Supporting Product Development through Cross-Discipline Dependency-Modeling – Novel Approaches for Traceability-Usage

Grischa Beier, Asmus Figge, Robert Müller, Uwe Rothenburg, and Rainer Stark
Fraunhofer IPK/Virtual Product Creation, Berlin, Germany
Email: {grischa.beier, asmus.figge, robert.mueller, uwe.rothenburg, rainer.stark}@ipk.fraunhofer.de

Rainer Stark
Technische Universität Berlin/Industrial Information Technology, Berlin, Germany
Email: rainer.stark@tu-berlin.de

***Abstract*—Driven by markets' competitiveness, new products have to permanently integrate innovations and higher functionality. Products are getting more complex, borders of subsystems become increasingly blurred, and unexpected interdependencies between subsystems emerge. Hence developers have difficulties maintaining their necessary understanding of the system. An approach helping to handle the interdependencies of complex systems is traceability, where dependencies being implicitly known to some developers are modeled explicitly.**

This paper presents the key findings of two surveys addressing traceability between system models conducted among German companies. The addressed research topic is the investigation of opportunities for supporting product development by making use of traceability. Novel methods for traceability usage are elaborated and evaluated by a prototypical proof of concept.

***Index Terms*— dependency modeling, traceability usage, house of quality, progress monitoring**

I. Introduction and Motivation

Driven by markets' competitiveness, new products have to permanently employ innovations and integrate higher functionality. Products are getting more complex from a systems interaction perspective, borders of subsystems become increasingly blurred, and unexpected interdependencies between subsystems emerge. For that reason developers have difficulties to keep up their necessary understanding of the entire system. Often systems knowledge is not available to engineers although it may exist within the company.

Following classical methodologies of product modeling (such as VDI 2206) several process steps have to be taken to successfully develop a product, in each one of which digital data artifacts[1] are created. Those artifacts are created in different specialized tools, resulting in isolated descriptions of the product. One approach to bridge this isolation of artifacts is traceability.

Traceability methods originate from requirements management in the development of software for safety critical systems. Mechatronic product development can profit from these approaches, by extending and adapting them accordingly. While some methods for Systems Engineering are available today (see Section II), they are not practiced comprehensively. This is primarily due to two reasons: high effort necessary to establish traceability and its unclear benefits [1], [2] (see also Section III).

Coping with these challenges was one of the major topics within the collaborative research project ISYPROM[2]. During this project methods were developed for an efficient modeling of tracelinks [3], [4]. However, this paper focuses approaches to support development activities by utilizing tracelinks. The addressed research topic is the investigation of opportunities for supporting product development by making use of traceability between different digital. Novel methods for traceability usage[3] are elaborated and evaluated by a prototypical proof of concept. Furthermore initial feedback from industry workshops is given. These detailed approaches will be described in Section V.

At first traceability theory is briefly introduced and related work is presented and discussed in Section II. In Section III the key findings of two surveys are presented, in which Fraunhofer IPK has interviewed several industrial enterprises on the topic of traceability. The paper finishes with evaluation information regarding the presented approaches, concluding remarks and offers an outlook on future research activities (see Section VI)

Manuscript received December 6, 2012; revised February 9, 2013

[1] In this paper the term artifact is used to describe any kind of digital result of a process step (e.g. requirements specifications, functional models, product structure or test cases). Each artifact consists of elements. All dependencies between these elements that are explicitly modeled are called tracelinks.

[2] The research and development project ISYPROM was funded by the German Federal Ministry of Education and Research (BMBF) within the Framework Concept "Research for Tomorrow's Production" (funding number 02PC105x) and managed by the Project Management Agency Karlsruhe (PTKA).

[3] Pinheiro has introduced the term "trace extraction" for the target-oriented usage of existing tracelinks. Since this section primarily addresses their benefit for Systems Engineering processes, the authors prefer the term "traceability usage" in this context [5].

©2013 Engineering and Technology Publishing

doi: 10.12720/lnit.1.1.21-28

II. TRACEABILITY- INTRODUCTION: STATE OF THE ART

Traceability was introduced in the late 1970s in software development, more specifically in the area of requirements engineering. It addresses the management of interdependencies between software requirements and other artifacts in software engineering. Gotel and Finkelstein define traceability as follows: "Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases)" [6]. This approach was later adapted to the needs of model based software development. In this context Paige et al. define traceability as "[…] the ability to chronologically interrelate uniquely identifiable entities in a way that matters […] [7]. In the context of Model Driven Engineering (MDE)[2], many of the artifacts of interest are models, conforming to a metamodel, and are constructed using a set of modeling tools. Traceability in MDE is therefore predominantly concerned with chronologically defined relationships involving models and elements of models" (see red arrows in Fig. 1 representing the concept of tracelinks). Traceability is regarded as a measure of system quality and process maturity and is mandated by many standards such as MIL-STD-498, IEEE/EIA 12207, IEEE Std. 1219, ISO 9000ff, ISO 15504, ISO/IEC 12207 or the new ISO 26262 [8].

