Graph-based Visualization of Requirements Relationships

Philipp Heim[1], Steffen Lohmann[1], Kim Lauenroth[2], Jürgen Ziegler[1]
[1]*Interactive Systems and Interaction Design*
[2]*Software Systems Engineering*
University of Duisburg-Essen, Germany
{philipp.heim, steffen.lohmann, kim.lauenroth, juergen.ziegler}@uni-duisburg-essen.de

Abstract

Understanding the relationships between requirements is important in order to understand the requirements themselves. Existing requirements management tools mainly use lists, tables, trees, and matrices to visualize requirements and their interrelations. However, all these visualization forms have a limited capability to show multiple relationships of different types. In this paper, we propose to extend traditional requirements analysis and management by a graph-based visualization that allows to represent multidimensional relations in a direct and flexible way. In particular, we propose a special presentation form that enables the exploration of requirements along their relationships and facilitates understanding of dependencies between requirements.

1. Introduction

Current requirements management tools such as *Telelogic Doors*, *IrQA*, or *IBM Requisite Pro* mainly provide spreadsheet-like user interfaces that use lists, tables, trees, and matrices to visualize requirements and their relationships. The combination of these visualization forms has proven to be valuable for managing large sets of requirements. However, these visualizations have the following shortcomings when used for analyzing requirements relationships:

- *Lists* are valuable for presenting large datasets in a clear and linear way when combined with sophisticated scrolling and paging techniques. At the same time, the linear way limits their expressive power to a single dimension in which to arrange requirements at a time.
- *Trees* provide a hierarchical visualization and thus are able to show two dimensions at a time. This doubles their expressive power compared to lists but still lacks the capacity to show multi-dimensional requirements relationships.
- *Matrices* can be used to show multi-dimensional relationships between requirements. However, their rigid way to arrange the matrix hinders the easy extension of the visualization and hence the exploration of requirements and their relations.

Against the background of these limitations in expressing requirements relationships, graphs seem to be a suitable extension to the visualization forms that are typically used in requirements management tools. A graph allows for flexible visualization of multidimensional relationships between requirements by representing requirements as nodes and relationships between requirements as edges. So far, graph-based requirements visualizations are solely applied in specific cases: for instance, some approaches use graphs to visualize goal models (e.g., [8], [10]); others provide graph-based visualizations to show requirements traceability (such as the *Traceline* extension for *Telelogic Doors* [2]). However, existing tools do not provide graph-based visualizations to support multidimensional exploration and analysis of various requirements relationships, thus far.

One reason might be that graph-based visualizations usually do not scale well to large datasets because their presentation tends to result in a complex structure that is hardly manageable or understandable by the user (cf. [7]). A graph-based visualization is particularly valuable if a small set of highly interrelated requirements has to be visualized. Therefore, we propose a focus and context approach that uses a graph-based visualization not for the presentation of a whole set of requirements but for a limited set that is related to the requirement of interest.

The remainder of this paper is structured as follows. In Section 2, we give a more detailed motivation for the need of an improved visualization for requirements relationships from the context of a current research project. In Section 3, we describe our graph-based visualization approach. We close the paper with a summary and outlook on future work in Section 4.

2. The Need for an Improved Visualization of Requirements Relationships

We have experienced the need for an improved visualization of requirements relationships within the context of the *SoftWiki* project [8]. A central goal of this project is the elicitation of requirements from a large number of geographically distributed stakeholders. For this purpose, we developed a tool set within the project that enables stakeholders to collaboratively collect and discuss requirements [5]. We recognized that clarifying relationships between requirements is a key to success for effective collaboration, avoidance of misunderstandings, and prevention of redundancy. Furthermore, the visualization of related requirements is highly valuable for exploration and analysis of requirements that have been collaboratively expressed by many stakeholders. In order to fully understand a certain requirement, it is important to understand in which way it interrelates with other requirements (cp. [6]).

