Process for Developing an Initial Version of a Shared Platform

by

Andrey Kornienko

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Applied Science

in

Technology Innovation Management

Carleton University
Ottawa, Ontario

Your file Votre référence
ISBN: 978-0-494-94644-2

Our file Notre référence
ISBN: 978-0-494-94644-2

Abstract

For most start-up companies it is not feasible to independently develop products. Instead, it is more realistic to share in the development of core assets (shared platform) as a member of a collective, and focus on differentiators independently. This research looks at how to jumpstart a collective.

In order to attract members, an initial version of a shared platform (credible commitment) that demonstrates the viability of the platform, needs to be built. This research proposes a conceptual process for building initial version of a shared platform. The process has been developed by extending traditional software product line engineering approach to serve the core asset needs of multiple platform owners. The process surveys existing market offers to build a feature model of the platform. Lastly, this research identifies several types of feature model evolutions.

This research will aid entrepreneurs and platform leaders to demonstrate viability of their platform to potential collective members.

Acknowledgements

I would like to thank my supervisor, Prof. Michael Weiss, for his tireless guidance throughout the process of researching and writing this thesis. With his support I was able to hone my research skills, persevere through setbacks and reach my goal of a Master's degree. I will always be grateful for his endless generosity with his time, patience and enthusiasm.

My sincerest gratitude to Ingrid Bongartz, PhD, my director at Kinaxis Corporations, for her faith in my ability to successfully pursue a Masters' Degree. The moral support of the entire Kinaxis team has had the most positive affect on my attitude towards combining work and studies. A humble thank you to Kinaxis management team for the company's financial support of my studies.

The research of this thesis has contributed to my crossing paths with Nadia Noori, a mentor and a friend. My genuine thanks to Nadia for her time and advice during the writing of this thesis.

I would like to thank my wife Nataliya and my daughter Christina Grace for the love, strength and patience that helped me carry on my studies. Special thanks to my wife for her guidance with the writing process and the endless hours of reviewing and editing this thesis.

To my parents in Russia, who have always supported by dreams of achievement; and more so to my father, Prof. Kornienko, for setting an example of value of research and education.

Last but not least, to my class mates and professors in the TIM program, for being a source inspiration throughout the course of the program.

Table of Contents

List of Tables

List of Figures

List of Appendices

1 Chapter: Introduction

Today, expectations for development efficiency are becoming increasingly high. Organizations are always on the lookout for ways to decrease time to market and development costs of their products. The traditional approach to development has been described as the 80/20 rule (Weiss, 2011), where 80% of resources are spent on activities that do not differentiate a company from its competitors. And only 20% of resources are dedicated to unique components that will attract customers (Weiss, 2011).

A solution is to form a collective—shared ownership and development of the core assets by a group of companies (Weiss, 2011). The resulting code is a shared platform. The shared platform is made available to all participants of the process and external organizations that wish to use it.

The reason for creating such a platform is that the companies (members of the collective) can focus on their differentiators. In terms of the life cycle of the technology, this is not the early stage where lots of innovation and different ways of solving the same problem, but the stage when a dominant design has already evolved. The platform contains that dominant design.

This thesis is motivated by the success that IBM achieved with the Eclipse project and accomplishing the similar goals of building a shared platform and a community around said platform using the limited resources of a group of small companies.

The Eclipse platform is an open software platform. It is managed and developed by Eclipse open source community (Smith & Milinkovich, 2007). The Eclipse project was started in 2001 as a spin-out of technology that IBM had acquired from Object

Technology International. "Spin-out" is referred to a case where a company externalizes an internal development project (West & Gallagher, 2006).

Initially, the Eclipse community was primarily made up of IBM and its partners. In 2004 the Eclipse Foundation - an independent, non-profit governance body, was created. Thus, IBM relinquished its control over the project and allowed other players, including IBM's competitors, to become equal members of the community (Spaeth et al., 2010).The Eclipse platform has evolved from an integration platform for tools (des Rivieres & Wiegand, 2004) to a general platform for integrating applications and services (Gruber et al., 2005).

Emerging companies are under immense time and resource pressures to establish positive cash flow and are unable to dedicate sufficient resources to activities that do not differentiate them from the competition. The need to drive down costs and time to market has motivated companies to seek other avenues to acquire the non-differentiating parts of their core assets elsewhere, for example, through open source software. Through active participation in the collaborative process, organizations are able to create a product that they can effectively tailor to build their individual market offers.

For the Eclipse project, IBM externalized already existing technology. From the beginning IBM had vast internal resources to support and evolve the Eclipse platform. For small companies the question becomes how to create a community around a platform without having a lot of resources and/or existing technology.

Research of open source software has shown that "an open source project must start with a Credible Promise" (Weiss, 2009). Without a credible promise the project will fail to attract developers. Further, sufficient functionality needs to be present to encourage other

developers to build on the open source project (Weiss, 2009). So in order to attract potential collective members we need to build an initial version of a shared platform (credible commitment).

The goal of creating an initial shared platform is to demonstrate to potential collective members that the platform is viable. By a viable platform (as similar to minimum viable product) we mean, a platform with a set of features that meets the basic needs of collective members and will gain their commitment in developing the platform.

In the absence of well-defined requirements and commitment of potential members of the collective, the approach presented here is based on analysis of a market segment to identify features/requirements/needs that are addressed by current competitors and from this analysis building an initial version of a shared platform. Majority of the existing products on the market are likely not modular. Some of the reasons for them to be not modular are: they were developed as standalone systems, not platforms; for performance reasons, they were developed as integrated systems; or the requirements were not yet well-understood, so they were built as an integrated system. However, a shared platform must be modular by necessity to allow collective members to use different subsets of the platform to build their market offers. Modularity of architecture means that the platform is designed in terms of components and their interdependencies. Modular architecture allows users to design individual modules separately and experiment with them without impacting the system as a whole. A system that is modular in architecture will be able to attract more developers (Baldwin & Clark, 2006).

The goal of this research is to define a process of building an initial version of a shared platform. The platform needs to be evolvable by the uncertainty in requirements (we only

know the features of current market offers, but do not know what features will be required), and by the uncertainty in technology (we know that as technology evolves, the platform needs to evolve as well).

1.1 Objectives

The objective for this research is to define a process to create an initial version (requirements view) of a shared platform designed with evolution in mind.

By evolution we mean that we want to design the platform that will not only meet current needs, but will be flexible to the collective's future needs. To describe the platform we will use a feature model. This term is coming from software product line engineering. A feature model is a set of features and their interdependencies (for definition, please see 2.2.2 Feature Modeling).

