The practical use of inconsistency information in engineering design tasks – first observations

Alfred Sadlauer

Institute of Mechatronic Design and Production,
Johannes Kepler University Linz,
Austria
Email: Alfred.Sadlauer@gmx.at

Markus Riedl-Ehrenleitner

Institute of Software Systems Engineering,
Johannes Kepler University Linz,
Austria
Email: marxriedl@gmail.com

Peter Hehenberger*

School of Engineering,
University of Applied Sciences Upper Austria,
Austria
Email: Peter.Hehenberger@fh-wels.at
*Corresponding author

Andreas Demuth and Alexander Egyed

Institute of Software Systems Engineering,
Johannes Kepler University Linz,
Austria
Email: Andreas.Demuth@jku.at
Email: Alexander.Egyed@jku.at

Abstract: Today's product development projects require collaboration across different engineering domains in order to be successful. For instance, a project may require software engineers to collaborate with electrical engineers and mechanical engineers. Even though engineers of different domains focus on different parts of the system-under-development, these parts typically cannot work in isolation. Therefore, coordination among these engineers is necessary to ensure that the individual parts of a system work together well when combined. The lack of such coordination leads to inconsistencies and hence the inability to integrate individual parts of the system. Even though approaches for finding such inconsistencies have been developed, it has yet to be shown whether the presentation of inconsistencies is of actual value to engineers. In this paper, we present the results of a practical experiment that assessed the effects of the presence of inconsistent information during development. The results indicate that specific feedback about inconsistency (when performing changes) leads to better engineering results than merely presenting general information about system interconnections.

Keywords: design inconsistency; multi-domain collaboration; system inter-dependencies.

Reference to this paper should be made as follows: Sadlauer, A., Riedl-Ehrenleitner, M., Hehenberger, P., Demuth, A. and Egyed, A. (2017) 'The practical use of inconsistency information in engineering design tasks – first observations', *Int. J. Product Lifecycle Management*, Vol. 10, No. 2, pp.171–190.

Biographical notes: Alfred Sadlauer is a doctoral student at the Institute of Mechatronic Design and Production at Johannes Kepler University (JKU) Linz. His current research interests are mechatronic product design with focus on design process models and model-based systems engineering.

Markus Riedl-Ehrenleitner is a Software Engineer at DynaTrace Linz, Austria.

Peter Hehenberger is a Professor for Integrated Product Development at the School of Engineering, University of Applied Sciences Upper Austria, Wels/Austria. Prior to this appointment, he was Assistant Professor and Deputy Head at the Institute of Mechatronic Design and Production of the Johannes Kepler University of Linz (JKU), Austria. His core competencies cover 'model-based mechatronic system design' and he has published over 100 peer-reviewed papers and been guest editor for special issues. He is a member of IFAC TC4.1 Mechatronics, IFIP WG5.1 ('Global Product Development for the Whole Life-Cycle') and Design Society, where he co-chairs a special interest group (SIG) on methodologies for design, integration, modelling and simulation of cyber physical systems.

Andreas Demuth was a Post Doc at the Institute for Software Systems Engineering at Johannes Kepler University (JKU) Linz, Austria.

Alexander Egyed is a Full Professor at the Johannes Kepler University (JKU), Austria and Head of the Institute for Software Systems Engineering (ISSE). He received his Doctorate degree from the University of Southern California, USA and previously worked for Teknowledge Corporation, USA (2000–2007) and the University College London, UK (2007–2008). He was recognised as a Top 1% scholar in software engineering in the Communications of the ACM, Springer Scientometrics, and Microsoft Academic Search. He was also named an IBM Research Faculty Fellow, received a Recognition of Service Award from the ACM, Best Paper Awards from COMPSAC and WICSA, and an Outstanding Achievement Award from the USC.

1 Introduction

During the development of technical products, knowledge from various domains is combined. For each domain, typically several models exist which represent the aspects of the product that are of direct relevance for the domain. However, these aspects cannot be seen in isolation as the combination of all models of all domains describe the relevant product properties, and individual pieces of information may be used by engineers from different domains in their respective models. It is therefore crucial that the different models, when combined, form a system that is free of contradictions (i.e., inconsistencies). The introduction of such inconsistencies is fostered by the nature of modern development practices: time-to-market should be minimised by developing different aspects of the product concurrently rather than sequentially (e.g., source code is developed by software developers while at the same time civil engineers calculate the properties of physical components of a system).

Models share information about the product in the form of product properties. The term property as used in this paper was defined by Hubka and Eder (1996, p.108f) as follows:

> "Properties (attributes) of the technical systems are all those features which belong substantially to the object – the object owns the property (it is proper to the object). The value of the property represents the measure of the property (size, condition, form of embodiment) in the concrete case. The value can be indicated quantitatively (x m/s – number and unit of measure) or only qualitatively (large, small)."

In the course of a concurrent development process, engineers tend to modify product properties. Inconsistencies arise when such changes affect product properties that are relevant to multiple domains and the properties are not correctly propagated to all models (and their subsequent models) that depend on them.

According to Nuseibeh et al. (2001, p.172), an inconsistency occurs in "any situation in which two descriptions do not obey some relationship that is prescribed to hold between them", for example, when a property has instances with different values assigned (i.e., different models that are both relying on the mass of a part use different values to model this property).

When inconsistencies arise, domain models may not combine correctly to represent the desired product, making the product inherently flawed. The faster inconsistencies are detected, the sooner and, intuitively, the more economically they can be resolved. Thus, early inconsistency identification should be an effective means to increase engineering efficiency – leading to accelerated system design. Immediate identification of inconsistencies is state of the art in individual software platforms (Nuseibeh et al., 2001). Examples include computer-aided software engineering (CASE) tools which are able to identify mistakes in code, such as undefined variables.

1.1 Research questions

This paper discusses the results of a practical experiment in which participants were provided with the model of a mechatronic system (i.e., a robot arm), relevant inconsistency information, and general system interdependency visualisations. The experiment built upon the results of Keller et al. (2006) and Ghoniem et al. (2005) to investigate whether node-link graphs or matrix representation are better for presenting system interconnections to engineers along with the information when inconsistencies occur. The hypothesis is that the information about the occurrence of an inconsistency supported through matrix representation of the system results in a shorter amount of time necessary to resolve inconsistencies than the usage of node-link representation of the system. As the respective research indicates (Keller et al., 2006; Ghoniem et al., 2005) the size of the system along with its density are of influence. In the present analysis, no variation in this direction was performed.

