Visualization of Feature Survival in Platform-Based Embedded Systems Development for Improved Understanding of Scope Dynamics

Krzysztof Wnuk[1], Björn Regnell[1], Lena Karlsson[2]
[1]*{krzysztof.wnuk,bjorn.regnell}@cs.lth.se, Lund University, Sweden*
[2] *lena.karlsson@dnv.com, Det Norske Veritas, Sweden*

Abstract

This paper presents a method for visualizing the scoping process in platform-based development of embedded systems. The proposed visualization shows the decision process of including or excluding features that are candidates for the next release. The presented visualization charts are evaluated in a large-size embedded system platform project. The evaluation indicates that the visualization of feature survival and scope dynamics can improve the understanding of the decision process of platform scoping in real industrial projects. Future work includes dealing with the relations between features and system requirements, improving user interaction as well as visualizing statistical measures of efficiency of the scoping process.

1. Introduction

One of the crucial decisions in platform based embedded systems development, where a common development project is a base for many products [1] is whether or not to select a set of requirements for a future release [10]. There are various factors that increase the complexity of scoping decisions, for example: decisions about new functionalities are made *a priori* with limited knowledge about their market value and implementation effort and they are often changed by key customers or management as results of late strategic decisions [11]. This paper presents a visualization support for understanding such decisions. The visualization support is evaluated in an industrial case study. In our study, the management is deciding upon not a single requirement but a bundle of requirements that is building up a new functionality in the product. This bundle in our case is called a *feature* and is an entity of which we can estimate market value and implementation effort and can be used for project scoping. In this rapidly changing environment the decision about the future has to be based on uncertain information, preventing us from making perfect decisions about the scope. Visualization of this decision process may be a valuable assistance and may enable management to see the overall picture of the scope. In this paper we have investigated how to get a simple but comprehensive picture of the current scope to understand the risk of taking too much features into the project and implications of this decision.

The paper is structured as follows. Section 2 provides background information about the context of our industrial case study. Section 3 gives a brief description of the methodology of gathering, processing and rendering information as well as the validation approach with industrial practitioners. Section 4 presents the results from the application of our visualization proposals in an industrial setting. Section 5 describes the results of the industrial validation and discusses limitations. Section 6 includes related work. Section 7 provides conclusions and ideas for future work.

2. Industrial Case Description

This section describes the environment in which the proposed visualization techniques were tried out. The visualizations are based on empirical data from an industrial project at a large company using a product line approach. The company has approximately 5000 employees and develops embedded systems for a global market.

2.1. Product line environment

The company develops a platform in several consecutive releases. Each platform release is the basis for one or more products that reuse the functionality and qualities of the platform. The first platform release has approximately two years lead-time from start to launch, and is focused on functionality growth and quality enhancements for a product portfolio. The following platform releases are shorter and more focused on adaptation to the different products that will be launched on the different platform releases. In order to be able to start the first release project as much as two years ahead of launch, it is necessary to build in some flexibility in the process. Therefore, there is a separate flow of requirements which were not known when the release project started. It is called a *secondary flow* as opposed to the *primary flow* which starts at the beginning of the release project. The secondary flow approach enables the project to start analyzing the functionality that is of highest importance in the primary flow and wait with functionality that appears later. The approach provides a balance between flexibility and stability in the projects. The secondary flow is connected to the product development project that builds on the first platform release. Therefore, many of the product requirements are entered in the secondary flow since they were not analyzed at the start of the project. Also late market requirements can be taken into account in the secondary flow.

2.2. Requirements Management Process

The requirements analysts groups in this company are called *Requirements Teams* (RT). They are responsible for eliciting and specifying requirements on system level within one or more technology areas. There are 20 RTs, with between 10 and 20 members in each, serving several parallel platform projects. The RTs hand over the requirements to the *Design Teams* (DTs), who designs and develops the software for the features. The company uses a stage-gate model with several increments [16]. There are *Milestones* (MS) and *Tollgates* (TG) to control the project progress. In particular, there are four MS for the requirements management and design before the implementation starts: MS 1, MS 2, MS 3, and MS 4. For each of these milestones, the project scope is updated and baselined. The milestone criteria are as follows:

- **MS 1:** High-level features are defined based on each RT's roadmap. At this stage the features usually contain a description, its market value and effort estimates. The features are reviewed, prioritized, and approved. The initial scope is decided and baselined per RT, guided by a project directive and based on initial resource estimates in the primary receiving DT.
- **MS 2:** Features are refined to requirements which are specified, reviewed and approved. One feature usually contains ten or more requirements from various areas of the products. The features are assigned to DTs that will take responsibility for designing and implementing the assigned features after MS 2. The DTs also allocate an effort estimate per feature.
- **MS 3:** The effort estimates are refined and the scope is updated and baselined. DTs refine system requirements and start designing.
- **MS 4:** The requirements work and design is finished, and ready to start implementation. The final scope is decided and agreed with the development resources.

