Proceedings of the TMCE 2012 Symposium, May 7 - 11, 2012, Karlsruhe, Germany.
© Organizing Committee of TMCE 2012, ISBN ----

DEVELOPMENT OF INDUSTRIAL VISUALIZATION TOOLS FOR VALIDATION OF VEHICLE CONFIGURATION RULES

Anna Tidstam
Department of Product and Production Development, Product Development
Chalmers University of Technology, Sweden
tidstam@chalmers.se

Lars-Ola Bligård
Department of Product and Production Development, Design & Human Factors
Chalmers University of Technology, Sweden
lars-ola.bligard@chalmers.se

Knut Åkesson
Alexey Voronov
Department of Signals and Systems, Automation
Chalmers University of Technology, Sweden
{knut.akesson, alexey.voronov}@chalmers.se

Fredrik Ekstedt
Fraunhofer-Chalmers Research
Centre of Industrial Mathematics, Sweden
fredrik.ekstedt@fcc.chalmers.se

Johan Malmqvist
Department of Product and Production Development, Product Development
Chalmers University of Technology, Sweden
johan.malmqvist@chalmers.se

ABSTRACT

This paper addresses the industrial visualization tools used when validating vehicle configuration rules. The configuration rules are logic expressions allowing vehicle configurations to be built, as well as which components these configurations should consist of. Valid configuration rules are permitting vehicle configurations according to the specialists' expectations. Those vehicles also have to have the correct components assigned. Currently, the validation of configuration rules partially relies on time-consuming manual inspection and calculations. Our aim is to find a demonstrator facilitating the validation of configuration rules by adding the calculations results to the industrial visualization methods. In doing so, it should also be possible to use one single user interface instead of the multiple used today. The work presented in this paper is rooted in user studies at the automotive companies including both interviews and observations. The identified typical rule queries when using the industrial visualization tools and the identified difficulties informed the development of a demonstrator. This paper describes the conducted usability tests of the demonstrator, showing how the users found the validation of configuration rules less error-prone, more time-efficient and easier to learn.

KEYWORDS

Validation of configuration rules, formative usability test, user interface, development process automation

1. INTRODUCTION

In a simple case, there may be a very direct link between a customer selection and a manu-factured/sourced item, e.g. a computer drive of a certain size. However, the configuration of more complex and variant-rich products, such as vehicles, introduces some additional challenges. Vehicle configuration requires two domains, both the feature domain for customer selections and the item domain for design and manufacturing. Therefore, there are both intra- and inter-domain configuration rules [1]:

- **Product model authorization rules** define for which product model variants (e.g. "Volvo V70", "BMW 3 Sedan" etc) a specific feature variant (e.g. "sunroof") is authorized.

- **Feature variant combination rules** define prescribed ("inclusions") or forbidden ("restrictions") combinations of feature variants.
- **Item usage rules** define for what feature variant combinations a certain item should be used.

The feature variants are organized into feature families. The "configuration" is created by selecting one of the mutually exclusive feature variants from the feature families; e.g. the feature variant "with cup holder" may be avoided by selecting the feature variant "without cup holder".

According to [2], validation of configuration rules is about "accuracy" and "representativeness". The configuration rules are assumed to be valid if they are consistent with domain specialists' perceptions of which vehicles that should be allowed to be built. The domain specialists are people with specialist knowledge about the domain, e.g. the design engineer of brakes. The configuration rules are validated during the vehicle development process. The process model for developing vehicle configuration rules is described in Fig. 1. The process is initiated with a product modification request, expressed in terms of the automotive information model. New or modified configuration rules may be required to fulfill the product modification request. In [3] there is classification between high-level (sales) and low-level (engineering) configuration, where the low-level is characterized by non-interactive procedures. Interactive in this context means that the user configures the vehicle by iteratively selecting feature variants. The validation of configuration rules instead relies on manual inspection, calculations and virtual or physical builds [4]. After iterations the configuration rules are released for production, where they may be used for accepting customer orders and manufacturing of vehicles.

The contribution of this paper is an evaluation of the current industrial visualization support for how well it supports the users during the validation of configuration rules, especially considering the activities of the design engineer. Another contribution is the suggested visualization method which has been found addressing factors causing the validation of configuration rules to be difficult and time consuming.

1.1. Motivation and objectives

The validation of configuration rules is not possible to fully automate since the complete knowledge and detailed reasoning of the domain-specialist is not formalized [5].

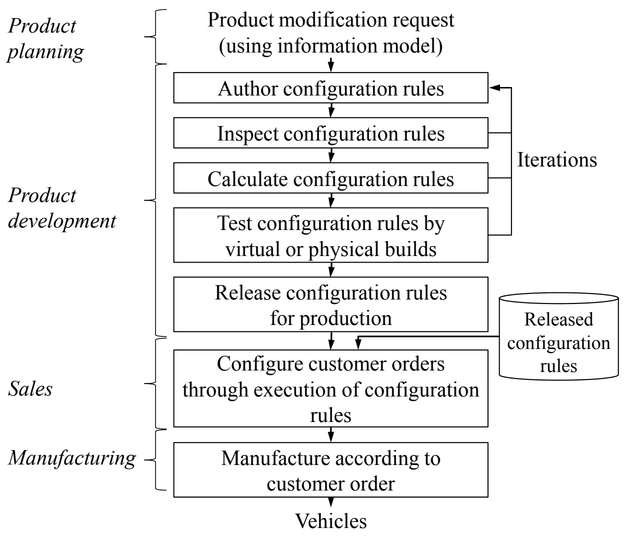

Product planning
Product modification request (using information model)
Author configuration rules
Inspect configuration rules
Calculate configuration rules
Test configuration rules by virtual or physical builds
Iterations
Product development
Release configuration rules for production
Released configuration rules
Sales
Configure customer orders through execution of configuration rules
Manufacturing
Manufacture according to customer order
Vehicles

Figure 1 Process model for the development of vehicle configuration rules.

The configuration rules may be compared to the large number of rules generated by data mining algorithms [6]. Three main areas of data mining are interestingness, rule query and visualization. It is difficult to user rule queries to find interesting configuration rules, if the user does not know what the rules are. Little research has been done studying how visualization could help the users to find interesting configuration rules. In [7] there is, however, a matrix-based visualization method that has been proven to support the user to fast identify interesting and actionable rules. Actionable means that the user is able to do something in order to obtain a desired effect.

Vehicle configuration rule sets can be very large, numbering several tens of thousands of rules, and the rules can be restricting combinations up to 100 hundred feature variants [8]. There is a potential for automating some steps during the validation of configuration rules by taking advantage of computations [9]. There is, however very little research on how to use those computations when visualizing the configuration rules. Given the high number and complexity of rules, as well as the low degree of automation, the manual inspection of configuration rules is currently a very time-consuming, difficult and error-prone activity. The users find the visualization tools for configuration rules difficult to use [10]. Often those users do not rely on their own capability of independently validating configuration rules with the visualization tool, or they do not trust the information stored there.