Both aforementioned application areas of traceability have something in common with Systems Engineering approaches: engineers create several artifacts describing the final product from different viewpoints, in a different degree of detail and maturity [9].

That is why there is a current trend to transfer these software engineering methods to mechatronic system development. Today, there are several tools, which provide functionalities for tracelink recording, and usage.

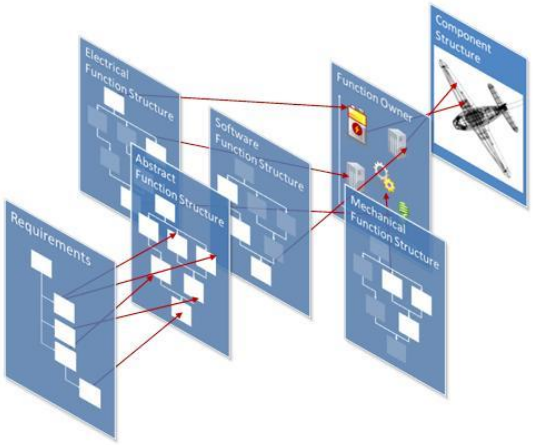

Figure 1. Dependencies between elements of different artifacts can be modeled with tracelinks.

A selection of these tools will be introduced in the following paragraphs.

Rational DOORS by IBM is a tool used for requirements management. In DOORS requirements are structured in hierarchies, managed as objects and stored in a database. This allows for their individual versioning as well as referencing between them [10]. Another approach to requirements management is followed by Reqtify (Dassault Systèmes): tracelinks are stored in several original requirements specifications. Reqtify is then used to visualize tracelinks, perform analyses and to generate reports [11].

Loomeo by TESEON GmbH is a tool for the analysis of system structures. For this purpose Loomeo uses dependency matrices. Dependencies between system elements are documented in these matrices. There are a number of analyzing techniques allowing for the identification of interrelated structures by rearranging matrices or for tracing of impact chains [12], [13]. The Cambridge Advanced Modeller (CAM) provides features for modeling and analyzing dependencies in certain types of systems. It offers special toolboxes to model products (based on a matrix approach) or processes, synthesize a product architecture or to estimate and visualize the propagation of changes [14], [15]. METUS provides functionalities to specifically model functions and components of a product and their dependencies. These tracelinks are used by several analysis functions in order to e.g. optimize costs of the system [16],[17].

Many PLM systems provide basic functionalities for the establishment of traceability, too. For example, in Teamcenter Systems Engineering and Requirements Managements by Siemens PLM, links can be modeled between requirements and all other items in the database [18]. Dassault Systèmes V6 RFLP-approach allows for creating many artifacts (requirements, functions, logics and physical components), which can be traced in an integrated environment [19].

Other approaches utilize traceability information for integration purposes. One example is ToolNet, developed in a research project by DaimlerChrysler and EADS, which acquires information from various existing sources. Tracelinks are then established between the different model elements as references and stored in a central database [20], [21]. Atego Workbench by Atego is a collaborative engineering framework which also allows for acquiring data from existing tools (e.g. requirements specifications or use case diagrams), providing functions for creating tracelinks between the elements as well as functions to analyze them, e.g. impact analysis [22].

III. SURVEYS ON INDUSTRIAL TRACEABILITY PRACTICE

In order to learn more about the industrial acceptance and application of traceability approaches, Fraunhofer IPK conducted two surveys: one in 2010 and another one in 2011. Both surveys were analyzed by cumulating and structuring the answers given by interviewees in order to identify underlying trends. However it should be noted that due to the limited number of participants of their specialized knowledge within their respective company, the presented findings cannot be entitled to completeness.

[2] We refer to MDE as Model-based Software Engineering.

©2013 Engineering and Technology Publishing

Nevertheless, the identification of representative tendencies seems very likely, as all interviewees were explicitly asked to consult other experts when uncertain. This was to ensure a representative view of the entire company. In the following paragraphs, the key findings from both surveys are presented separately. Subsequently, common conclusions, which were drawn from their results, are summarized.

The participants of the first survey were representatives of six German automotive companies. The scope of the survey involved four topics from the early phases of product development, one of which being traceability. The method of the survey consisted of a two-step process. First, every contact person received a questionnaire with five questions regarding traceability. In an in-depth telephone interview the answers and other remaining questions were discussed.

This survey revealed that traceability is a well known but rarely practiced topic in the automotive industry. Its application ranged from "not used at all" (one company) over "hardly used" (two companies) to "little use" (three companies). Even when traceability is applied, its implementation is rather selective and not continuous throughout the development process. The used level of abstraction is mainly system or component layer (both mentioned by three companies). Only one company traces more detailed layers such as parameters. In order to document dependencies, the applied methods are matrices, references and naming conventions.