1.2 Deliverables

The following are the deliverables of the research:

- Process of creating a feature model for an initial shared platform
- Feature model for an initial version of a shared platform of the example
- List of changes to the feature model and more importantly, types of changes to the feature model

1.3 Relevance

We believe that this research is relevant to two groups: entrepreneurs and potential platform leaders; and academia.

Entrepreneurs and potential platform leaders need to understand how to capture the needs/requirements to build an initial shared platform or credible commitment. Research of open source software has shown that "an open source project must start with a Credible Promise" (Weiss, 2009). Without a credible promise the project will fail to attract developers. Further, sufficient functionality needs to be present to encourage other developers to build on the open source project (Weiss, 2009).

Researchers need to find a way of managing evolution of a shared platform. Gawer & Cusumano (2012) argue that current research does not fully understand how industry platforms emerge, as researchers assume that the platform already exists.

1.4 Contribution

This research makes contribution to the body of knowledge for:

- Entrepreneurs and potential platform leaders: by using the process they will be able to attract members to their platform by demonstrating that the platform is viable.
- Researchers: the process extends traditional software product line engineering in the following way:
    - Designed to serve the core asset needs of multiple organizations (not owned by one, but by many organizations)
    - Includes surveying existing applications/products
    - Designed with future evolution in mind

1.5 Organization

The thesis is organized into seven chapters. Chapter 1 presents the introduction. Chapter 2 covers the review of the relevant literature. Chapter 3 describes the method used to produce deliverables of this research. Chapter 4 describes data collection and data analysis. Chapter 5 describes the results of this research. Chapter 6 represents the discussion of the results. Finally, Chapter 7 provides the conclusions, describes the limitations of the research, and identifies opportunities for future research.

2 Chapter: Literature Review

The review of the literature is organized into two streams: Platforms, Software Ecosystems and Collectives (Section 2.1), and Software Product Line Engineering, Feature Modeling and Evolution (Section 2.2). Section 2.3 describes the lessons learned from the literature review. The table below provides a summary of the literature review streams and key references.

Table 1 Literature Review Streams

[illegible]	[illegible]	[illegible]
Platforms, Software Ecosystems and Collectives	• Majority of platforms are opened by established organizations when platform already exist and successful • Mostly in software ecosystem a platform is owned by one organization • Collectives can be started by small organizations to develop initial core assets • In a collective your benefits are correlated with contribution level	Bosch (2009) Bosch & Bosch-Sijtsema (2010) Eisenmann, Parker, & van Alstyne (2009) Gawer & Cusumano (2012) Weiss (2011)
Software Product Line Engineering, Feature Modelling, Evolution	• Product Line consists of core assets and applications • Market Driven vs Technology Driven • Designing with evolution in mind is not common practice in Product Line Engineering	Gomaa & Webber (2004) Kang et al. (1990) Kang et al. (2002) Kotonya & Lee (2010) Medeiro, de Almeida & Meira (2009) Voget & Becker (2002)

2.1 Platforms, Software Ecosystems and Collectives

2.1.1 Platforms

Platforms have become a popular way for organizations to build their product base around reusable core assets. Internal platforms are core assets that are shared between products in an organization's product line (Gawer & Cusumano, 2012). Figure 1 shows a platform (core assets) that is shared between multiple products of one organization, the platform is owned by one organization.

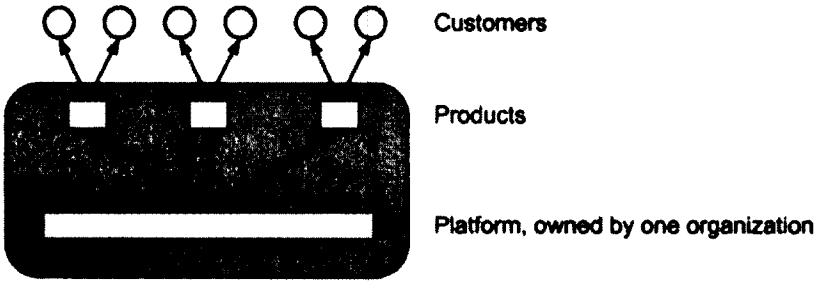

Figure 1 Platform owned by one organization

In terms of business strategy, it has become common to utilize "platform thinking" to understand commonalities between an organization's products, customers and internal processes, and to use these commonalities to increase profitability (Sawhney, 1998).

External platforms are similar to internal platforms. However, instead of providing shared assets between internal products, external platform is the foundation of products for multiple external organizations (Gawer & Cusumano (2012)). Boudreau & Hagiu (2009) concur with Gawer and Cusumano and define platforms in a very similar way, with the only difference that they do not distinguish between internal and external platforms. More specifically, Boudreau & Hagiu (2009) define platforms as core assets that are

shared by community members who extend platform functionality to build unique market offers.

At its inception, software platform development was a relatively rigid process. This plan-based approach was organized as a value chair, with minimal cross functional interactions and responsibilities. As technology became more advanced, the needs of users changed dramatically. Today, users demand faster feedback and stronger influence on the final product and the development process. To keep the client base engaged, organizations have to become more receptive to external input and more efficient at getting the final product to market. (Hanssen, 2010)

2.1.2 Software Ecosystems, Definition and Types

Bosch & Bosch-Sijtsema (2010) take the definition of a shared platform and extend it to describe the community of organizations that use a shared platform. They define this community as a software ecosystem, where a software platform is used by members of the community to create their own market offers. Bosch (2009) also provides a more technical definition of a software ecosystem: "a platform, products build on top of platform and application build on top of platform that extend the product with functionality develop by external developer". Bosch (2009) describes existence of three types of software ecosystems: the operating system-centric software ecosystem, application-centric software ecosystems and end-user programming software ecosystems. The operating system-centric software ecosystems are independent and expect that a multitude of applications will be developed to operate within their environment and will offer value to the customer. The success of the operating system-centric software

ecosystem is defined by the number and sale of applications that will operate within their environment (Bosch, 2009).

The application-centric software ecosystems come from the success of an application, whereby the owner organization is not able to handle the large volume of customization requests from its clients. As a result, the owner organization releases the application's APIs so that the users are able to develop their own in-house customizations (Bosch, 2009).

The end-user programming software ecosystems are instinctive environments created with end user in mind that allow the end user to build their own application. These software ecosystems are extremely labour intense, the tools available to end user are narrow (ex. Blogger.com). (Bosch, 2009)

2.1.3 Openness of Platforms

The above defined ecosystem types speak to the difference in the degree of openness of platforms to outsiders. Gawer & Cusumano (2012) extend this concept and define the characteristics of openness of a software platform as: the level of access to information on interfaces, the type of rules governing use of the platform, or cost of access.