Also, a change in the feature variant combination rules may result in necessary modifications to the item usage rules etc. This leads to issues since the different classes of configuration rules are interconnected but are currently developed in different user interfaces. The basic idea of this paper is to develop a demonstrator visualizing the different classes of configuration rules in one single user interface; see Fig. 2. The currently used visualization tools force the users to go back and forth between multiple user interfaces, which results in a higher cognitive load.

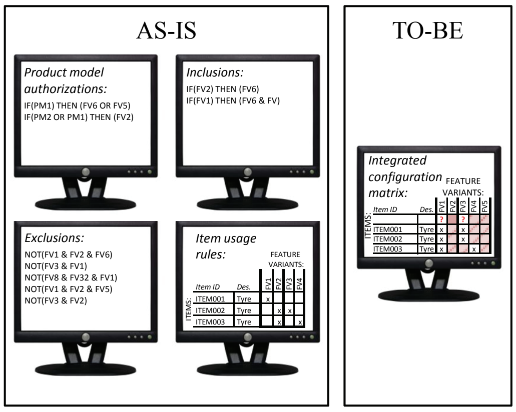


Figure 2 From multiple user interfaces and viewing format (AS-IS) to one single integrated configuration matrix (TO-BE).

The hypothesis is that a demonstrator based on a single user interface would facilitate the validation of configuration rules, resulting in fewer errors and less time-consuming process. The usability of the demonstrator may be measured by errors (how many and how severe), learnability (how easy it is to accomplish task the first time) and efficiency (how quickly) [11]. More specifically, the stated research questions are:

RQ1: What are the strengths and weaknesses of industrial visualization tools used when validating vehicle configuration rules?

RQ2: Which visualization tool addresses those weaknesses?

The remainder of the paper is organized as follows. The research method is presented in Section 2. The results are described in Section 3 followed by a discussion in Section 4. Finally, Section 5 presents the conclusions and future work.

2. RESEARCH METHOD

The starting point for this research paper is the results from a user study described in [9]. One of the outcomes of the user study was a formalization of the authoring methods used, which is an automation potential that initiated the further development of the industrial visualization tools. The user study included both interviews and observations, and was conducted at three large automotive companies operating in-house developed visualization tools for configuration rules. The interviewees were either design engineers or product structure specialists. The description of current visualization tools, typical rule queries, visualization needs and design considerations are based on the data collected during the user study. The literature on visualization tools and methods helped in defining the industrial visualization tools as well as motivating the research. The pres-study findings are described in section 3.1 in the result chapter, see Fig. 3. The second phase described in section 3.2 was the creation of the demonstrator. The applied visualization method was proposed by several interviewees during the user study. The demonstrator was then used developing and evaluating through formative usability tests. Those tests are described in section 3.3. A positive outcome of the formative usability tests [12] would be that all test participants are able to perform the tasks with the demonstrator without errors as well as doing it time-efficiently. The tests were based on industrial data and the participants were real users for the application, i.e. design engineers and product structure specialists from the studied automotive companies. A small number of users were selected for the formative

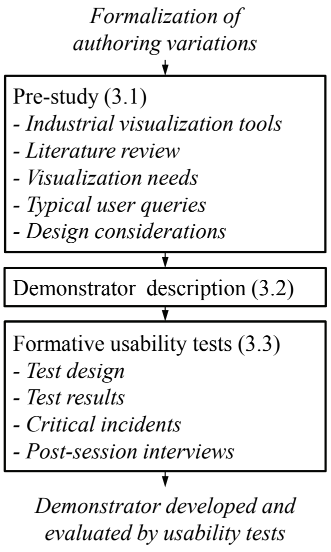

Formalization of authoring variations
Pre-study (3.1)
- Industrial visualization tools
- Literature review
- Visualization needs
- Typical user queries
- Design considerations
Demonstrator description (3.2)
Formative usability tests (3.3)
- Test design
- Test results
- Critical incidents
- Post-session interviews
Demonstrator developed and evaluated by usability tests

Figure 3 Outline of research process.

usability evaluation, with a focus on extracting as much information as possible from every test user, which is following the guidelines described in [11]. A pilot test with one user was conducted to estimate the adequate number of tasks. Then three usability tests were conducted, separated into two test series by a redesign of the demonstrator. The redesign of the demonstrator is an important step, as iterative design is the best method according to usability engineering to increase the quality of user experience [11].

The tests were formative; e.g. the purpose of the tests was to improve elements of the visualization support. The data collection consisted mainly of critical incidents notations. A critical incident is something that happens during the test that has a significant positive or negative impact on task performance or user satisfaction. A typical negative critical incident is something that causes an error or something that blocks the progress.

The usability test consisted of 14 tasks from four typical rule queries, as well as three post-session interview questions. The computer screen was captured for being able return to the critical incidents as well as to some extent measuring response times. The errors were measured when the wrong answer to the tasks were filled in on the test sheet. The tests also considered learnability and time efficiency. The time efficiency was measured when the user had some experience from performing tasks with the demonstrator.

The usability test contained post-sessions interviews to further let the users estimate the usability of the demonstrator. The formative usability tests did not contain any benchmark tasks comparing the current visualization support with the demonstrator, but the response times were measure in case it was possible to do so.

3. RESULTS

The three main sections describing the results are the pre-study, the description of the demonstrator description and the formative usability tests.

3.1. Pre-study

The pre-study started with a description of industrial visualization tools used when validating vehicle configuration rules and a literature review on visualization methods. Then the typical rule queries are listed and the design considerations discussed.

Industrial visualization tools

The studied automotive companies are using visualization tools similar to the old mainframe applications in terms of displaying information with very limited use of colours and very extensive use of text. The visualization tools are using various visualization methods, defined as "systematic and rule-based graphical representations aiming to acquire insights, develop an elaborate understanding or communicate experiences" [13]. Examples of visualization methods are tables, pie charts, mind maps, Gantt charts, decision trees etc. This paper is discussing the lists, tables and matrices, and how these methods are used in visualization tools for validation of vehicle configuration rules. The demonstrator is based on a matrix, but may easily be transformed to a tree graph which is sparsely used visualization method found at one of the studied companies.

