

Building Roadmaps: A Knowledge Sharing Perspective

Antony Tang
VU University Amsterdam
Department of Computer Science
Amsterdam, the Netherlands
atang@cs.vu.nl

Taco de Boer
Océ Technologies
Venlo, the Netherlands
taco.deboer@oce.com

Hans van Vliet
VU University Amsterdam
Department of Computer Science
Amsterdam, the Netherlands
hans@cs.vu.nl

ABSTRACT

Roadmapping is a process that involves many stakeholders and architects. In an industry case, we have found that a major challenge is to exchange timely knowledge between these people. We report a number of knowledge sharing scenarios in the roadmapping process. In order to address these issues, we propose a codification mechanism that makes use of a semantic wiki to facilitate knowledge sharing.

Categories and Subject Descriptors

D.2.1 [Requirements/Specifications], I.2.4 [Knowledge Representation Formalisms and Methods].

General Terms

Documentation, Human Factors, Verification.

Keywords

Roadmapping, Requirements, Architecture, Knowledge Sharing.

1. INTRODUCTION

Product line software architecture design is typically a forward looking activity for planning new features and functions that will be incorporated into the future products. A roadmapping process is typically used to achieve this goal [7]. From the product and marketing perspectives, marketing managers and product managers need to understand the potentials, development and planning of the product lines in order to stay competitive. From a software architecture perspective, roadmapping is necessary for two reasons. Firstly, software architects can evaluate the feasibility and the development costs of potential development; secondly, software architects can plan platform architecture enhancements to provide the necessary flexibility and compatibility for future features [10]. Such an architecture design approach encourages systematic reuse of software, instead of opportunistic reuse, leading to a better quality software with a lower development cost [3].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SHARK'11, May 24, 2011, Waikiki, Honolulu, HI, USA
Copyright 2011 ACM 978-1-4503-0596-9/11/05 ...\$10.00

Roadmapping is an approach to manage high-level and future requirements. The outcome of the process is a roadmap that describes the product features that are planned, their priorities and their interdependencies. Each feature is described in terms of a new product release. This process typically involves three types of stakeholders: (a) product and marketing stakeholders would provide inputs on requirements and market needs; (b) architects at different levels with different product responsibilities would assess technical feasibility, provide inputs on technology trends, and plan the architecture design; (c) project or program managers would provide inputs on software and product development schedule and planning. In a large organization such as the one we report here, some twenty or more people are directly or indirectly involved in the roadmapping process for each product line.

These stakeholders plan future products periodically, typically annually or biannually. Regular updates from the stakeholders also take place when market, technology and development conditions change. Roadmapping is an ongoing process that comprises periodic meetings, i.e. product requirement analysis and planning, as well as continuous information exchanges.

In order for the roadmapping process to work, stakeholders must share their product and design knowledge effectively. However, there are several issues in knowledge sharing. Firstly, the knowledge of the stakeholders may be unavailable or uncertain at the time when the roadmap is drawn up [1]; secondly, some of the knowledge is internalized and tacit [12], and cannot be shared readily; thirdly, explicit knowledge may not be distributed effectively [9]. This presents a major challenge to the completeness and the integrity of roadmaps. This situation is more challenging in the case of product line development because there are competing goals between developing a product for market release and developing a reusable software in a line of products.

The aim of this paper is to analyze the complexity of software architecture knowledge exchange under such circumstances and propose some ways to overcome the issues. We study a roadmapping case in Océ Technologies. We have analyzed their current process and have identified where knowledge is created and how it needs to be shared between roadmap stakeholders. From this analysis, we have identified three main aspects that are important to roadmapping: knowledge ownership, knowledge distribution and the timing of knowledge sharing.

Knowledge can be distributed by pushing or pulling. The roadmapping process requires both mechanisms. The pushing of knowledge can be challenging because the one who pushes the knowledge needs to understand who and when to push knowledge to. The success of the push mechanism relies on the proactiveness

of the individuals. To address this issue, we propose a new way to codify and semi-automate knowledge sharing.

2. BACKGROUND

Roadmapping is important to the future development of a product. It generally defines a future environment such as the direction of a product; it also defines objectives such as what features a product will offer and how it will compete in the market place [1]. For technology products such as the high-end printers that Océ Technologies produces, a roadmap also drives architecture design because software architects would have to evaluate that features that are planned for a future product are feasible, and they also need to know if the current software architecture can support the planned features.

2.1 Related Work

The roadmap process serves many purposes. It enables product planning, which is the most common purpose [13, 14]. It also serves in capability planning, strategic planning, knowledge asset planning and program planning. Roadmaps can be classified into product roadmaps, technology roadmaps and product-technology roadmaps [11]. Product roadmapping is about the planning of product developments based on the understanding and prediction of the business markets. Technology roadmapping is about the planning of product developments based on the understanding of how technology will change in the future [7]. A product-technology roadmap is a combination of both roadmaps together.