The secondary flow starts approximately at MS 2 and is connected to the start of the product projects. The same MS and MS criteria as described above are used also for the secondary flow, and the two flows run in parallel until they are merged when the secondary flow MS 4 is passed.

The requirements are written in domain-specific natural language, usually containing 1-5 sentences with many special terms that require contextual knowledge to be understood. The abstraction level of a requirement can vary from detailed implementation-oriented descriptions to high-level customer-oriented descriptions, but are most often at a relatively high level.

2.3. Scoping Process

The project starts with a roadmap extraction, where the different RT groups look into their long-term roadmaps of high-level requirements, and extract information that is suitable for the coming platform project. The RTs have a project directive to guide the extraction so they know what functionality and qualities to focus on for this particular platform project.

The content of the high-level roadmap is then used as basis for creating features. Features are used to describe and decide which new functionality shall be implemented in a platform project. A feature groups requirements that constitute a new functional enhancement to a platform. A feature is thereby on such a level that it is possible to judge the market-value of scoping it in to a certain platform, and also the effort of implementing it. The market-value and effort estimates are obtained using a cost-value approach based on pair-wise comparisons [8]. The method provides the opportunity to calculate the *Return on Investment* (ROI) (dividing the market-value by the effort) and is used for decision making. The scope is decided based on the ROI in relation to the available development resources within the DTs.

All requirements and features are contained in a requirements database. There are approximately 25000 system-level requirements, some of which are grouped into several hundreds features (legacy requirements are not connected to features). Each requirement is described with a set of attributes such as Id, Name, Status, Source, etc. Each feature has attributes such as Name, Description, Justification, Scope, Market-value, Effort estimate, etc. The content of the database is baselined regularly.

The scope is controlled in a baselined document called *Feature List* (FL) contained in the requirement database. The FL is updated and baselined each week after decision in the *Change Control Board* (CCB). The CCB accepts suggestions for adding and removing features from the scope based on the input from the RTs and other stakeholders such as Product Planners, DTs, etc. There is a decision log, describing each de-scope of feature with a motivation.

The case in this paper investigates the first release of one particular platform project, including both *primary flow* and *secondary flow*. The project has just reached MS 4 and set its final scope. Some of the features in the database are already planned for following releases but since the focus was on the first release those were considered as *out-scoped* in the investigation.

The scope is changing drastically and frequently in the investigated project, creating a lot of turbulence. Project members are frustrated about the situation and feel that management is changing the direction of the scope without realizing the effect on the project. There is a need to analyze and understand the scope changes in order to improve the scoping process. In addition, there is a need to visualize the amount of effort that is wasted because of constant scope changes, and illustrate how the scope is reduced because of inaccurate effort estimates and unrealistic stakeholder expectations.

3. Methodology

In this section we describe the process of gathering and analyzing industrial data. As described in [3] we can consider a simplification of a visualization process as a sequence of the following steps: gathering, processing, pictorial rendering, analyzing and interpreting of data. In this section we focus on the first three stages. Our conceptual model considers visualization in a form of a two dimensional one page graph, where features can be seen as horizontal lines and states are expressed by different colors of the lines. This simple idea encapsulates both the survival of the feature and if we consider the whole set of features also the dynamics of scope changes over the project life-cycle. In order to create a mathematical representation of this idea we have created an exporter of the data from the *Scope* parameter for each feature in the FL document. In this data the information about including or excluding certain features is stored. In total we have extracted 81 data points, where each data point is one baseline of the FL document containing all features existing in it. The investigated baselines include between 300 and 600 features.

The next step was to process the data. Validation of gathered data with requirements experts resulted in a sampling policy to set the minimum time interval for baselines to 3 days, since the document was baselined more irregularly than once a week. The expert's explanation for this irregularity is that the decision process takes usually more than one step, where some additional information or estimates are added to prepare a full view for the CCB. Next, data was transformed into a representation acceptable by Matlab, which is used as rendering tool [5]. The transformation is based on a coding scheme, where each feature is mapped into one row, and each value of the *Scope* attribute is mapped to an integer value. The resulting matrix is sorted to show surviving features at the top of the graph. As a color scheme we have decided to follow what we consider a natural way of describing positive and negative states by drawing green lines for all features that were considered as in scope for a certain baseline and drawing red lines for ones that were considered as out. We have also used gray color to show the features that were present in the FL but with empty attribute values indicating yet undecided features. Our final data set consisted of 531 features over 39 data points (from the preliminary 81 data points described in the previous paragraph).