1.2 Paper outline

The next section of this paper provides background information regarding consistency checking and the role of inconsistencies. The third section (method) includes a feasibility study, along with an introduction to DesignSpace as a tool to record and communicate inconsistencies and shows an ensuing experiment linking the usage of inconsistency information to design efficiency, inconsistencies, and existing approaches to integrated mechatronic design. The fourth section explains the experiment setup. The fifth section (results section) describes, and reports the results of the practical experiment in which DesignSpace was used to present inconsistencies. It also discusses the findings and outlines further applications and improvements planned. Finally, the sixth section concludes the paper.

2 Consistency checks: concepts and approaches/methods

2.1 Concepts and approaches/methods

Inconsistencies are part of the iterative nature of product development (Ballard, 2000; Maier et al., 2014; Leon et al., 2013; Wynn et al., 2007). Iteration and redesign can have different purposes and goals (Safoutin, 2003; Wynn et al., 2007). A critical reason for iteration and redesign is the difference between current solution and requirements of the final product (Follmer et al., 2012), which constitutes an inconsistency between current state and goal. Early identification of inconsistencies that require iteration or redesign is crucial to reducing their effects on cost (Ehrlenspiel et al., 2007).

Prior to inconsistency identification, a basis for comparing the different models must be defined. An approach rooted in software development is the systems modeling language (SysML), which is applied, for example, in systems engineering (Friedenthal et al., 2014). The specific connection of models from different domains with shared properties is at the core of mechatronic system models (Follmer et al., 2011, 2012). Functions can serve as a very specific type of property for the connection across engineering domains (Lucero et al., 2014).

Consistency checking during the concept development of mechatronic design models aims to identify inconsistencies early in order to accelerate the design process (Hehenberger et al., 2010) and reduce the cost of changes (Boehm, 1984; Ehrlenspiel et al., 2007). For these considerations it is not only necessary to look at the properties causing the inconsistency, but also at the other properties they are connected to. Propagation of a change can have considerable consequences, as it can result in an avalanche of ensuing changes (Jarratt et al., 2011).

2.2 Benchmark of tools and methods

Several approaches address consistency as part of model-based system engineering. Moser and Biffl (2012) used their engineering knowledge base (EKB) as a framework to store engineering knowledge for use across multidisciplinary engineering projects. This engineering knowledge is stored in web ontology language (OWL) ontologies to facilitate data exchange between different engineering tools. The EKB consists of six steps:

1 extraction of tool data
2 storage of extracted tool data
3 description of domain knowledge
4 description of tool knowledge
5 mapping of tool knowledge to domain knowledge
6 usage of EKB

The EKB allows for the inclusion of automated consistency checks regarding syntax and plausibility for semantic correctness. Such checks must be defined by human experts. The interfaces for the experts are the tools of their respective domains, with the results also being retransformed for use in the respective tools.

Coatanéa et al. (2014) presented a methodology that identifies contradictions (inconsistencies) within models through the creation of a graph of dependencies of variables. This graph can be analysed with respect to the dimensions of the variables thereby identifying variables that cause contradictions. Similarly, Qamar and Paredis (2012) suggested a dependency network as a feasible approach to identifying and avoiding as many inconsistencies as possible. An approach using graphical representation for different models, where the graphs undergo pattern recognition in order to identify inconsistencies, was also employed by Herczig et al. (2014).

The design compiler 43 (Böhnke et al., 2009) represents a different paradigm. It uses UML as a general-purpose modelling language to create design languages for specific purposes. The language is created by analysis of the purpose (e.g., aircraft or satellite design), which leads to a vocabulary in the form of a class diagram. Application of the vocabulary through rules instantiates these classes, and the design compiler creates models based on these rules. This approach requires a significantly larger effort upfront for creating the vocabulary that represents the design space. However, once this vocabulary exists, all possible solutions within it are available within a fraction of the time needed by traditional development approaches. Thus, the approach using design compiler 43 supports high variability, especially in areas where the design space is already well known.

DesignSpace provides a flexible engineering environment for efficient knowledge sharing. Various tools and approaches exist that provide parts of DesignSpace's functionality, such as artefact (property, model, value, …) linking and versioning. However, existing tools fail in combining key features of DesignSpace: the integration of arbitrary development artefacts, traceability across artifacts, change impact analysis, or fine-grained versioning. For example, IBM RELM, available at http://www-03.ibm.com/software/products/en/ratiengilifemana (accessed 16 December 2015), allows arbitrary artefacts to be stored, linked, and queried for reasoning. It is based on the Open Services for Lifecycle Collaboration (OSLC), available at http://open-services.net/. OSLC is a set of specifications that enable the integration of engineering lifecycle tools. Based on a core specification, domain-specific specifications (e.g., configuration management, quality management, requirements management and architecture management) are built on top. OSLC integrates tools based on RDF and HTTP, allows tracing among RDF resources, modelling as so called resource shapes, and not only data integration but also user interface integration of different tools. However, IBM RELM generally does not provide the fine-grained versioning of DesignSpace.

All approaches that collect engineering knowledge that can in turn be used for analysis and inconsistency information require engineering information input. This input must come from the designers, who are both 'contributors and beneficiaries of the system' (Alavi and Leidner, 1999) and must accept both roles. Collaboration through the combination of models from different engineering domains therefore not only requires traditional designers, but also an expert who connects the respective disciplines, that is, a system architect (Komoto and Tomiyama, 2012). These observations from literature lead us to the conclusion that inconsistency information is a crucial part of efficient collaborative design. However, its proper representation is the key to its successful usage.

3 Method

As part of the experiment, the participants were provided with a model of a cyber-physical system (i.e., a robot arm). The experiment addressed the role of a designer as the recipient of inconsistency information and evaluated different visualisations of model interdependencies. Inconsistency information was provided by a software prototype called DesignSpace. General system interdependencies were visualised by either a node-link graph or a design structure matrix.

The participants in the experiment were mostly mechatronics students from Johannes Kepler University Linz (JKU). They were asked to perform a defined engineering change task in the model and to repair any inconsistencies that arise because of performed changes (i.e., that were caused either by the initial change or any subsequent change that was made to propagate the initial change). The model consisted of four sub-models, each of which modelling different aspects of the overall system. These sub-models were not directly connected by existing engineering tools but by means of the DesignSpace prototype. The initial model provided to students was free of inconsistencies.

In this section we address a feasibility study of inconsistency information, introduce the DesignSpace tool to record and communicate inconsistencies and present a design experiment that related design efficiency to the usage of certain representations of inconsistencies.