Figure 1 shows a part of the user interface of SoftWiki's collaboration platform. Since all the requirements and relationships cannot be shown properly on one screen, the exploration is executed in top-down manner similar to other requirements management tools. The stakeholders start from a list of all requirements and navigate to a subset they are interested in by using different access points and ways of navigation, such as a hierarchical topic structure (visualized in a tree view, Figure 1, A), certain keywords (visualized in a tag cloud, Figure 1, B), or a full-text search. Having found the requirements of interest in the result list (Figure 1, C), the stakeholders can select one of them to make it the focus requirement and see it in more detail (Figure 1, D).

In order to present the focus requirement within its context, related requirements are displayed below the detail view (Figure 1, E). We differentiate the following top-level relation types that can exist between requirements:

- *User-defined relations*: relations that have been explicitly set by a stakeholder, such as "conflicts with", "details", "depends on", or "redundant".
- *Content relations*: automatically derived relations based on the text similarity regarding the title and description of the requirements.
- *Shared metadata relations*: automatically detected relations based on shared metadata (such as author, keywords, organizational unit…)

The related requirements are initially visualized as a list. By clicking on one of the related requirements, this becomes the new focus requirement. In comparison to the top-down exploration that allows to globally navigate from one set of requirements to any other set, the local navigation follows the relations that are ex-

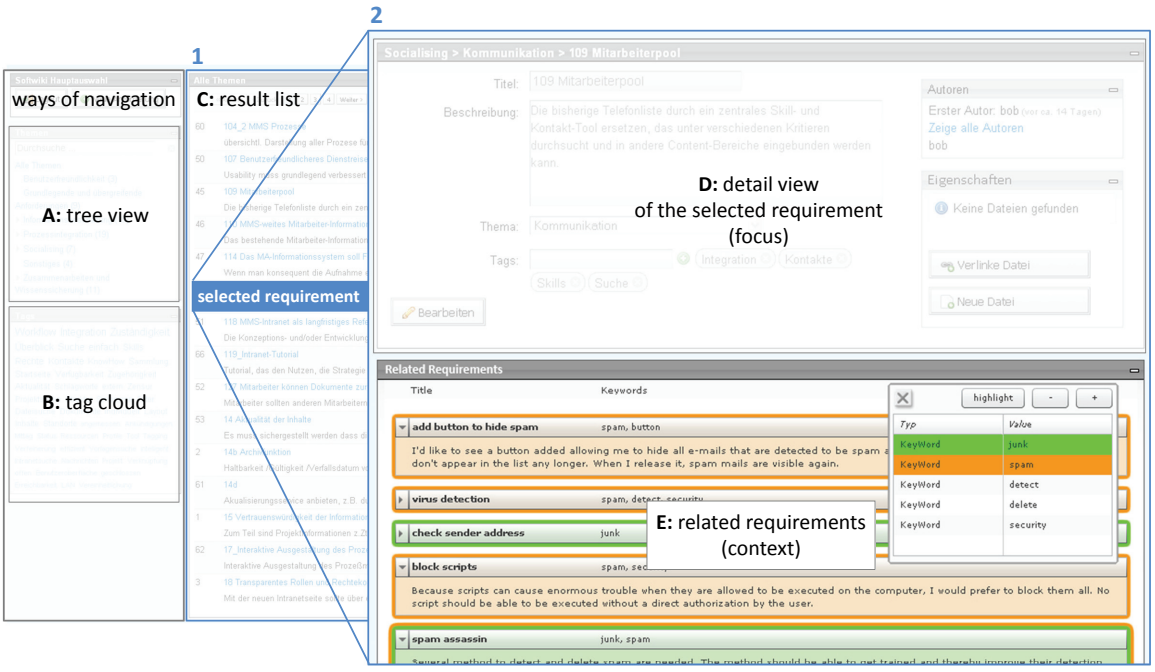

Figure 1: User interface of the SoftWiki collaboration platform

plicitly defined or automatically detected between requirements. In Figure 1, the user has chosen that only related requirements are shown that share certain metadata with the focus requirement (cp. the overlay in Figure 1, E). Furthermore, the user has highlighted those requirements that share the metadata "keyword:spam" and "keyword:junk" by the colors green and orange.

Whenever a related requirement is selected and shown in the detail view, the list of related requirements is updated accordingly. This way, information about requirement relationships is partitioned over multiple screens, placing substantial cognitive load on the user to keep track of the requirements and how they are interrelated. To reduce the cognitive load and facilitate understanding of relationships, we propose a graph-based visualization of related requirements that can be used as an alternative to the list visualization.

