Department: Head
Editor: Name, xxxx@email

Managing Traceability Information Models: Not such a simple task after all?

Salome Maro, Jan-Philipp Steghöfer, Eric Knauss, Jennifer Horkoff, Rashidah Kasauli
Chalmers | University of Gothenburg, Sweden

Rebekka Wohlrab
Systemite AB and Chalmers | University of Gothenburg, Sweden

Jesper Lysemose Korsgaard, Florian Wartenberg, Niels Jørgen Strøm
Grundfos, Denmark

Ruben Alexandersson
Volvo Cars, Sweden

***Abstract*—Practitioners are poorly supported by the scientific literature when managing traceability information models (TIMs), which capture the structure and semantics of trace links. In practice, companies manage their TIMs in very different ways, even in cases where companies share many similarities. We present our findings from an in-depth focus group about TIM management with three different systems engineering companies. We find that the concrete needs of the companies as well as challenges such as scale and workflow integration are not considered by existing scientific work. We thus issue a call-to-arms for the requirements engineering and software and systems traceability communities, the two main communities for traceability research, to refocus their work on these practical problems.**

A CRUCIAL BUILDING BLOCK in a successful traceability strategy for any organisation is a well-defined and goal-oriented traceability information model (TIM). A TIM defines the relationships between the diverse artifacts created during system development, often by different teams and often different parts of an organisation. For these relationships, it defines structural aspects such as trace links types and cardinalities. Additionally, a TIM's semantic information can be informed by organisational structures, information about the workflow, the interaction between products and platforms, and the collaboration between different parts of an organisation. This makes a TIM a crucial artifact in an organisation and, at the same time, very difficult to change once established. The main question when defining a TIM is thus how to capture these aspects and

translate them into a model that the individual engineer can successfully work with and use to create value. TIM management includes activities such as defining which relationship types are needed, implementing the designed TIM in a tool and managing its evolution. Because of the complexity of what a TIM captures and its importance for traceability, each company, and sometimes each project, requires a tailored TIM.

Due to this specificity, practitioners lack concrete guidelines when creating TIMs especially in large systems engineering companies. To address this issue, we conducted a workshop with three companies developing systems from three different domains to understand their current and emerging TIMs and the challenges they face when managing them. The workshop revealed that each of the three companies manage their TIMs in radically different ways but that they face common challenges which are yet to be addressed in literature. We summarise our findings in this paper and describe research challenges that result from them as a call-to-arms for the RE and software and systems traceability community.

Important terms

Traceability is is defined as the ability to interrelate any uniquely identifiable software engineering artifacts to any other, maintain required links over time, and use the resulting network to answer questions of both the software product and its development process [1].

A traceability strategy is a plan of action leveraged to design a traceability solution, consisting of a traceability process, tooling, and a traceability information model (TIM).

A traceability information model (TIM) defines all the artifact types that need to be traced to and the different relationship types between the artifacts types. Figure 1 shows an example of a simple TIM.

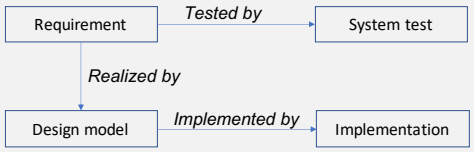

Figure 1. A simple TIM as an example. It connects four different artefact types (requirements, system tests, design models, and implementation) with three different trace link types (tested by, realized by, and implemented by).

Background on TIM management

The literature on TIMs management is split into three main themes: research proposing TIMs for specific purposes e.g., TIMs for safety certification [2] or for managing non-functional requirements [3]; research proposing generic TIMs that can be reused, e.g., [4]; and finally tools that support concrete implementation of TIMs that have already been designed, such as [5]. Regardless of theme, however, many of the studies are abstract (e.g, [6]) or conducted using small examples that are isolated from realistic company environments. Their generalisability is thus dubious. In both the research community and in industry, it is agreed that no single TIM fits every development context, therefore, the number of studies proposing generic TIMs has decreased over time, and research is focused on project-specific or task-specific TIMs [7].