3.1 Pilot survey among engineers regarding inconsistency information

At a professional practitioner training course at the University of Applied Sciences of Upper Austria, we conducted a survey to:

1 assert the readiness of designers to be informed about existing inconsistencies in their designs

2 determine how much time they would be willing to invest in a technology that indicates inconsistencies.

One of the courses as part of this training addressed systems engineering in practice. All 67 participants in this course were employed by companies that defined themselves as mechatronic companies. Of these 67 people, 61 identified a benefit of consistency information, one identified no benefit and six did not respond.

The majority (60) stated that they use or would use a tool that helps them to identify inconsistencies in their products if its application requires a reasonable small amount of time. As reasonable they marked 1 up to 4 hours of work per average 40-hour work week. The results of this survey indicated that improved consistency checking is a need in industry.

The commitment of 1–4 hours is a significant amount of time. However, this front-loading effort would significantly reduce errors due to inconsistencies in later design phases, which accrue significantly higher costs, than the time effort needed in the early design phases to implement the consistency network (Ehrlenspiel et al., 2007). As inconsistencies commonly occur during changes, such information should be an important part of change management. The opportunity costs (what has to be sacrificed in order to address the inconsistency information) are difficult to estimate. Currently, we expect the time to be added to the early design phases, resulting in increased up-front effort. Nevertheless, due to the better handling of changes and lower number of errors to be fixed in the later design phases, the later phases should take significantly less time, resulting in a net benefit of the inconsistency information. Such considerations are part of future research.

3.2 Procedure and material

Tasks and support measures and order were assigned randomly to participants by drawing without returning in order to achieve equal distribution and minimise selection bias. The participants had to perform the respective tasks individually.

Two slightly different experimental protocols were used. The first pilot group received an explanation of both experiments at the very beginning. For the second group, the protocol was different as general information was provided from the beginning, but specific information (i.e., the task description) was provided immediately before each of the two experiments, along with oral information including a questionnaire to assess their task comprehension. This questionnaire was discussed with the supervisors of the experiment to ensure that the participants understood the tasks correctly. The protocol changes between the first and the second group were made because feedback from the first group indicated that the task was insufficiently clear. For example, one participant stated that the initial amount of information was overwhelming.

The independent variables were the visual support measure in the form of a matrix or graph representation of the overall system, along with the respective time ratio these representations were used in comparison with the general inconsistency information and the detailed inconsistency information provided by DesignSpace. The following list presents all the dependent variables measured in the experiment:

1 duration of problem solving in minutes

2 number of inconsistency flags present within DesignSpace at the end of the provide time for completion

3 number or remaining steps for completion of the task after 30 minutes

4 number of errors made during the problem solving

5 usage ratio of the respective means to identify inconsistencies.

The first four variables were measured directly after an individual participant reached the end of the task either through completion or time expiration. The time was stopped once the task was completed or the time ran out. DesignSpace showed the number of unresolved inconsistencies. The number of overall changes necessary was known and compared to the number of changes performed. The errors (false values entered) were identified through deviation to the known solution. The usage ratio was identified through video analysis of each of the individual participants. The first four variables were used to calculate a combined time value which served as an indication of the quality of the performance of the participant. The usage ratio was used as an indication for the application of the support measures. The respective effects of the independent variables on the dependent variables were evaluated statistically with correlation coefficients.

Task fulfilment was video recorded using CamStudio Recorder v2.7.2. Not recorded were the preparations. The recorded dependent variables were used to evaluate the performance of the individuals. The basic measure to evaluate the participants was the timeframe of 30 minutes allocated for task completion. For the participants who were not able to finish the task, the flags, errors and remaining steps were added as a time penalty. The reference for a standard penalty was the overall time allocated divided by the number of steps (i.e., 30/17). An outstanding change, a flag still active, and an error had penalty factors of 1, 0.5 and 2, respectively, reflecting their severities. A property value that was set neither to its initial value nor to the value calculated in another file was considered an error. The after effects of an error were not considered to be additional errors. Errors typically occurred when participants either inserted the value of an incorrect property or guessed rather than used values from existing files.

4 Experiment setup

The purpose of the experiment was to gain further insights into how designers can take advantage of inconsistency information. In addition, the experiment served as a means of identifying potential areas of further improvement in communicating inconsistency information.

The goal of the experiment was to identify which of the two different representations of inconsistencies provided better results. Consequently we designed the experiment to provide an overall value measuring the performance of the participants in measuring the time needed for finishing the task with added penalties for errors and unfinished steps, along with an indication for the usage of the provided support measure which was identified through video analysis.

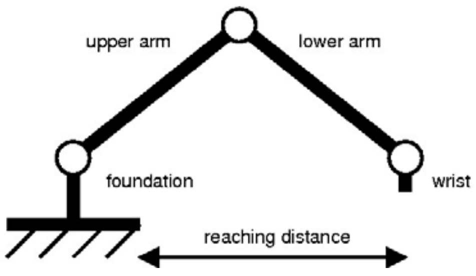
4.1 Participants

The experiment was supported by 22 participants who volunteered from the student body and staff of the Institute of Mechatronic Design and Production and the Institute for Software Systems Engineering at JKU Linz. All of the participants had a technical background in the fields of mechatronics (18), computer science (2), mechanical engineering (1) and material engineering (1) pursuing their bachelor (8), master (5) or PhD (9) degree. The participants were not compensated for their participation. None of the volunteers were female. All personal information obtained (including age, highest level of education, university credits, work experience and particular knowledge of engineering tools) was anonymised. None of the personal information obtained showed any significant correlation with the results of the experiment.

4.2 Task

The task the participants were asked to perform was about adapting the grasping distance of a robot arm in an engineering model – and the help they received while tackling it. Figure 1 shows a schematic of the robot under consideration. The scenario the participants of the experiment were presented with was described as follows: an important customer wants to buy robots from our company, but the current specifications within our existing portfolio do not fit his requirements. However, there is a robot which comes close and could be adapted accordingly. The person usually in charge of this specific robot design is currently not available and the participant has to adapt the grasping distance of this specific robot according to the requirements of the customer and apply the necessary ensuing changes so that the robot is functional. The results have to be available within the next 30 minutes.