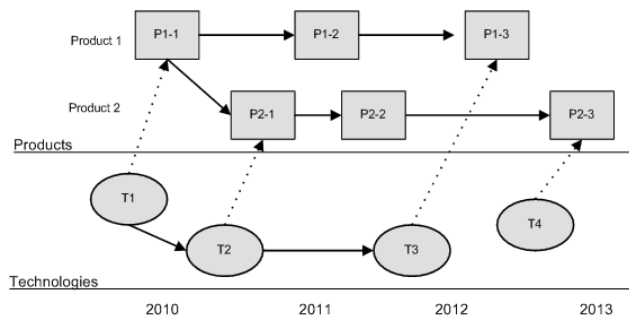


Figure 1. Product and Technology Roadmapping

Figure 1 is a simplified example to depict the relationships between products and technologies in a roadmapping process. The arrows indicate dependency between two products, or between a product and a technology. Product *P1-1* is based on technology *T1*. Technology *T1* is the basic enabler which the product is built on. Product *P2-1* reuses some technology that are developed in product *P1-1* as well as technology *T2*. Both products 1 and 2 belong to the same product line, and each product evolves into new versions over time as technologies and market needs change. Technologies *T1* evolve into *T2* and *T3* over time. *T4* is a (new) technology, not evolved from *T1-3*. A technology may also be applied to multiple products with its only evolution. For instance, a power supply is used in different products with slightly different features. In our case, we focus on product-technology roadmaps because Océ Technologies has adopted some of this practice [7].

The technologies that are used for product development can be developed externally or internally. In the case of employing external technology, roadmap planning is about anticipating the availability and the maturity of such technologies. An example is

anticipating if a certain platform such as an operating system could be used in a product, based on factors such as availability, maturity and serviceability. Alternatively, technologies can also be developed in-house. For instance, an in-house R&D department may develop architecture components as its core software assets to build a product. In which case, the roadmap planning process would also need to cater for the development planning and progress of the in-house R&D technologies.

A roadmapping process requires different parts of an organization and different stakeholders to participate [15]. It is a process in which the knowledge, explicit or tacit, of these stakeholders will need to be combined and exchanged in order to create a viable and implementable plan. A major and practical challenge in this process is to exchange knowledge in a timely manner between these stakeholders to create and evolve a roadmap. Based on the knowledge sharing model of [12] in which knowledge sharing takes place by *socialization*, *combination*, *externalization* and *internalization*, one suggestion is to organize stakeholders into contributors (of knowledge), controllers (of process), and distributors (of knowledge) and define their responsibilities and dependencies for the roadmapping process [9].

2.2 About the Industry Case

Océ Technologies produces high-end printers to serve the business markets in high-volume printing, wide-format printing and office printing. Printer software is one of the main components in a printer. The software renders images, controls the print engine and other devices such as scanners and finishers. The company is in a highly competitive market place where providing new features to the market in a timely fashion is important to product success. We were engaged with one business unit that has two development teams, one in the Netherlands and one abroad.

The business unit has been practicing roadmapping for a number of years. They follow a roadmapping process that closely resembles the product-technology process depicted in Figure 1. They produce technology roadmaps and product roadmaps at the printer level, sub-system level and component level. A roadmap defines what features are to be built, at what phase of the product development the features are built, and the costs and feasibility of building those features. Although the efforts that have gone into roadmapping appear to be worthwhile in terms of creating a vision and a plan for the product, they also face many issues, especially in keeping the roadmaps up-to-date and sharing roadmaps effectively amongst the stakeholders. A major issue is the coordination of information amongst the stakeholders and architects, mostly in different locations and different parts of the organization, to keep roadmaps accurate and up-to-date. This study is based on studying the roadmap process of one product.

2.3 Research Method

In order to study and analyze the situation, we take an action research approach that involves the look-think-act cycle [16]. The research steps taken are as follows:

- **Look.** Firstly, we examined the roadmapping documentation produced; secondly, we conducted a survey to investigate how roadmapping knowledge is shared amongst key stakeholders; thirdly, we use a focus group meeting to identify issues experienced by the architects (see Section 3).

- **Think.** We analyze the data that we have gathered, articulating and uncovering the cooperative issues of building a roadmap. We analyze this data with regards to the roadmapping steps and the stakeholder communication. The analysis of these aspects allows us to theorize where the gaps are (see Section 4).
- **Act.** To rectify the situation, we suggest a knowledge sharing model that enhances the current roadmapping process and addresses the cooperative issues that are currently experienced by the stakeholders (see Section 5).

This kind of engineering process cannot be studied quantitatively because there is only a single case. Instead we use a qualitative approach to gather evidence and to gain an understanding of the situation. We are careful to gather objective information without researchers' biases. For instance, in the survey, we ask participants to name the roadmaps that they know about and measure the recall rate of the roadmaps that they name. In this way, the researchers are not biased by incomplete knowledge. Furthermore, in the focus group meeting, stakeholders were asked to discuss their issues freely, and the researchers observe and understand their situation.