While it is true that no single TIM fits every development context, concrete guidelines are needed to support practitioners to define, implement and maintain their specific TIMs. This part of research is lacking and practitioners have to manage TIMs with little support from scientific literature. A few studies exist, e.g., the work by Ramesh and Jarke [6] provides some reference TIMs that practitioners could use as a starting point for designing their own TIMs. The paper discusses issues encountered when using the reference models in few case studies. Additionally, Mäder et al. [8] gives a few guidelines for defining TIMs and provides guidelines for traceability in safety-critical environments which also include some guidelines on TIM design [9]. Overall, different aspects are still missing such as as alignment of TIMs with different processes and workflows, TIM evolution and in general TIM management in large organisations.

Our long-term goal is to fill this gap. This paper presents the first step in this endeavour, i.e., identifying practitioner's challenges when

Table 1. Overview of the three companies that participated in the workshop. In order to protect each company's anonymity, the text refers to Company 1, 2, and 3 instead.

	Company A	Company B	Company C
Domain	Industrial automation	Telecommunication	Automotive
No. of participants	3	1	3
Participants' roles	2 Lead Systems Engineers 1 System Test Engineer	1 Systems Engineer	1 Information Architect 2 Process, method, and tools specialists

managing TIMs, in order to synthesize useful solutions and guidelines for these challenges.

Methodology

We conducted a focus group with five researchers and seven practitioners from three companies from the systems engineering domain. All participants from the companies have experience with traceability. Additionally, participants from two of the three companies have practical experience designing and managing TIMs. During the focus group, one of the researchers presented examples of TIMs and a summary of guidelines on how to define TIMs extracted from literature. This was followed by presentations from the three companies (Table 1) on their TIM and their TIM management process. Each presentation was followed by a discussion of the specific challenges presented and the relationship to literature. The discussion was open such that all participants could ask questions about the presented TIM and TIM management process. The researchers took notes during the presentation and discussions.

The notes were later analyzed in a brainstorming session with five researchers. The challenges identified were sorted and grouped into three categories and member checked with the industry participants of the focus group.

In order to protect each company's anonymity, we refer to Company 1, 2, and 3 instead, when reporting results. These numbers are not in any particular order; specifically, Company 1 does not necessarily correspond to Company A.

Practitioners' Challenges

The three companies that presented their TIMs in the workshop all have a different approach towards the design of TIMs. In Company 1, the TIM is not formally documented as a model, but implicitly defined as part of the definition of done and integrated in the continuous integration tooling. Developers are thus aware of which trace link types to create. The TIM focuses on linking development artifacts in an agile environment, e.g., test cases, user stories and code. *Company 2* defines an overall information model that models all artifacts and aspects used in the development environment. The traceability strategy is derived from this information model. *Company 3* follows a formal process and defines a company-wide TIM with a centralized approach via a feature model and including traceability in two dimensions: traceability between platform artifacts and between the product instances of the platform artifacts.

While the companies follow different approaches, the challenges can be ascribed to three main drivers:

- The companies develop *complex products* using a *product line approach*.
- The companies are *large and distributed* and different teams need to work together on the same end product.
- The companies have *long-lasting products* which means traceability needs to be maintained over a long period of time.

The challenges are presented in the sections below and summarised in Figure 2.

Complex products

The focus group allowed us to identify challenges that arise due to the complexity of the products developed by the companies.

- **Inherent complexity of TIMs** A TIM for a large and complex system containing a large