Our approach is similar to the model of transformation of data into graphical representation described by Haber and McNabb [4], where firstly the data is processed or filtered, secondly the Abstract Visualization Object (AVO) is defined and finally the data is mapped onto attribute fields of this AVO. In our case as attributes we consider the geometry (we consider the feature scope attribute as a horizontal line). We also use color as an attribute to represent scoping of the features by coloring them green while in and red while out. Another used attribute is the position since each feature has its unique and constant

Y axis value. The time attribute is mapped directly on the X axis with scaling according to the number of days between the baselines.

Before the actual visualization we conducted interviews with some RTs to gain understanding about the challenges of setting a realistic scope in the early phases of the project. The focus was on investigating the reasons, why the features that were in scope really early were then de-scoped just right before MS 4. We also discussed the features that appeared later on in the project, to find out why there were not known before.

After creating visualizations we showed and discussed our result with some RTs. These RTs were chosen because they had different scoping approaches. During these meetings, opinions about the proposed visualizations were gathered and proposals for future improvements were discussed.

4. Feature Scoping Visualization

This section presents results from applying our visualization approach to a large industrial set of 531 features. Our contribution includes two types of graphs: *Feature Survival Chart (FSC)* and *Feature Growth Chart (FGC)*. We also present results from a deeper analysis of two different RTs.

4.1. Feature Survival Chart (FSC)

This chart shows feature scope changes over time, which is illustrated on the X axis. Each feature is positioned on its specific place on the Y-axis so that the complete lifecycle of a single feature can be followed by looking at the same Y-axis position over time. The various scope states are visualized by

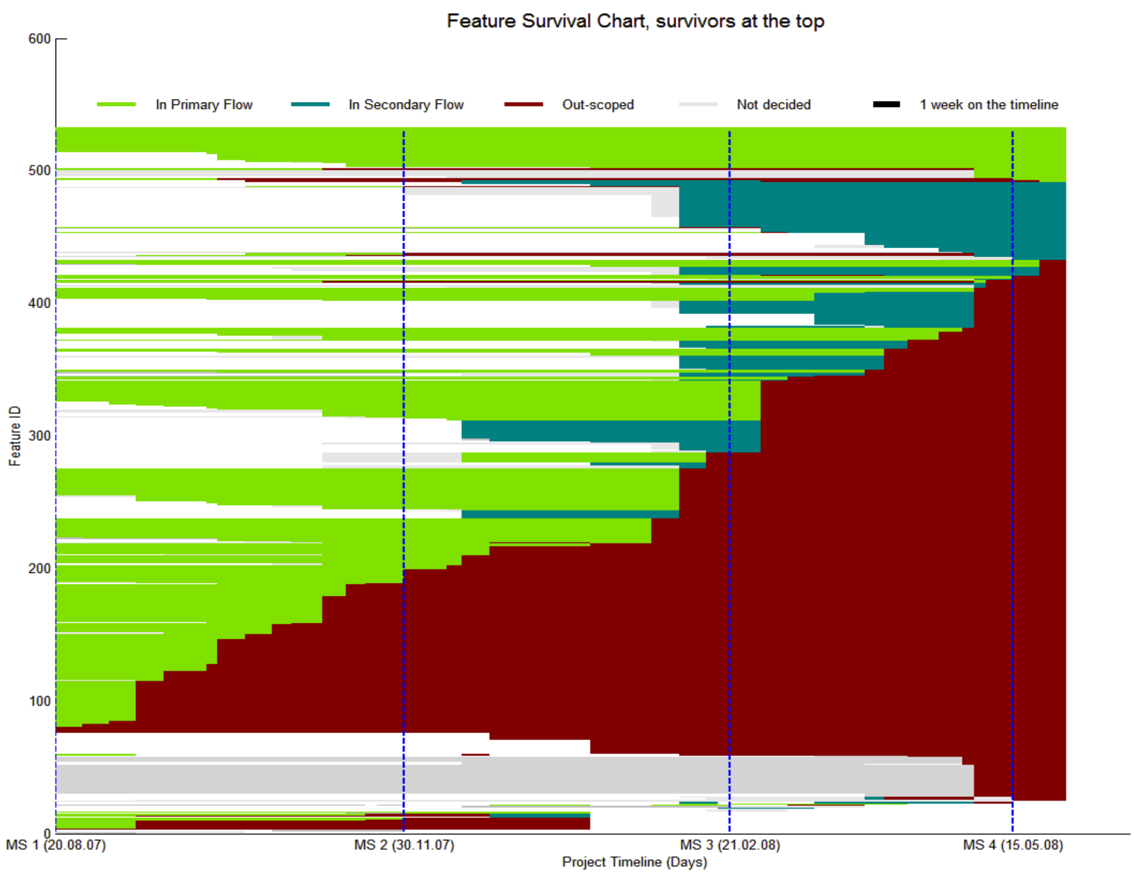

Figure 1. Feature Survival Chart. The red lines show out-scoped features. The green lines show features in scope (light green for primary flow features and dark green for secondary flow features). The survivors are placed at the top, as the graph is sorted on duration in scope from last baseline. The full size color picture can be found at http://www.cs.lth.se/home/Krzysztof_Wnuk/rev08/Feature_Survival_Chart.bmp