Three item usage rules are presented as a list, table and matrix in Fig. 4. The first item usage rule in the list is equal to the first row in the table and the matrix. The list example has one heading "Item usage rules", followed by the logic expressions. The table may be used if the data may be classified into groups, e.g. "Item ID" and "Feature variants". The

Item usage rule list:

Item usage rules
IF (18inchtyre) THEN (ITEM001)
IF (stdwheel) THEN (ITEM002)
IF (20inchtyre & sparewheel) THEN (ITEM003)

Item usage rule table:

Item ID	Feature variant 1	Feature variant 2
ITEM001	18inchtyre	
ITEM002	stdwheel	
ITEM003	20inchtyre	sparewheel

Item usage rule matrix:

			Feature families:			
			A		B	
			Feature variants:			
	Item ID	Description	18inchtyre	20inchtyre	stdwheel	sparewheel
Items:	ITEM001	Tyre	x			
	ITEM002	Tyre			x	
	ITEM003	Tyre		x		x

x = feature variant included in item usage rule referring to the row's item.

Figure 4 Item usage rules presented both as list and matrix.

matrix may be used if more column- and row-headings are introduced, similar to the pivot table, e.g. feature family “A” and feature variant “18inchtyre” etc. The matrix rows contain crosses referring to the feature variants written as column headings.

Literature review

There is very little research on industrial visualization tools for validating vehicle configuration rules. The few screenshots of user interfaces found in the literature are shown for motivating a change of the information model, as in [8]. Visualization methods are more extensively covered, especially the matrix-based methods, e.g. the dependency structure matrix, henceforth DSM [14]. This “intra-domain” matrix consists of square matrices with empty diagonal elements [15, 16]; see Fig. 5. The purpose of the DSM is either sequencing or clustering analysis [17]. The DSM has not been used for validating vehicle configuration rules. Instead, the purpose is an optimization of the product structure, e.g. finding potential modularization [18].

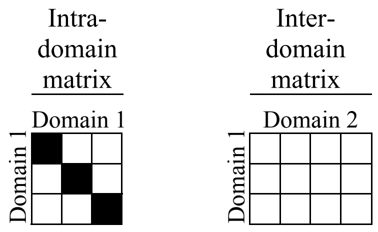


Figure 5 Comparison between intra-domain matrix and inter-domain matrix, adapted from [17].

Another visualization method is to combine inter-domain matrices with intra-domain matrices. It is then possible to create matrices with computational values from intra/inter-domain matrices, as e.g. when using the “K- and V-Matrix” method [15]. The K- and V-Matrix method use three matrices; see Fig. 6. The V-Matrix is an intra-domain matrix showing pairwise compatibilities between feature variants. The configuration rules is commonly longer than stating pairwise relations, so the V-Matrix is capable of showing a simplification of the feature variant combination rules. Then the K-Matrix shows the item usage rules by an inter-domain matrix. The correspondence to the V-Matrix for items does not exist as a documentation of product structure at the studied automotive companies. An item usage rules matrix with 4 item usage rules using in total 10 different feature families with an average of 4 feature variants would create a matrix with 4 rows and 40

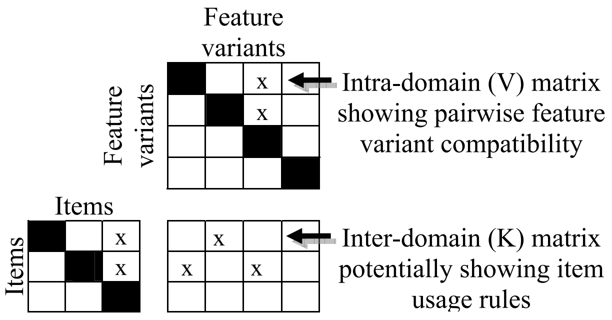

Feature variants
Feature variants
Intra-domain (V) matrix showing pairwise feature variant compatibility
Items
Items
Inter-domain (K) matrix potentially showing item usage rules
x

Figure 6 K- and V-Matrix using both intra-domain and inter-domain matrices, adapted from [15].

columns. By adding the V-matrix for feature variants, there is an additional 40 rows required. The size of the K- and V-matrix fast becomes unmanageable. Using more than one matrix is therefore not aligned with this paper’s aim to not drastically increase the size of the currently used matrices.

There are also visualization methods for how to display the calculated configurations, see Fig. 7. The table of configurations should be read as truth tables: one column for each feature variant (e.g. “a1” and “b2”), and each row contains one configuration of the feature variants (e.g. “a1” and “b1”, and “c3”).The table of configurations potentially grows exponentially as the number of included feature families increases. The large size of the table causes it to become time-consuming to inspect. One method for reducing the number of table rows is to use the Cartesian product representation [19]. The Cartesian product representation reduces the number of rows by introducing the “or” operator within feature families, e.g. “a1” and “b1” and (“c1” <u>or</u> “c3”). The demonstrator’s visualization method is further reducing the number of rows into one single row. The demonstrator is using pink fills for the feature variants that are never allowed in the configurations.

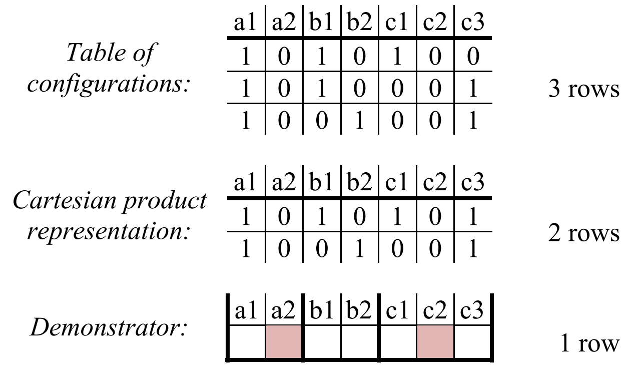


Table of configurations: 3 rows

a1	a2	b1	b2	c1	c2	c3
1	0	1	0	1	0	0
1	0	1	0	0	0	1
1	0	0	1	0	0	1

Cartesian product representation: 2 rows

a1	a2	b1	b2	c1	c2	c3
1	0	1	0	1	0	1
1	0	0	1	0	0	1

Demonstrator: 1 row

a1	a2	b1	b2	c1	c2	c3
	🟥				🟥	

Figure 7 Comparison of visualization methods for showing configurations.

In this example it is “a2” and “c2”. The demonstrator is not capable of showing which feature variants are conditionally forbidden; e.g. “b2”, is forbidden if “c1” is selected.

The conclusion of the literature review is that there are inter/intra-domain matrices capable of visualizing the configuration rules. There are also visualization methods for reducing the number or row, e.g. the Cartesian product representation. There is however little research on how the visualization method is supporting the user during the validation of configuration rules. The next two sections discuss the design considerations and typical rule queries which describe the users’ needs during the validation of configuration rules.

