Using Requirements Management Tools in Software Product Line Engineering: The State of the Practice

Danilo Beuche[1], Andreas Birk[2], Heinrich Dreier[3], Andreas Fleischmann[4], Heidi Galle[5], Gerald Heller[6], Dirk Janzen[7], Isabel John[8], Ramin Tavakoli Kolagari[9], , Thomas von der Maßen[10], Andreas Wolfram[5]

[1] *pure-systems GmbH, D-39106 Magdeburg, Germany, danilo.beuche@pure-systems.com*
[2] *Software.Process.Management, D-70197 Stuttgart, Germany, andreas.birk@swpm.de*
[3] *SynSpace, D-79102 Freiburg, Germany, hd@synspace.com*
[4] *Technical University of Munich, Software and Systems Engineering Group, D-85748 Garching, Germany, andreas.fleischmann@in.tum.de*
[5] *Continental, Automotive Systems Division, D-88671 Markdorf, Germany, {heidi.galle, andreas.wolfram}@contiautomotive.com*
[6] *Hewlett-Packard, 71034 Böblingen, Germany, gerald_heller@hp.com*
[7] *Harman/Becker Automotive Systems GmbH, D-76307 Karlsbad, Germany, djanzen@harmanbecker.com*
[8] *Fraunhofer IESE, D-67663 Kaiserslautern, Germany, isabel.john@iese.fraunhofer.de*
[9] *Technical University of Berlin, Software Engineering Group, D-10587 Berlin, Germany, tavakoli@cs.tu-berlin.de*
[10] *arvato direct services GmbH / Bertelsmann AG, D-33310 Gütersloh, Germany thomas.vondermassen@bertelsmann.de*

Abstract

A software product line oriented approach to developing systems has an impact on the development process as a whole. Although the basic idea of product lines is simple, systematic implementation of the approach is a challenging task. In fact, today's software product lines have such complex variability that their handling must be supported by tools, otherwise a systematic approach would not be possible. The same holds for requirements management of product lines. On the basis of specific industrial scenarios the paper derives important requirements that have to be observed if requirements management tools are to be usefully applied to product lines. Current requirements management tools are evaluated on the basis of these requirements in their daily industrial use. The presented requirements can indicate the future direction of tool development and method research.

1. Introduction

Why should we pay more attention to a product line requirements management process than to a single project environment and why is requirements management for product lines much more difficult than for a single project environment?

The question is easy to answer. Like the product itself, the specifications for it will change, evolve and vary as well. In addition, product line requirements are generally more complex, live longer, and involve more critical dependencies than single-product requirements.

In order to keep track of the changes and to control and categorize the requirements, requirements management tools are necessary. Most if not all of the requirements management tools on the market are focused on a single-project scope. But efficient product development will need to reuse as many components and artifacts as possible to keep expenses low for new developments. Reuse by copying all components and requirements of a previous product to form the basis of a new one is a pragmatic approach, but it is unsystematic and short-time oriented. If the number of product derivates increases, it is almost impossible to select the best matching template for a new product in the family tree. Systematic reuse is one of the goals of product line development, meaning that the components must not only be copied but actually shared among several subprojects. But the product will not only consist of the shared requirements for the whole product line, some of them will be specific to one product. Commonality

0-7695-2888-0/07 $25.00 © 2007 IEEE
DOI 10.1109/SPLINE.2007.15

IEEE computer society

and variability are the magic words that modern product development has to deal with.

In this area of conflict between commonality and variability, general and specific requirements, requirements management for product lines has special needs. The differences between one product derived from the product line and another may consist in:

- changed requirements
- obsolete requirements
- additional requirements

The single fact that requirements can change and different versions of the same requirement are part of two derivates of a product makes high demands on requirements management tools and their configuration management functionality.