Figure 1 Schematic of the robot model used for the design experiment



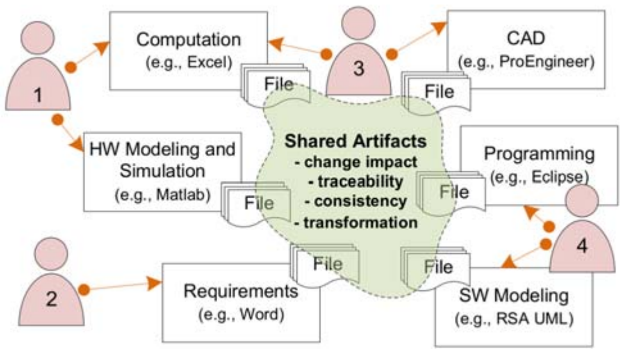
The change of the grasping distance, among other properties has effects on the length of the arms, the weight of the robot, the size of the drives at the joints, the power supply and on how robust the foundation has to be. The respective equations and corresponding calculations were part of the Excel files provided for the participants. Excel stands as representative for other used tools in mechatronic design (e.g., MATLAB®, SysML®, Modelica®, 3D-CAD) as it is widely used in industry and it is fair to assume that it does not require any special training for engineers in its usage.

The presented scenario requires changes to be performed which result in inconsistencies. The research aim is to identify which of the presented methods to represent the inconsistencies are more efficient to reach the goal to eliminate them. The collaboration aspects in the experiment address the combination of the different disciplines included in the field of mechatronics and sequential collaboration. Both are a necessary foundation for concurrent collaboration.

4.3 Experiment tool

The cloud-based DesignSpace (Demuth et al., 2015) was used in the experiment because it enables engineers to share knowledge and record interdependencies across engineering tool boundaries – as depicted in Figure 2.

Figure 2 Overview of DesignSpace (see online version for colours)



DesignSpace makes development artefacts publicly available such that cross-tool interdependencies can be established. Hence, it enriches the knowledge available in development tools with how this knowledge is interrelated. DesignSpace provides three types of services: data, engineering, and collaboration services.

- *Data services:* a central feature of DesignSpace is that it integrates the knowledge of development artefacts by continuously mirroring the artefacts. Whenever an artefact is changed by an engineer, this change is immediately and automatically propagated to DesignSpace and hence visible to others. The knowledge in DesignSpace thus reflects the current status of the system under development at all times. Note that the actual integration of individual artefacts is application-specific. This results in different domain-specific models (i.e., metamodels or ontologies). However, DesignSpace does not hard-code a specific metamodel or ontology; these may vary and can be changed like any other artefacts. In order to support this, DesignSpace offers an editor/visualisation tool called the WorkBench, which provides tool-independent visualisation and editing services to let engineers view and manipulate artefacts at will. This is not only useful for linking artefacts but also useful for navigation and visualisation to engineers
- *Engineering services:* in addition to representing and linking engineering knowledge, DesignSpace has the ability to detect inconsistencies through arbitrary user-definable constraints. For example, if a property derived from a calculation must be reused correctly in a drawing then this relationship can be effectively represented in form of an equality link that links these two artefacts. Another form of connection between properties is provided by a trace link, which indicates a general, unspecified dependency. The WorkBench provides the means to define such constraints, which are then verified by a consistency checker built into DesignSpace. The consistency checker reacts to changes and evaluates predefined constraints in a scalable manner – constraints that may range from simple equality constraints to more complex well-formedness rules across multiple artefacts.
- *Collaboration services:* collaboration is at the heart of DesignSpace. At the very least, DesignSpace supports asynchronous collaboration, as it embodies a revision control system that tracks at a fine-grained level how engineering artefacts came to be (i.e., their revision histories). A developer must be able to first adapt artefacts in private and then deliberately decide to share changes. However, we omit a more

detailed discussion of the collaboration services because they were of very limited relevance for the experiment presented in this paper.

4.4 Experimental setup

The robot model was distributed across four different Excel files. The first model was the requirements model, which contained the stakeholder requirements and their desired values (requirements model, 12 properties). The second model contained the calculations of lengths and angles (kinematics model, nine properties). The third model contained the calculations of the accelerations necessary to fulfil the requirements (dynamics model, 18 properties), and the fourth model contained the calculations of the resulting mass and power requirements of the robot (energy model, 43 properties). Output properties were indicated in pink and input properties in light green. For properties which were used both within a model and propagated to further models, both colour codes (pink and light green) were possible.

The cells in the Excel files that contained values were not connected using Excel formulas, but rather by means of DesignSpace. The cells of an Excel file could be exported using a provided tool adapter (realised with an add-in and comments in Excel). As soon as a cell was annotated with a comment in a specific syntax, it was exported to DesignSpace with the given name, unit and data type. The integration of the Excel files into DesignSpace is schematically depicted in Figure 3. Step 1 indicates the creation of the models along with their connection to DesignSpace. In addition to the four models, the computation and the arbitrary model emphasise the possibility for extension of the models.

After exporting all properties from the Excel files, the interdependencies were prepared for the experiment using WorkBench – a tool for editing and viewing the contents of DesignSpace. The interdependencies were realised as two different types of links:

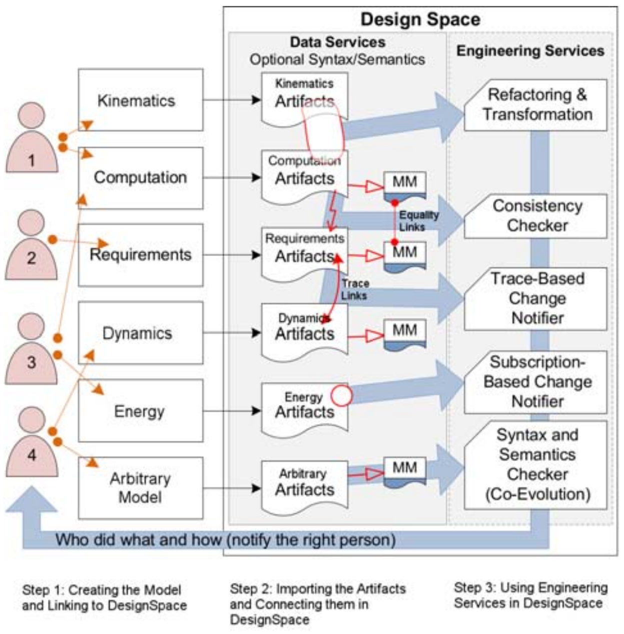
1 equality links, which indicate properties that must have the same value

2 trace links, where properties are connected to each other, for example, by belonging to the same mathematical equation.

These connections are indicated in the middle of Figure 3 (Step 2). The integration of the artefacts, along with their connection in DesignSpace, allows to form the METAMODEL (MM). Besides to the two types of links mentioned and used for the robot example, the design space allows for calculations with the artefacts and subscription to change notification.