Design considerations

Matrix-based visualization tools using only one user interface have to have restrictive size limitations to fit approximately within a computer screen. Microsoft Excel displays about 50 rows and 30 columns if opening a worksheet with the standard but high screen resolution 1920x1200 (100% zoom). The rows required for representing the vehicle configurations potentially grows exponentially with the number of feature families. With at least 2 feature variants from every feature family the theoretical number of configurations is $2^{\text{number of feature families}}$. The configurations for 10-20 feature families used in one single configuration rule require approximately 30,000 rows. The demonstrator uses a simplification adding columns equal to the number of feature variants visualized. As one specialist in visualization studies stated: *an approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem*, or *graphical excellence is that which gives to the viewer the greatest number of ideas in the shortest time with the least ink in the smallest space* [20]. This design decision will later be tested and evaluated in the next section.

In [8] the stated conclusion is that the vehicle configuration rules can be just barely managed if including more than three feature variants. During the user study for this paper, a design engineer demonstrated how the inspection of allowed configurations takes place. The demonstrated checks were simple in terms of how many feature variants were involved. This is an important design consideration when studying the validation of configuration rules. The pink fills showing forbidden feature variants in Fig. 7 are one example of how to keep the involved number of feature variants low. The feature variant “a2” is always forbidden, no matter what involvement of feature variants “c1”-“c3”.

The demonstrator will be based on the item usage rule matrix for several reasons. The main reason for this starting-point is the process when configuration rules are modified and updated from a technical viewpoint. It is the design engineers that know what is technically feasible to build, and should therefore have a visualization method developed from their most common user interface.

As the aim is to visualize interesting and actionable configuration rules, the selection of feature families and item usage rules visualized has to be user-specified. Some design engineers know exactly which feature families to visualize, while others rely on support from others. The tests that developed and evaluated the demonstrator were based on the actionable configuration rules for one of the design engineers participating in the user study.

Typical rule queries

The rule queries are used during the tests as a framework for defining the test tasks. The purpose of the typical rule queries is to study if the users may answer them by using the demonstrator. The demonstrator aims at visualizing interesting and actionable configuration rules, which has failed if the users are not able to understand the user interface. Correct answers for the typical rules queries are a prerequisite for the success in facilitating the validation of configuration rules.

There are standards for describing the information model for vehicle configuration rules, e.g. AP214 [21]. There is, however, no standard describing typical rule queries asked during the validation of vehicle configuration rules [22]. Four typical rule queries were instead found during the user study and from the literature:

Rule query A: Is ITEMXX assigned to configurations including feature variant YY?
Missing or faulty configuration rules are discovered when the items are assigned to configurations.

Rule query B: Is it possible to take away/add feature variants in an item usage rule without modifying the item assignment to configurations?
This question checks whether an item usage rule may be shortened (reduced number of feature variants)

without modifying to which configurations the item is assigned.

Rule query C: Are there any configurations with more than one item assigned from a user-specified selection of items?
If the actionable configuration rules represent e.g. the “steering wheels”, this rule query asks whether there are any configurations with more than one steering wheel. It is possible to test any user-specified set of items to see if they are mutually exclusive. In [5] this rule query is defined as to ask if there are “ambiguities in the item list”.

Rule query D: Are there configurations without any item assigned from a user-specified selection of items?
The last typical rule query is similar to previous rule query in the sense that it is about a user-specified set of items. This rule query, however, asks for vehicle configurations without any items instead of duplicates. In [23] this rule query is defined as to ask if the set of items is “exhaustive”.

To answer these typical rule queries there was a demonstrator developed following the design considerations described in the next section.

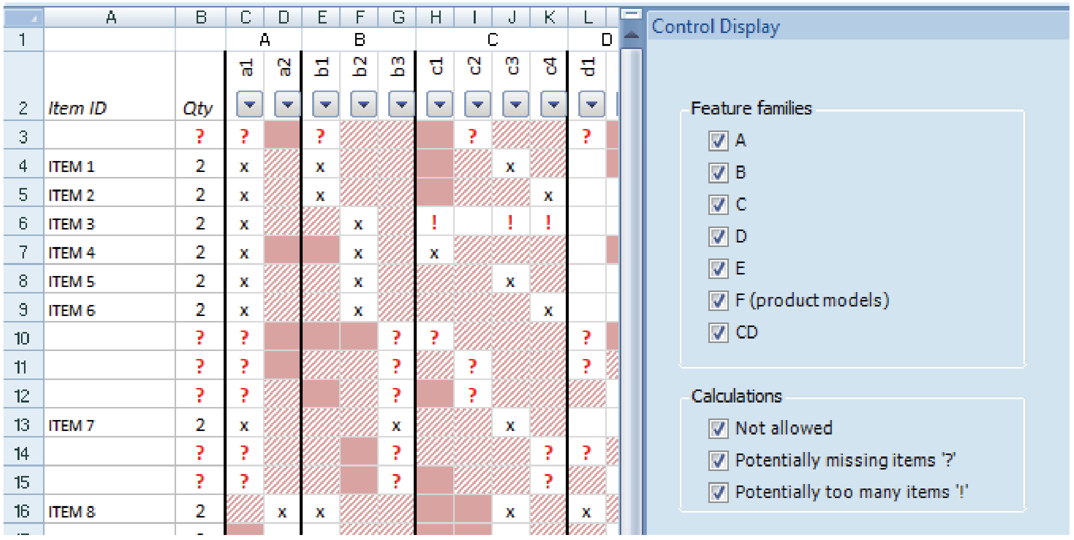
3.2. Demonstrator description

The user study described in [9] showed that it is the industrial visualization tools’ user interfaces showing item usage rules which is the most commonly used user interface for design engineers. The aim of the visualization method implemented in the demonstrator is to display all classes of configuration rules at the same time. The suggested visualization method is shown in Fig. 8. By using the classifications for intra- and inter-domain matrices [14], this is an inter-domain matrix with two types of values (either crosses or colours). The crosses are from the binary item usage rule matrix [16], and the colours are computed values derived from the product authorizations as well as feature variant combination rules.

The feature variants are organized in feature families. In the figure, the feature families are separated with thicker black lines. Everything shown in black is imported from the item usage rule matrix. The “x” symbol signifies that the row’s item usage rule includes the column’s feature variant, fully according to previous explanations of the currently used item usage rule matrix. The set of items in this example (“ITEM001”-“ITEM007”) has been selected based on a design engineer’s specified request of items. In this case, the items have variations in thickness and length which are defined by feature families “B” and “C”.

The configurations using “?” symbol(s) signify that there are allowed configurations without any assigned item from the user-specified item set. The first row in Fig. 8 contains red question marks and should be read (“a1” & “b1” & “c2” & “d1”).