different colors. As a result, each scope change can be observed as a changing color. The colors are decided as follows: green for the primary flow, darker green for the secondary flow, red for out-scoped and gray for not yet decided features. In Figure 1 we can see the visualization of 531 features in 39 baselines, spawning almost a year of work. The data is sorted to put *survivors* (features that were in scope at MS 1 and are still in scope at MS 4) on the top and visualize the following aspects:

- The number of features in the secondary flow turned out to be larger than in the primary flow. See in Figure 1 the secondary flow after MS 4, where the area of dark green color at the last baseline is larger than the light green area.
- The dynamics of de-scoping can be seen in the FSC, where the features were de-scoped in bigger sets, rather than in single features. This sharp increase of out-scoped features emphasizes the extent and timing of decisions. We can easily spot three large scope reductions, one after MS 1, one before and one after MS 3.
- The effort spent on the features that have not survived to MS 4. An estimated indicator is the total size of the green area that changed to red before MS 4. If we compare this information to the size of the survivors' area we can construct valuable process efficiency measures. We anticipate that such measures can be used by management to assess the requirements process efficiency and address areas of improvements. Moreover, our additional statistical analysis of this project revealed, that the survivors were only 18 % of the total number of analyzed features. The more features survived, the more effective is the elicitation and market analysis process and less effort is spent on features that were not implemented. The earlier the features are de-scoped, the more effort is saved for other features and other parallel projects.

4.2. Feature Growth Chart (FGC)

The second type of visualization is the FGC presented in Figure 2 that also shows an overall view of the whole scope of the project but generated in a different way. This time we have visualized the number of features in a particular state as a function of

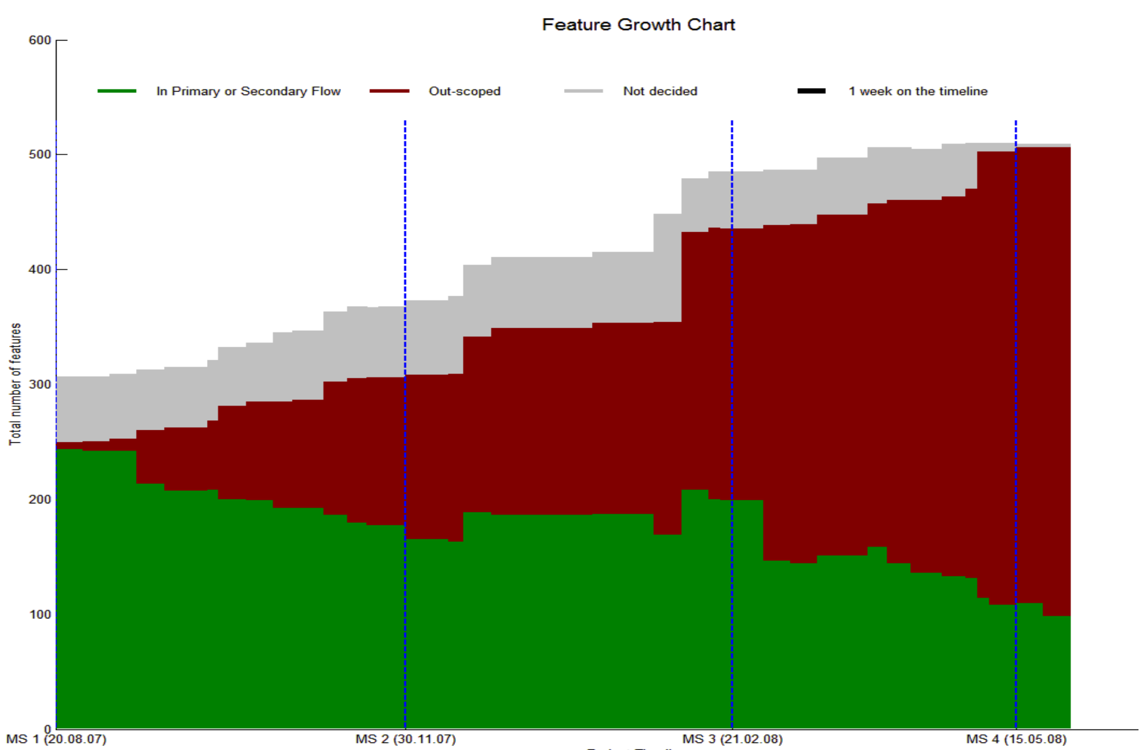

Figure 2. Feature Growth Chart. The green area represents the number of in-scoped features, red the number of out-scoped and gray the number of undecided. The full-size color picture can be found at http://www.cs.lth.se/home/Krzysztof_Wnuk/rev08/Feature_Growth_Chart.bmp

time. All 3 states were put one above each other to create a combined view, visualizing the total amount of features of each state at each point in time.