The second survey was conducted with developers from four companies from different industry sectors plus four software vendors. The interviews covered six open questions regarding aspects of traceability.

Of the four developing companies interviewed, one company had never applied traceability and thus could not estimate demands or share experiences. All others identified the huge efforts needed to document and maintain tracelinks as the main challenge of the approach. In this context, an interviewee from one company pointed out that within his company a single person is responsible for tracelink recording (by checking and filling out cells in dependency matrices) subsequent to artifact completion. Whenever changes occur, these matrices have to be re-evaluated. At the same time the accuracy in these recording and maintenance tasks have to be especially high, as "90 % correctness is not sufficient". Depending on the complexity of the development project, this can be a full-time job for a developer. Another problem that was mentioned by one person is the manner in which traceability is established in industry: high-level requirements are captured, further detailed and structured according to systems, assemblies or components. Tracelinks are then established between those higher and lower levels but cross-links between the lower levels are missing. As a result, the impact of changes in one system on another one cannot be evaluated, ruling out one important benefit of traceability. Despite those difficulties in practice, all vendors have identified traceability as an important aspect of system development. However, they express the need for guidelines on how to apply traceability theories in an industrial context.

Both surveys revealed that the business integration of traceability between development artifacts is not evenly distributed throughout the disciplines. As a rule of thumb, we found out that, the more software and electronics engineering involved in the development, the more common the application of traceability. Reasons for this are an easier automation in model based software design and the regulations by standards for safety-critical functions. According to the interviewed companies and vendors a major challenge is the required amount of interfaces when recording tracelinks between artifacts from different tools. In practice this hinders the establishing of continuous traceability throughout the development process. Not specialized traceability tools but authoring tools like DOORS or Microsoft Excel are used to document dependencies, constraining traceability to artifacts, which are managed inside these tools. For the same reason recording support and further usage of tracelinks are uncommon.

In summary, there are several approaches (especially in requirements management of software and electronics development) for traceability being used in industry. Furthermore the tools of all interviewed vendors provide functionalities to trace artifacts during development. Yet, the sole purpose for traceability in industry is documentation in order to comply with standards. No further potential has been identified. Essentially, there are two reasons for the lack of comprehensive use of traceability: too high an effort is required for tracelink recording, and that additional benefits to documentation are not clear enough.

IV. Introducing ModelTracer – a Traceability Tool for Systems Engineering

During the joint research project ISYPROM a number of methods have been developed aiming at a traceability of development data in Systems Engineering. The prototypical traceability software tool ModelTracer was implemented for the purpose of evaluating the developed methods.

ModelTracer is an integrative approach. It acquires original data from authoring tools via common standards (such as ReqIF) or specific adapters. This authoring data cannot be modified within ModelTracer as the philosophy is to leave the authoring exclusively to the respective engineering tool. Once the different artifacts are acquired, tracelinks between their elements are recorded and stored in a central database. The data model for ModelTracer is depicted in Fig. 2.

ModelTracer is supporting Systems Engineering by establishing a consistency between else isolated data artifacts. It harmonizes development processes by providing information regarding dependencies in complex systems to applying developers. The impact of development changes on elements from other disciplines can be estimated and thus respective developers can be informed. Information regarding developed methods for

©2013 Engineering and Technology Publishing

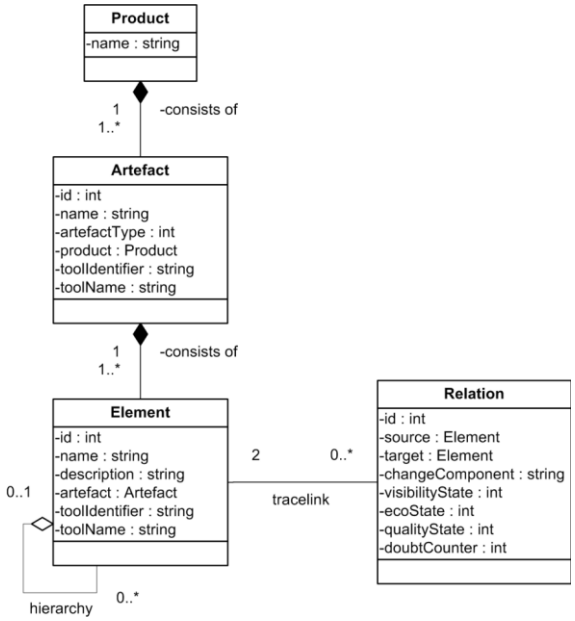

Figure 2. Data model of ModelTracer.

efficient tracelink recording is not subject of this paper (consult [3], [4]). The following section focuses on novel methods for traceability usage in order to support Systems Engineering processes, being the second hindrance for higher traceability acceptance.