The paper is the result of a working group and bases on research findings of a previous working group identifying essential challenges in the context of requirements engineering and software product lines. In [7] major challenges in product line engineering that were the outcome of the previous working group are presented (excerpt):

- justification of the platform approach as a process model by a cost/benefit-analysis
- independent platform team
- cooperation between platform and product development teams
- proof of justification of the platform team
- communication overhead
- configuration management
- influence of the architecture on requirements negotiation
- effective tool support

The working group 'Requirements Management Tools for Product Line Engineering', consisting of the authors of this paper was set up in the context of the Requirements Engineering section of the German Computer Society GI (Gesellschaft für Informatik) in January 2004 and met eleven times until the end of 2006. We decided to investigate the challenges of effective tool support for product lines for the following reasons:

- Until now there does not exist any systematic analysis of requirements for product line tools.
- Tools play the key role when installing a new approach (like product lines) because tools are a cross-sectional issue. If people are convinced of the effectiveness and usability of the tools then they will also be more willing to support the overall process – which is in fact represented by the tools.
- We see product lines from a practitioner's point of view and useful and usable tools are of a high importance for us

In existing methodical investigations on the product line approach, tools play a minor role, but to bring a product line approach to live in practice, tools become crucial. The paper is oriented at practical issues and tries to support product line approaches in industrial settings. The tool requirements we collected are quite abstract and don't describe technical solutions. They are thought of as a list of demands that a tool developer can take as a starting point and that can be refined. The motivation of this working group was to initialize a discussion on this (often neglected) topic and to push the convincing product line approach one step forward into the direction of a wide-spread industrial use.

Some related investigations that also look at tooling questions and that touch the area of software product lines are:

1. The Software Engineering Institute (SEI) at Carnegie Mellon University conducted a survey in 2002 asking 31 companies about their strategies for dealing with product lines, which tools they used for product lines and how they used them:
http://www.sei.cmu.edu/publications/documents/02.reports/02tn017.html
2. In 2003, the SEI held a workshop that 'explored the area of tool support for product lines with representatives from technically sophisticated organizations, having direct experience in software product lines':
http://www.sei.cmu.edu/publications/documents/00.reports/00tr002.html
3. The International Council on Systems Engineering (INCOSE) has a requirements management tools survey site dating back to the 1990s. They have published and still maintain a table evaluating some 30 requirements management tools along with about 15 detailed attributes. However, since the vendors are evaluating their own products, the results of this evaluation should be treated with caution: http://www.paper-review.com/tools/rms/read.php
4. The Atlantic Systems Guild offers 'a survey of requirements engineering tools [that] gives you a review of most of the current tools.' It covers a broad selection of requirements management tools and is frequently updated. However, no specific product line aspects are covered in this list:
http://www.volere.co.uk/tools.htm

None of these investigations focuses on the problems of requirements management for product lines and none of them went so far to derive concrete requirements in order to improve existing tools. This is our goal with this paper. The following sections give an

overview of the research approach of the work group and the main research findings, especially the list of requirements for requirements management tools. The research approach we followed was the following:

1. Analysis of existing publications: we analyzed the investigations given above and further publications like [3] but did not find adequate requirements
2. Collection of real usage scenarios for requirements engineering tools in a product line environment (Section 2)
3. Derivation of product line requirements (Section 3)
4. Proposal for functionality in requirements engineering tools regarding product line needs (Section 3 and 4)
5. Benchmark of existing requirements engineering tools regarding these requirements (Section 5)

Section 6 concludes the paper with remarks on how interested readers and experts working with requirements management tools can contribute to the presented work.

2. Industrial use scenarios

In this section, we shortly introduce the main results of industrial scenarios of the adoption and use of tools in the context of product line engineering. The collected scenarios (please refer to [1] for a complete description of all investigated scenarios) are not fictitious but describe real practice based on daily routine of members of the work group and reflect real problems in the context of product line engineering. All the scenarios were gathered by experts who use the tool in their daily work. The documentation of the scenarios helped us in deriving new requirements, detailing requirements and prioritizing them with respect to product line engineering. (Section 3).

In the following, three exemplary usage scenarios are outlined; scenarios investigated by the work group but not presented here address the usage of DOORS on the automotive OEM side and usage of RequiLine [5] (both of these scenarios are research scenarios rather than actual practice scenarios):

Using CaliberRM at hp's OpenView Business Unit

The OpenView organization develops its product line concurrently at different locations around the world. OpenView products are typically multitier products (UI clients, management servers, database servers and agents). The products support a wide range of operating-system platforms. With time, the following challenges became increasingly apparent:

- Products started to overlap in functionality.
- Customers who bought more than one product faced with consistency and efficiency problems.
- Development and maintenance costs rocketed.