3. Graph-based Visualization

A graph-based visualization can be expanded node by node and hence presents information in a single visualization that is otherwise distributed over multiple screens. This prevents the stakeholders from getting "Lost in Hyperspace" [3] and facilitates them to gradually explore the related requirements. Relations that exist between requirements can be represented as labeled edges helping the stakeholders to comprehend multiple relationships at a time.

3.1 Visualizing Different Relation Types

Since requirements might be connected by an arbitrary number of relationships that can be of all three top-level types mentioned above, a proper visualization of these relationships is required. In order to make it easy for the stakeholders to see how requirements are interrelated, the different relation types are also visualized differently.

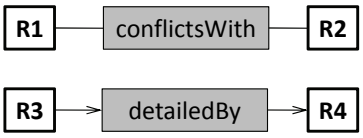

Figure 2: User-defined relations (undirected and directed)

User-defined relations are represented as directed or undirected labeled edges depending on the symmetry of the relation. For instance, the "conflictsWith" relation between requirement "R1" and "R2" in Figure 2 is a bidirectional relation and is therefore represented as undirected labeled edge. The "detailedBy" relation is in comparison unidirectional and is hence represented as directed labeled edge (Figure 2, bottom).

Content relations, which are based on automatically calculated text similarities between two requirements, are bidirectional and therefore represented as undirected edges that are labeled with "similarTo" (Figure 3). Depending on the degree of similarity, the edge is differently weighted, visually represented by its thickness. (The mechanisms used for calculating the text similarity and thickness of the edges are beyond the scope of this paper). To visually distinguish content relations from user-defined relations, the labeled edge is not filled with a color.

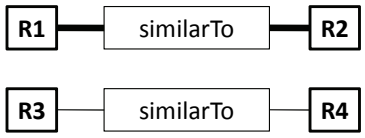

Figure 3: Content relations (with different weights)

In contrast to user-defined and content relations that both represent direct connections between requirement pairs, *shared metadata relations* represent indirect connections between requirement pairs based on shared metadata. For instance, the requirements "R1", "R2", and "R3" in Figure 4 all refer to the keyword ("KW") "spam". This commonality constitutes an indirect connection between all three requirements. Metadata that is only referred to by a single requirement, in contrast, is not of interest when analyzing relationships between requirements. Consequently, such metadata is not visualized in order to reduce the complexity of the graph (such as the keyword "security" in Figure 4).

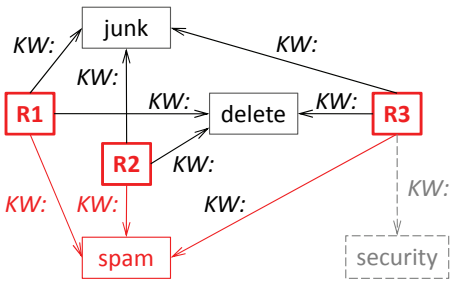

Figure 4: Indirect connections based on shared metadata

To better comprehend indirect connections based on shared metadata and to identify similarities, conflicts, and dependencies between requirements more easily, indirect connections are visualized as direct relations in

our graph. This direct representation by shared metadata relations between requirements is illustrated in Figure 5.

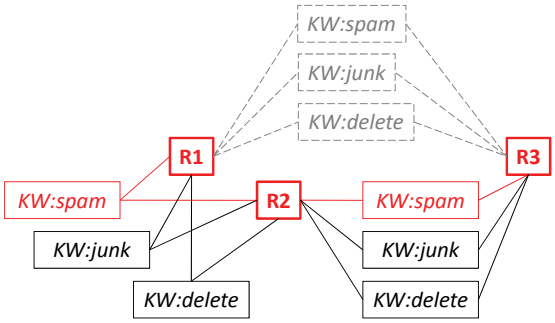

Figure 5: Visualization of shared metadata relations

However, the direct visualization of all connections based on shared metadata between all requirements of interest quickly results in a graph with a lot of edge crossings and hence does not facilitate but rather impede understanding (cp. Figure 5). This is because the number of relations grows exponentially with the number of requirements that share certain metadata.