3. CURRENT ROADMAPPING PROCESS

3.1 Project Structure

When developing a roadmap for a product, e.g. a printer with some specific features, a project is created. Each project has architects at three levels. The product architect, also called the print systems architect, is responsible for developing all features in a printer product, including hardware, software and service components. At the next level, there are many software and hardware systems. One of the key systems is the controller software system. Each system is developed by a team of engineers with a team leader and system architect. The owner of the roadmap process at this level is the system team leader, supported by a system architect. At the next level there are subsystem teams. For instance, the controller software system has nine software sub-systems such as printer driver, image processing, service diagnostic software and so on. Each software sub-system is the responsibility of a function team and a function team has a range of between four to eighteen people, including an architect. At each of the three levels, there is someone, typically an architect and/or team leader, responsible for producing a roadmap. A software system sub-team typically handles between three to seven products simultaneously. Figure 2 shows the relationships between team responsibilities and products in a product line.

Figure 2 illustrates 2 products, A and B, that belong to the same product line. Each product is organized into a separate project, with a product architect who looks after the development of the roadmap of that product. The product architect is supported by software architects, electrical engineers, mechanical and other engineers. The architect of each major system is supported by architects and developers that are grouped by their specializations. In controller software system, for instance, there are nine teams, each of them is responsible for a sub-system. Architects at each level develop roadmaps appropriate to their level and field of expertise. Together the roadmaps define the features that will be developed for a particular product.

It is important to note that each sub-system team supports more than one project / product. For instance, the user interface team develops user interfaces for all products in the product line. The idea is to leverage the expertise of one team and share knowledge amongst all products within a product line. As shown in Figure 2, sometimes architects at the system level may work on one or multiple products. Since the organizational structure is arranged by product, architects and developers are very much product focused.

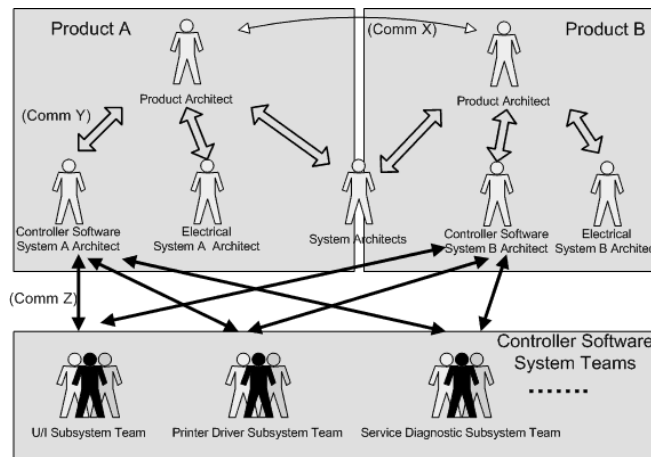


Figure 2. Product Roadmapping and Team Structure

3.2 Initial Investigation

To begin with, we identify roadmapping documents at Océ. There are two types of documents. Firstly, a roadmap chart depicts all features that are planned for the future. This chart is a two dimensional matrix: (a) product features in one dimension; (b) the time when a feature is planned for implementation, feature priority and some interdependency information as the other dimension. This chart is updated at least annually. Secondly, for each feature, there is a detailed description of the analysis and design of the feature. This document is prepared by the architects at different levels to assess the feasibility, cost and benefits of each feature on the roadmap.

The roadmapping process consists of the following steps:

- Feature ideas are created by architects or marketing managers
- Technical feasibility and the impact analysis of a feature idea is studied by the architect(s) to ensure that planned product features are implementable
- Market and product analysis are studied by marketing group
- Regularly, all stakeholders meet to update the roadmap and to plan for a product release. Planning is based on prioritization of the features and availability of the required (human) resources. The roadmap process is product-based and each product in a product line has its own roadmap.
- Once the roadmap is adopted in the development program, the roadmap is baselined and used in development & marketing

At this stage of the investigation, we are not sure if roadmap knowledge is sufficiently circulated between the stakeholders and

if that knowledge is kept current. In a three day product roadmap workshop, we have surveyed the participants to find out how well they know about their roadmaps. Three R&D department managers, one project leader, nine architects and two team leaders participated in this workshop. We asked the fifteen participants at the beginning of the workshop to identify unique product and technology roadmaps that the group collectively knows about. Eighteen technology roadmaps and ten product roadmaps have been identified collectively. From the two types of roadmaps that are collectively identified, we asked questions in three areas:

- Identify the roadmaps that you know about
- Identify the roadmaps that you have read
- Who should know about the roadmaps

The results from the first two questions are summarized in Table 1. Arguably, some of the roadmaps that some architects have identified are unofficial documents or do not exist. The official product and technology roadmaps are 4 and 10, respectively. It indicates that the architects are unclear about what roadmaps exist. Out of the identified roadmaps, the product roadmap that is most identifiable is identified by 13 out of the 15 participants, and the same holds for the technology roadmap.