Given this situation OpenView's management decided to reengineer the products into a more tightly integrated product line. Requirements engineering and management was one of the main improvement areas. The Volere Requirements Template (see [6]) was selected as a standard requirements structure for all projects. Borland's requirements management tool CaliberRM [2] was chosen to support the distributed development needs of the organization. The tool was customized to ensure that every project had the same requirements structure following the Volere Template. The basic principles for the requirements process and tool were:

- allow broad sharing of information
- promote consistency to support efficiency, sharing and reporting
- enable flexibility
- allow individual empowerment

Using DOORS at Harman/Becker

Developing automotive embedded systems as a supplier for a multitude of car manufacturers asks for a comprehensive product line approach. In most of the cases system specifications are handed over by the manufacturers to the suppliers as textual requirements descriptions. In the recent years it can be observed that car manufacturers tend to provide the requirements specification in the format of the tools the specifications are developed with. The requirements management tool that is often used by (German) automotive original equipment manufacturers (OEMs) is Telelogic DOORS (cf. [8]). At first sight, a single tool might be expected to produce a single exchange process. Unfortunately a single tool does not result in a single requirements management and exchange process.

To avoid unnecessary parallel development, a common platform has recently been developed to make available basic functions for all customer-specific development projects. From a requirements management view, two tasks have to be performed:

- identification and description of common requirements for the platform
- matching customer-specific requirements to the platform requirements

The complexity of this task grows owing to the continuous changes in the requirements for both the platform and the customer-specific systems.

Using IRqA at ContiTemic

Similar to the scenario described for Harman/Becker, the challenge of the automotive supplier ContiTemic posed by requirements management of several product lines for different customers was handling many reusable sets of requirements. Tool support is needed to tackle the following:

- General rules provided by customers (nonfunctional quality requirements, environment requirements …) are valid for all projects of this customer.
- General internal rules (quality rules, development process rules …) are valid for all projects (of all customers).
- Product line spanning requirements are valid for each project of this product line.
- Project specific requirements are only valid for this project.
- Variant specific requirements are only valid for the respective variants of a product (in a project).
- Type-or-model based requirements are valid for all products of a special series type or model of a car.

The current strategy is to solve these challenges using a single database in IRqA [4] containing all requirements and by constructing projects as sets of requirements.

3. Requirements for requirements management tools

Based on the above industrial usage scenarios , we derived requirements for requirements management tools for software product lines. Our aim is to provide the requirements engineering community, and especially requirements management tool developers, with a catalogue of current requirements to adapt their tools to make them suitable for product line engineering.

This section decribes how the requirements have been derived, and explains those that are specific to product line development.

3.1 Starting point

The basis for the requirements catalogue is a detailed description of tool requirements for single product developments, written by Hoffmann et al. in [3]. Additionally to the requirements in the paper, they developed a comprehensible requirements catalogue for single product development tool. This catalogue comprises about 100 requirements, which are hierarchically structured. The top level serves as a grouping by stakeholders, the second level contains categories, which group related requirements:

- **Requirements categories from tool users**: information model, views, formatting, multimedia and external files, documentation of history, baselining, traceability, analysis functions, tool integration, import, change management and comments, document generation, collaborative work, checking out for offline use, web access
- **Requirements categories from project administrators**: users, roles and rights, size restrictions, workflow management, extensibility
- **Requirements categories from tool administrators**: database, encryption

Each individual requirement within these categories was prioritized by the authors.

In our approach we took the existing catalogue as a basis and adapted it in as described in the following way in order to make it suitable for product line development:

1. The documented requirements were revised and partially reformulated.
2. Requirements that become relevant in the context of product line engineering and in our own scenarios have been added to the catalogue. About 20 new requirements for the existing categories and three new requirements categories ("configuration management", "discussion support" and "priorities") have been added and filled with requirements.
3. The requirements are (re)prioritized as follows: for each requirement, it has been defined whether it is equally important, more important or most important in a product line context compared with single product development (cf. Section 4).

The full catalogue of requirements and their priorities can be found in [1]. The following subsections describe the newly elicited requirements and the prioritization. The requirements titles are set in italics font.

3.2 Requirements specific to product line engineering

Most new requirements specific to product line engineering (11 in total) belong to the newly introduced "configuration management" category. However, the requirements in this category mainly deal with issues that are of similar importance as in single product development. In the initial requirements list they were contained implicitly in other categories. The first subset of these new requirements deals with *basic version and configuration management* like object identification and versioning. These requirements are relevant in every kind of software development. *Support for baselining of the requirements database* was considered

more important in a product line context. The focus here is to be able to baseline arbitrary parts of the database in order to capture easily the state of several projects/products at once.