To prevent edge crossings, we reduce the number of relations by arranging all requirements that share certain metadata in a chain that connects each requirement only with its predecessor and successor. All other edges are not shown any longer. For instance, the shared metadata relations between "R1" and "R3" are not directly shown in Figure 6 but in a transitive way via "R2". We call the resulting type of graph *ChainGraph* due to the chain arrangement of the nodes.

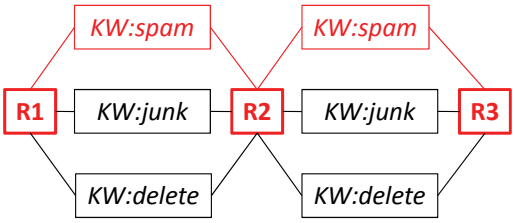

Figure 6: ChainGraph visualization of shared metadata relations

3.2 The ChainGraph Approach

The ChainGraph directly visualizes shared metadata between requirements by shared metadata relations that are organized in chains. Since requirements often refer to different metadata, they are connected by different chains that ultimately result in a network of requirements relationships (Figure 7). User-defined and content relations can easily be added to the graph; as they connect only two requirements at a time, they need not to get arranged in a chain.

In order to facilitate following shared metadata relations, the chains can be highlighted in different colors.

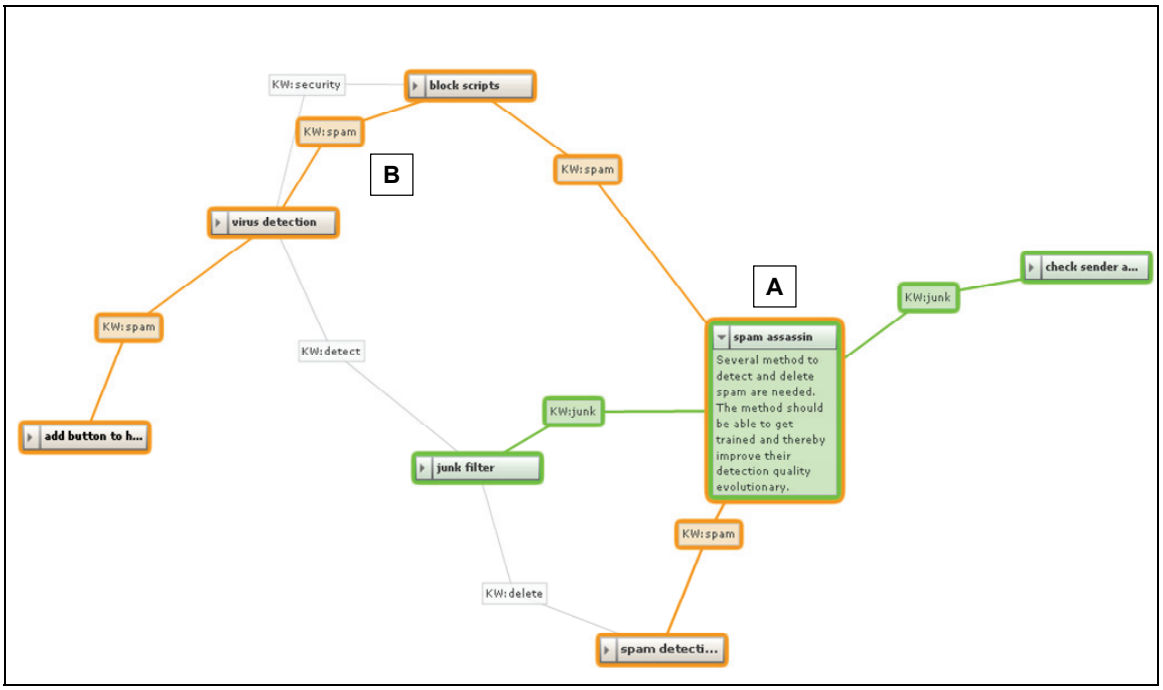

Figure 7: Application of the ChainGraph approach for visualizing shared metadata relations

In Figure 7, the two chains that are based on the shared metadata "KW:spam" and "KW:junk" are highlighted in green and orange, respectively. If a requirement interrelates with other requirements by more than one highlighted relation, it is surrounded by as many colored rings as there are relations with this requirement (Figure 7, A).