Eisenmann, Parker & van Alstyne (2009) also speak to the measure of openness or degree of sharing of platforms. They characterize this practice by saying that platforms may be opened with and without restrictions. The measure of restrictions is judged by four characteristics: (1) openness to demand-side users (or end users); (2) openness to supply-side users, those who provide complements to the platform; (3) openness to platform providers who support the maintenance of the platform for the end user; (4) and

openness to platform owners, those who create the platform and oversee its strategic development.

When platform owner organizations open their platforms to external developers, the result is a shift in ownership structure of the final product. By opening the platform the owner organization is seeking new opportunities in meeting the needs of own users and maintaining the relevance of their platform in the market.

Open platforms are associated with "network effects": the platform becomes more and more valuable to its original owner and the members of the ecosystem the more members join the group. This is due to the increased access the members have to each other and each other's products which complement the platform. (Gawer & Cusumano, 2012)

A very significant benefit of opening an internal platform to external users is that ideas from external sources became available for an organization to capitalize on. Through open innovation the tasks of research and development became intertwined and allow external stakeholders to drive innovation. (Hanssen, 2010)

With these new opportunities come additional costs. In addition to costs of the technical aspect of opening the platform, the owner organization also incurs costs of managing their relationship with the community of external developers. Therefore, until recently, only well-established organizations were able to afford the costs of opening a platform to create a software ecosystem. (Weiss, 2011)

2.1.4 Extension/Complement Quality

Organizations need to be careful when opening platforms, as in addition to extra costs, there are risks to quality of complements associated with the process. (Thomke & von Hippel, 2002; Gawer & Cusumano, 2008; Boudreau & Hagiu, 2009)

Boudreau and Hagiu (2009) state that it is the role of the platform owner organizations to ensure that negative effects are reduced or eliminated entirely. The dependencies that are created between extensions/complements did not exist prior to the creation of an ecosystem and pose risks to the already established processes and patterns. Platform owner organizations need to carefully oversee both internal and external development process to ensure product quality does not slide. (Bosch & Bosch-Sijtsema, 2010)

There are various approaches to managing quality which will have different effects on four main characteristics of an ecosystem: (1) governance, (2) openness, (3) quality and (4) flow of ideas. Internal extensions are controlled for quality more than external extensions. Low quality of internal extensions will have a very significant impact on the platform and on all dependent internal extensions/complements, while low quality of external extensions will have minimal effect on the platform. By controlling the flow of ideas, organizations are able to ensure high quality of internal extensions; or by allowing the flow of ideas, organizations are not able to supervise the quality of extensions (Noori & Weiss, 2012).

Owner organizations can also regulate the ecosystem environment through such measures as pricing, legal agreements, toolkits, design rules and sandboxes (Hagiu, 2009; Bosch & Bosch-Sijtsema, 2010).

2.1.5 Collectives

Previous discussions focused on opening an existing platform by an already established organization. However, with the slowdown of economy and move towards small business development, the methodology of opening an existing platform is outside the realm of possibilities for small companies. Without a solid platform to contribute to, small organizations are not able to reap the same benefits as those in an ecosystem with an established platform.

By focusing resources on creating own core assets, the small organization will be investing large amounts of money into a portion of their market offer that will not differentiate them from their competition. In the process, the small company will compromise viability of the organization through erroneous distribution of company resources.

The idea of a software collective was born to allow small organization to reap the benefits of shared core asset development. In contrast to the ecosystem that revolves around an existing strong product, the purpose of a software collective is to share in the development of the core platform. As a group, small organizations are able to deliver a complete product to their customers or use the platform to build their individual public offers on its foundation. Members of the collective share in the risks and the benefits of having access to the basic code and the increased quality that results from collaborative development (Weiss, 2011).

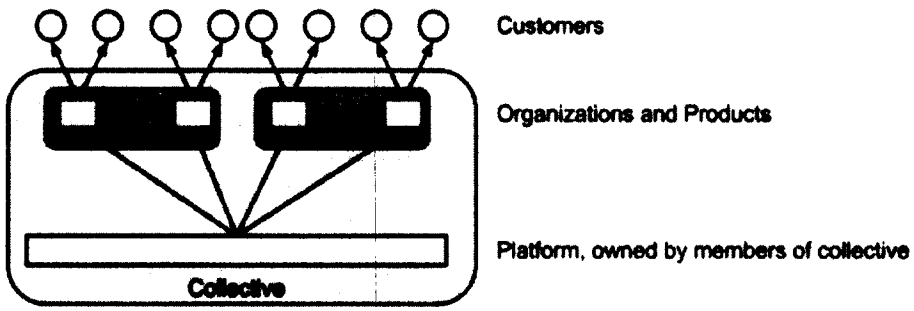

Figure 2 Platform owned by members of collective

Figure 2 shows a shared platform that is owned and developed by members of a collective; members of the collective build their individual market offers (products) on top of the platform and sell those products to customers. Members of a collective benefit from being part of the effort by getting access to the source code and adding own unique proprietary value. By sharing in the development of core assets, organizations are able to dedicate more resources to differentiating themselves from the competition (Weiss, 2011).

To reap the most benefits organizations need to commit to a collaborative effort and invest in its success. Studies show that collective members who contribute significantly to the development of the shared platform are able to benefit the most from the process (von Hippel & von Krogh, 2003). Through strong leadership and meaningful contributions, collective member, and the leader organization specifically, are able to benefit most from the functionality of the product (Weiss, 2011).

While the benefits are well defined and understood, there are great challenges in organizing a collective. The main challenges are to identify what the product requirements are and how to build a credible commitment with interested parties.

One of the methods to identify requirements for the shared platform is to survey collective members themselves. However, this is not feasible to do in the initial stages of

the process as the leader organization might not have access to organizations who are interested in participating and the organizations themselves might not have a clear understanding of the product. Involving potential collective members at a very early stage can have serious negative consequences for the project. Lack of a clearly viable idea can cause members to lose interest in the product.

Similarly, leader organization may approach too many potential members or attract a group of members without complementary skills. Wrong choices in member group can lead to disfuctionality in the collective, waste of resources and failure of the project.

In summary, software ecosystem and collectives can be successful tools in developing products efficiently, if the development process is strategic and well controlled.

2.2 Product Line Engineering, Feature Modeling and Evolution

2.2.1 Software Product Line Engineering

Software Product Line Engineering (SPLE) is a well-known approach for building highly customizable platforms. Traditional Software Product Line Engineering is applied within the platform-owner organization where many of the product functionalities are shared between products. In most cases it is engineering driven, i.e. based on introspection about what a product should be able to do.

The goal of Software Product Line Engineering is to develop core assets that can be reused. This goal is achieved by identifying product line commonalities and variabilities to specify product line requirements.

In traditional product lines, "a central unit is responsible for integrating and testing reusable assets and making them available to product groups (Bosch & Bosch-Sijtsema,

2010). The central unit is usually able to control requirements, product roadmaps, features, and variability.