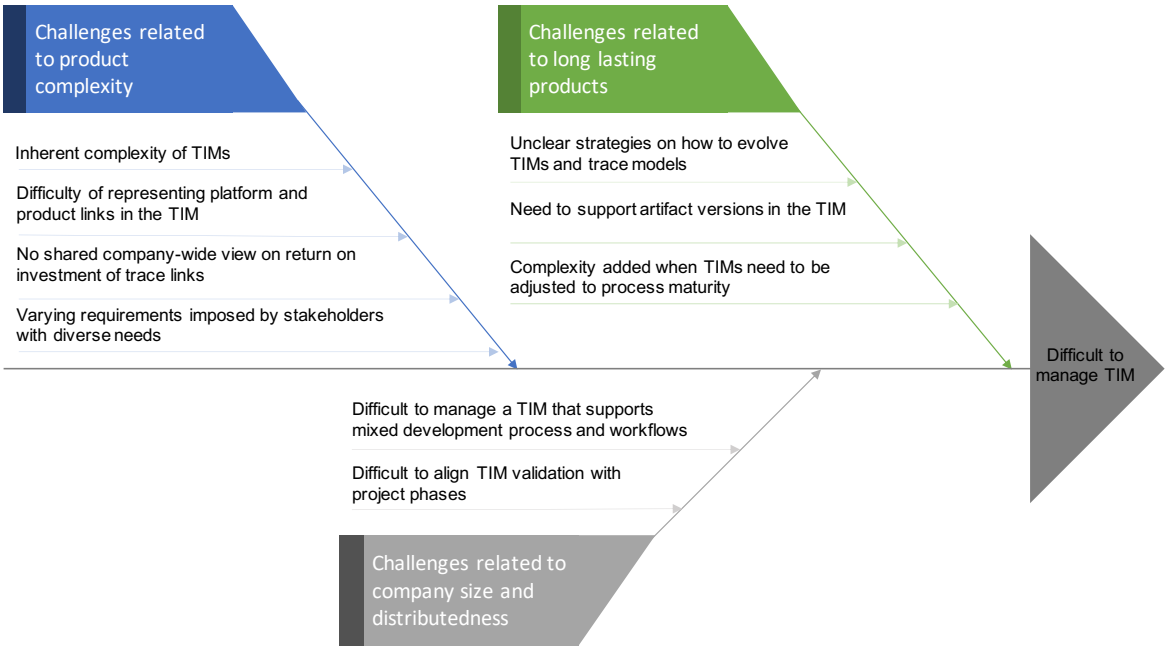

Figure 2. Overview of Challenges that make design of TIMs difficult in practice.

number of artifact types needs to encompass the entire product and is therefore complex itself with a large number of artifacts types and relationship types. Thus managing such a TIM is difficult.

Company 2 addresses this challenge by creating an overall information model. It not only defines all artifact types that exist along with their relationship types, but also captures additional information such as tool capabilities and role responsibilities. While one might argue that the information model constitutes a TIM, it contains aspects that are not relevant for traceability of software development artifacts which are not clearly delineated. For instance, the information model used by *Company 2* contains more than 50 artifact types and also defines workflows and responsible roles. This makes it difficult to reason about traceability with such information models.

One of the engineers from *Company 2* reported that the complexity of the overall information model makes it difficult to define a subset of the information model as a TIM and plan an overall traceability strategy for the company based on this.

- **Difficulty of representing platform and product links in the TIM** In a product line environment, traceability needs to support maintenance of the shared assets in the platform as well as the assets for specific products. In such a scenario, companies struggle with defining a TIM in a way that caters to product lines activities (e.g., variability tracking and maintenance) and activities related to individual products (e.g., change impact analysis and project progress monitoring) alike. Ideally, every feature is first implemented on the platform level and only reused at the product level so that trace links from the platform are inherited by specific product instances to achieve full traceability. However, in practice, as reported by Company 2 and 3, this is rarely the case and product-specific features exist that make traceability on the product level necessary. It is a challenge to represent platform and product traceability in the TIM to make the traceability as visible and efficient as possible.
- **No shared company-wide view on return on investment of trace links** Stakeholders defining a TIM need reasons for including certain relationship types in the TIM. A common way to motivate a relationship type is to link it to a stakeholder need by eliciting the traceability needs of the different stakeholders in the company [8], [9]. For smaller systems,

it is easier to understand which trace link types are needed since knowledge of how the different artifact types fit together is common, however for complex systems, this elicitation leads to many suggested relationship types due to the large number of artifact types. They thus need to be filtered by the expected value they create, but it is difficult to predict the concrete value of certain relationship types. Moreover, different link types have different value for different stakeholders. While this is a general traceability challenge, it is also TIM related since for complex systems it is unclear which combination of trace link types provide the most value. All the three companies in the workshop struggle with making decisions on which relationship types to include in the TIM. For example, in Company 2, there are more than 50 artifact types, which makes it difficult to decide which trace link types between these artifacts are actually valuable.

- **Varying requirements imposed by stakeholders with diverse needs** A TIM is typically designed according to the stakeholder needs in the company. For large organisations, a single TIM needs to address the needs of many stakeholders. For instance, different stakeholders may not only require different link types but also require same link types but on different abstraction levels. This results in large and complex TIMs. The representation of such TIMs to stakeholders is difficult since different stakeholders need to derive different information from the TIM. For example, performing a system level safety analysis requires trace links on a higher abstraction level compared to low level safety analysis of specific functions. In practice, it is challenging to design a TIM that is flexible enough to support these different needs and still provide guidance to developers so that they create useful trace links using such a TIM.