Table 1. Roadmap Identification and Circulation

	10 Product Roadmaps	18 Tech. Roadmaps
A roadmap that is most known in terms of the number of participants	13	13
Max. number of roadmaps a single participant can identify	6	10
Average number of roadmap a participant can identify	4	6
Average number of roadmaps a participant has read	1	2

There are, however, many roadmaps, official and unofficial, that are unknown to many participants. On average, 4 out of 10 product roadmaps and 6 out of 18 technology roadmaps are identified. Some roadmaps are only known to one person, indicating that some roadmaps are hardly shared. We further ask the participants what the status of the roadmaps is. The responses are: (a) they don't know; (b) they are in a draft form; (c) they are approved. The responses are equally spread across all three categories. It indicates that either the architects do not know the status of the roadmaps or the status of the roadmaps is ambiguous.

There is no single person who knows of all the (official and unofficial) roadmaps, the most a single person knows is 6 out of 10 product roadmaps, and 10 out of 18 technology roadmaps. When we ask how much they have read the roadmaps, an average of 1 out of 10 product roadmaps and 2 out of 18 technology roadmaps have been read. When asked why this is so, one reason given is that the participants are not able to locate the documents. Since the participants are key stakeholders of the roadmaps, this result is surprising and indicates that the sharing of roadmaps is quite limited. The low read rate may have implications for the implementation of the planned features as well as architecture design and planning. Although it cannot be assumed that everyone should read everything, the fact that no one, not even product nor system architects, know about all the roadmaps when they should

have an overview of the product is an indication of potential knowledge sharing issues.

Currently roadmaps within the business unit are considered as confidential information because the documents describe future product details that need to be protected before product releases. For this reason, these documents have limited circulation. When we ask the participants who should know the information contained in the roadmaps, participants indicate that almost all development roles, including user interface designer, team leaders, developers, testers, integrators and configuration managers, need to know about them. The coordination to distribute the knowledge to all these roles on a need-to-know basis with confidentiality consideration is a challenge.

3.3 Roadmapping Process

From our survey in the roadmap meeting, we have found evidence to suggest that the roadmap knowledge is not circulated adequately. Many architects and stakeholders do not know about the roadmaps that are useful to them. We postulate that there may be weaknesses in the process or in the ways people communicate.

In order to understand the situation, we have arranged a focus group meeting where three groups of stakeholders who contribute to the creation of roadmaps present their processes. The three groups of architects are: product architect, system architects from controller software system and sub-system architects from service diagnostic sub-system.

Roadmapping Process Steps. There is a standard process for product level roadmap creation across the organization. In a focus group meeting, the following process steps are outlined by the product architect for her product:

- Identify business and architecture drivers. Start creating a long list of options (1.5 week).
- Generate design options. Shortlist product options based on drivers, and value each feature (3 weeks)
- Create initial roadmap. Link design options to product timeline and identify resource and timing conflicts (1 week).
- Refine roadmap. Define product packages according to development timeline, resolve timing and architectural conflicts (1 week).
- Validate roadmap. Review options and resource utilization (1 week).

Inputs to a Roadmap. All architects provide inputs to the product level roadmap. These inputs can be knowledge that they have or new ideas that from their inventions. The key inputs that architects at each of the three levels provide are:

- The product architect gets information from sales and marketing about new features and market trends that might be desirable in a product. She prepares a long list of potential features for a product, prioritizes them and then reduces them to a short list of 50-90 items. This is an iterative process between a product architect and other stakeholders. This product roadmap exercise is performed periodically.
- The system architect specializes in a particular field, e.g. controller software. She would understand the architecture

design of the product as well as the technology trends, standards and legislative considerations in her area. She will contribute new technological ideas to the product. She would consider the budget and the current architecture design to evaluate the impacts, dependencies and feasibilities of implementing new product features. The roadmapping exercise is periodic but some information about the implementation of each feature may become available afterwards. For instance, a legislative change may impact the requirements and the design of a feature.

- Sub-system architects are more specialized than system architects. They contribute new feature ideas from market trends, new tools and their research & development. Since sub-system architects and their team work on multiple products in a product line, they not only have the knowledge of how to implement a feature but also knowledge of the intricate details of how things work in different products. Architects at this level develop technology roadmaps independently of the product roadmap cycle.

Communication Channels. Based on the information that we have gathered, we outline three communication channels between architects in Figure 2:

- Between product architects across projects (Comm X) – product architects are to share roadmap knowledge in a product line such that new product features and technologies can be planned and aligned between products.
- Between a product architect and system architects (Comm Y) – a product architect is supposed to coordinate the creation of the product roadmap. She communicates sales and marketing ideas to her architects. This is achieved through meetings, personal communication and document preparation.
- Between system architect and sub-system architects (Comm Z) – when sub-system architects see opportunities for technology improvements, either by software reuse across different products or introducing new technologies, these can be developed into a mini-roadmap that is shared with system architects of one or many products. Additionally, these teams are asked to carry out impact analysis when product ideas are passed from the product architects through system architects.