V. Traceability Usage Opportunities

Most companies cannot see sufficient benefits to justify the significant workload required to establish traceability between development data (see findings in Section III or [2]). That is one of the reasons why few companies, that are not legally obliged to, practice traceability. The presented surveys in section 3 have shown that traceability is mainly used for documentation purposes. This section introduces a selection of opportunities how development processes benefit from traceability usage of already recorded tracelinks.

Every recorded tracelink represents a piece of development knowledge. Hence every developer increases the knowledge represented in the traceability model when recording a tracelink and thereby confirming the dependency between a combination of elements. In that manner the traceability model aggregates an extensive expert knowledge regarding the interdependencies of a system. This knowledge is needed at various stages of the development process. It can be used to support the utilization of many established or new development methods (such as "Failure Mode and Effects Analysis" or "Quality Function Deployment" [23]).

A. Existing Support For Development Methods Through Traceability

A well researched opportunity to make use of the traceability model is the analysis and restructuring (e.g. in the sense of a modularization) of product structures [13,24,25]. As mentioned earlier, METUS [16] offers analysis functions in order to e.g. optimize costs or the weight of the system under consideration using links between two hierarchically structured artifacts: functions and components. Tools like Loomeo [12] and CAM [14] provide functionalities to restructure products using Matrices. In order to run these analyses the respective matrix need to be filled with traceability information first, not making use of existing traceability models. Additionally, CAM [26] provides a toolbox to predict the impact of changes. For that purpose each element of the product structure must be given a value expressing how much it amplified or hindered the propagation of a past change [26,27].

[28] first suggested to map requirements to software functions with the means of a dependency matrix [28]. The aim is to capture whether all requirements are covered by the implemented functions. Although this idea rather resembles a coverage analysis the basic construct us suited to be used for the method House of Quality[3]. Additionally, Gotel and Finkelstein suggest to use well established methods such as the House of Quality to acquire traceability information [6]. Although some authors have noticed the close link between traceability and the House of Quality no actual implementation of a traceability-based House of Quality has been published. The AID method in [29] provides a coupling between three dependency matrices and a QFD matrix. The actual benefit is limited to importing the elements from the dependency matrix into the QFD matrix without making use of the actual traceability information.

An analysis based on existing traceability information is provided by Reqtify. Among others they can be used to generate progress reports measuring to what degree requirements are already linked [11].

In conclusion, it must be stated that there are only a limited number of methods to support Systems Engineering processes based on existing traceability information. For the majority of DSM-based applications tracelinks are recorded for particular purposes only. These tracelinks then cover a limited fraction e.g. a small part of one artifact or a particular combination of artifacts. Traceability that must be proven for legal reasons is often not compatible with these approaches.

In this paper a traceable consistency of development data between various artifacts containing development data is propagated. Therefore a broad variety of traceability usage opportunities for different development phases and combinations of artifacts are conceivable. Some traceability usage opportunities are presented in the following paragraphs.

B. Novel Support For Development Methods Through Traceability

For the application of many development methods, knowledge regarding interdependencies of the system under consideration is essential. Most of these interdependencies are already available in the traceability model as tracelinks. For that reason the authors examined to what extend traceability information can support each of the following methods: Fault Tree Analysis (FTA),

3 For detailed information regarding the method House of Quality see Section V, Subsection B.

©2013 Engineering and Technology Publishing

Failure Mode and Effects Analysis (FMEA), Event Tree Analysis (ETA), House of Quality (HoQ) as a part of the Quality Function Deployment (QFD), Pareto Analysis and tracelink-based Progress Monitoring. Three of these methods were found most suitable considering the extent to which traceability information supports their application on the one hand and the potential impact to industrial processes on the other hand: FMEA, HoQ and tracelink-based Progress Monitoring. Details of the FMEA wizard have already been published in [30] whereas this section presents the detailed concepts for HoQ and tracelink-based Progress Monitoring. The presented methods are based on a traceability model containing three artifacts (requirements, functions and product structure) as well as the tracelinks between their comprising elements.

House of Quality: The overall aim of the House of Quality (HoQ) is to ensure the future product suites the customers' demands, wherefore these two information pieces are mapped in a common matrix [31]. The method originates from quality management and typically relates customers' wishes to product features.

The HoQ method requires developers to initially select a set of customers' wishes and prioritize them according to their importance for the product. In a second step all product features need to be identified that actually fulfill the set of customers' wishes. Subsequently, the dependencies between all selected elements (customers' wishes and product features) need to be identified and assigned a value saying how much a specific product feature fulfills a customers' wish. Based on these two values a parameter can be calculated for every product feature indicating its importance for the future product and allowing to derive a ranking in between the product features. This is meant to help prioritizing future development activities and resources.

Some of these steps can be (partially) automated by using already existing traceability information. Please note, that for the remains of this section customers' wishes are represented by the requirements artifact while product features are represented by functions (as a verbal description of the product features to be designed). The operation of the House of Quality as it is implemented in the ModelTracer is divided into six steps (see Fig. 3).