In a similar fashion, *support for multi project/multi product status and progress reporting* was considered a necessary new requirement for product line tools.

Also important, but not covered in the original requirements list, is *good support for tool-based communication among users*. In product line engineering, artifacts have to be discussed more intensively across a larger (and often distributed) group of people. An integrated but relatively informal type of discussion support is considered necessary in addition to the formalized change process, which the tool should support anyway. Examples are wiki-like discussions or forum discussions, which are directly linkable or linked to a specific (set of) requirement(s).

In order to break down work on different products, the *definition of arbitrary, named subsets of requirements* is also a more important issue for product lines because the number of requirements is usually higher than in single product developments and some partition can be helpful.

Various additional changes have been made to existing requirements to clarify their meaning. Table 1 shows some example requirements with a high product line priority. The points were given by the workgroup during the prioritization process (see section 4).

The following two subsections address those newly added requirements categories that are of highest importance to product line development: requirements specific to variability modeling and relationships to product and project management.

3.2.1 Requirements specific to variability modeling. We added five requirements that are related to variability. It is important that the tool supports *shared multi project and multi product information models* in order to provide a consistent modeling infrastructure for related projects or products that are part of the same product line development. A simple information model copy when a new derived project starts is not sufficient here because changes made afterwards have to be made individually to all related projects, which can be problematic if there are a lot of them.

Additionally, the information model provided by the tool must support the *expression of variability and variation.* In other words there must be ways of expressing variation points, rules for variation point instantiation and the description of relations between product variants in the tool's information model.

(Defined) *variants must have a first class representation* in the tool, which allows the addition of meta-attributes to the variant. Examples of such meta-attributes are a list of stakeholders for this specific variant (customer, account manager, etc.) or links to other variant-specific artifacts.

Given an information model that provides support for variability definition, a tool needs not only offer support for variant representation but must also *support the variant creation process* using the stored variability information. Dependencies between variants must be expressed, e.g. it should be possible to check, in which variant a requirement is realized or to compare variants with each other.

A similar requirement is the *need for multidimensional prioritization of requirements* in the tool with respect to different stakeholders. This can be used for tracking of individual realizations of several products using the same requirement. Use cases for such functionality are manifold. For instance, one project might decide that a feature described in a requirement is a "nice-to-have" item, while it might be essential for other projects.

3.2.2 Relationships to product and project management. The management of variability in requirements is not a task in a delimited area of requirements engineering. In fact, product and project management must be considered major stakeholders within requirements management. Both need special information about variability.

Product management:

One of the main tasks of product management is to plan and control the process of product development. Within this context, requirements engineering is a consumer of information produced by product management (e.g. new market-driven requirements) and a supplier of information needed by product management to make strategic decisions in product development.

The following questions therefore come to mind:

- Should the new product be developed from scratch or should it enhance an existing product?
- Which relationships exist between the new product and existing products?
- Can the basic functionality be conserved while new features are implemented?
- Should the new product be a variant of an existing product?
- How do the changes influence the product life cycle?

These are just a few of the questions that need to be considered during product management, but they are fundamentally related to requirements management in

the context of product line engineering. How can a product manager answer these questions without knowing the requirements? In addition to the number of features, the number of variants must also be considered, to make accurate cost estimation.

Furthermore, not only the development and product costs are influenced by the number of variants, but also the costs that must be considered during the product life cycle, e.g. update effort, maintenance costs and staff training.

Requirements engineering must therefore *support the product management by providing information about dependencies between features*. The more complex the requirements are, the more adequate tool support is needed to facilitate the management of information.

Project management:

The task of the project management in the context of software development is to plan and control the project. The main focus is on the functionality that should be provided by the new product. A change in functional requirements has a strong impact on the project plan.

If products are being developed in the context of a product line, multiple projects often use a common set of requirements concurrently. Without planned coordination of the projects, multiple implementations of the same requirement will be the result, which will lead to an enormous overhead in development. Since the additional effort consumes additional resources, project managers have even less time for coordination between the projects – a vicious circle.

Again, the information about dependencies between requirements across products must be provided by requirements engineering. Integration of requirements engineering can help to distribute the different tasks across projects and therefore to optimize the use of resources. The main advantage can be achieved by verification and validation. These two tasks are typically performed half-heartedly at the end of a project.