In a software ecosystem, a product line is shared with external partners; the "central unit" within the owner organization is no longer responsible for the product line. The final product is assembled by external developers, who are the customer (Bosch, 2009). Opening a product line to external developers allows the external developers to participate in the creation of products that best suit their needs and allows external developers to claim ownership to a portion of the product (Bosch, 2009). Hanssen (2010) claims that external developers can, not only customize the product to their needs, but contribute core assets to the product line.

While product line engineering has been around for some time, today, there is an increased interest in researching reusable and dynamically reconfigurable core assets – dynamic software product lines (DSPLs), which are an extension of traditional product lines applied to a new type of product. "Dynamic product reconfiguration involves making changes to a deployed product configuration at runtime. A service-oriented product line (SOPL) is a DSPL application domain that is built on services and service-oriented architecture" (Kotonya & Lee, 2010).

Service orientation's main goal is system agility, where software itself is the service provided. To cope with ever-changing business environments, runtime flexibility of applications is crucial (for example, adding third-party functionality on demand). To achieve these goals, special attention must be paid to design considerations and constraints for matching between service consumers and providers (Kotonya & Lee, 2010).

"Service-based systems are distributed and composed of various services that can be discovered and replaced at runtime." Service-based systems view qualities as constraints on functionality and require tools which guarantee system quality at runtime (Kotonya, Lee, & Robinson, 2009). Quality issues are generally addressed statically during system design and implementation. At first, quality of services is defined in terms of features and a limit is placed on available resources per product. This information is used when product starts to negotiate with service providers to select available services at runtime.

In the past, the products were developed to be static; all possible variations were built into the product at the initiation stage before being delivered to the customer. This made the process of making changes to the product difficult for the customer. More recently, the need for products that allow for variability after product is delivered to the customer has increased exponentially (Kotonya & Lee, 2010).

The process of changing functionality of a deployed software product during runtime is described as dynamic product reconfiguration. Possible changes that can be implemented at runtime include, but are not limited to: adding, deleting, or modifying product features and making changes to architectural structures (Kotonya & Lee, 2010).

2.2.2 Feature Modeling

The activities of product line engineering process have two stages: domain and application engineering. At the domain engineering stage activities involved are: requirements analysis, variability analysis and design of architecture (Lee et al., 2008; Medeiros et al., 2009). Domain engineering stage looks at feature models to identify

points of variability. FODA describes a feature as "a prominent and distinctive user visible characteristic of a system".

A feature model "identifies product line's features by identifying externally visible product characteristics in a product line and organizing them into a model". (Kotonya & Lee, 2010) A feature model represents the following: (1) standard features; (2) relationships between standard features; (3) logical grouping of features; (4) alternative or optional features of each grouping (FODA). By analyzing the commonality and variability of products, functions, objects, etc can be understood in their reusable form (FODA).

A feature model is essentially a layered model. It allows you to discover layers of abstraction.

"Feature modeling is the activity of identifying externally visible characteristics of products in a product line and organizing them into a feature model". Features that are shared between multiple products are identified as common or mandatory, while all other features fall either into an option or alternative feature category. A feature diagram systematizes all features into three distinct categories: composed-of, generalization/specialization, and implemented-by. Composition rules are attached with the feature model and identify dependencies and exclusion relationships between options and alternative features. (Kotonya & Lee, 2010)

In the strategic business sense, a feature model is a communication tool. By providing a breakdown of common products, a feature model allows cross functional, and cross area teams to discuss the model and core assets with use of a common language. "To the users, the feature model shows what the standard features are, what other features they

can choose, and when they can choose them. To the developers, the feature model indicates what needs to be parameterized in the other models and the software architecture, and how the parameterization should be done" (FODA).

What the feature model does not do, in comparison with other modeling techniques, is describe all details of products. (FODA)

2.2.3 Variability management

As stated above, a feature model describes both commonalities and variabilities present in a product line. Variation Point Model (VPM) is an approach to modeling variability and builds on the basic definition of variability: a variation point identified one or more locations at which the variation will occur" (Jacobson, Griss & Jonsson, 1997). The VPM approach provides a mechanism for modeling variability for adaptive and evolvable software product lines. There are four different approaches to modeling variability: (1) parameterization; (2) information hiding; (3) inheritance; (4) variation points. (Gomaa & Webber, 2004)

Modeling variability via the parameterization approach results in a very short time to market for products. The core assets work within the existing architecture and only parameters to be parameterized are made available to the re-user. The parameterization allows the reuser to populate attributes and is limited in that no functionality can change. Using the parameterization approach successfully to complete more complicated tasks is not a viable solution. Gomaa & Webber (2004) suggest use of other approaches in such cases.

The information hiding approach provides higher variability than parameterization and a relatively short time to market. The variability of the information hiding approach is higher as both the functionality and parameters can be varied. The choices are still limited as the information hiding approach allows the reuser to choose from a set, and usually limited, number of choices. While the implementations will be variable and depend of reuser choices, the interface will be common among new applications (Gomaa & Webber, 2004).

The inheritance approach builds upon the information hiding approach, in that while the reuser may choose variants from a limited set it also allows the core asset developer to manager the variation points. The developer has access to a larger (no longer limited) number of variants. This is accomplished by extending the superclass interface in variant subclasses. The inheritance approach also allows the developer to make changes to the core assets (Gomaa & Webber, 2004).

Lastly, the approach that allows the most variability and flexibility in creating an application is the variation points approach. The inheritance approach already allows for some flexibility here, and the variation points approach expands on this. The variation points approach allows the reuser to create unique variants by registering them with the common core assets at runtime. The variation points approach allows for minimum development of the core assets, but requires high investments into development of variants as these are not chosen from core assets, but created individually. As a result, the time to market for application is increased. The variation points approach is recommended for products in a product line that are low on commonality (Gomaa & Webber, 2004).

The owner must understand the different approaches to be able to choose the most appropriate and resource saving method for their product. "Variability management... is fundamental to run a sustainable successful product line in practice" (Lee & Muthig, 2006).

2.2.4 Market Driven PLE

Kang et al (2002) describe the importance of marketing and product plans in developing high value product line assets. Lee et al (2009) also speak to the importance of market analysis of competitor's applications in identifying current trends and understand market needs.

While product line requirements are essential to product asset development, they are insufficient in and of themselves. Marketing and product plans are used as key design drivers in identifying crucial components in product line architecture and design.

Kang et al (2002) identify five reasons for using marketing and product plans as a crucial design driver:

1. Generally there is lack of cross-functionality in organizations. Traditionally, there is lack in communication between marketing and product plan developers and product developers. This may lead to important information/requirement/ parameter being dropped from product design.
2. When product assets are developed for a product line with customers in multiple market segments, marketing and product plans provide information on commonalities and variabilities of needs and characteristics of users in each segment.