This view visualizes the total amount of features per data point. For this graph we consider features in both primary and secondary flow as in scope, here given a light green color. The out-scoped features are red, while the non-decided features are gray. This form of visualization reveals the following interesting aspects of the studied requirements process:

- The overall trends are (1) a growing amount of de-scoped features, (2) a decreasing amount of features in scope, and (3) a more or less constant number of undecided features. The changes in feature growth are sometimes abrupt.
- The ratio between in, out-scoped, and undecided features at the beginning and at the end of the project is very different. As we can see in Figure 2, this project had a much larger scope in the beginning compared to the final scope, and as a result had to go through a drastic scope reduction.

4.3. Graphs per Requirements Team

The graphs described in previous sections are useful for analyzing the scoping dynamics for the whole project, but also for coherent sub-sets of features. In order to visualize and understand the reasons and implications of a single decision we generated graphs, which include the data from only one RT each. The generated graphs gave us information that we could validate with the specific RTs in order to understand why certain feature was de-scoped and discuss the issues related to de-scoping, as well as address possible areas of improvements. We here present 2 examples of FSCs for RTs that followed different scoping policies which resulted in different sets of survivors. These examples demonstrate that FSCs can be useful for analyzing interesting differences among RTs.

4.3.1. Visualization of RT A. In the first RT example, showed in Figure 3, we can see several features put in scope at the beginning of the project that were later de-scoped in several steps. These steps included also adding more new features either to the primary flow or to the secondary flow. After a more in-depth consideration we observed the following phenomena:

- Several features were not considered in the scope until after MS 1, which resulted in shorter life time for those features.
- Most of the de-scoping is also done rather late in

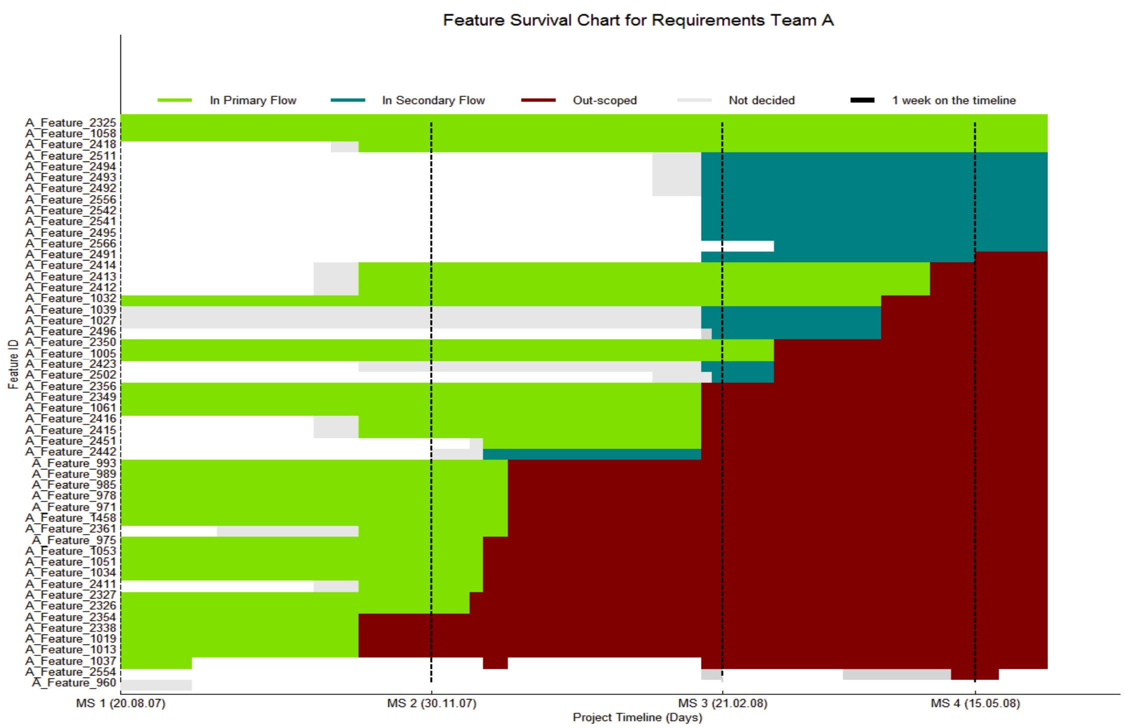

Figure 3. FSC for RT A. The labels on the Y axis are unique feature identities. The full-size color figure can be found at http://www.cs.lth.se/home/Krzysztof_Wnuk/rev08/RT_A.bmp

the project, resulting in a much effort spent on features that never survived.