4. KNOWLEDGE SHARING ISSUES

Using the survey and the focus group meeting, we have observed a number of common issues in coordinating and sharing of knowledge. Roadmapping in a high-tech environment can be typified by three conditions: (a) a large number of people with diverse and specialized knowledge; (b) high variability of the end results; (c) fast time-to-market. Under these conditions, we have observed challenges in coordinating and distributing knowledge.

4.1 Knowledge Ownership

From the focus group meeting, we have learned that architects at the product level, system level and sub-system level have their own expertise. The product architect has an overview of the product and the requirements from sales and marketing. However, the product architect does not have intimate knowledge of how these ideas can be implemented, so she requires the system architects and the sub-system architects to provide that

knowledge. On the other hand, the system architect has overview knowledge of the system and the different sub-systems within it. She has the knowledge about the interdependencies between those sub-systems, but she doesn't know the implementation details of each sub-system. Sub-system architects have the detailed implementation knowledge, but they may not be aware of the constraints or the integration issues with other sub-systems.

Currently when roadmaps at various levels are prepared, an individual architect does not possess all the knowledge necessary to implement a system and therefore they have to assume that the others can fill that gap. Product architects may assume that a feature can feasibly be developed. System architects may assume that the requirement of a feature is definite. As information becomes known over time, the owner of the knowledge may not share it or she may not know that it is important to share it with the others. On the receiving end, those who may need it don't know that the knowledge relevant to them is available. During the roadmapping process, this issue of "I don't know what I need to know" becomes a major issue because often a decision to implement a new feature can trigger changes to different parts of a system. The architect with the new idea may not know who may be impacted by such a change, and the architect whose designs are affected is not aware of the change.

4.2 Knowledge Distribution

Many roadmaps have been created in the business unit, but from the survey we have learned that roadmap identification and sharing is low even though almost everybody agrees roadmaps should be shared more widely (see Section 3.2). One issue is knowing whom to share roadmap information with, i.e. pushing knowledge; and knowing whom to ask for roadmap information, i.e. pulling information [5].

Although the roadmapping team resembles some form of hierarchical structure (Figure 2), the organization uses an agile development practice which lends itself towards an unstructured form of communicating knowledge. From this study, we have not found evidence to indicate that the organization has defined a standard process for producing and sharing roadmaps and knowledge. It appears that the coordination and exchanging of knowledge happens through personal contacts between interested parties.

This exchange can happen within the same level (e.g. between two sub-system architects) or across two levels (e.g. between product architect and system architect). The manner in which exchanges happen can use three basic mechanisms: (a) requesting information from someone due to a need, e.g. ask if anyone knows about something; (b) pushing information voluntarily to someone when it is expected that the information may be useful in some ways; (c) through meetings in which knowledge is exchanged instantaneously. Due to the constraints of organizing too many meetings, much knowledge is exchanged using the first two mechanisms. The knowledge that is delivered can be in response to a pull request or the knowledge may be proactively pushed. In case of proactive push, the target audience may or may not be known. For instance, the Service Diagnostic Team prepared a roadmap to build operating platform software to be used by multiple products; the knowledge was pushed to the system architect without knowing who the target audience was.

Figure 3 illustrates the roadmap knowledge distribution scenarios across time. Roadmaps are organized into product, system and sub-system levels that correspond to the roles of the architects at each level. Each roadmap has a create and update cycle across time, indicating the beginning and the end of its relevance as a roadmap. An oval indicates the period when roadmap milestones have been scheduled. The terminator at the beginning signifies the inception point, and the terminator at the end signifies the end point. During the life-cycle, an architect may pull, or request knowledge to build a roadmap. In response to the explicit request, system architects provide the knowledge. This form of knowledge transfer is simple to manage because the requestor would know and take action if the knowledge is not delivered.

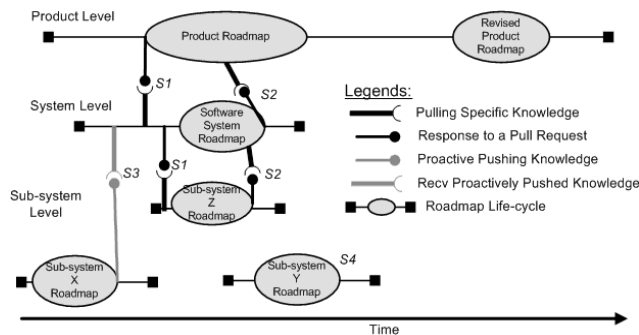


Figure 3. Knowledge Distribution in Product Roadmapping

Request Requirement Knowledge (scenario S1). An architect requests requirements information from a higher-level architect, e.g. system architect pulls product requirements from a product architect. A request is made and the information is supplied.

Request Detail Knowledge (scenario S2). A request is made by high level architect to pull roadmap details, e.g. product architect request system architect to supply specific knowledge such as the evaluation of the impact of a product feature, and a request by a system architect to a sub-system architect, and their responses (Figure 3).