Once an artefact or a connection was included into DesignSpace, the respective engineering services (see Step 3 on the right side of Figure 3) for it were activated. Of the five mentioned engineering services, the consistency checker and the trace-based change notifier were used for the robot example. At all times, DesignSpace determined and visualised the state of the consistency of the established equality links, and issued an alert when changes affected a trace link. The other three services (refactoring and transformation, subscription-based change notifier and the syntax and semantics checker) were not used in the robot example.

Figure 3 Schematics of integration of robot example into DesignSpace (see online version for colours)

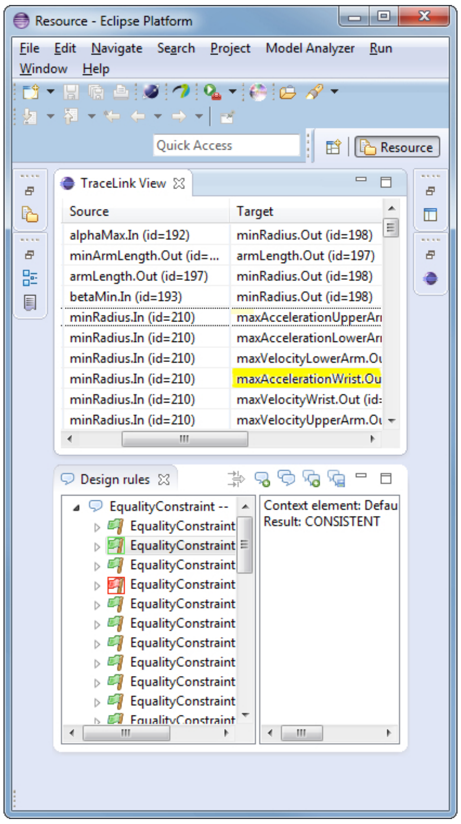

Design Space
Data Services
Optional Syntax/Semantics
Engineering Services
Kinematics
Computation
Requirements
Dynamics
Energy
Arbitrary Model
Kinematics Artifacts
Computation Artifacts
Requirements Artifacts
Dynamics Artifacts
Energy Artifacts
Arbitrary Artifacts
MM
Equality Links
Trace Links
Refactoring & Transformation
Consistency Checker
Trace-Based Change Notifier
Subscription-Based Change Notifier
Syntax and Semantics Checker (Co-Evolution)
1
2
3
4
Who did what and how (notify the right person)
Step 1: Creating the Model and Linking to DesignSpace
Step 2: Importing the Artifacts and Connecting them in DesignSpace
Step 3: Using Engineering Services in DesignSpace

In addition to the Excel files and DesignSpace, the supervisors of the experiment prepared two different visualisations of the interdependencies within the robot model: a node-link graph and a design structure matrix printed out on paper size A3.

4.5 Experiment flow

The experiment was carried out as follows. The participants of the experiments (the present robot experiment and another one regarding iteration) each had a pre-arranged appointment. First they had to sign a consent form. Second they had to choose the specific tasks they had to perform through an urn draw. Afterwards, they filled out an entry questionnaire regarding their age, gender, education level, etc. During this time, the experiment supervisor prepared the specific support measures and the experiment setting. Once the participants were finished with the questionnaire, they received the introductory information for the experiment. They had as much time as they wanted for studying this introductory information. Once they were satisfied with this information, they were asked to fill out a comprehension questionnaire, during which they were also allowed to ask questions. After the first experiment, a break of approximately 15 minutes was scheduled, during which the participants had to refrain from any outside contact. Following this break, the participants received the introductory information for the second experiment, again followed by a comprehension questionnaire during which it was again allowed to ask questions. After the second experiment, the participants had to fill out a feedback form and were questioned by the supervisor regarding their experience and opinion of the experiment, to evaluate possible hypothesis awareness.

Figure 4 DesignSpace information about trace links and inconsistency (see online version for colours)


Resource - Eclipse Platform
File Edit Navigate Search Project Model Analyzer Run
Window Help
Quick Access
Resource
TraceLink View
Design rules

Source	Target
alphaMax.In (id=192)	minRadius.Out (id=198)
minArmLength.Out (id=...	armLength.Out (id=197)
armLength.Out (id=197)	minRadius.Out (id=198)
betaMin.In (id=193)	minRadius.Out (id=198)
minRadius.In (id=210)	maxAccelerationUpperAr
minRadius.In (id=210)	maxAccelerationLowerAr
minRadius.In (id=210)	maxVelocityLowerArm.Ou
minRadius.In (id=210)	maxAccelerationWrist.Ou
minRadius.In (id=210)	maxVelocityWrist.Out (id:
minRadius.In (id=210)	maxVelocityUpperArm.Ou

EqualityConstraint --
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint
EqualityConstraint

Context element: Defau
Result: CONSISTENT

To perform the task, the participants received introductory material and one of the representations of the interdependencies. Half of the participants received a node-link graph and the other half a design structure matrix visualisation (Keller et al., 2006). The model was explained in the introductory material that was given to the participants along with instructions on how to use the model. Details of the model were accessible to the participants only after the experiment had started. The introductory material also explained DesignSpace, which provided information on inconsistencies as soon as they occurred.

The participants in the experiment had to change the property value of the grasping distance of a robot from 2 m to 3 m and clear all the inconsistencies resulting from this change. As previously mentioned, properties were shared between models. Thus, a change in an output property required changes in the corresponding input properties (and subsequent properties) to maintain consistency. The grasping distance had to be changed in the requirements model, which affected all other files. The time allocated for the completion of this task, which required 17 correct changes across 4 Excel files, was 30 minutes.

The names of the properties used within the models were similar in some, but not in all, cases – reflecting realistic situations where naming conventions cannot be trusted fully. Hence, it was not sufficient to merely ensure that all the properties with the same name had the same values. All calculations were done automatically within the Excel files. The only changes to be made by the participants were to manually enter the property values calculated within one file into another file if value changes were necessary. As soon as DesignSpace detected an inconsistency in the equality links, a red flag indicated an inconsistency. Figure 4 shows a view of DesignSpace's workbench used in the experiment. The trace link view in the upper half of Figure 4 indicates changes in connected properties by yellow highlighting. Inconsistencies are shown in the bottom. By clicking on these artefacts, more information is revealed.

In a test run, a test person randomly changed properties in the hope of guessing correct values. To avoid such behaviour, cells not intended to be changed were locked (write-protected). This protection was not visible to the participants unless they sought to change a locked cell. Properties that were allowed to be changed were indicated by a light blue colour in the cell to their left.

After the experiment, the participants were asked to complete a questionnaire to assess goal awareness. All participants were told that the task was to evaluate DesignSpace in practice. The participants were unaware of the other goal, which was to evaluate the effects of the two different visualisations of the model interconnections. The intent was to ensure that the participants had no bias as to either visualisation style.