3. There are three important pieces of information that must be gathered from marketing and product plans before product design can begin: (1) features that will be included in the product; (2) how the features/product will be delivered to customers during installation and adaptation, (3) what extensions are allowed to customers. Analyses of these facts should inform decisions during the product design stage.
4. Product delivery method affects engineering feasibility and associated costs and reflected in product design.
5. By introducing marketing plan information early in the asset development process, analysis of technical feasibility and financial implications are recycled and used back in the product planning and marketing plans.

Market driven approach to product line engineering will support business goals of the owner organization, will ensure that product satisfies needs of customers, and will minimize development costs and time to market.

2.2.5 Product Line Evolution

Developing product lines and product families with high reusability characteristics requires that the domain of these products be stable and well defined. In domain engineering, domain is a set of concepts and terminology of a certain area of study and knowledge of product development in that area. Domain in itself are the parameters that identify what and how a software system is to be built, and is expected to be known by its mere definition.

Vranic & Marko (2006) discuss how the traditional organized development methods can be used for domains that are yet to be concretely identified. They claim that stable domains are not always readily available in practice, and that there is great value in understanding how to develop software when characteristics of a domain are yet to be understood.

Voget & Becker (2002) describe the risks associated with unstable or immature domains in the following ways:

1. Vagueness – difficulty in deciding on domain level prevents effective decision-making on which core assets need to be available during what part of the product line's life cycle.
2. Extensions – identifications of which are critical
3. Reductions – elements of the product line disappear before they become obsolete
4. Product and market related risks—new products or new functions to existing products are developed first for the high end market. Only once they become established technologies, they are made available to the general market. The risk is in keeping the general market engaged during the time the technology becomes mainstream.
5. Knowledge related risks—new technologies have very few customers, and so collection of requirements and client needs is not exact. As a result, there may be misunderstandings between product and client. Developing products that are not fully understood always entails a high level of risk.

Vranic & Marko developed an approach that meets the needs of a project in an environment of heterogeneous information resources. By taking known and understood

factors/characteristics of a domain, project objectives are feasible to achieve in spite of the fact that the domain is under constant change. The objective of a project may be achieved by several steps (Vranic & Marko, 2006):

- Create an initial feature model using use cases
- Identify archetypal entities of the domain and relationship between them through analyzing main concepts and corresponding use cases
- Refine the feature model – separation of concept into independent vs dependent based on identified archetypal entities
- Generalize use cases

The risks of developing product lines in domain under construction are the risk of uncertain technological evolutions. By isolating unstable sub-domains and excluding them from the project line, while simultaneously stabilizing such subdomains in order to preserve them as part of the product line, evolution can be factored into the development process in a strategic way. By understanding specifics of a subdomain, developers are able to apply the same concepts and principles to the entire domain.

Product lines are complex entities that require rapt attention when it comes to variability and evolution. Changes implemented within a product line are part of a long term strategic plan implemented by company management. To more efficiently and strategically plan product lines, management must keep in mind that the products will undergo evolution. Product evolution is part of the long term business strategy and is a cross functional activity that includes marketing, management and development (Lee & Muthig, 2006). A sign of good planning is a product line that is designed with future requirements in mind.

As discussed in the previous section, variability management is important for sustainability of the product line (Lee & Muthig, 2006). Sustainability as a characteristic of a product line, states that the product line will successfully adapt and evolve over time. Therefore we can conclude that, designing with variability in mind leads to managing future evolution of the feature model and the product line (Gomaa & Webber, 2004).

Botterweck et al (2010) speak to evolution in the context of feature-oriented model-driven product line engineering and propose the use of an Evolution Feature Model (EvoFM). The EvoFM systematizes information on addition and subtraction of feature from the feature model and provides a timeline for evolution activities. EvoFM does not replace the feature model, but is an additional tool that complements the feature model.

The simplest evolutions are changes to features, such as add, remove and replace. There are also additional actions that may take place, such as move, replace, rename or change of feature type. Following are a list of things to keep in mind when thinking about feature model and EvoFM (Botterweck et al, 2010):

- EvoFM does not speak to the variability within the product line, it only describes the options for evolution changes
- Mapping allows constructing a feature model from a given instance of EvoFM configuration. A specific configuration states presence or absence of feature in the feature model
- If a feature in EvoFM is specified as mandatory, than it is mandatory in the feature model
- EvoFM does not describe an existing feature's constraints in the product line
- Optional feature in EvoFM describes variability within the feature model

- Each configuration of EvoFM represents one evolution step of the product

2.3 Lessons Learned

Lessons learned from the literature are below.

- Very limited or no literature on how to start developing an initial version of a shared platform.
- Rationales for a company to open up their internal platform and share the development costs and benefits with external developers are:
    - To retain current customers and expand market share by meeting the growing needs and demands of clients through expansion of functionality of their products
    - To meet the growing needs and demands of clients in a reasonable timeframe as not to lose market share. A lack of resources may be the core reason for opening up an internal platform, if an organization relied solely on own resources there would be a significant delay in delivery and potential loss of customers.
    - To meet current trend and customer expectation for custom tailored products. Today this trend applies to all products/services regardless of their size or complexity. Customization requires a significant investment in R&D, something that many companies are not able to finance independently.
- A company may open their platform to four different kinds of developers and use public or platform regulators to control the ecosystem environment

- There are different levels of openness of a platform.
    - As the platform is more open to external developers, the less control the owner organization has over the development process.
    - The benefit of less control is increased flexibility that allows room for unique solutions.
- There are significant challenges with opening up an internal platform to external developers.
    - Maintain high degree of product and development process quality through regulation
    - Incur costs associated with managing the relationships with the members of the newly formed ecosystem
- Managing quality is tied strongly to the ecosystem environment, the way it is governed, how it is opened, who enters the ecosystem and how the flow of ideas is managed.
- Creating a software product line, and even more so a software ecosystem, increases the effort required for software development by as much as 1.5-3 times.
- Software Product Line Engineering mostly applied in stable domains
- Designing with evolution in mind is not a common practice in Software Product Line Engineering
- A collective is set up when a group of organizations wants to achieve a goal they cannot achieve on their own

3 Chapter: Research Method

This chapter describes the method that was used to produce the deliverables of this research. It is organized in two sections. Section 3.1 describes the unit of analysis. Section 3.2 describes the method and steps undertaken to produce deliverables of the research.

3.1 Unit of Analysis

The unit of analysis is an application feature. A feature is a characteristic of a system which is of interest to a stakeholder (Czarnecki & Eisenecker, 2000). Of our particular interest are high level features or functional features, i.e. features that are used and are understandable by end users of an application.