Requirements engineering must thus be based on an adequate tool to support project management by providing the necessary information. Furthermore, it is important that information in projects be swiftly available to enable effective control of projects.

To summarize, *the coordination of product, project and requirements management is necessary to effectively handle variability in requirements*. Adequate tool support is essential.

4 Prioritization of requirements for the context of product lines

This section presents the prioritization of tool requirements defined by the working group. The starting point was the list of tool requirements for single product development and the associated prioritization suggested by [3]. The working group rated each tool requirement with regard to product lines (see Figure 1). Each tool requirement from single product development was rated as (1) equally important, (2) more important, or (3) highly important in a product line context. The working group also identified further requirements specific to product line development, as described above, and rated their importance.

The prioritization was derived by a vote among the members of the working group. It was validated through subsequent discussions. As a result of this approach, some priorities are not unique. This is due to the different backgrounds of the working group members in terms of application domain and project type. This paper of the working group's results documents the variation in priorities, allowing the reader to make a well-informed evaluation of the proposed requirements list.

This section describes the peculiarities of each category of requirements for requirements management tools listed in [3] in a product line context. Figure 1 visualizes the average priorities given by the group members

Table 1 Example Requirements

	Category	Description	Prio (single sys.)	Points	max. Points	rel. Prio SPL
1.3.1.	Views	The tool must support multi-project and multi-product requirement views.		40	40	1,00
1.4.	Change Management	The tool must support change management. This can either be done by the tool itself or the tool should provide a suitable interface that conforms to the following requirements.	++	28	40	0,70
1.7.5.	Traceability	Links must connect any objects in the database, not only in the same subset (module, project, etc.)	++	28	40	0,70
1.8.2.	Analysis Functions	The tool should provide information about status and progress of multiple projects/products.		40	40	1,00
1.12.	Collaborative Work	It must be possible for many users to work on the same data at the same time.	++	35	35	1,00

for each requirement, labeled by requirements categories. To improve readability, the requirements are represented by diamonds. The sectors represent the requirements categories and the concentric circles represent the priority of the requirements in the context of product lines. The innermost circle contains requirements that are most important in the product line context, the outermost circle contains requirements which are equally important compared to single product development.

An analysis of Figure 1 reveals that it is mainly requirements from the categories "information model", "views" and "analysis" that become most important in the product line context. A detailed analysis in extracts is given below (please refer to [1] for a complete analysis overview).

User requirements: Information model

Priorities of tool requirements are the same as for single product development. For product lines, there are two additional requirements, both of which are highly important:

- The tool must support multi project and multi product requirements management information models (highly important).
- The tool must support variability mechanisms (highly important).

User requirements: Views

Views-related requirements are more important, but only rarely much more important, than for single product development. They are particularly relevant with regard to the following aspects:

- definition of user-specific views
- availability of graphical diagrams
- configuration and rule-based control of views
- modification of views in the course of a project

For product lines, there are two additional requirements:

- The tool should allow views to be predefined for user roles (important).
- The tool must support multi project and multi product requirement views (highly important).

User requirements: Documentation of the history

For product line development, the following three requirements are slightly more important:

- All changes to the requirements must be tracked.
- All objects managed in the tool must be versioned.
- There must be a distinction between major and minor versions regarding objects.

User requirements: Traceability

Traceability is slightly more important for product lines than for single product development, in particular with regard to the following requirements:

- It must be possible to define attributes for links.
- It should be possible to create rules governing what kinds of objects must have links to what other kinds of objects.
- Links must connect all objects in the database, not only in the same subset (module, project, etc.). (important to highly important).
- The tools must feature a practical, user-friendly and concise graphical representation, and navigation of the traces (e.g. matrices, trees or graphs).

User requirements: Analysis functions

Analysis functions are slightly more important for product lines than for single product development, in particular with regard to the following requirement:

- The tool should allow inconsistencies in the link structure to be analyzed (e.g. find gaps in traces).

For product lines, there is one additional requirement:

- The tool should provide information on the status and progress of multiple projects and products. (highly important).

User requirements: Tool integration

Also these requirements are slightly more important to much more important than for single product development, in particular with regard to the following requirements:

- The tool must have open interfaces to other tools used in the development process and make information stored in them visible and linkable. (more important to much more important).
- Access rights to the external objects must be recognized. (partly more important).