The experiment was performed on a computer with an Intel(R) Core(TM) i5 CPU 650 @ 3.20 GHz and 4 GB RAM using the Windows 7 Professional 64-Bit operating system. The working files were created and used with Excel 2010. DesignSpace software was a prototype without release number. A current version of the prototype of DesignSpace is available at http://isse.jku.at/tools/dsspc/xadr.zip (pw: dsisse).

5 Results

5.1 Measured data

This section presents the results of the design experiment. To support the hypothesis, it was expected that the usage of the matrix representation would already provide significantly better results (based on the calculated combined time value) than the representation with the node-link graph.

The solution strategies which the participants applied were identified and categorised through analysis of the video recordings of each participant. Participants' usage of the information sources varied, as three different solution strategies were observed: The first and intended solution strategy with respect to the hypothesis was to use DesignSpace solely as an information source for inconsistencies and the graphical representation of the system for identifying the interdependencies for the propagation of changes (see top of Figure 5).

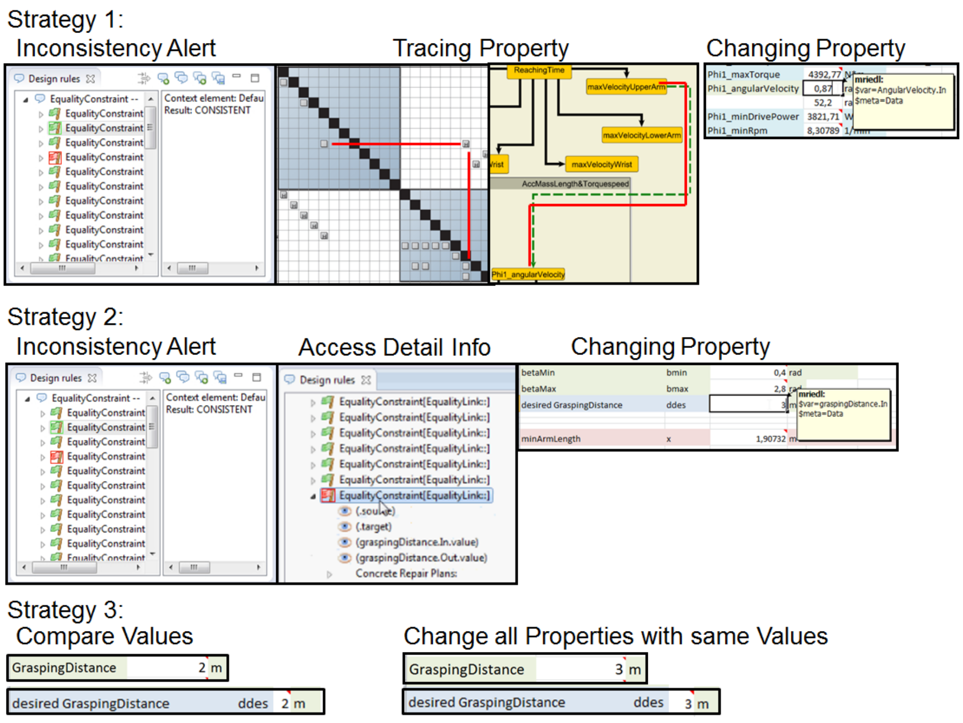
The second approach we observed was to access detailed information about the task through DesignSpace. When an inconsistency showed up with a red flag, the name of the properties causing the inconsistency could be retrieved by two mouse-clicks at the right position. Even though the filename was not provided, this information still narrowed down the search considerably. This solution approach was not foreseen. DesignSpace allowed identification of properties via the names used in the equivalence connections. While this could have been avoided by using random names, it would have made creating the interconnections in WorkBench significantly more complicated. The participants were not informed about the additional information within DesignSpace, but about the possibility to explore DesignSpace (see middle of Figure 5).

The third approach was to compare the property values and change them under the assumption that values that were initially identical should remain identical. In combination with the inconsistency flags of DesignSpace, this strategy also allowed the completion of the task. This last approach was used only by two participants and was identified in the discussion following the experiment (see bottom of Figure 5)

All participants used the intended solution approach at least to some degree. Interestingly, the implicit requirement that units of properties which must have identical values must also be identical (see also Coatanéa et al., 2014) was not recognised or used by any of the participants.

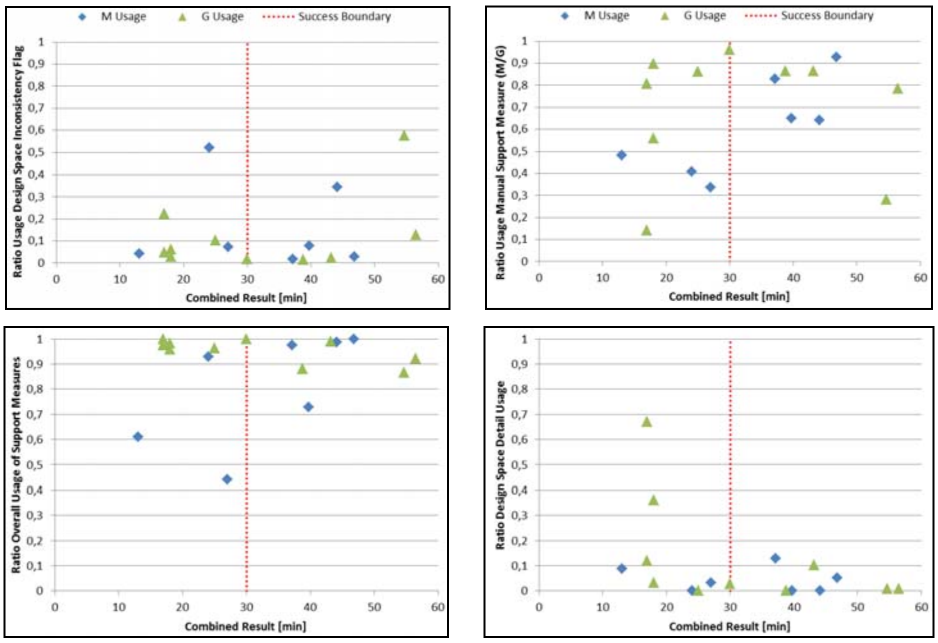
Distinguishing between the three different approaches proved difficult in the video analysis. To have clear boundaries, we proceeded as follows: Whenever the participants bowed their heads to study the visual representation of the system interconnection, the corresponding time was assigned to graph or matrix support. When the participants accessed the details of DesignSpace, time was assigned to DesignSpace. Whenever the mouse cursor was over DesignSpace showing solely inconsistency information, this time was assigned to use of DesignSpace for inconsistency information. Other behaviors, such as taking notes, were not assigned to any of the three behaviors of interest.