At first the developer is requested to select specific customer requirements – this inevitably has to be a manual activity. The selection of the related functions is then performed automatically based on the traceability

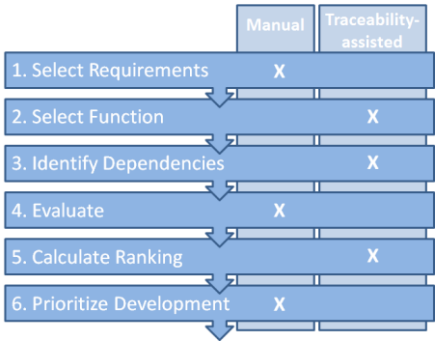

Figure 3. Traceability-supported House of Quality method.

information. For that purpose the database is searched to find all linked functions of each selected requirement. All identified functions are marked in the tree structure (see Fig. 4), signaling their status as selected.

If the developer is not fully happy with the suggested pre-selection, corrections can be performed through manual (de)selection of elements. When the selection is complete the House of Quality is generated automatically, displaying all selected requirements and functions as well as their connecting tracelinks.

This is implemented by drawing a matrix, with all selected requirements elements forming the first column (augmented by a pre-defined default value for the importance of the requirement for the product) and all selected functions forming the heading line. If a tracelink exists between a requirements and a function, the pre-defined default value for the degree to which the function is fulfilling the respective requirement is filled in the respective cell automatically (see Fig. 5). In that manner developers save a lot of time allowing them to focus their attention on the following evaluation. In the fourth step developers are enabled to change the values (through a drop-down showing a pre-defined scale) for the general importance of every requirement and the attribute to every element pair expressing how much a function is actually fulfilling a particular requirement. Based on this information the HoQ wizard automatically calculates a ranking of the listed functions. This ranking can work as decision support for project managers when eventually assigning priorities to the given functions. To recapitulate taken decisions the House of Quality can be saved and reloaded.

Progress Monitoring: The second presented opportunity for traceability usage addresses the management aspects in product development, where the traceability-model can assist in information aggregation [32], [33]. Nowadays progress reports require a lot of manual work. When writing the report many different people have to be contacted and asked about the respective status of their development, although in many cases this information is already available. In many tools developers have to set attributes describing the current status of an element e.g. maturity statuses like "released" in PLM or the fulfillment of a test case. The interfacing

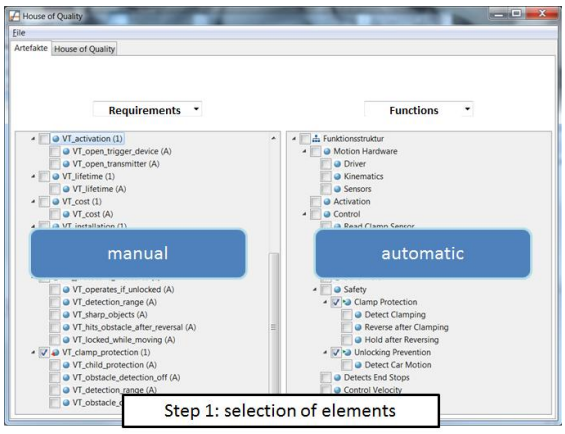

Figure 4. Manual and automatic selection of elements in the House of Quality wizard.

©2013 Engineering and Technology Publishing

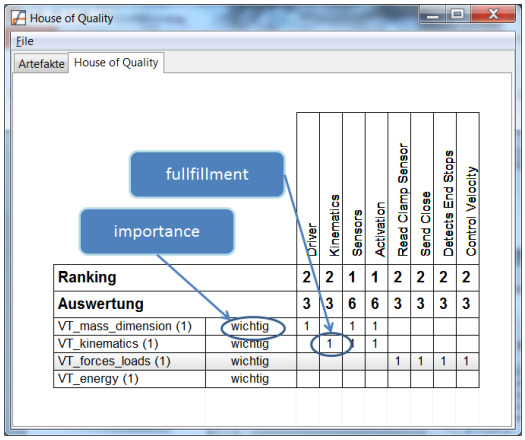

Figure 5. House of Quality matrix displaying relations between requirements (left) and function (top).

of these authoring tools with the ModelTracer allows the status attributes to be easily extracted from such tools, following the requirement of [34] to interoperate with a multitude of engineering tools.

The aim of tracelink-based Progress Monitoring is to enable management to generate an overview of the current development progress at any given time – objective and without any additional human effort. Its basic idea is to estimate the current status of an element (see Fig. 6: actual status of X1 in artifact X is requested) by aggregating status information from all linked elements of a specific artifact Y (in Fig. 6: Y1 and Y2) under the prerequisite that the authoring tool of artifact Y contains up to date status information. The knowledge which information needs to be aggregated is taken from the traceability model. Such functionality is recommended in literature for traceability tools [35], [36].