Sunk Knowledge (scenario S3). It depicts a case when knowledge is both pushed and received, but the knowledge is sunk or not used, so it does not reach the right audience. For instance, a sub-system architect pushes knowledge to a system architect, the knowledge is not relevant when it is received, and it is forgotten by the system architect without circulating it further.

Unshared Knowledge (scenario S4). It depicts new knowledge that is not passed on to anyone, either because an architect is not aware that the knowledge needs to be shared or she does not know who to push that information to. The knowledge remains undistributed. This happens under many circumstances. For instance, a sub-system architect works with many products, she may see an opportunity to build a general platform for a product line. However, the products are organized along a project and when the focus is to meet a project deadline, the idea of building a common platform as a technology feature is harder to sell. So the knowledge that a product line approach is a better design does not surface to the top level. It can also happen when the owner of the knowledge does not know who may be interested in the knowledge that she possesses, or she is not aware that the information may have impact on the roadmaps. A sub-system

architect, for instance, knows about an interface design change but she is not aware that a change may impact on the implementation of a newly planned feature. This is similar to the use cases of adding and validating architectural decisions suggested in [8].

If the potential receiver of this piece of knowledge is unknown, then it is likely that the knowledge will not be distributed to the right person. If the knowledge is stored in a repository where someone can find it, then the issue is about searching for the knowledge when it is needed.

4.3 Timing of Knowledge Sharing

From the focus group meeting, we have observed two general issues with regards to the timing of knowledge preparation and sharing. Firstly, there is insufficient preparation time. When new feature requests from sales and marketing arise, the product architect would request system and sub-system architects to prepare a technical impact analysis. The analysis would show the feasibility of implementation and the impacts on other parts of the system. Sometimes there is insufficient time for architects to prepare a thorough analysis and to review all the options before making a decision. In section 3.2, we report that the refinement and validation cycles are typically limited to two weeks. The severe time limitation can create sub-optimal roadmap decisions.

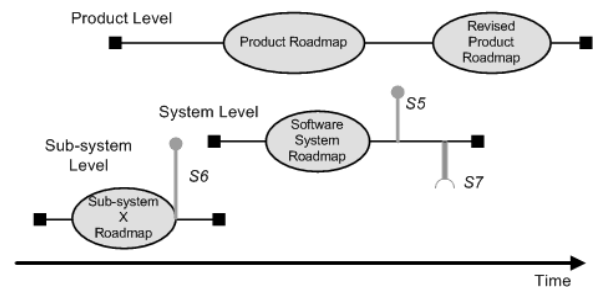


Figure 4. Timing of Problematic Knowledge Push

Secondly, if the knowledge is known outside the main roadmap preparation period, the information may not be deemed useful and has a tendency to be overlooked. There are two scenarios in this case (see Figure 4).

Un-received Post-roadmap Knowledge (scenario S5). This is a case when new knowledge is found after the product roadmap has been created. This knowledge is proactively pushed but there is no receiver and therefore does not go into the revised roadmap. This can be at the system or sub-system levels.

Un-received Pre-roadmap Knowledge (scenario S6). This is a case in which the knowledge is found and proactively pushed before the roadmap process officially begins, and there is no receiver. When the roadmap process starts, people have omitted to take it into account.

Asynchronous Knowledge Request (scenario S7). A system architect may need to pull or be alerted when certain relevant knowledge becomes available. For instance, an architect can inform others that if there is any new features that have anything to do with service diagnostics, she would want to be notified. Such an event may or may not happen. By advertising the need to know about certain events helps the architect to be notified of the

arrival of such asynchronous knowledge, provided if the others keep this request in mind.

4.4 Process Coordination

In the focus group meeting, it is generally agreed that currently there is no well-defined roadmapping process. There are a number of general issues. Firstly, there are too many roadmaps at different levels, instead of a single product roadmap that presents a single vision of the product. The drawbacks of having many roadmaps at different levels are that relevant knowledge may be lost and it is harder to integrate dispersed information into one single coherent roadmap. A matrix is used to track features and do analysis, and this matrix is the main tool to track features and their interdependencies. The knowledge contained in this matrix is spread across many people. Product and system architects have a good idea of what they want to achieve and they know the interdependencies between the components. On the other hand, sub-systems architects have detailed working knowledge of the component. They know which parts of the software need to be modified to make a feature work. The knowledge coordination process is to connect the architects who know what to achieve with the architects who know how to achieve it.

Secondly, the coordination of the roadmap process is minimal and no one has the sole responsibility for the distribution of knowledge. The current practice is to use a workshop to discuss the roadmap. A workshop may be an effective way of communicating roadmap knowledge but it cannot be held regularly, so other means of coordinating roadmap knowledge are required. Thirdly, at the system and sub-system levels, there are no processes to guide the roadmap life-cycle. Their creation, update and distribution are unstructured. Outside a workshop, the stakeholders rely on the individual's proactiveness to distribute knowledge. If all the knowledge owners know whom to distribute the knowledge to, then there would be no issue in knowledge sharing. In reality, knowledge owners are not aware that certain roadmap knowledge is relevant to other people, so the right knowledge does not go to the right person.