Large and distributed companies

In this section, we describe challenges that are due to the companies being large and distributed but still having to coordinate and work together to create the same end product.

- **Difficult to define a TIM that supports mixed development processes and workflows** In large systems engineering companies, different disciplines (e.g., mechanical engineering, electrical engineering, and software engineering) need to be coordinated and integrated. Teams that focus on one of those disciplines tend to follow processes that are aligned with the way the parts of the product are developed. A mechanical engineer, e.g., is more likely to use a waterfall-ish process whereas a software engineer might work in an iterative-incremental fashion. Each process requires different kinds of artifacts, e.g., a product requirement document for the mechanical engineer and a backlog item for the software engineer. A TIM that is used in such an organisation needs to cater to all processes or workflows and therefore integrate all artifacts prescribed by all used processes, independent of discipline, thus contributing to the TIM's complexity. This challenge was reported by company 2 and 3.
- **Difficult to align TIM conformance checks with project phases** Constraints defined in the TIM such as cardinality are used to check if links exist and are syntactically correct. For instance, if the TIM defines a one-to-one cardinality for a link from requirement to test, then this should be enforced to ensure that the link actually exists and the cardinality is adhered to. However, in practice, the enforcement of such conformance checks needs to be aligned with the current project phase or current phase within the sprint. For instance, the companies reported that enforcing that there is a link between a requirement and a test during requirements elicitation through the TIM is not helpful, but is helpful at a later stage when requirements are already gathered and need to be maintained. This challenge becomes more pronounced when mixed development processes are used and there is no clear global definition of phases. Additionally, constraints on the TIM should not limit the agility of the different teams.

Long lasting products

The products developed by the three companies have to be maintained over a long period of time. Thus the TIM and the traceability links in

place also need to be maintained. This introduces three main challenges with respect to TIM management as organisations, development processes and system engineering techniques evolve.

- **Unclear strategies on how to evolve TIMs and trace models** All the companies acknowledged that once the TIM is defined and used, it is hard to change both on a technical and process level, since existing links conforming to the current TIM need to be migrated. Additionally, there are no strategies in place to analyse the existing traceability strategy in the companies to decide if the TIM needs to be updated. This leads to TIMs that are defined once and not updated, even though some of the relationships they capture have evolved.
  Currently, TIMs are changed in big bang approaches: when a new platform is created at Company 2, e.g., the TIM is changed as well. The old platform is maintained with the old TIM and the new platform will be maintained with the new TIM. However, the company now wants to move to a development strategy in which their platform will change less frequently, therefore the company needs a new strategy on how to continuously evolve the TIM.
- **Need to support artifact versions in the TIM** In a typical software and systems engineering product line, various versions of an artifact coexist and are used in products at the same time. Which version is used in which product needs to be tracked. In the automotive domain, e.g., regulations state that the manufacturer needs to trace which versions of the components are deployed in a specific vehicle. Trace links to specific artifact versions thus need to be established both at the platform level and product level (cf. Figure 3).
  Company 2 and 3 reported that it is a challenge to define a TIM that explicitly supports versioning information for the artifacts connected by trace links.
  Company 1 reported creating traceability links from requirements and test cases to specific commits but having issues with tracking artifact versions when multiple artifact repositories are used.
- **Complexity added when TIMs need to be adjusted to process maturity** The stability of a TIM reflects the stability of the development process. For companies transitioning from one development process to another, TIM management is challenging. Product development continues in the midst of process changes and traceability still needs to be maintained. In such situations, it is necessary to manage and evolve multiple TIMs in parallel as different development teams have different schedules for adopting a new development process. For instance, Company 3 is transitioning from single product development to product-line development. In this transition, three TIMs are required: a TIM that supports a single product approach; TIM that supports a product-line approach; and a TIM that supports teams that are "in between", i.e., teams that have started to transition but are not yet fully transitioned. Such teams end up creating more traceability links than needed in either of the development paradigms, as they must trace to artifacts in the old and new process. Moreover, as teams migrate from one paradigm to another, trace link migration is difficult due to the challenge of evolving TIMs and trace models discussed before.

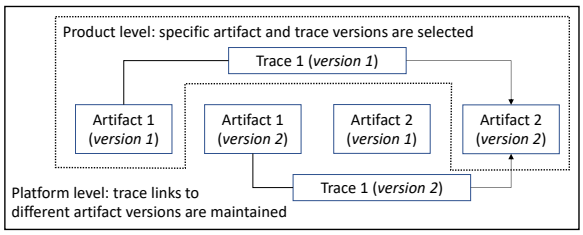

Figure 3. Versioning of trace links in a product line environment.