The row for “ITEM003” in Fig. 8 should be read as IF(“a1” & “b2”) THEN(“ITEM003”). The consequence of the following three red exclamation marks “!” for “c1”, “c3” and “c4” is that at least one item more than “ITEM003” is assigned to those configurations. At the current state of development it is up to the user to find these other items. From



	A	B	C	D	E	F	G	H	I	J	K	L
1			A		B			C				D
2	*Item ID*	*Qty*	a1	a2	b1	b2	b3	c1	c2	c3	c4	d1
3		?	?	🟥	?	▨	▨	🟥	?	▨	▨	?
4	ITEM 1	2	x	▨	x	▨	▨	🟥	▨	x	▨	
5	ITEM 2	2	x	▨	x	▨	▨	🟥	▨	▨	x	
6	ITEM 3	2	x	▨	▨	x	▨	!		!	!	
7	ITEM 4	2	x	🟥	🟥	x	▨	x	▨	▨	▨	
8	ITEM 5	2	x	▨	▨	x	▨	▨	▨	x	▨	
9	ITEM 6	2	x	▨	▨	x	▨	▨	▨	▨	x	
10		?	?	🟥	🟥	🟥	?	?	▨	▨	▨	?
11		?	?	🟥	▨	▨	?	▨	?	▨	▨	?
12		?	?	▨	🟥	▨	?	🟥	?	▨	▨	▨
13	ITEM 7	2	x	▨	▨	▨	x	▨	▨	x	▨	
14		?	?	▨	▨	🟥	?	▨	▨	▨	?	?
15		?	?	▨	▨	🟥	?	🟥	▨	▨	?	▨
16	ITEM 8	2	▨	x	x	▨	▨	🟥	🟥	x	▨	x

Control Display

Feature families
- ☑ A
- ☑ B
- ☑ C
- ☑ D
- ☑ E
- ☑ F (product models)
- ☑ CD

Calculations
- ☑ Not allowed
- ☑ Potentially missing items '?'
- ☑ Potentially too many items '!'

Figure 8 Demonstrator’s user interface.

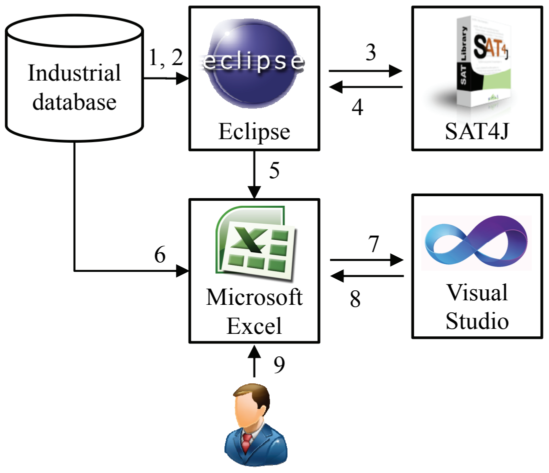
manual inspection it can be realized that it is “ITEM004”-“ITEM006” that are assigned on vehicle configurations together with “ITEM003”.

The pink fill signifies that the row’s item usage rule is not allowed to include the column’s feature variant according to the feature variant combination rules and/or the product model authorization rules. The pink fills have been calculated, and may be a result from more than one rule. As an example, the feature variant “c1” has a pink fill caused by the configuration “a1” and “b1” for “ITEM001”.

The cells with the pink diagonal stripes signify that the row’s item usage rule is not allowed according to mutually exclusive feature families to include the column’s feature variant. Every row with a black cross or a red question mark in a feature family will have the pink striped fill for all other feature variants from that feature family. The first row in Fig. 8 has a red question mark in “c2”, which gives fields with pink diagonal stripes for “c1”, “c3” and “c4”. In this case, however, there are feature variant combination rules and/or product model authorization rules restricting “c1” which gives the solid pink fill.

To summarize, the demonstrator is displaying item usage rules exported from the company’s configuration rules database, while the other classes of configuration rules are processed to generate the pink fills, “?” and “!” in the integrated configuration matrix. Additional rows with question marks are added if there are allowed vehicle configurations with no assigned items from the analyzed set of items. The system architecture used for the demonstrator is described in Fig. 9. The exports from the industrial database are transferred by text-files to the configuration software written in java. The project within eclipse is using an imported library as a configuration engine (SAT4J). A further description may be found in [24]. The output of the configuration software is shown in excel where the interaction with the user is controlled by a program written in C#. It is possible to hide all the added functionality of the demonstrator and return to the current user interface used at the automotive company.

The demonstrator’s capabilities will now be further described by showing how it can help to answer the previously mentioned typical rule queries.



1. Configuration rules for a product family (.txt)
2. Selection of feature families and item usage rules (.txt)
3. Configuration question (function inside java program)
4. Configuration answer (yes/no)
5. Table of allowed feature variant combinations (.txt)
6. Item usage rules (.txt)
7. Call for Excel Add-In control (function inside C# program)
8. Result of Excel Add-In control (.xlsx)
9. Mouse click on Excel Add-In controls

Figure 9 System architecture for the demonstrator.

Answering rule query A: Is ITEMXX assigned to configurations including feature variant YY?
The demonstrator shows forbidden feature variants with pink fills, e.g. “d2” is forbidden for “ITEM001”. It would be possible to add more rows and show a table of allowed configurations instead. However, the risk is that the matrix becomes inefficient to use because of its size.

Answering rule query B: Is it possible to take away/add feature variants in an item usage rule without modifying the item assignment to configurations?
Using Fig. 8, it should be realized that for example the item usage rule for “ITEM001” may be more precise by including “d1” and/or “e3”. These feature variants are the only allowed options from their corresponding feature families. Both “e1” and “e2” are filled with pink colour for “ITEM001” which means that these feature variants are not allowed to be used according to the feature variant combination rules and/or product model authorization rules. In [9] it is concluded that in general it is easier to interpret the meaning of longer item usage rules since they are more precise. Also, the more precise item usage rules may limit the consequences of potentially faulty feature variant combination rules (but it is not possible to allow or restrict feature variant combinations with the item usage rules as many

design engineers believe). It was also found that it is in general more time-efficient to do logically reasoning with shorter rules. The pros and cons of how to author item usage rules result in authoring variations. The demonstrator reduces the need of discussing authoring variations as the visualization shows were the authoring variations occur.

Then there are the errors of either having too many or too few items on an allowed vehicle configuration. This is frequently verified at the automotive companies by testing a high number of real or simulated vehicle configurations.

Answering rule query C: Are there any configurations with more than one item assigned from a user-specified selection of items?
The exclamation marks show when more than one item from a user-specified item set is assigned for some configuration, e.g. “ITEM003” is assigned as well as one item of “ITEM004”, “ITEM005” or “ITEM006”.