5 Evaluation of tools

This section presents the approach chosen for tool evaluation and the main evaluation results. The evaluation is based on case evidence from the organizations participating in the workgroup. This might appear only fairly systematic at first sight. However, this approach is more rooted into practice than any other comparative evaluation of tool requirements we know about. The validity of company-specific experiences has been secured through intensive group discussions and comparisons with other organizations' experiences.

5.1 Explanation of the evaluation approach

The tool evaluation started with an identification of requirements engineering tools that had been available within the workgroup's participating organizations, and that were being applied for managing product line requirements. These tools were Borland CaliberRM, Telelogic DOORS, QA Systems IRqA, and RequiLine of RWTH Aachen.

Table 2. Evaluation values and their meaning

Value	Semantics
0	Don't know.
1	The requirement isn't supported by the tool.
2	The requirement isn't supported in a way that is suitable for day-to-day operations. Custom solutions (extensions) are required (and possible) to address the requirement.
3	The requirement isn't supported adequately but it can be addressed with organizational conventions to use the tool in a specific way.
4	The requirement is addressed well enough to do daily work.
5	The tool's implementation of the requirement is perfect (at least we cannot imagine a better way of doing it).

To structure the evaluation process in a way as objective and transparent as possible, we created an evaluation schema based on numerical values with associated semantics (see Table 2).

At first, it seemed that the criteria were sufficiently well defined. However, when trying to evaluate the tools, a few issues arose. The first issue was that a certain degree of subjective interpretation was still possible. Do to our time and resources constraints, we were not able to mitigate this issue (e.g., through assignment of fit criteria for each requirement). However, we believe that the approach chosen makes the decisions sufficiently transparent. It shall enable users of the rating to derive informed decisions.

Another reason for limited objectivitiy are the configuration and customization capabilities of the tools. Most tools can be extended to some degree to suit requirements for which they obviously were not designed initially. During the rating, it was allowed to apply the values 2 to 4 of the of the evaluation schema, in order to reflect the tool's customization capabilities.

5.2 Visualization

Once we had obtained the data, the challenge was to visualize the various dimensions:

1. Show the importance of the requirement with regard to product line support.
2. Show how well the tool addresses the requirement.
3. Be able to compare tools.
4. Visualize requirement categories (the table has a series of categories for requirements).

We experimented with various display formats and finally decided that spider charts are best suited because they address all the dimensions described above. Bubble charts and 3D column graphics failed to visualize all dimensions.

The following chart shows how well the analyzed tools address the specific requirements. The categories "view", "analysis functions", "information model", "user rights" and "roles" and "configuration management" have been chosen because requirements belonging to these categories are most important in a product line context.

The red line indicates the product line priority of the specific requirements. Using this line, it is easy to evaluate how well the respective tools perform. Figure 2 gives an exemplary visualization of the evaluation.

5.3 Results

None of the investigated tools (DOORS, CaliberRM, IRqA, RequiLine) supported all or most of the presented requirements best, but each tool has its strengths and weaknesses. In order to decide which tool meets best the demands of a specific organization or development approach, one has to decide his or her specific prioritization of product line tool requirements using the developed requirements list.

For an overall comparison of the investigated requirements management tools regarding their suitability for product line development, we have made the following findings:

- All tools support views more or less in a way that is needed by product line engineering. Most of the tools have a configurable view concept, but they do not support multi project views. No tool allows views to be predefined for user roles.
- The analysis functions of the tools do not support product lines adequately. Especially the requirement on status and progress analysis of multiple projects that is very important in product line context is not fulfilled by any tool.
- All tools have an explicit information model but only one of the tools supported variability mechanisms in the information model. For all tools, it is possible to somehow define a multi-project and multi product information model.
- All tools have configuration management support, but the tools miss to fulfill the requirements that are important for product line engineering. The creation of a new variant and the tracking of dependencies between variants is only partially supported by one tool and not at all by the others. On the other hand, all tools but one support baselining (fixing a requirements set at a given point in time).