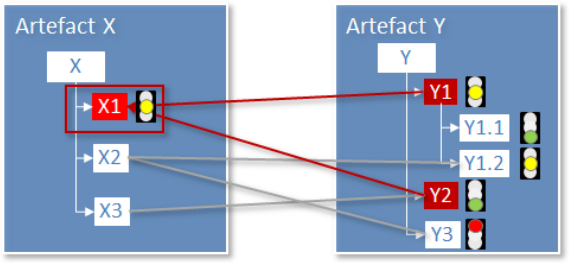

Figure 6. Progress Monitoring: defining the current status of element X1 by aggregating the status information of linked elements Y1 and Y2.

For example, usually there are a number of parts from the product structure necessary to fulfill a certain function. This function and the respective product elements are related by tracelinks, being available in the traceability model. Therefore aggregating the statuses of the linked parts and assemblies provides an estimation of the current implementation status of the considered function.

In order to enable ModelTracer to perform this kind of Progress Monitoring it has to be defined, which artifact is defining the status of another artifact's elements. In this case the status of product structure elements define the status of the function they are linked to. Additionally it has to be specified, which object in the data model is carrying the desired status information: here the status of a part or an assembly in the PLM system.

As soon as these prerequisites are met, the wizard can be used. Developers are requested to choose a pair of artifacts from a drop-down menu (here functions and product structure). In the next step all functions, whose current status is requested, have to be manually selected from a listed tree structure. The selection of the related parts and assemblies is then performed automatically based on the traceability information. For that purpose the database is searched to identify all linked elements of the product structure for each selected requirement and their respective status in the PLM system. All identified parts and assemblies are checked in the tree structure, signaling their status as selected. Subsequently status information is aggregated and displayed in a small diagram next to every selected function.

A similar example is the close relation of requirements and test cases. It is often necessary to test multiple aspects of a system before being able to declare if a requirement is fully satisfied. Assuming all test cases are linked to the particular requirement and their current status is documented, one can easily aggregate the test status of a requirement.

The procedure is comparable to the one described before. The developer selects the artifacts requirements and test cases from the drop-down list. Subsequently he selects the requirement, which current status is of interest (R1.1 in Fig. 7). The Progress Monitor automatically identifies all linked Test Cases (TC 1.1 and TC 2) through searching the saved tracelinks between the artifacts (red arrows in Fig. 7). All Test Cases have a current status documented in their authoring systems (symbolized by green lights). TC 1.1 and TC 2 each have a status of "passed". Since all of its Test Cases have successfully passed their test, Requirement 1.1 can consequently be assigned the status satisfied (also symbolized by green lights here).

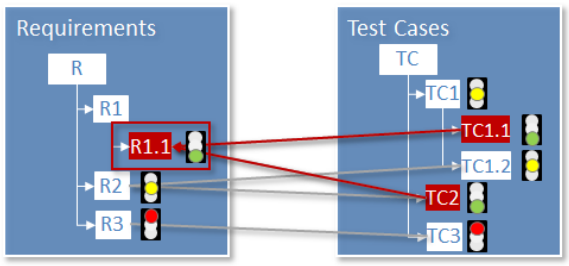

Figure 7. Requesting the current status of a requirement by automatically aggregating status information of its respective test cases.

All traceability usage concepts depend on a correct and complete traceability model. Missing as well as incorrect tracelinks can lead to insufficient results of the described methods and thereby misguide developers in their decision process. Nonetheless the authors believe there is a broad variety of already established methods in product development, whose utilization can be supported and accelerated by supplying traceability information. In that manner traceability contributes to virtual product development by allowing developers to focus on creative

©2013 Engineering and Technology Publishing

work rather than on repeatedly filling in spreadsheets with already existing information.

VI. Conclusion & Outlook

Handling the transdisciplinary development of complex systems and the dependencies between their comprising elements is an enormous challenge. An approach coping with this challenge is to explicitly model these dependencies and thus establish traceability.

The present paper has presented the key findings of two industrial surveys and the introduction of novel opportunities for traceability usage through a traceability-based House of Quality wizard and Progress Monitoring.

Both surveys investigated the industrial acceptance and application of traceability approaches. They revealed that traceability is not evenly distributed throughout the disciplines: the more software and electronics engineering is involved, the more common is the application of traceability. Additionally, it was discovered that the sole purpose for traceability in industry is documentation in order to comply with standards. No further potential is currently exploited, even if traceability information is available. Generally, two reasons for the rarity of this practice were identified: too high an effort needed for tracelink recording and non-obvious additional benefits.

Addressing the second deficit, detailed concepts for a tracelink-based Progress Monitoring and for supporting the House of Quality with traceability information were introduced. The elaborated methods have been prototypically implemented in the ModelTracer as a proof of concept. A hypothetic product example has been used to verify their effectiveness. Both methods were presented to industry representatives during workshops. Their initial feedback is very promising. In that regard the initial research question can be answered: There are a number of opportunities how traceability models can support product development. Nonetheless in a next step both introduced methods still need to be validated in a real industry environment. An applicable guideline on how to integrate which traceability aspects in which Systems Engineering processes is also an open research topic.