Refocusing TIM design research

The existing guidelines on TIM management such as those from [6], [8] and [9] are a good starting point for practitioners. However, these guidelines do not cover all the challenges practitioners face when creating TIMs in large systems engineering companies. Particularly there is little support for understanding the implications of the development process aspect on the TIM, especially when mixed development processes are used. Our findings show that this is a problem in

practice.

For some of the challenges, e.g., TIMs for product lines, changeability of TIMs and TIM modularisation, research such as [10], [11] and [12] exists respectively. However, this research only investigates the tooling aspect with restricting assumptions such as that all artifacts are persisted in certain formats and specific tools are used. This limits the transferability of such research to practice.

Moreover, some of the guidelines are conflicting in practice. For instance, some literature suggests that the level of abstraction between two artifacts connected by a trace link should match [8] while organisations that develop safety-critical products need to trace between artifacts on different levels of abstraction to be able to perform various safety analysis tasks required by safety standards. One of the solutions suggested is to trace to different levels of abstraction depending on the use of the trace links and have tool support to represent the links at different levels of abstraction [6]. However, it is not clear how this can be handled in practice in a cost-efficient and effective manner. Additionally, existing TIM design guidelines are scattered in various publications making them hard to find, and there is no guidance on how practitioners can operationalize these guidelines.

From our research, we believe that among many possible research directions, the following research is needed in the context of the challenges we discussed:

- Research on managing complex TIMs such as TIM modularization, use of multiple TIMs in an organization and adoption of TIMs in a cost-efficient and effective manner.
- Creation of guidelines for designing TIMs that support product-line platform related activities and product-specific activities, as well as how TIMs can support derivation of product-specific links from platform links.
- How to evolve TIMs and existing trace models such as using model transformations to transform existing links to match a new TIM, and how to evolve TIMs when processes change.
- Research on how organisations can design TIMs that support multiple development processes and workflows, e.g., by defining process-agnostic TIMs and providing guidelines on mapping the different artifact types and links to specific development processes.

While some of the above topics are not new to the requirements and traceability community, our aim is to explicitly point out these challenges to draw more attention to the practical complexity that exists in companies that deploy traceability. We acknowledge the usefulness of technical and tool-related research on traceability. However, from the perspective of managing TIMs, we believe solving many of the challenges requires a process-oriented research perspective. We therefore suggest more empirical studies specifically on TIM management focusing on large organisations developing complex product lines with long-lasting products. Only if research on traceability and TIM management acknowledges these practical challenges will it yield beneficial lessons for practitioners.

Summary

This paper reports on challenges that practitioners face when managing TIMs. The challenges were identified using a focus group with three companies from the automotive, industrial automation and telecommunication domain. The challenges are due to three drivers; complexity of the products developed, the size and distributedness of the companies and the long lasting products that need to be maintained. Our study calls for more research on TIM management targeted to companies with such characteristics.

ACKNOWLEDGMENTS

We thank all company participants for their contributions to this paper. This work has been partially supported by Software Center Project 27 on RE for Large-Scale Agile System Development, the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, the Sida/BRIGHT project 317 under the Makerere-Sida bilateral research program 2015-2020, and the ITEA 3 Call 6 project 17003 PANORAMA.

REFERENCES

1. COEST, “Center of Excellence for Software Traceability (CoEST),” 2015, accessed: 2020-04-29. [Online]. Available: http://www.coest.org