Answering rule query D: Are there configurations without any item assigned from a user-specified selection of items?
The demonstrator uses question marks for this rule query. Question marks show configurations without items from the user-specified selection of items.

The next section will describe the users’ reactions to the usability test.

3.3. Formative usability tests

This section will start by describing the test design, which is followed by the quantifiable test results. Finally, the evaluation and development of the demonstrator is further described by the critical incidents and the post-session interviews.

Test design

Vehicle configuration rules from the three studied automotive companies are shown in Fig. 10. All three companies share the characteristics that the item usage rules are not displayed in the same user interface as the other classes of configuration rules. Company B is special in the sense that it has one single user interface showing longer but fewer logic expressions combining product model authorizations, inclusions and exclusions. Still, the item usage rules reside in another visualization tool. The demonstrator is based on the data from Company A, which has the highest number of configuration rules (~500 000) as well as the most item usage rules per item (10:1).

	Company	**A**	**B**	**C**
	Product family	1	1	1
	Product models	20	6	3
	Feature families	500	200	500
	Feature variants	3 000	800	4 000
Configuration rules	Product model authorizations	20 000	1 000*	4 000
Configuration rules	Inclusions (IF-THEN, AND)	300		8 000
Configuration rules	Exclusions (NOT, AND)	30 000		1 000
Configuration rules	Item usage rules (IF-THEN, AND)	400 000	9 000*	20 000
Configuration rules	Length limitations	10	100	5
	Items	40 000	7 000	20 000
Analysis	Item usage rules/Items	10:1	1:1	1:1
Analysis	Configuration rules excl. item usage rules /Items	1:1	1:7	1:1

*) IF-THEN, AND, NOT and OR are used.

Figure 10 The configuration rules for one product family from three automotive companies.

The test case used all configuration rules for calculations but the user interface showed only 15 item usage rules and 6 feature families. The selection of configuration rules was defined as actionable for a design engineer participating in the user study. The user interface has a control display with check boxes where it is possible to hide all the demonstrator’s visual elements that have been added to the item usage rule matrix. In that sense, the demonstrator is an extension to industrial visualization tools.

The users were told to freely use the demonstrator and to answer 14 tasks which were based on the four typical user queries.

Quantifiable test results

The quantifiable test results were achieved for number of errors, learnability and time efficiency. The number or errors were defined as the number of faulty answers handed-in at the end of the test session on the task sheet. Few faulty answers were initially filled-in on the task sheet, but they were all corrected during the test by the users.

Repetition of a task is likely to make the users more confident and eventually result in a more time-efficient operation. Since the tasks outnumbered the typical rule queries it was possible to study the learning curves [25] for each typical rule query. Both the learnability and the time efficiency were possible to measure, but with subjectivity as the time was spent on both critical incidents and fulfilling the tasks. It was especially difficult to measure the response times for the tasks which differed greatly in difficulty as more critical incidents occurred. The tasks for typical rule query A (Is ITEMXX assigned to configurations including feature variant YY?) were similar in difficulty and were therefore chosen for studying the response times, see Fig. 11. Four users conducted the tasks, but unfortunately one of the test recordings was lost due to IT issues.

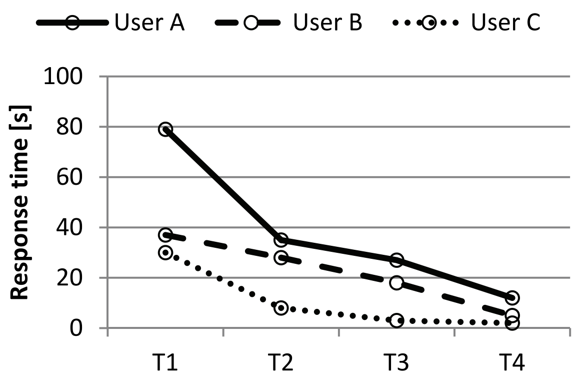


Figure 11 Response times for tasks T1-T4 from typical rule query A.

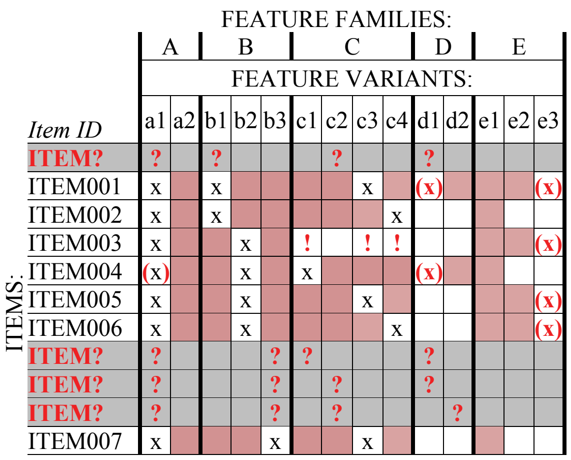
The learnability is the time spent on answering the typical rule query for the first time, i.e. the response times for the task T1. The users' response times varied between 30-80 seconds. The time spent on typing the rule query in the industrial visualization tool is comparable to this response time. Then, with the current visualization tool, there is no automation of the calculations required for fulfilling the tasks.

The time efficiency is measured at the last repetition of the typical rule query, i.e. the task T4. The response times for this task are in the order of a few seconds for all users.

The next section describes both positive and negative critical incidents occurring during the test.

Critical incidents

The usability tests resulted in identification of numerous critical incidents. A critical incident is something that happens while the user is fulfilling the tasks and has a significant positive or negative impact on performance or user satisfaction [11]. The occurrence of critical incidents increased the understanding of the conceptual critique and which elements of the demonstrator should be modified or taken away since they caused confusion. To be able to discuss which types of modifications that have taken place, the initial version of the demonstrator is shown in Fig. 12. Many of the modifications were done to obtain a higher consistency. This follows the golden rules of user interfaces described in [26]. An excerpt of the list of critical incidents:



		FEATURE FAMILIES:													
		A		B			C				D		E		
		FEATURE VARIANTS:													
	Item ID	a1	a2	b1	b2	b3	c1	c2	c3	c4	d1	d2	e1	e2	e3
ITEMS:	ITEM?	?		?				?			?				
	ITEM001	x		x					x		(x)				(x)
	ITEM002	x		x						x					
	ITEM003	x			x		!		!	!					(x)
	ITEM004	(x)			x		x				(x)				
	ITEM005	x			x				x						(x)
	ITEM006	x			x					x					(x)
	ITEM?	?				?	?				?				
	ITEM?	?				?		?			?				
	ITEM?	?				?		?				?			
	ITEM007	x				x			x						

Figure 12 The initial version of demonstrator before the usability tests were conducted.