Features were extracted from the descriptions provided by the vendor. We tried to be objective during feature extraction and used only information that was provided by the vendor.

3.2 Steps

We used the case study research approach (Eisenhardt, 1989) due to the novelty of this research. The research method consists of the following steps:

Table 2 Research Method Steps

No	Step	Summary
1	Identification of cases	• Define population of software applications
2	Data Collection	• Define how to collect data
3	Initial feature model	• Select initial number of cases • Extract features from the applications • Build an initial feature model • Document all steps how features were extracted and built an initial feature model

4	Evolution of feature model	• Refine feature model, keep track of changes • Collect metrics of changes
5	Process definition	• Generalize what we did into the process

In the following sections these steps are discussed in more details.

3.2.1 Identification of cases

In this research nine cases of applications were examined. These cases were selected from web collaborative applications.

Collaborative applications allow individuals to coordinate their work in achieving common goals. Majority of teams using collaborative applications are geographically dispersed and require multi-functional tools to track task assignments and project schedules. The invaluable function of collaborative applications is their ability to provide various communication venues that allow for efficient and effective group decision making. Lastly, the software allows for tracking progress of deliverables, deadlines and rationales behind decision-making. Synchronous collaborative applications are collaborative applications that support instantaneous communication. The distinction brought in by the term "synchronous" signifies that interaction occurs instantly, in real-time. This is an important distinction, as some collaborative applications may allow sharing of information with a time lag between the deposit of information into the system and its retrieval.

A form of synchronous collaborative applications are web collaborative applications that combine multiple features such as private and public chats, video, voice, presentations, etc. Web collaborative applications have many uses and have been applied with great success for remote meetings and education. Web collaborative applications or web

conferencing solutions as they are also called, have become a popular tool for students to participate in class presentations and discussions remotely; as well as in business applications to host meetings.

Existing applications of the selected type were collected by using information available on the internet and from the literature (please see section 3.2.2 Data Collection).

As we discussed in introduction, a shared platform needs to be modular to successfully attract potential collective members. Most of the applications of the selected type are not modular, but many of them provide APIs, can be configured or customized. For this research we have categorized applications by this first dimension – how extensible/configurable/customizable an application is. The second dimension was chosen using the following rationale: as organizations joining collectives are usually small to mid-sized, at least at the initial stage of product development members of collective will target other small to mid-sized organizations as the target market for their products. This is a logical conclusion, as smaller organizations tend to not have capacity to deliver products on a magnitude required to meet the needs of enterprise sized organizations.

To summarize the above, we categorized applications using the following two dimensions:

1. How an application is extensible/configurable/customizable
2. Targeted clients - small/mid-sized companies or enterprises.

Based on this categorization, the set of applications was narrowed down to applications that are more extensible/configurable/customizable and that target small/mid companies.

From the selected list of applications nine applications were chosen: Table 3 represents the applications that were chosen for analysis.

Table 3 Applications included in the research:

No	Application	Website
1	Banckle Meeting	http://banckle.com/apps/meeting/overview.html
2	OpenMeetings	http://incubator.apache.org/openmeetings/
3	OpenTok	http://www.tokbox.com/opentok
4	WizIQ	http://www.wiziq.com/
5	ISL Groop	http://www.islonline.com/web-conference/
6	FuzeBox Meeting	http://www.fuzebox.com/products/fuzemeeting
7	eLecta Live	http://www.e-lecta.com/webconferencing.asp
8	iLinc	http://www.ilinc.com
9	Teleskill Live	http://www.teleskill.it/en/services/teleskill-videoconference-live

3.2.2 Data Collection

An initial list of web collaborative applications was created by using information available on the internet. This list of applications was narrowed down to nine applications using classification by two dimensions.

For each of the selected nine applications, information about its features available to the user was collected through websites and manuals. For the data collection we refer to features that are provided by vendors. This information was represented in the form of a table with the following columns:

- Feature – ability of an application to perform a specific function or functions.
- Description – is a detailed explanation of the functionality/ability of a capability. This column is populated with data from the website or a user/administrator guide. Word for word is used for most of the cases, with very occasional rephrasing for clarity.

3.2.3 Initial feature model

To create an initial feature model, features and their descriptions from the following applications were analyzed: Banckle Meeting, OpenMeetings, and OpenTok.

First, we extracted features for each application (we extracted only high level features that are used by the end users). Extracted features for each application were represented in a table. Second, we created a cross table with correlated features (as the same feature might have different names for different applications) from different applications. Based on this table an initial feature model was created.

A feature model is essentially a layered model. It allows you to discover layers of abstraction.

A feature model is represented in a diagram (tree) – a feature diagram (tree). Feature diagrams are a simple, well-understood, graphical notation for feature models. They use a hierarchical tree structure to present features and their child features as well as their cardinality.

We use a definition of a feature cardinality that is provided by Czarnecki & Kim, 2005: "feature cardinality is an interval of the form [m..n]. Feature cardinality denotes how many clones of the feature (with its entire sub-tree) can be included as children of the

feature's parent when specifying a concrete configuration." Two common feature cardinalities are [1..1] and [0..1], which represent a mandatory feature and an optional feature respectively.

The root of a tree is called – root feature. We will call children of the root feature – top level features. Any feature that is child of another feature is called – sub-feature.

Another definition that we will use is an attribute (feature attribute). A feature may contain a number of attributes. Feature attributes allow for more concise representation of a feature diagram. Each attribute has a name, a type (for example, string or number) and a value. Czarnecki et al. (2002) introduced attributes as a way to represent a choice of a value from a large or infinite domain such as integers or strings (you can also think about an attribute as a parameter of a feature).

Figure 3 represents an example of a feature model (you will find more description about this example in section 4.1 Initial feature model). WebCollabApp is a root feature; Session Management, Participants Management, Participants List, Room Management and Audio are top level features; Start Session, Finish Session and Join Session are sub-features of Session Management.

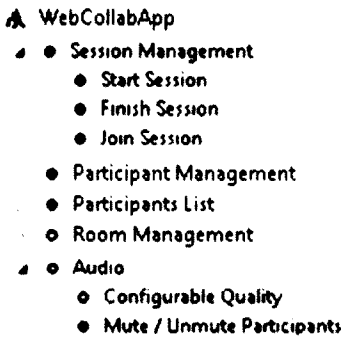

Figure 3 Example of Feature Model

To create a feature diagram "fmp" Eclipse plugin was used. This plugin was developed by Generative Software Development Lab at the University of Waterloo. Table 4 represents notation for feature modeling used in fmp (Hwan et al, 2005).