We use a force-directed algorithm [4] in order to layout the graph in an aesthetically pleasing way. The algorithm optimizes the positions of the nodes so that all edges are of more or less equal length and there are as few crossing edges as possible. Since even in simple settings the problem of automatic label placement turns out to be NP-hard [1], we treat the labels as additional nodes to get the placement solved along with the computation of the force-directed layout. This divides a relation between two requirements into two edges with the label as an articulated joint in between. That way, the shape of an edge gets more flexible and hence reduces overlapping when two requirements are connected by more than one relation (Figure 7, B).

4. Conclusion and Future Work

In this paper, we proposed a graph-based approach for visualizing requirements relationships. The approach allows for a flexible and extensible representation of multi-dimensional requirements relations and hence enables a better understanding of these relations. Together with a specific type of graph, that we call ChainGraph, we achieve the following advantages for large sets of requirements:

- *Single visualization:* Requirements relationships are displayed in a single visualization instead of being spread over several screens or windows.
- *Direct representation of relations:* Existing (also indirect) relationships between requirements are explicitly visualized as labeled edges.
- *Fewer number of crossing edges:* Labeled edges are treated as nodes and requirements that share the same metadata are arranged in a chain, thus reducing the number of edges as far as possible.
- *Following paths*: The flexible extensibility of a graph-based visualization in combination with highlighting capabilities and the ChainGraph approach provides sophisticated support for the analysis of relationships and multidimensional dependencies.

At the moment, the presented graph-based visualization is a standalone application prototype. Our current work includes a seamless integration of our approach into the SoftWiki project that was introduced in Section 2. Based on this integration, we plan to evaluate the benefits of the proposed visualization through experimental studies with the help of eye tracking. The goal of the evaluation will be to show that the graph-based visualization allows for faster understanding of multidimensional relationships between requirements than other visualization types such as lists or matrices. Furthermore, we plan to extend the approach to also consider the interrelations between requirements and other resources, such as software artifacts (e.g. test cases or design artifacts) or externally available domain knowledge.

5. References

[1] Christensen, J., Marks, J., and Shieber, S., "An Empirical Study of Algorithms for Point-Feature Label Placement", *ACM Transactions on Graphics*, 14(3), 1995, pp. 203-232.

[2] DOORS/TraceLine: Visualizing Requirements Traceability, http://www.telelogic.com/products/doors/doorstraceline

[3] Edwards, D.M. and Hardman, L., "Lost in Hyperspace: Cognitive Mapping and Navigation in a Hypertext Environment", *Hypertext – Theory into Practice*, 1999, Intellect Books, Exeter, UK, pp. 90-105.

[4] Fruchterman, T., and Reingold, E., "Graph Drawing by Force-Directed Placement", *Software – Practice & Experience*, 21(11), John Wiley & Sons, New York, 1991, pp. 1129-1164.

[5] Lohmann, S., Heim, P., and Lauenroth, K., "Web-based Stakeholder Participation in Distributed Requirements Elicitation", *Proceedings of the 15th IEEE International Requirements Engineering Conference (RE '08),* Barcelona Spain, IEEE, to appear

[6] Pohl, K., *Process-Centered Requirements Engineering*, John Wiley & Sons, New York, 1996.

[7] Schraefel, M.C., and Karger, D., "The Pathetic Fallacy of RDF", *Proceedings of the 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, USA, 2006.

[8] Sebastiani, R., Giorgini, P., and Mylopoulos, J., "Simple and Minimum-Cost Satisfiability for Goal Models", *Proceedings of the 16th Conference on Advanced Information Systems Engineering (CAiSE'04)*, Springer, 2004, pp. 20-35.

[9] SoftWiki – research project, funded by the German Federal Ministry of Education and Research (BMBF) – see http://www.softwiki.de

[10] Tran Van, H., van Lamsweerde, A., Massonet, P., and Ponsard, C., "Goal-Oriented Requirements Animation", *Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04)*, Kyoto, 2004, pp. 218-228.