5. KNOWLEDGE SHARING MECHANISM

The analysis of the roadmapping process shows that the knowledge coordination between architects to create a roadmap is a challenge. It is typically not an issue when specific information is pulled and one knows precisely where to pull that information from. The key challenge is when knowledge is created and the creator of the knowledge does not know who may be interested in it. The usefulness of this knowledge would be reduced (e.g. scenario *S3* and *S4*). Additionally, if there is no receiver ready to receive this knowledge when it is created, the relevance of the knowledge drops and may not be retained over time (e.g. scenarios *S5* and *S6* in Figure 4). Therefore, it is necessary to retain this knowledge explicitly so that it can be retrieved later.

For architectural knowledge (AK) to be retained over the roadmap life-cycle, it is necessary to document and codify the knowledge in some way so that it can be searched and retrieved at different times. At Océ, architecture knowledge retention uses both codification and personalization strategies [2], with a heavy reliance on personalization. Architects often communicate verbally to exchange design information. A personalization strategy relies on the workers to know who their peer knowledge

workers are and what they know. If an architect wants to find something, she would ask around to find the person who might possess this tacit knowledge. In a situation such as roadmap creation and revision, this can be difficult because there are too many stakeholders and it is unclear who might know what, or what might have an impact in another part of a system.

5.1 Retaining Knowledge

Roadmaps evolve over time. When parts of a roadmap change or new information becomes available, then someone needs to be informed. If a pure personalization strategy is used, then someone needs to coordinate and facilitate knowledge sharing because the many knowledge pushers and receivers situation can be chaotic otherwise. The current process does not have a specific person appointed to this role. If such a person is appointed, much reliance would be placed on her ability to gather both documented and tacit knowledge; and knowing who needs what knowledge so that the right knowledge is distributed to the right person at the right time. This can be difficult to achieve and it may be risky to the roadmapping process because of potential oversights.

On the other hand, a pure codification strategy would require documentation of all knowledge, this would be difficult to achieve also as the stakeholders have much tacit knowledge, and not all of them would be documented or cost effective to document. Even with documented knowledge, it can be difficult to anticipate what other people should know to push that knowledge to them. A hybrid strategy that involves codification and personalization has been found to be suitable in some industry cases [4]. A coordinator would facilitate knowledge assembly and distribution. She would be a point-of-contact to all of the roadmap participants. Codified knowledge is retained and is reusable to help architects find knowledge when they need it.

Conventional file-based documentation has been used as the codification method but it has its limitations. We have found from another study that the retrieval of documented information for subsequent use to be ineffective [6]. Firstly, people believe that documents often do not contain up-to-date details and so they become less reliant on them. Secondly, the information is often scattered over a number of files and cross references are often missing or not up-to-date. Keyword search is used as a means to find information but searches may not yield accurate knowledge retrieval results because of versioning issues and the use of synonyms and homonyms. Thirdly, many inter-dependent roadmap features, designs and analyses exist but their dependencies are not recorded and may be overlooked.

Some of these issues can be partly addressed by having a better defined roadmapping process and by improving knowledge communication. We propose to use semantic wikis with a suitable ontology to improve knowledge sharing. We propose to index software documents with a lightweight ontology. An ontology is a system that classifies concepts. For instance, a "paper cutting controller" software function is a concept about *finishing*. Users can use one or more concepts to search for the relevant documents that contain those concept instances. We have implemented a prototype system using a generic and adaptable ontology to support knowledge indexing and retrieval from software documentation. We draw an analogy between an ontology-based knowledge retrieval system and a relational database system,

where the ontology is the index and software documents are the data contained in tables.

Let us use a simple ontology that defines the concept of a system, the types of features and the knowledge area based on feature requirement, feature behavior, feature design and mutual dependency. The knowledge that architects create during the roadmapping process can be stored in a centralized repository system containing many wiki-pages. Key phrases in each document can be indexed using the concepts defined in an ontology. In this way, the ontology enables architects to search for knowledge using defined concepts. For instance, a product architect can search for a specific roadmap feature in the Service Diagnostic sub-system and relate it to the other features that have dependencies on it. This provides a mechanism for someone to find knowledge that has been deposited. We have prototyped a system to enable architects to create semantic wiki pages and indexing key concepts contained in those semantic wiki pages. Users can custom-define software engineering and domain specific ontological concepts. It allows them to index knowledge according to their searching needs. We have implemented a prototype system using Semantic Media Wiki and Cliopatria [17].

5.2 Just-in-time Knowledge Sharing

The issue of distributing just-in-time knowledge cannot be solved by just an ontology-based wiki system. Currently there is no notification mechanism in any current semantic wiki system. An event-based notification system similar to the concept of Twitter needs to be developed to alert architects when relevant knowledge becomes available. With such a system, architects can register their interests using defined concepts. When knowledge instances of those concepts are created in the knowledge base, the system would notify the architects of the arrival of the knowledge.