Overall, one can conclude that the available tools can in principle be used for product line development,

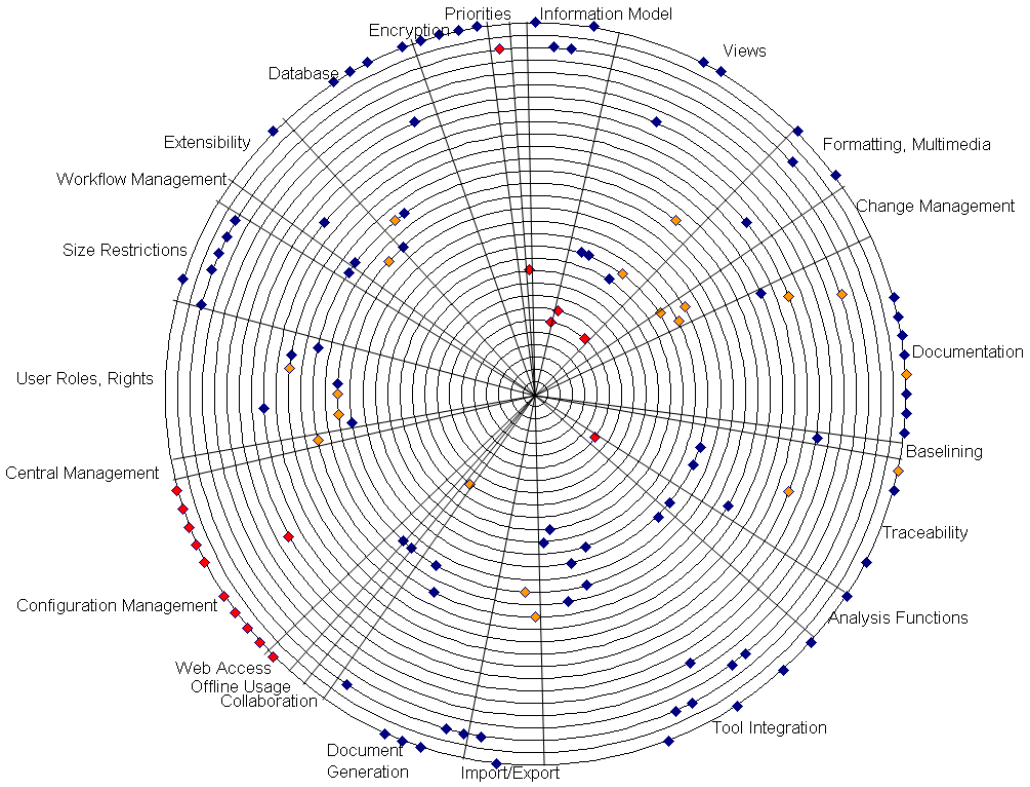

Figure 1: Prioritization of requirements in the context of product lines

but they impose severe limitations or cause extraordinary high effort for customization and work-around solutions.

6. Conclusion

In this paper, the requirements for requirements management tools were gathered and prioritized. The result is a comprehensive analysis of requirements and requirements management tools in the context of software product lines based on practical experience. The paper gives an overview of the key requirements – some are newly introduced to support product lines. Especially scalability is the main driver to identify requirements or to prioritize requirements in the context of product line engineering, because in this case a huge set of data items must be handled.

The presented analysis and the results are so far unique. This paper is intended to direct the attention of both researchers and tool developers to the current problem of inadequate requirements management tools for software product lines.

Today's software product lines have such complex variability that their handling must be supported by tools, otherwise a systematic approach would not be possible. The same holds for requirements management of software product lines. The evaluation of requirements for requirements management tools presented in this paper revealed that all requirements management tools currently used in industry need to be improved if they are to be successfully used in a product line setting. Nevertheless these tools are used in practice and are also used in product line settings and as the scenarios described in this paper show, they are also used successfully – but this is only achieved because workarounds are in place: either the tool itself is enhanced or related processes bypass shortcomings of the tool. But this situation is not satisfying, because especially small companies need to rely on all-embracing tool support because enhancements or heavy processes to overcome shortcomings of tools are too expensive.

Mature software development that is geared to product lines requires more than pragmatic, stop-gap solutions. Systematic software product line engineering calls for systematic tool support especially in the context of requirements management, mainly because one

