus", *O'Reilly*, 2004.

[7] E. Estevez and M. Marcos, "Model-Based Validation of Industrial Control Systems", In: *IEEE Transactions on Industrial Informatics*, vol 8(2), pp302-310, 2012.

[8] P.T. Eugster, P.A. Felber, R. Guerraoui, and A-M. Kermarrec, "The many faces of publish/subscribe", In: *ACM Computing Survey*, 35(2), pp.114-131, 2003.

[9] W.W. Gibbs, "Considerate Computing", *Journal of Americal Research*, Volume 292(1), pp.54-61, 2005.

[10] P. Grünbacher, "Collaborative Requirements Negotiation with EasyWinWin", In: *Proc. of the 23rd Int. Workshop on Database and Expert Systems Applications*, pp. 954-958, 2000.

[11] J.A. Gulla, "Applied Semantic Web Technologies", *CRC Press*, 2012.

[12] P. Hehenberger, A. Egyed, and K. Zeman, "Consistency checking of mechatronic design models", In: *Proc of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE)*, 2010.

[13] G. Holl, D. Thaller, P. Grünbacher, and C. Elsner, "Managing Emerging Configuration Dependencies in Multi Product Lines", In: *Proc. of the 6th Int. Workshop on Variability Modeling of Software-Intensive Systems (VaMoS)*, Leipzig, Germany, pp.3-10, 2012.

[14] G. Holl, P. Grünbacher, C. Elsner, T. Klambauer, "Supporting Awareness During Collaborative and Distributed Configuration of Multi Product Lines", In: *Proc. of the 19th Asia-Pacific Software Engineering Conf. (APSEC)*, Hong Kong, 2012.

[15] G. Hohpe and B. Woolf, "Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions", *Addison-Wesley Longman*, 2003.

[16] J. Kontio, "The riskit method for software risk management", *Computer Science Technical Reports*, University of Maryland, College Park, MD, US, 1997.

[17] O. Kovalenko, D. Winkler, M. Kalinowski, E. Serral, and S. Biffl, „Engineering Process Improvement in Heterogeneous Multi-Disciplinary Environments with Defect Causal Analysis", In: *Proc. of the 21st EuroSPI Conference*, Luxemburg, 2014 (to appear).

[18] T. Moser, R. Mordinyi, D. Winkler, M. Melik-Merkumians, and S. Biffl, „Efficient Automation Systems Engineering Process Support Based on Semantic Integration of Engineering Knowledge". In: *Proc. of the 16th IEEE Int. Conf.on Emerging Technologies and Factory Automation (ETFA)*, pp.1-8, 2011.

[19] T. Moser and S. Biffl, "Semantic Integration of Software and Systems Engineering Environments". In: *IEEE Transactions on Systems, Man, and Cybernetics*, Part C, Volume 42(1), pp.38–50, 2012.

[20] T. Moser, R. Mordinyi, D. Winkler, and S. Biffl, „Extending Mechatronic Objects for Automation Systems Engineering in Heterogeneous Engineering Environments", In: *Proc. 17th IEEE Int. Conf. on ETFA*, pp.1-8, 2012.

[21] T. Moser, S. Biffl, W. Sunindyo, and D. Winkler, "Integrating Production Automation Expert Knowledge Across Engineering Domains". In: *International Journal of Distributed Systems and Technologies (IJDST)*, Special Issue on Emerging Trends and Challenges in Large-Scale Networking and Distributed Systems, Volume 2(3), pp. 88–103, 2011.

[22] R. Mordinyi, T. Moser, D. Winkler, and S. Biffl, "Navigating between Tools in Heterogeneous Automation Systems Engineering Landscapes", In: *Proc. of 38th Annual Conf of the IEEE Industrial Electronics Society (IECON 2012)*, pp 6182-6188, 2012.

[23] R. Rademakers and J. Dirksen, "Open-source ESBs in action", *Manning Publications*, 2008.

[24] R. Reichert, A. Kunz, R. Moryson, K. Wegener, „A Key Performance Indicator System of Process Control as a basis for relocation planning", In: *Proc of the 43rd CIRP Conf. on manufacturing systems*, pp.805-812, 2010.

[25] L. Resende, "Handling heterogeneous data sources in a SOA environment with service data objects (SDO)", In: *Proc. of the ACM SIGMOD Int. Conf. on Mgmt of Data (SIGMOD)*, pp.895-897, 2007.

[26] A. Wiesner, J. Morbach, and W. Marquardt, "Information integration in chemical process engineering based on semantic technologies", In: *Computers & Chemical Engineering*, volume 35, pp.692–708, 2011.