To solve Asynchronous Knowledge Request, an event alert can be defined in which the knowledge consumer registers her alert criteria in a semantic wiki system. When the producers create knowledge (e.g. scenarios *S5* and *S6*), the system automatically notifies the consumer. The timing issue and knowledge distribution issue can both be addressed. We argued earlier that a knowledge coordinator is also required. The role of this person is essential to facilitate the use of personalization strategy.

6. CONCLUSION

In this paper, we study the roadmapping process in Océ Technologies. Roadmapping involves many stakeholders, including many architects. Despite the many successful products that Océ Technologies have planned and launched, one major challenge is to further improve the effectiveness of exchanging timely knowledge of requirements, design and impact analyses between architects. From a survey and a focus group meeting, we have found that much of this knowledge has been created but they are not shared effectively. In order to find the right people to share the information with and in a timely manner, a personalization strategy that is heavily relied on currently is insufficient. So we propose to complement the roadmapping process with a codification method. A semantic wiki system with an appropriate ontology can be used to index roadmapping concepts in roadmap documents. The consumers of knowledge can either search the

knowledge base directly or receive notifications when relevant contents become available.

7. ACKNOWLEDGMENTS

We thank Philippe Kruchten for his participation and inputs in the focus group meeting. This research has been partially sponsored by the Dutch "Regeling Kenniswerkers", project KWR09164, Stephenson: Architecture knowledge sharing practices in software product lines for print systems.

8. REFERENCES

- [1] R. E. Albright and T. Kappel, "Roadmapping in the corporation," *IEEE Engineering Management Review*, vol. 31(3), pp. 31-40, 2003.
- [2] M. Ali-Babar, R. C. d. Boer, T. Dingsøyr, and R. Farenhorst, "Architectural Knowledge Management Strategies: Approaches in Research and Industry," in *Second Workshop on SHaring and Reusing architectural Knowledge - Architecture, Rationale, and Design Intent (SHARK/ADI 2007)*, 2007.
- [3] G. Böckle, P. C. Clements, J. D. McGregor, D. Muthig, and K. Schmid, "Calculating ROI for Software Product Lines," *IEEE Software*, vol. 21(3), pp. 23-31, 2004.
- [4] K. C. Desouza, Y. Awazu, and P. Baloh, "Managing Knowledge in Global Software Development Efforts: Issues and Practices," *Software, IEEE*, vol. 23(5), pp. 30-37, 2006.
- [5] N. M. Dixon, *Common knowledge: how companies thrive by sharing what they know*, 1st ed.: Harvard Business Press, 2000.
- [6] T. Gerrits, P. Nacken, A. Tang, and H. v. Vliet, "Expectations in Architecture Knowledge Sharing," in *Practical Product Lines Conference*, 2010.
- [7] P. Groenvelde, "Roadmapping Integrates Business and Technology," *Research Technology Management*, vol. 50(6), pp. 49-58, 2007.
- [8] A. Jansen, J. v. d. Ven, P. Avgeriou, and D. K. Hammer, "Tool Support for Architectural Decisions," in *6th Working IEEE / IFIP Conference on Software Architecture (WICSA 2007)*, 2007, p. 4.
- [9] S. Jantunen and K. Smolander, "Challenges of Knowledge and Collaboration in Roadmapping," in *International Workshop on Software Product Management (RE'06)*, 2006, pp. 19-26.
- [10] I. John and M. Eisenbarth, "A decade of scoping: a survey," in *Proceedings of the 13th International Software Product Line Conference (SPLC)*, 2009, pp. 31-40.
- [11] T. A. Kappel, "Perspectives on roadmaps: how organizations talk about the future " *Journal of Product Innovation Management*, vol. 18(1), pp. 39-50, 2001.
- [12] I. Nonaka, "A Dynamic Theory of Organizational Knowledge Creation," *Organization Science*, vol. 5(1), pp. 14-37, 1994.
- [13] R. Phaal, C. Farrukh, and D. Probert, "Technology Roadmapping: linking technology resources to business objectives," University of Cambridge, 2001.
- [14] R. Phaal, C. J. P. Farrukh, and D. R. Probert, "Technology roadmapping—A planning framework for evolution and revolution " *Technological Forecasting and Social Change*, vol. 71(1-2), pp. 5-26, 2004.
- [15] R. Phaal, C. J. P. Farrukh, and D. R. Probert, "Developing a technology roadmapping System," in *Portland International Conference on Management and Technology (PICMET)*, 2005, pp. 99-111.
- [16] E. T. Stringer, *Action Research*, Third ed.: Sage, 2007.
- [17] A. Tang, P. Liang, V. Clerc, and H. v. Vliet, "Supporting Co-evolving Architectural Requirements and Design through Traceability and Reasoning," in *Relating Software Requirements and Software Architecture*, P. Avgeriou, P. Lago, J. Grundy, and I. Mistrik, Eds., 2011.