Traceability is not practiced comprehensively when developing complex systems in industry. This will only change, if novel approaches are available reducing the required recording effort and improving the achievable benefits through their usage. The functionalities of ModelTracer presented in this paper are one step towards this direction.

References

[1] A. Egyed, P. Grunbacher, M. Heindl, and S. Biffl, "Value-Based Requirements Traceability: Lessons Learned," in *Proc. 15th IEEE International Requirements Engineering Conference*, New Delhi, India, October 15-19 2007, pp. 115–118.

[2] S. Winkler and J. Pilgrim, "A survey of traceability in requirements engineering and model-driven development," *Software Systems Modeling*, vol. 9, pp. 529–565, 2010.

[3] R. Stark and A. Figge, "Eco tracing - A systems engineering method for efficient tracelink modelling," in *Proc. 18th International Conference on Engineering Design*, Product and Systems Design 2011, vol. 4, pp. 21–32.

[4] S. F. Königs, G. Beier, A. Figge, and R. Stark, "Traceability in systems engineering: Review of industrial practices, state-of-the-art technologies and new research solutions," *Advanced Engineering Informatics* vol. 26, no. 4, pp. 924–940, 2012.

[5] F. A. C. Pinheiro, "Requirements Traceability," in *Perspectives on Software Requirements*, J. Sampaio do Prado Leite, J. H. Doorn (Eds.), Springer, Berlin, 2003, pp. 91–113.

[6] O. Gotel and A. Finkelstein, "An analysis of the requirements traceability problem," in *1st International Conference on Rquirements Engineering*, Colorado Springs, CO, U.S.A, pp. 94–101.

[7] R. Paige, K. Goran, D. Kolovos, S. Zschaler, and C. Power, "Building Model-Driven Engineering Traceability Classifications," in *Proc. ECMDA Traceability Workshop*, Berlin, Germany, 2008, pp. 49–58.

[8] N. Aizenbud-Reshef, B. T. Nolan, J. Rubin, and Y. Shaham-Gafni, "Model traceability," *Browse Journals & Magazines*, vol. 45, pp. 515–526, 2006.

[9] J. Buur and M. M. Andreasen, "Design models in mechatronic product development," *Design Studies*, vol. 10, no. 3, pp. 155–162. 1989.

[10] IBM (October 12 2011). [Online]. Available: http://www-01.ibm.com/software/awdtools/doors/

[11] Geensoft. (December 17, 2010). Reqtify: Effective Solution for Managing Requirements Traceability and Impact Analysis across Hardware and Software Projects Lifecycle. [Online]. Available: http://www.geensoft.com/en/article/reqtify.

[12] Teseon GmbH, Loomeo. (December 20, 2010). [Online]. Available: http://www.teseon.com

[13] M. S. Maurer, Structural Awareness in Complex Product Design, Dissertation, München, 2007.

[14] Cambridge EDC. (September 19, 2011).Cambridge Advanced Modeller. [Online]. Available: https://www-edc.eng.cam.ac.uk/cam/.

[15] C. Eckert, P. J. Clarkson, and W. Zanker, "Change and customisation in complex engineering domains," *Research in Engineering Design*, vol. 15, no. 1, pp 1–21, 2004.

[16] ID-Consult GmbH, METUS - The System Design Method.

[17] G. Tretow, J. Göpfert, and C. Heese, "In sieben schritten systematisch entwickeln," CAD-CAM Report, 2008, pp. 36–39.

[18] Siemens PLM. Teamcenter. (November 30, 2012). [Online]. Available: http://www.plm.automation.siemens.com/de_de/products/teamcenter/index.shtml

[19] Dassault Systèmes. V6. (November 30, 2012) [Online]. Available: http://www.3ds.com/de/products/ v6/welcome/,

[20] P. van Gorp, F. Altheide, and D. Janssens, "Traceability and fine-grained constraints in interactive inconsistency management," in *Second ECMDA Traceability Workshop*, T. Neple, J. Oldevik, and J. Aagedal (Eds.), Bilbao, Spain, 2006.

[21] G. B. Rosenauer, "A standards-based approach to dynamic tool integration using java business integration: A redsign of the toolnet framework built on entrerprise integration standards," Diplomarbeit, Wien, 2008.

[22] Atego. (November 30, 2012). Atego Workbench. [Online]. Available: http://www.atego.com/downloads/support/data-sheets/AtegoWorkbench.pdf.

[23] S. Mizuno, Y. Akao, and K. Ishihara, "QFD, the customer-driven approach to quality planning and deployment," *Asian Productivity Organization*, Tokyo, Japan, 1994.

[24] P. J. Clarkson, C. Simons, and C. Eckert, "Predicting change propagation in complex design," *Journal of Mechanical Design* vol. 126, 2004.