2. J. Cleland-Huang, M. Heimdahl, J. H. Hayes, R. Lutz, and P. Maeder, "Trace queries for safety requirements in high assurance systems," in *International working conference on requirements engineering: Foundation for software quality*. Springer, 2012, pp. 179–193.
3. M. Kassab, O. Ormandjieva, and M. Daneva, "A traceability metamodel for change management of non-functional requirements," in *2008 Sixth International Conference on Software Engineering Research, Management and Applications*. IEEE, 2008, pp. 245–254.
4. A. Espinoza Limón and J. Garbajosa Sopeña, "The need for a unifying traceability scheme," in *ECMDA Traceability Workshop Proceedings*, 2005, pp. 175–184.
5. N. Drivalos, D. S. Kolovos, R. F. Paige, and K. J. Fernandes, "Engineering a DSL for software traceability," in *International Conference on Software Language Engineering*. Springer, 2008, pp. 151–167.
6. B. Ramesh and M. Jarke, "Toward reference models for requirements traceability," *IEEE transactions on software engineering*, vol. 27, no. 1, pp. 58–93, 2001.
7. J. Cleland-Huang, O. C. Gotel, J. Huffman Hayes, P. Mäder, and A. Zisman, "Software traceability: trends and future directions," in *Proceedings of the on Future of Software Engineering*. ACM, 2014, pp. 55–69.
8. P. Mader, O. Gotel, and I. Philippow, "Getting back to basics: Promoting the use of a traceability information model in practice," in *Proceedings of the 2009 ICSE Workshop on Traceability in Emerging Forms of Software Engineering*. IEEE Computer Society, 2009, pp. 21–25.
9. P. Mäder, P. L. Jones, Y. Zhang, and J. Cleland-Huang, "Strategic traceability for safety-critical projects," *IEEE software*, vol. 30, no. 3, pp. 58–66, 2013.
10. N. Anquetil, U. Kulesza, R. Mitschke, A. Moreira, J.-C. Royer, A. Rummler, and A. Sousa, "A model-driven traceability framework for software product lines," *Software & Systems Modeling*, vol. 9, no. 4, pp. 427–451, 2010.
11. R. F. Paige, N. Matragkas, and L. M. Rose, "Evolving models in model-driven engineering: State-of-the-art and future challenges," *Journal of Systems and Software*, vol. 111, pp. 272–280, 2016.
12. P. Mäder and J. Cleland-Huang, "A visual traceability modeling language," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2010, pp. 226–240.

Salome Maro is a PhD candidate at Chalmers | University of Gothenburg, Sweden. Her research is on software and systems traceability tools and processes in industry. She holds an M.Sc. degree in Software Enginering from University of Gothenburg. Contact her at salome.maro@cse.gu.se.

Jan-Philipp Steghöfer is an Associate Professor at Chalmers | University of Gothenburg, Sweden. His research interests include software and systems traceability, large-scale and distributed agile development, software engineering education, and self-adaptive sytems. He holds a PhD from the University of Augsburg, Germany. Contact him at jan-philipp.steghofer@gu.se.

Eric Knauss is an Associate Professor at Chalmers | University of Gothenburg, Sweden. His research focuses on managing requirements and related knowledge in large-scale and distributed software projects. He holds a PhD from Leibniz Universität Hannover, Germany. He is member of program and organization committees of top conferences and reviewer for top journals. Contact him at eric.knauss@cse.gu.se.

Jennifer Horkoff is an Associate Professor at Chalmers | University of Gothenburg, Sweden. Her research focuses on requirements engineering, requirements modeling, quality and value modeling, model analysis and creativity. She holds a PhD from the University of Toronto. Contact her at jennifer.horkoff@gu.se.

Rashidah Kasauli holds a PhD from

Chalmers | University of Gothenburg, Sweden. Her research focuses on requirements, safety-critical systems and large-scale agile development. She holds an Msc in Data Communication and Software Engineering from Makerere University, Uganda. Contact her at rashida@chalmers.se.

Rebekka Wohlrab holds a PhD from Chalmers University of Technology, Sweden. Her research focuses on requirements engineering and software architecture in large-scale agile development. Contact her at wohlrab@chalmers.se.

Jesper Lysemose Korsgaard is Lead Systems Engineer also working with software requirements at GRUNDFOS Holding A/S. Contact him at jlysemose@grundfos.com

Florian Wartenberg is a System Test Engineer at GRUNDFOS holding A/S. His interests focus on requirements, traceability, verification and validation. He holds a diploma in Computer Science from Humboldt University of Berlin. Contact him at fwartenberg@grundfos.com

Niels Jørgen Strøm has an SE-Zert Level B certification and has a background as an electronics engineer and ISO assessor. He is a Chief Systems Engineer at Grundfos Holding A/S and has been with the company for 25 years. He works with product lines, embedded systems, configuration management and process management. Contact him at njstroem@grundfos.com.

Ruben Alexandersson holds a Ph.D. from Chalmers University of Technology, and a background as teacher in Software Engineering. He is currently a Technical Specialist at Volvo Cars and has been with the company for ten years. Contact him at ruben.alexandersson@volvocars.com.