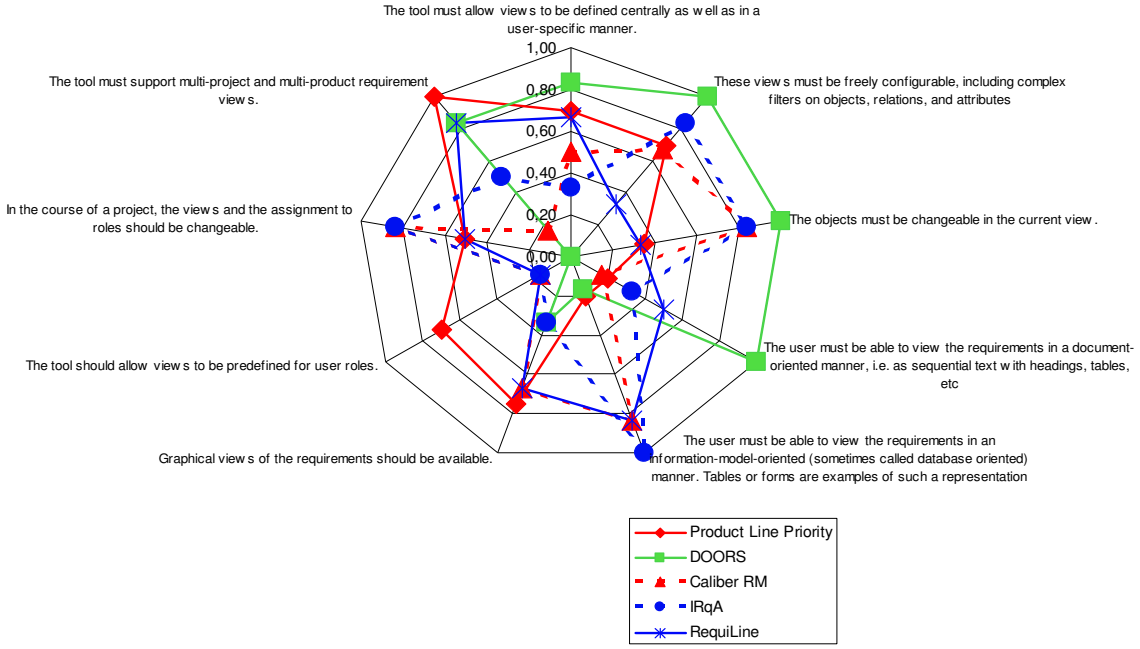

Figure 2: Evaluation overview of the category "View"

is confronted with the task of handling a huge number of requirements for multi projects and multi products.

A reader with interest in tool supported requirements management for software product lines can gain the following from the paper at hand:

- Current tools can be used in product line settings as the described scenarios show. But this is only possible with a huge amount of extra effort for customization, extensions, and work-arounds.
- The list of requirements given in this paper can be a basis to implement requirements management tools supporting software product lines.
- Current organizational processes can be analyzed for actual requirements and the fulfillment of these requirements in different tools can be seen in the requirements list at hand. The adequate tool can then be selected.

To substantiate the findings of the paper, the working group has set up a website (http://www.gi-ev.de/fachbereiche/softwaretechnik/re-pl/) where interested people with practical experience in using one of the listed tools are invited to evaluate them on the basis of the requirements catalogue. The website also provides more detailed information about the working group, its members, and the detailed results.

7. References

[1] D. Beuche, A. Birk, H. Dreier, A. Fleischmann, H. Galle, G. Heller, D. Janzen, I. John, R. Tavakoli Kolagari, T. von der Maßen, and A. Wolfram, "Report of the GI Work Group 'Requirements Management Tools for Product Line Engineering'", Aachener Informatik Berichte, AIB-2006-14, ISSN 0935–3232, Aachen, 2006. http://www.gi-ev.de/fachbereiche/softwaretechnik/re-pl/

[2] Borland, Caliber website: http://www.borland.com/de/products/caliber/index.html

[3] M. Hoffmann, N. Kühn, M. Weber, and M. Bittner, "Requirements for Requirements Management Tools", *Proceedings of 12th IEEE International Conference on Requirements Engineering* (RE 2004), IEEE Computer Society, Kyoto, Japan, 6-10 September 2004.

[4] QA Systems, IRqA website: http://www.qa-systems.de/ html/deutsch/produkte/irqa/ irqa.php.

[5] Research Group Software Construction, RequiLine website: http://www-lufgi3.informatik.rwth-aachen.de/ TOOLS/ requiline/

[6] J. Robertson, and S. Robertson, *Mastering the Requirements Process*, Second Edition, Addison-Wesley, 2006.

[7] K. Schmid, A. Birk, G. Heller, I. John, S. Joos, K. Müller, T. von der Maßen, "Report of the GI Work Group 'Requirements Engineering for Product Lines'", IESE-Report No. 121.03/E, Kaiserslautern, November 2003.

[8] Telelogic, DOORS website: http://www.telelogic.com/corp/products/doors/index.cfm