[25] U. Lindemann, M. Maurer, and T. Braun, "Structural complexity management: An approach for the field of product design," *Springer Berlin Heidelberg*, Berlin, Heidelberg, 2009.

[26] D. C. Wynn, N. H. M. Caldwell, and P. J. Clarkson, "Can change prediction help prioritise redesign work in future engineering systems?" in *Proc. 11th International Design Conference*, Dubrovnik, Croatia, 2010, pp. 1691–1702.

[27] M. Giffin, O. de Weck, G. Bounova, R. Keller, C. Eckert, and P. J. Clarkson, "Change Propagation Analysis in Complex Technical Systems," *Journal of Mechanical Design*, vol. 131, 2009.

[28] J. MacMillan and J. R. Vosburgh, "Software quality indicators," Defense Technical Information Center, Ft. Belvoir, 1986.

©2013 Engineering and Technology Publishing

[29] V. Holley, B. Yannou, and M. Jankovic, "Towards the Predicition of multiphysic interactions using MDM and QFD matrices," in *Proc. 11th International Design Conference*, Dubrovnik, Croatia, 2010, pp. 1109–1118.
[30] G. Beier, A. Figge, T. Lehner, and A. Metin, "Durchgängige Nachverfolgbarkeit in der Systementwicklung: Datendurchgängigkeit für die Entwicklung von Fahrzeugfunktionen," in ZWF: Zeitschrift für Wirtschaftlichen Fabrikbetrieb, Carl Hanser Verlag, München, 2011, pp. 462–465.
[31] J. R. Hauser and D. Clausing, "The house of quality," *Harvard Business Review*, vol. 66. pp. 63–73, 1988
[32] B. Ramesh and M. Edwards, "Issues in the development of a requirements traceability model," in *Proc. International Symposium on Requirements Engineering*, San Diego, USA, 1993, pp. 256–259.
[33] P. Lago, H. Muccini, and H. Vanvliet, "A scoped approach to traceability management," *Journal of Systems and Software*, vol. 82, pp. 168–182, 2009.
[34] A. Marcus, X. Xie, and D. Poshyvanyk, "When and how to visualize traceability links?" in *Proc. Third international Workshop on Traceability in Emerging Forms of Software Engineering*, Long Beach, USA, 2005, pp. 56-61.
[35] C. Duan and J. Cleland-Huang, "Visualization and Analysis in Automated Trace Retrieval," in *Proc 1st International workshop on Requirements Engineering Visualization*, Minneapolis, USA, 2006, pp. 5-13.
[36] International Organisation for Standardization, Information Technology - Systems and Software Engineering: Guide for requirements engineering tool capabilities, Technical Report, 2009.

Robert Müller studies Informatics at the Humboldt Universität Berlin. He is currently working for the department for Model-based Engineering at the Fraunhofer Institute for Production Systems and Design Technology, Division Virtual Product in Berlin. His personal research interests include information visualization and traceability in product development processes.

Dipl.-Ing. Grischa Beier studied mechanical engineering with a specialization in engineering design (Technical University Ilmenau - Germany, Universidade Federal de Santa Catarina – Brasil, St. Petersburg State University of Information Technologies, Mechanics and Optics – Russia). Since 2006 he works as a research fellow at the Fraunhofer Institute for Production Systems and Design Technology, Division Virtual Product Creation. His research interests include modeling procedures in product creation and information provision in cross-domain development processes.

Dipl.-Ing. Asmus Figge studied mechanical engineering with a specialization in engineering design at the Technische Universität Dresden. From 2008 until 2010 he worked as a research fellow at the Chair for Industrial Information Technology at the Technische Universität Berlin. Since 2011 he works as a research fellow at the Fraunhofer Institute for Production System and Design Technology, Division Virtual Product Creation. His research interests include efficient modeling and maintenance of tracelinks.

Dipl.-Ing. Uwe Rothenburg studied Automotive Technology at the Technische Universität Berlin. After completion of his Dipl.-Ing. degree he started working as a research fellow at the Fraunhofer Institute for Production Systems and Design Technology, Division Virtual Product Creation. Today he is head of the department for Model-based Engineering. His research interests include the simulation of dismantling processes and the adaptation of geometric models for functional validation of virtual prototypes.

Prof. Dr.-Ing. Rainer Stark is head of the Division Virtual Product Creation and director of the Chair for Industrial Information Technology at the Technische Universität Berlin since 02/2008. After his studies in mechanical engineering (Ruhr University Bochum and Texas A&M University) he received the Dr.-Ing. degree from the Universität des Saarlandes Saarbrücken. During his industrial activities he worked in different leading positions in the automotive industry. His research topics are the intuitive and context-related information modeling, intuitively usable and functionally experienceable virtual prototypes, the function-oriented Virtual Product Creation as well as development processes and methodologies for the product modeling.

©2013 Engineering and Technology Publishing