- *(Positive/Learnability):* Positive critical incidents during the usability tests were the users' spontaneous statements when first exposed to the demonstrator's user interface. The pink fills were found to be self-explanatory and the users showed appreciations with words such as "powerful".
- *(Negative/Learnability):* One example of a higher consistency is the removal of grey fills that indicate configurations without any items from the user-specified selection of items. These grey fills were replaced with pink or white fills according to the same logic as for the item usage rules.
- *(Negative/Learnability):* In the initial version of the demonstrator, as soon as there was a black cross in a feature family, the other feature variants from that feature family were filled with pink. Then it was not possible to judge whether the pink fills were caused by the black cross or other restricting configuration rules.
- *(Negative/Learnability):* The red crosses and parenthesis are also taken away since the introduction of the striped pink fill made the symbols redundant.
- *(Negative/Learnability):* The fields with pink diagonal stripes brought more confusion than clarity before the test participants were supplied with the explanation of the feature. The test participants struggled when guessing what the

fields with pink diagonal stripes describe. However, when the explanation was supplied there were no problems with using them for the benchmark tasks. Positive statements such as "was it that simple" were made. An earlier version of the integrated configuration matrix was tested to use pink fills exclusively but was then criticized for not showing the distinction. To make the pink diagonal stripes self-descriptive requires further development.

- *(Negative/Efficiency):* One of the product structure specialists found it disturbing that it is not possible to find the explicit formulation of feature variant combination rules from the demonstrator's user interface. Finding the formulation of single feature variant combination rules is difficult and in most cases impossible. This was neither the aim of the demonstrator. The product structure specialist instantly answered correctly on the typical rule queries, but spent a considerably long time on verifying his answers trying to use his old methods. The user acted in the same way throughout the test.

Most of the critical incidents for the final version of the demonstrator concerned the learnability. More iteration has to be done to potentially remove them. The critical incident for the efficiency is more severe, however, since this has a negative impact of the product structure specialist's efficiency even after the first use of the demonstrator. The user suggested introducing a "debugger" explaining the pink fills as an idea for removing this critical incident. The debugger was also mentioned as an important feature for not forgetting how to formulate feature variant combination rules.

More user statements about the demonstrator were found from the interviews in the next section.

Post-session interviews

The results of the post-session interviews gave predictions about the usability of the demonstrator. The answers to the post-session questions were mainly positive, and is here described following the usability factors errors, efficiency and learnability:

Errors: The users predicted that the greatest value of the demonstrator was the increased confidence of the users on finding the correct answers to the typical rule queries. All of the typical rule queries that were tested with the demonstrator are possible to answer today by using the industrial visualization tools. However, the demonstrator automates some of the calculations and therefore reduces the risk of finding wrong answers to the typical rule queries.

Learnability: The demonstrator opens up some of the typical rule queries that only skilled users of configuration rule analysis were able to answer. All activities during the development of the configuration rules benefit from an increased understanding of the configuration rules.

Efficiency: Also, a major positive predicted outcome is the increased time efficiency. This is due to the review process of the configuration rules, which often starts with a request from the design engineers. Getting it right the first time and avoiding iteration loops in the review process is predicted to be the main source of increased time efficiency. By having the item usage rules as a starting point, it is considered to enable the design engineers to request valid configuration rules without iteration loops with the product structure specialists. The demonstrator's user interface, however, clearly came as a surprise to one of the more experienced (>10 years) product structure specialists, who found it "unnatural". The industrial visualization tool based on separate user interfaces for different classes of configuration rules is deeply rooted among the people who have been using the same methods for many years.

The identified risk of misuse that was mentioned during the post-session interviews resulted in modifications prior the final version of the demonstrator. The misuse concerned the re-formulation of item usage rules. The initial version of the demonstrator showed red crosses for every feature family where all feature variants except one were restricted. In Fig. 13, the item usage rule for "ITEM001" has pink fills for "e1" and "e2", but not "e3" where a red cross "**(x)**" is placed instead. The red crosses perhaps would have been misinterpreted as a signal to the user to add black crosses everywhere the red crosses show up. This is in direct conflict with the wish to keep the number of feature variants within item usage rules as low as possible. The misuse would continue until all feature families with only one allowed feature variant contained a black cross. This drawback, however, was not mentioned in the later tests due to changes in the demonstrator's user interface. The red crosses were easily taken away, as they were only redundant information to the pink fills.

4. DISCUSSION

The discussion will start with answering the research questions, and then discuss the logic used in the demonstrator followed by a generalizability discussion.

Answering RQ1: What are the strengths and weaknesses of industrial visualization tools used when validating vehicle configuration rules?
The common major weakness of the industrial visualization tools is that they force its users to shift between several user interfaces causing a higher cognitive load than necessary. Also, there is a potential of using a higher degree of automation.

The most common industrial visualization method for configuration rules is the list, but exceptions may be found – e.g. matrices and trees. The strength of matrices, compared to the table or list, is their clarity when comparing several configuration rules. The feature variants are used as column headings, which makes it possible to use symbols for enhancing the clarity.

In this paper, only lists and matrices are discussed when presenting the industrially used visualization methods. However, in early stages of development projects, it has been found that other visualization methods exist. At two of the three studied automotive companies, there are manually created tree visualizations, e.g. for judging at an early stage whether the number of items to be developed is adequate. Whether these planning documents can be used for also validating the configuration rules is yet to be investigated, but it is a feasible task to transform the data presented in Fig. 4 to a tree structure. Trees have not been further studied for evaluation.

RQ2: Which visualization tool addresses those weaknesses?
The demonstrator has one single user interface for visualizing all classes of configuration rules at the same time. The typical user queries identified are possible to answer by using only the demonstrator. The user interface is based on an extension of the item usage rule matrix. From a logical perspective, the demonstrator's major drawback is that it does not visualize all configurations for every item usage rule, as e.g. the Cartesian Product Representation does. Because of this simplification, the demonstrator is capable of using one single row for displaying the configurations.

The demonstrator has been proven to automate some steps for the users when answering typical rule queries. The predicted outcome of the demonstrator is fewer errors, increased time efficiency and higher number of users capable of answering the typical rules queries. This is due to the higher degree of automation, in combination with the extended user interface that now includes all classes of configuration rules.

4.1. Generalizability

The demonstrator has been tested on data from one single automotive company. When conducting some trial tests at several companies, it was clear that most time was spent on clarifying small differentiating elements between the automotive companies' information models. The most strategic test setup would have been to use three different tests, so that each automotive company would be able to conduct tests on its own configuration rules. However, the test environment was only using one company's data and therefore all tests were conducted by employees from that company. To the study's defense, the main weakness of the industrial visualization tools is the required shifts between several user interfaces, which is a weakness found in a user study involving all three studied automotive companies [1].