- The balance between the primary and secondary flow is uneven. The final MS 4 scope has about 75% features from the secondary flow and 25% from the primary flow.
- The points in time when the largest de-scoping was done are visualized as large increases in the red area. Based on this graph, management can check CCB logs for particular dates in order to determine the reasons of such a big de-scoping.

The graph shows several sets of features being de-scoped a few weeks after MS 2. Analyzing the change log, it is found that there is new information from the market taken into consideration at this point in time. It resulted in a re-prioritization within this RT so that many features were de-scoped. The large amount of de-scoped features right before MS 3 is due to lack of resources within the primary DT.

4.3.2. Visualization of RT B. In the second RT example, showed in Figure 4 we can see a different scoping policy compared to the first one. In particular, we can observe the following interesting facts:

- Few features appeared after the initial scope was set at MS 1.
- Drastic scope reduction occurred just after two weeks from MS 1, where around 34% of all considered features where canceled.
- There was another significant scope reduction around two weeks before MS 3.

The change log reveals that there was a strategic business decision changing the focus of the scope right after MS 1. Later on, before MS 3, it was discovered that the resource situation was worse than expected among the DTs that were impacted by these features, so another set of features was de-scoped.

By comparing the graphs from the two RTs we can see that there are resource problems discovered in both groups before MS 3. Thus we can also use the FSC to see commonalities between the RTs. By connecting the FCS to the logs from CCB issues we can identify reasons behind the differences between the groups; a strategic business decision resulting in large early de-scoping in RT B, and in RT A, a new market information that is changing the scope late in the

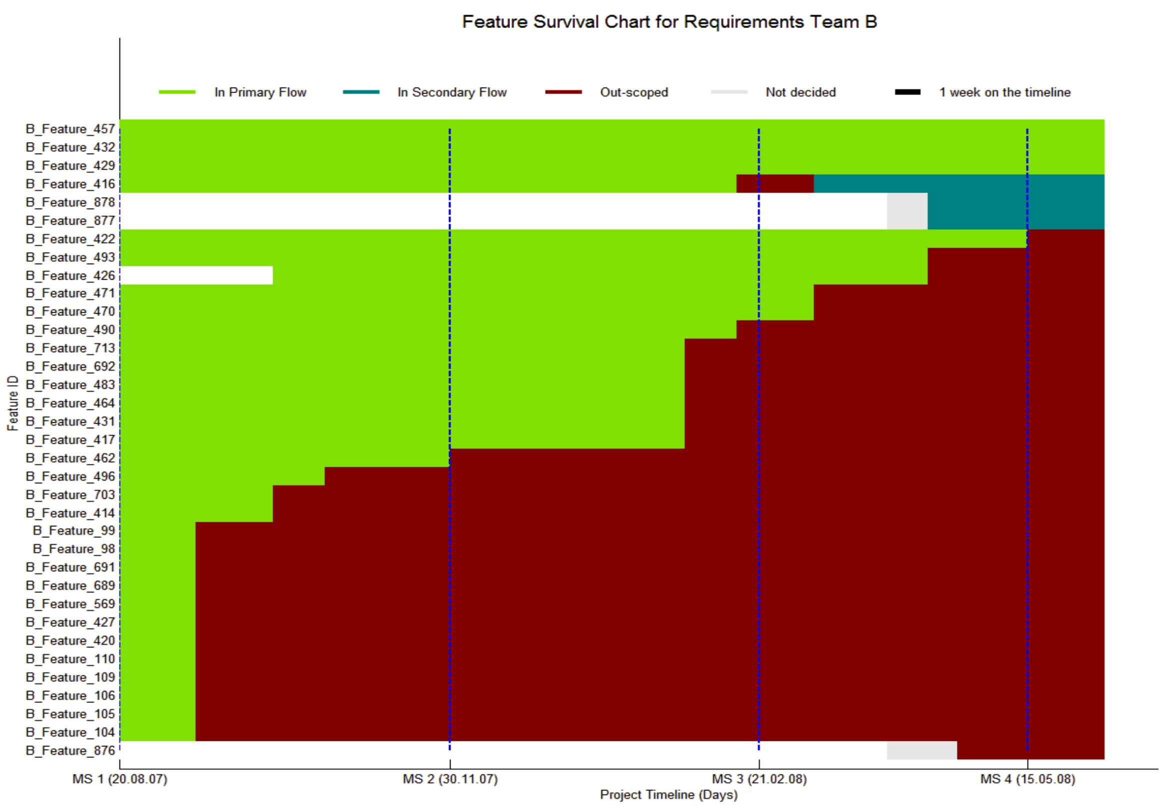

Figure 4. FSC for RT B. A large scope reduction can be easily spotted after MS 1. The full-size color figure can be found at http://www.cs.lth.se/home/Krzysztof_Wnuk/rev08/RT_B.bmp

project.