Table 4 Feature Modeling Notation

Icon	Description
	Root feature
F	Solitary feature *F* with feature cardinality [1..1] (i.e. mandatory feature)
G	Solitary feature *G* with feature cardinality [0..1] (i.e. optional feature)
[0..m] H	Solitary feature *H* with feature cardinality [0..m], $m > 1$ (i.e. optional cloneable feature)
[m..n] J	Solitary feature *J* with feature cardinality [m..n], $m > 0$ and $n > 1$ (i.e. mandatory cloneable feature)
K	Grouped feature *K* with feature group cardinality [0..1]
L	Grouped feature *L* with feature group cardinality [1..1]
M ('value' : T)	Feature *M* with attribute of type *T* and value of *value*
	Feature group with group cardinality <1-1> (i.e. exclusive-or group)
	Feature group with group cardinality <1-k>, *k* = group size (i.e. inclusive-or group)
<i-j>	Feature group with group cardinality <i-j>

3.2.4 Evolution of feature model

After an initial feature model was created additional applications were added for analysis: WizIQ, ISL Groop, FuzeBox Meeting, eLecta Live, iLinc and Teleskill Live. Based on this additional application the feature model was refined and changes to the feature model were captured. We also kept a metric of changes:

- Number of new top level features.

- Number of new sub-features
- Number of new attributes

For definition of "top level feature", "sub-feature" and "attribute" please see the previous section (3.2.3 Initial feature model).

3.2.5 Define the process

As was expected, after the previous steps were performed, it became evident that the steps were not specific to web collaborative applications themselves, except that our cases were drawn from this domain. As a result, all steps that were performed to obtain the final feature model can be generalized for use with other types of applications.

4 Chapter: Data Analysis

Chapter 4 describes analysis of the web collaborative applications in terms of their features available to the user.

This chapter is organized into two sections. Section 4.1 describes construction of an initial feature model; Section 4.2 shows evolution of the feature.

4.1 Initial Feature Model

An initial feature model was created by analyzing features of the first three applications: Banckle Meeting, OpenMeetings and OpenTok. Based on the information that was collected for each application we extracted features for each application. We tried to identify high level features that are visible to users or usable by users. After features were extracted a cross table with correlated features from different applications was created. Using this table an initial feature model was created.

4.1.1 Banckle Meeting

Banckle Meeting boasts ability to provide a feature-rich conferencing experience. The company's claim is that Banckle Meeting covers all collaborative requirements. It supports any flash-enabled browser and provides a separate application for Android based devices. Banckle Meeting works well for web meetings, scheduled meetings, eLearning sessions and webinars. Banckle Meeting provides a well-documented API. The application provides a user interface which is built inside one window.

Based on the information provided on their website (Appendix A.1) features in Table 5 were extracted.

Table 5 Banckle Meeting Extracted Features

Extracted Features	Description
Session Management • Start Session • Finish Session • Join Session	Session is a period of interaction between a group of people online. It's not described explicitly, but it's mentioned in the contexts of features. Session Management is an abstract feature that allows working with sessions.

Participant
• Name
• Role
  o Host
  o Presenter
  o Attendee
	An individual or group of individuals who enter the session via a single connection to the server. Participants may take on host, presenter and attendee roles over the session and to contribute to the session content at any given time or when it is their turn. Host participant is an initiator of a session. He/she can change roles of other participants.

Participants List • Add Participant • View Participants • Remove Participants • Participants Permission	Participants List is a feature that shows a list of individuals inside the sessions and allows for updates to their status. Permissions to access Whiteboard, Chat, Participant List and other features can be controlled by the Host.
Audio • Mute / Unmute Participants	This feature allows participant to talk between each other. From our understanding Banckle Meeting supports only voice over IP (no incoming calls from landline phone).
Video • FullHD or normal quality • Show All or Selected Participants	Banckle Meeting provides Video capability. It supports FullHD mode. It also has ability to show video of all or selected participants
Whiteboard • Upload slides • View uploaded slides • Multiple pages • Multiple drawing tools • Typing tools • Virtual laser pen	Whiteboard is a space on the user screen that is part of the software application. The space is separated from other features (such as chat, participant list, etc) and allows for a common area for content presentation and shared content modification.

• Annotation capabilities • Export as an image	
Screen sharing • Full screen • Particular area • Application • Remote control	Sharing of individual desktop screen with session participants. Shared screen function allows sessions participants to view information on the desktop of another participant and even receive remote control of the desktop.
Chat	Chat feature that is available to all session participants for use and view. Runs simultaneously with other features and allows additional avenue for communication.
File Depot • Upload files • Download files • Share files	File distribution feature that stores uploaded files and allows access to these for participants.
Polls and Votes • Configure questions • View results	To substitute roundtable voting or answering to questions, poll and votes feature allows for feedback to be received simultaneously to other features being run within the session. Statistical analysis of answers is provided automatically.
Scheduler • Select Participants • Reminders	Allows for scheduling of sessions with various lists of participants.
History / Archive • Record sessions • Browse recorded sessions • Play recorded session • Download recorded session • Browse saved chats • Download saved chats • Attendance reports • Files uploaded during sessions • Session notes	Stored historical data on past sessions that allows participants to access recorded instances of sessions, download information/files disseminated during the session and access attendance reports.
Raise Hand	Attract attention of presenter and receive control over audio function. This is a coordination feature.
Customizable Layout	Ability to move windows/features within the session environment

Encryption	Safety measure that protects information shared during a session from outsiders.

4.1.2 OpenMeetings

OpenMeetings is an open source web-conferencing solution with a wide range of functionalities, such as video conferencing, instant messaging, white board, collaborative document editing and other groupware tools. The solution is built on the Red5 Media server. As an open source solution, it allows the users to customize their web conferencing environment. It also provides a well-documented API. OpenMeetings provides a user interface which is built inside one window.

Based on the information provided on their website (Appendix A.2) features in Table 6 were extracted.

Table 6 OpenMeetings Extracted Features

Extracted Features	**Description**
Session Management • Start Session • Finish Session • Join Session	Control over access to and duration of sessions.

User Management
• Name
• Access Level
  o Administrator
  o Moderator
  o Simple
	Predefined profiles of participants and their roles and permissions within a session.
	Administrators are users who manage and configure the system (general parameters, rooms, users, etc.).
	Moderators are users who usually organizers of conferences. They have the right to do everything in a room and have the ability to give the rights to interact to other users in the room. They can also give the moderator status to a regular user of the room.
	Simple users are users who may enter a room and interact according to the rights granted to them and dependent on the type of room in which they are in

Participant
• Name
• Role
  o Moderator
  o Attendee
	An individual or group of individuals who enter the session via a single connection to the server.
	Participant is linked to User and initial role is inherited from User.