5. CONCLUSIONS AND FUTURE WORK

The analysis of industrial visualization tools and the related user study have shown that there is a potential for facilitating the validation of configuration rules. Usability tests of a demonstrator have shown that the identified weaknesses are possible to address. The outcome of the usability test was successful:

- *Errors:* All users fulfilled the tasks correctly. A decrease in errors is predicted during the post-session interviews due to automation and an improved understanding of the different configuration rule classes.
- *Learnability:* Negative critical incidents occurring could mostly be eliminated through development of the demonstrator.
- *Efficiency:* The measurement of response times showed that test participants who had gained some experience with the demonstrator performed the benchmark tasks in the order of seconds. This is an improved time efficiency according to the post-session interviews.

The conclusion is therefore that the demonstrator shows a visualization tool that will result in a more time-efficient and less error-prone validation of configuration rules.

Our plans for future work include further development of the demonstrator in order to conduct more comprehensive usability tests. Especially, the industrial data from the two remaining automotive companies will be visualized and tested.

Additionally, the demonstrator shows only one snapshot in time of the configuration rules, and is not yet capable of showing how the configuration rules develop over time. This capability will be addressed in future work. Substantial effort has to be made in describing how the time effectivity is managed for vehicle configuration rules. More functionality that may be added is the requested debugger and aiding elements such as mouse popup menus.

One major difference between the industrial databases is that the company A uses mainly exclusions and company C uses mainly inclusions, see Fig. 8. Also, the length limitations vary greatly between the studied companies (100 compared to 5). Another difference is that the number of item usage rules per item varies (10 compared to 1). If those factors have an impact of the demonstrator's success is also left as future work.

ACKNOWLEDGEMENTS

This work was carried out at the Wingquist Laboratory VINN Excellence Centre for Efficient Product Realization at Chalmers University of Technology, supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA). The support is gratefully acknowledged.

REFERENCES

[1] Tidstam, A. & Malmqvist, J. (2010), "Information Modelling in Automotive Configuration", Proceedings of NordDesign, Vol. 2, Gothenburg, pp. 275-288.

[2] O'Keefe, R. M. (1993), "Expert system verification and validation: a survey tutorial", in Artificial Intelligence Review, Vol. 7-1, pp 3-42.

[3] Haag, A. (1998), "Sales configuration in business processes", in IEEE Intelligent Systems, Vol. 13-4, pp. 78-85.

[4] Meseguer, P. & Preece, A. D. (1995), "Verification and validation of knowledge-based systems with formal specifications", in Knowledge Engineering Review, Vol. 10-4, pp 331-343.

[5] Sinz, C., Kaiser, A. & Küchlin, W. (2003), "Formal methods for the validation of automotive product configuration data", in Artificial Intelligence for Engineering Design, Analysis and Manufacturing, Vol. 17-1, pp.75-97.

[6] Agrawal, R. & Srikant, R. (1994), "Fast algorithms for mining association rules", Proceedings of International Conference Very Large Data Bases, Vol. 1215, pp. 487-499.

[7] Kaidi, Z., Liu, B., Tirpak, T. M. & Xiao, W. (2005), "A visual data mining framework for convenient identification of useful knowledge", Proceedings of 5th International Conference on Data Mining, Houston, pp.8-16.

[8] Hami-Nobari, S. & Blessing, L. (2005), "Effect-oriented Description of Variant rich products", Proceedings of International Conference on Engineering Design, Melbourne, pp. 219-220.

[9] Tidstam, A. & Malmqvist, J. (2011), "Authoring and verifying configuration rules", Proceedings of International Conference on Product Lifecycle Management, Eindhoven.

[10] Pak, C. (2011), "Hantering av dataflöde i produktutveckling" [In English: Management of information flow in product development], Master Thesis, Royal University of Technology, Stockholm, Sweden.

[11] Nielsen, J. (1993), "Usability engineering", Academic Press, San Diego.

[12] Hix, D. & Hartson R. H. (1992), "Formative Evaluation: Ensuring Usability in User Interfaces", Technical Report, Virginia Polytechnique Institute & State University Blacksburg.

[13] Lengler, R. & Eppler, M. (2007), "Towards A Periodic Table of Visualization Methods for Management", Proceedings of the Conference on Graphics and Visualization in Engineering, Clearwater.

[14] Steward, D. (1981), "The Design Structure System: A Method for Managing the Design of Complex Systems", in IEEE Transaction on Engineering Management, Vol. 28-3, pp. 79-83.

[15] Bongulielmi, L., Henseler, P., Puls, C. & Meier, M. (2001), "The K- and V-Matrix Method – An Approach in Analysis and Description of Variant Products", Proceedings of the 13th International Conference on Engineering Design, Glasgow, Vol. 28, pp. 571-578.

[16] Lindemann, U., Maurer, M. & Braun, T. (2009), "Structural Complexity Management – An Approach for the Field of Product Design", Springer-Verlag Berlin.

[17] Malmqvist, J. (2002), “A Classification of Matrix-Based Methods for Product-Modeling”, Proceedings of International Design Conference, Dubrovnik, pp. 203-210.

[18] Maurer, M. (2007), “Structural Awareness in Complex Product Design”, PhD Thesis, Department of Product Development, Technical University of Munich.

[19] Madsen, J. N. (2003), “Methods for Interactive Constraint Satisfaction”, Master Thesis, Department of Computer Science, Technical University of Denmark.

[20] Tukey, J. W. (1977), “Exploratory data analysis”, Addison-Wesley, Reading Massachusetts.

[21] ISO (2004), “ISO 10303”, International Organization for Standardization, http://www.iso.org, Visited 2011-03-04.

[22] Viel, C. (2003), “Management of product’s diversity: industrial validation of AP214 reference model”, in International Journal of Computer Applications in Technology, Vol. 18-(1-4).

[23] Astesana, J. M., Cosserat, L. & Fargier, H. (2010) “Constraint-based modeling and exploitation of a vehicle range at Renault's: requirement analysis and complexity study”, ECAI 2010 Configuration workshop, pp.33-39.

[24] Voronov, A., Åkesson, K. & Ekstedt, F. (2011), “Enumeration of valid partial configurations”, Proceedings of International Joint Conferences on Artificial Intelligence, Barcelona, pp. 25-31.

[25] Wright, T. P. (1936), “Learning curve”, in Journal of the Aeronautical Sciences, Vol. 3-2, pp. 122-128.

[26] Fleischanderl, G. (2000), “User Interface Requirements for Knowledge Acquisition and Modeling”, Proceedings of European Conference on Artificial Intelligence, Berlin, pp. 41-43.