Figure 5 Observed solution strategies (see online version for colours)



As no eye-tracking technology was applied, there is some uncertainty regarding the usage of the specific support measures. It cannot be guaranteed that the position of the mouse actually reflected a participant's area of interest. It is possible that a participant had the mouse cursor over DesignSpace but was actually looking at the values in the Excel files. However, we feel confident that our observations were mostly accurate – in part since our findings reflect subsequent discussion with participants. Since complete assignment of all activities shown in the video recordings was not possible, the sum of the respective ratios can be below the maximum of 1. Using DesignSpace for detailed information required interaction with the mouse cursor. Figure 6 shows the combined result on the X-axis and the respective usage ratios on the Y-axis.

Figure 6 Ratios of support measures over the combined results (see online version for colours)



5.2 *Data analysis*

Statistical analysis was applied to help to identify potential tendencies for future research. All statistical analyses were performed with RStudio (version 0.98.1028) using R (version 3.1.1, 2014-07-10).

Of the 20 participants considered for evaluation, 15 used DesignSpace for more than 10% of the time available. To evaluate this result in more detail, a future experiment should apply eye-tracking technology in order to identify which aspect of the representation is used. This result is taken as an indication that the tool is generally acceptable in an engineering environment.

All recorded data were analysed to identify correlations. No item of personal data showed a significant correlation with the results. The main finding was that the ratio of detail usage is significantly correlated with the combined result (Pearson's correlation coefficient of –0.503, statistically significant at 0.05 level). Thus, the detailed information provided within DesignSpace clearly showed a statistically significant positive influence on the results.

An interesting and unexpected observation was that, in the course of the experiment, contrary to the goal of the supervisors of the experiment, the participants started to use the details of DesignSpace increasingly as an information source (Kendall's tau of –0.627, statistically significant at 0.001 level). We expected that this could be explained by age and greater work experience. However, independently, age and work experience did not show any significant correlation with the combined results or detail use of DesignSpace. We tend to rule out an unintentional influence of the supervisor of the experiment towards the detail use of DesignSpace as it was not in line with the experiment hypothesis. In addition, the data was only analysed after the experiment, which allows us to rule out result awareness of the detail use of DesignSpace to the supervisor, who did not observe the participants in detail, but just controlled the experiment environment. The correlation likely is coincidental and the participants just used the medium they found the most useful. Nevertheless, this result is vulnerable to outliers, (Figure 6, bottom right) due to the small sample size and requires further investigation for validation. The correlation coefficient between combined results and order of participants was not statistically significant (Kendall's tau of –0.292).

Both the material and the supervisor of the experiment mentioned that the purpose of the experiment was to evaluate DesignSpace in practice. This could have introduced a potential bias (Cash et al., 2012) towards DesignSpace.

The initial hypothesis that a matrix representation provides a higher success rate than a node-link graph could not be verified with the experiment. Neither of the two visualisations proved superior. Hence, our hypothesis must be rejected. However, the experiment showed that the online usage of detailed information provided by DesignSpace resulted in higher success rate. While this was not the subject of the study, it appears that ready availability to design knowledge is more important than any particular offline visualisation style in particular. This assumption requires further study. Of the observed variables, only the detail usage of the information provided within DesignSpace shows a statistically significant influence.

5.3 Threats to validity

The colour indication which cells were allowed to be changed constitutes a threat to validity. Obviously, in a real-world scenario, such indications are not available, although research exists which aims to provide them. In general, a change task comprises two steps: identifying which properties need to be changed (the where) and how these need to be changed (the how). In our experimental setup, the where was to some extent already answered due to the light-blue highlighting. However, this highlighting did not inform the participants about the interdependencies between the properties, which were presented by the two visualisations and DesignSpace. The colour highlighting only served as a hint within the Excel sheets.

Considering the surprisingly small success rate – only 11 of the 20 participants completed the task – it can be argued that the task was sufficiently difficult and that the explanations may have been deficient. Furthermore, the participants did not know the robot model and had to perform the task under time pressure. Both these boundary conditions are not necessarily found in a real-world scenario.

Despite the colour indication, the participants still needed to find a feasible way of performing the assigned task (the how). Overall, there were 82 possible property values.

Depending on the sub-model under consideration, the numbers of options were 39, 42, 48 and 73, respectively. Therefore, we can conclude that, although the highlighting was present, the task was sufficiently difficult for the participants, and the validity of our results is not diminished.

5.4 Future changes and improvements

After the experiment, the participants had the opportunity to discuss the answers to the questionnaire and their overall experience with the experiment with the supervisors of the experiment. The general consensus was positive, the risk of seeking to please the experimenter notwithstanding. However, in addition to the positive feedback regarding DesignSpace, there was important feedback on the presentation of the information. The participants would have preferred the inconsistencies to be presented interactively. Currently, the inconsistency information is not directly linked to the user interface of the software where the specific models are implemented. Such an inclusion would significantly improve usability. In addition, some participants expressed the wish for the inconsistency information to be highlighted in the general visualisation.

This wish for more specific representation of inconsistency information also reflects the outcome of the experiment in relation to the positive effect of the detailed and specific information on performance. The participants performed better when using the detail information provided in DesignSpace. With increasing user experience, the handling is expected to improve. Ideally, an inconsistency can be presented either in the native environment of the specific model or in the form of a graphical visualisation of the overall system model with the option to indicate both the specific models and the specific properties affected.

Further improvements include better usability of WorkBench for industrial use. The current prototype must be improved in terms of robustness and stability. For optimal usage, WorkBench has to be included in the engineering process to reduce complexity for the system architect. An enhancement currently under consideration is to provide information on how to overcome and solve an inconsistency beyond the trivial solution of changing one of the two properties causing it. Further, we are planning to incorporate aspects of change propagation in the inconsistency management.

In addition to the tool adapter for Excel, DesignSpace already features tool adapters for E-Plan, Creo, Rational Software Architect and Eclipse. These are being used in industrial and academic test applications, and further adaptations are in planning.

6 Conclusions

The initial hypothesis that in case of indication of inconsistencies, matrix representation of interconnections result in a shorter time necessary to solve the task in comparison to node-link representations of interconnections, was not verified with the experiment. However, the experiment showed that online, on-demand access to design knowledge through DesignSpace successfully supported engineers in identifying inconsistencies across domains. The information provided about inconsistencies is acceptable to engineers, and the detailed information on model interconnections and inconsistencies improved their performance when executing engineering changes.