5. Validation and Limitations

Our validation approach contains three steps. Firstly, before generating graphs we conducted interviews with RTs to understand the challenges of setting a realistic scope early in the project. The most important conclusion from this validation was that we would benefit from visualizing the scope and scope changes in order to be able to discuss specific cases and points in time. Interviews revealed also, that some RTs used an approach to set a big scope, although knowing there was not enough resources for all features. At the same time one more issue was discovered: the difficulty for DTs to set a good effort estimate early on. While the features can not be analyzed with desired precision it is also difficult to say how much effort is needed for design and implementation.

The second step was to do the visualization keeping in mind the issues and input received from the first step. The results confirmed previously described problems, like for example we found an increase of de-scope features between MS 2 and MS 3 which according to the log from CCB was caused by inaccurate DTs effort estimates. In our graphs we also saw, that during the period from MS 1 to MS 4, the project directive was changed a number of times, resulting in instability in the scope setting process. Some RTs had to remove several of their planned features in order to make room for the new ones based on the new project directive.

Finally we have presented our work to RTs and project management in order to gain comments, feedback and improvements proposals about the usage of our visualization technique. The graphs turned out to emphasize what was previously only a "gut-feeling" among the people in the organization. The RTs are frustrated about the situation with drastic shifts in focus of the platform, the inability to set correct effort estimates, the challenge of getting the correct stakeholders involved, etc. Therefore, the RTs were pleased to be able to use the graphs to show their situation to others within the company. Line managers and process engineers were also interested in using our visualization to motivate and enforce the process on getting the right things into the scope and to improve the quality of effort estimates, and scope decisions.

The approach has also some limitations. First of all it is limited into a static two-dimensional figure, where the user is getting a ready image with no interaction. Secondly the approach has limited end-user's configurability so that it is not possible to change the set of features for visualization or coloring scheme. Thirdly, our solution is not yet taking into consideration other important attributes, such as size of features in terms of number of sub-requirements, their criticality or implementation cost. Finally, our approach is lacking interactive zooming features, so that the user can after analyzing the whole scope of the project be able to focus only on the most crucial RTs without changing the graphs (currently we produce separated graphs per RTs and the whole project).

The results are tightly coupled with the specific requirements and the requirements engineering practices of this particular case. We have not yet had the opportunity to empirically evaluate our solution in other software development organization with different roles, deliverables and practices. Thus there are yet unaddressed threats to external validity that, we will try to address in future work.

6. Related Work

One of the literature studies about current practice in requirements engineering visualization revealed that visualization is supporting three aspects of requirements engineering [3]. The first aspect is focusing on visualization of the structure and relationships between requirements described for example in [12][13][17][18]. Secondly requirements visualization is supporting elicitation [20] and decision making activities [19]. Finally, some work is done to create a visual representation of requirements and their attributes based on a formal language, like in [14][15] or even visualize these representations [21].

Our approach, which is using similar types of 2 dimensional bar charts as described in [20] presents visualization of another aspect in requirements management process, visualizing the scope changes over time. The scoping process in our method is visualized and can provide valuable input for process improvement frameworks such as REPT [22]. Although some work exists in visualization of release planning [23] our approach extends this work by focusing on showing the implications of a non-perfect release planning rather than providing decision support for the activity itself [24]. Finally, we provide important information for project stakeholders about the progress of the project. We focus here on the project scoping activity, rather than all project activities describe e.g. by Hansen [25].

7. Conclusions and Further Work

One of the issues in project scoping is the risk of setting a big scope, although knowing there might not be enough resources for all features. Our visualization technique can easily spot this problem and increase awareness of balancing between setting limited scope early and setting a too large scope, as well as spotting the risk of spending too much resources on features that anyway is later de-scoped. Another important situation visualized by our approach is the difficulty for designers to set a good effort estimate early on. While the market value of a set of features can not be analyzed with desired precision it is also difficult to say how much effort is needed for design and implementation. Our visualization technique helps to identify what features and what time frames to analyze in order to find scoping problems related to uncertainties in the estimations that decisions rely on. Finally, we conclude that our approach may be useful in visualizing instability of the scope setting process. The decision to remove several features in order to make room for extra features from a new directive can be easily spotted.

In our industrial study we have demonstrated the applicability of our method in a large industrial environment. Our method can be also used to visualize other attributes in other environments as long as the containing documents are regularly baselined and there is a way of exporting data to a format that allows for rendering according to our proposed graphs.

The proposed approach has also some unaddressed issues of further work. In particular, on the following issues are of interest in future extensions of the Feature Survival Graphs and the Feature Growth Charts:

- Extending our implementation to include interaction with the user so that the user can zoom in to particular areas of the graph.
- Improving configurability of the solution in order to enable users to define their own coloring scheme, as well as the set of requirements to be visualized and the time range.
- Extending the current feature model by Y axis scaling which will directly show how many underlying system requirements that were included into a certain feature. This information, extracted using traceability links among requirements documents can enable visualization of de-scoping of partial features.
- Extend the current feature model with additional attributes, such as criticality and implementation cost. This might be visualized graphically by special markers.
- Extending the *Feature Survival Chart* by also sort or grouping the features based on feature dependencies for visualizing simultaneous scoping or de-scoping of related features.
- Defining, calculating, and visualizing statistical measures such as average time to de-scoping of a feature and the total effort spend on non-survivors.
- Continued empirical investigations of additional industrial cases that apply visualizations of the scope from inception to release of the platform.

8. References

[1] J. Bosch, "Software product lines and software architecture design", *Proceedings - International Conference on Software Engineering*, IEEE Computer. Soc, Toronto, 2001, pp. 717.

[2] Earrnshaw, R.A. *An Introductory Guide to Scientific Visualization,* Springer-Verlag, 1992.

[3] O. C.Z. Gotel, F. T. Marchese, S.J. Morris "*On requirements visualization*", Second International Workshop on Requirements Visualization (REV 2007), IEEE, New Delhi India, 2007.

[4] Haber, R.B and McNabb, D.A. *Visualization Idioms: A Conceptual Model for Scientific Visualization Systems, Visualization in Scientific Computing,* G.M. Nielson, B. Shriver and L.J. Rosenblum (Eds.), IEEE Computer Society Press, 1990.

[5] Matlab webpage http://www.mathworks.com last visited June 2008.

[6] Larkin, J. and Simon, H. (1987). Why a Diagram is (Sometimes) Worth Ten Thousand Words; *Cognitive Science*, 11, pp. 65-99.

[7] Glenberg, A. and Langston, M. (1992). Comprehension of Illustrated Text: Pictures Help to Build Mental Models; *Journal of Memory and Language*, 31 (2), pp. 129-151.

[8] Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information; *Psychological Review*, 63, pp. 81-97.

[9] J. Karlsson, K. Ryan , "Cost-value approach for prioritizing requirements", *IEEE Software,* IEEE, Los Alamos, 1997, pp. 67-74.

[10] B. Regnell, L. Karlsson, M. Höst, "An analytical Model for Requirements Selection Quality Evaluation in Product Software Development", *Proceedings 11th IEEE International Requirements Engineering Conference,* IEEE, Monterey Bay USA, 2003, pp. 254-263.

[11] J.-M. DeBaud, K. Schmid, "A systematic approach to derive the scope of software product lines", *Proceedings of the 1999 International Conference on Software Engineering,* ACM, Los Angeles, CA, USA, 1999, pp. 34-43.

[12] Telelogic Doors webpage http://www.telelogic.com/Products/doors/doors/index.cfm (last visited July 2008).

[13] C. Duan and J. Cleland-Huang, "Visualization and Analysis in Automated Trace Retrieval", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[14] Teyseyre, A. A 3D Visualization Approach to Validate Requirements, *Proc. Congreso Argentino de Ciencias dela Computacion,* Argentina, October 2002.

[15] Unified Modeling Language webpage, www.uml.org last visited July 2008.

[16] R.G. Cooper, "Stage-Gate Systems: A New Tool for Managing New Products", *Business Horizons,* May-June 1990, pp. 44-54.

[17] O. Ozakaya, "Representing requirements relationships" ", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[18] D. Sellier, M. Mannion, "Visualizing Product Line Requirements Selection Decision Inter-Dependencies", Second International Workshop on Requirements Visualization (REV 2007), IEEE, New Delhi India, 2007, pp. 1-10.

[19] M. S. Feather, S. L. Cornford, J. D. Kiper, T. Menzies, "Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[20] M. Pichler, H. Humetshofer, "Business Process-based Requirements Modeling and Management", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[21] S. Konrad, H. Goldsby, K. Lopez, B. H.C. Cheng, "Visualizing Requirements in UML Models", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[22] B.Palyagar, F. Moisiadis, "Validating Requirements Engineering Process Improvements – A Case Study", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[23] K. Herrmann, **"**Visualization of Release Planning", First International Workshop on Requirements Visualization (REV 2006), St. Paul MN United States, September 2006.

[24] D. Nobelius "Enpowering project scope decisions: introducing R&D content graphs" , R&D Management, 31, 3, Backwell Publishers, 2001.

[25] K. T. Hansen, "Project Visualization for Software", *IEEE Software,* IEEE Computer Society, July-August 2006.