While this paper supports and validates practical effectiveness of DesignSpace, it also reveals critical areas for improvement. In the experiment, specific information about inconsistencies provided more useful support than general information about system interconnections, independent of the type of representation.

Acknowledgements

This work was supported in part by the Austrian COMET-K2 programme of the Linz Center of Mechatronics (LCM), and was in part funded by the Austrian Federal Government and the Federal State of Upper Austria.

References

Alavi, M. and Leidner, D.E. (1999) 'Knowledge Management Systems: Issues, Challenges, and Benefits', *Communications of the AIS*, Vol. 1, Article 7.

Ballard, G. (2000) 'Positive vs negative iteration in design', *Proceedings of the Eighth Annual Conference of the International Group for Lean Construction*, pp.44–55.

Boehm, B.W. (1984) 'Software engineering economics', *IEEE Transactions on Software Engineering*, Vol. 10, No. 1, pp.4–21.

Böhnke, D., Reichwein, A. and Rudolph, S. (2009) 'Design language for airplane geometries using the unified modeling language', in Horváth, I. and Rusák, Z. (Eds.): *Proceedings Tools and Methods of Competitive Engineering TMCE 2012*, pp.985–997.

Cash, P., Elias, E., Dekoninck, E. and Culley, S. (2012) 'Methodological insights from a small scale design experiment', *Design Studies*, Vol. 33, No. 2, pp.208–235, doi: 10.1016/j.destud.2011.07.008.

Coatanéa, E., Nosiri, S., Mokammel, F., Christophe, F., Calonius, O. and Ryynänen, L. (2014) 'Contradiction graph search: a tool for discovering for physical contradictions', in Horváth, I. and Rusák, Z. (Eds.): *Proceedings Tools and Methods of Competitive Engineering TMCE 2012*, pp.993–1006.

Demuth, A., Riedl-Ehrenleitner, M., Nöhrer, A., Hehenberger, P., Zeman, K. and Egyed, A. (2015) 'DesignSpace: an infrastructure for multi-user/multi-tool engineering', *Proceedings of the 30th ACM/SIGAPP Symposium on Applied Computing*, in press.

Ehrlenspiel, K., Kiewert, A. and Lindemann, U. (2007) *Cost-Efficient Design*, Springer, Berlin.

Follmer, M., Hehenberger, P. and Zeman, K. (2012) 'Application of the mechatronic-system-model-approach for the development of an industrial annealing simulator', in Marjanovic, D., Storga, M., Pavkovic, N. and Bojcetic, N. (Eds.): *Proceedings of the 12th International Design Conference DESIGN 2012*, pp.637–646, The Design Society, Glasgow.

Follmer, M., Hehenberger, P., Punz, S., Rosen, R. and Zeman, K. (2011) 'Approach for the creation of mechatronic system models', in Culley, S., Hicks, B., McAloone, T., Howard, T. and Lindemann, U. (Eds.): *Proceedings of the 18th International Conference on Engineering Design ICED'11*, pp.258–267, The Design Society, Glasgow.

Friedenthal, S., Moore, A. and Steiner R (2014) *A Practical Guide to SysML: 3rd Edition: The Systems Modeling Language*, Morgan Kaufmann, Waltham.

Ghoniem, M., Fekete, J-D. and Castagliola, P. (2005) 'On the readability of graphs using node-link andmatrix-based representation: control experiment and statistical analysis', *Information Visualization*, Vol. 4, No. 2, pp.114–135.

Hehenberger, P., Poltschak, F., Zeman, K. and Amrhein, W. (2010) 'Hierarchical design models in the mechatronic product development process of synchronous machines', *Mechatronics*, Vol. 20, No. 8, pp.864–875, doi: 10.1016/j.mechatronics.2010.04.003.

Herczig, S.J., 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, doi: 10.1016/j.procs.2014.03.044.

Hubka, V. and Eder, W.E. (1996) *Design Science: Introduction to the Needs, Scope and Organization of Engineering Design Knowledge*, Springer, London.

Jarratt, T.A.W., Eckert, C.M., Caldwell, N.H.M. and Clarkson, P.J. (2011) 'Engineering change: and overview and perspective on the literature', *Research in Engineering Design*, Vol. 22, No. 2, pp.103–124, doi: 10.1007/s00163-010-0097-y.

Keller, R., Eckert, C.M. and Clarkson, P.J. (2006) 'Matrices or node-link diagrams: which representation is better for visualising connectivity models?', *Information Visualization*, Vol. 5, No. 1, pp.62–76, doi: 10.1057/palgrave.ivs.9500116.

Komoto, H. and Tomiyama, T. (2012) 'A framework for computer-aided conceptual design and its application to systems architecting of mechatronics products', *Computer-Aided Design*, Vol. 44, No. 10, pp.931–946, doi:10.1016/j.cad.2012.02.004.

Leon, H.C.M., Farris, J.A. and Letens, G. (2013) 'Improving product development performance through iteration front-loading', *IEEE Transactions on Engineering Management*, Vol. 60, No. 3, pp.552–565.

Lucero, B., Viswanathan, V.K., Linsey, J.S. and Turner, C.J. (2014) 'Identifying critical functions for use across engineering design domains', *Journal of Mechanical Design*, Vol. 136, No. 12, pp. 121101-1-121101-11, doi: 10.1115/1.4028280.

Maier, J.F., Wynn, D.C., Biedermann, W., Lindemann, U. and Clarkson, P.J. (2014) 'Simulating progressive iteration, rework and change propagation to prioritize design tasks', *Research in Engineering Design*, Vol. 25, No. 4, pp.283–307, doi: 10.1007/s00163-014-0174-8.

Moser, T. and Biffl, S. (2012) 'Semantic integration of software and systems engineering environments', *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 42, No. 1, pp.38–50, doi: 10.1109/TSMCC.2011.2136377.

Nuseibeh, B., Easterbrook, S. and Russo, A. (2001) 'Making inconsistency respectable in software development', *Journal of Systems and Software*, Vol. 58, No. 2, pp.171–180.

Qamar, A. and Paredis, C.J.J. (2012) 'Dependency modeling and model management in mechatronic design', *Proceedings of the ASME 2012 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*.

Safoutin, M.J. (2003) *A Methodology for Empirical Measurement of Iteration in Engineering Design Processes*, PhD diss., University of Washington.

Wynn, D.C., Eckert, C.M. and Clarkson, P.J. (2007) 'Modelling iteration in engineering design', *Proceedings of the International Conference on Engineering Design ICED'07*.