Participants List • Add Participant • View Participants • Remove Participants • Participants Permission	Participants List is a feature that shows a list of individuals inside the sessions and allows for updates to their status. Participants Permissions: • Allow/Deny moderation • Allow/Deny to draw on whiteboard • Allow/Deny screen-sharing/record screen • Allow/Deny Remote Control Screen • Give exclusive audio to others or self • (Re-) Start Audio, Video or Device settings

Room Management
• Name
• Type
  o Conference
  o Restricted
  o Interview
	Profile of a session with predefined chosen features, participants and permissions for participants.
	Conference room is a room where all participants can share their video and audio and access a whiteboard.
	Restricted room is a room where participants must ask permission to share their video
	Interview room is a room where only two participants can share their video and audio and it does not have a whiteboard.

Audio • Configurable quality	From our understanding. OpenMeetings supports only voice over IP. Quality of audio can be configured.
Video • Configurable quality • Camera resolutions: 4:3, 16:9 or 3:2	Communication via video camera that allows participants to view each other in real time. Video has a configurable quality and it supports different camera resolutions.
Whiteboard • View files from File Explorer • Drawing and typing • Multiple instances • Save into File Explorer	Whiteboard is a space on the user screen that is part of the software application. The space is separated from other features (such as chat, participant list, etc) and allows for a common area for content presentation and shared content modification.
Screen sharing • Full screen • Particular area • Configurable quality	Sharing of individual on location desktop screen with session participants. Shared screen function allows sessions participants to view information on the desktop of another participant and even receive remote

• Remote control	control of the desktop.
Chat	Chat feature that is available to all session participants for use and view. Runs simultaneously with other features and allows additional avenue for communication.
File Explorer • Upload files • Private and public drive	File distribution feature that stores uploaded files and allows access to these for participants
Polls and Votes • Configure up to 10 questions • View result	To substitute roundtable voting or answering to questions, poll and votes feature allows for feedback to be received simultaneously to other features being run within the session. Statistical analysis of answers is provided automatically.
Planner • Invite attendees • Notifications	Allows for scheduling of sessions with various lists of participants.
Record and Playback • Record sessions • Browse recorded sessions • Download recorded sessions • Watch recorded sessions	Stored historical data on past sessions that allows participants to access recorded instances of sessions.
Backup	Storing of all data associated with the system, which allows for re-install of software with all current data intact.

4.1.3 OpenTok

OpenTok is a unique video communications solution that allows the users to hold video conversations via a group video through any web or mobile device. OpenTok is provided by TokBox Inc. who claims to be a leader in global online video communications. This solution provides only two communication features audio and video and it does not provide features like whiteboard or chat. OpenTok does not provide a user interface. The

user interface must be developed by a third party with the help of API provided by OpenTok.

Based on the information provided on their website (Appendix A.3) feature in Table 7 were extracted.

Table 7 OpenTok Extracted Features

Extracted Features	**Description**
Session Management • Start Session • Finish Session • Join Session	Control over access to and duration of sessions.
Audio	OpenTok supports only voice over IP
Video	Communication via video camera that allows participants to view each other in real time.
History / Archive	Stored historical data on past sessions that allows participants to access recorded instances of sessions, download information/files disseminated during the session and access attendance reports.
Participant • Name	An individual or group of individuals who enter the session via a single connection to the server.
Participants List • Add Participant • View Participants • Remove Participants	Participant list is provided through API, it does not have any user interface.
Encryption	Safety measure that protects information shared during a session from outsiders. Please note, that video streams are not sent on a secure socket.

4.1.4 Feature Model

Based on these three applications a table of feature was created to capture similarities in features. This table is represented in Table 8. The first, the second and the third column represent extracted features from applications: Banckle Meeting, OpenMeetings and OpenTok respectfully.

Table 8 Cross table of features

Banckle Meeting	**OpenMeetings**	**OpenTok**
Session Management • Start Session • Finish Session • Join Session	Session Management • Start Session • Finish Session • Join Session	Session Management • Start Session • Finish Session • Join Session

Participant
• Name
• Role
  o Host
  o Presenter
  o Attendee
	Participant
	• Name
	• Role
	  o Moderator
	  o Attendee
		Participant
		• Name

Participants List • Add Participant • View Participants • Remove Participants • Participants Permission	Participants List • Add Participant • View Participants • Remove Participants • Participants Permission	Participants List • Add Participant • View Participants • Remove Participants
Audio • Mute / Unmute	Audio • Configurable quality	Audio
Video • FullHD mode is supported • Show All or Selected Participants	Video • Configurable quality • Camera resolutions: 4:3, 16:9 or 3:2	Video
Chat	Chat	

Whiteboard • Upload slides • View uploaded slides • Multiple pages • Multiple drawing tools • Typing tools • Virtual laser pen • Annotation capabilities • Export as an image	Whiteboard • View files from File Explorer • Drawing and typing • Multiple instances • Save into File Explorer	
Screen sharing • Full screen • Particular area • Application • Remote control	Screen sharing • Full screen • Particular area • Configurable quality • Remote control	
File Depot • Upload files • Download files • Share files	File Explorer • Upload files • Private and Public drive	
History / Archive • Record sessions • Browse recorded sessions • Play recorded session • Download recorded session • Browse saved chats • Download saved chats • Attendance reports • Files uploaded during sessions • Session notes	Record and Playback • Record sessions • Browse recorded sessions • Download recorded sessions • Watch recorded sessions	History /Archive
Scheduler • Select Participants • Reminders	Planner • Invite attendees • Notifications	
Polls and Votes • Configure questions • View results	Polls and Votes • Configure up to 10 questions	

	• View results	
Raise Hand		
Customizable Layout		Layout depends on client application
Encryption		Encryption

	User Management
	• Name
	• Access Level
	  o Administrator
	  o Moderator
	  o Simple	

	Room Management
	• Name
	• Type
	  o Conference
	  o Restricted
	  o Interview	

	Backup	

After the features were extracted an initial feature model was created. The initial feature model is presented in Figure 4 and Figure 5 (please see Table 4 for feature model notation).

- WebCollabApp
  - Session Management
  - Participant Management
  - Participants List
  - Room Management
  - User Management
  - Audio
  - Video
  - Chat
  - Whiteboard
  - Screen Sharing
  - File Depot
  - History / Archive
  - Scheduler
  - Raise Hand
  - Polls and Votes
  - Customizable Layout
  - Encryption
  - Backup

	Root feature
F	Solitary feature *F* with feature cardinality [1..1] (i.e. mandatory feature)
G	Solitary feature *G* with feature cardinality [0..1] (i.e. optional feature)

Figure 4 Initial Feature Model, collapsed view

- WebCollabApp
  - Session Management
    - Start Session
    - Finish Session
    - Join Session
  - Participant Management
    - Name
    - Role
  - Participants List
    - View Participants
    - Add Participants
    - Remove Participants
    - Participant Permission
  - Room Management
    - Name
    - Type
  - User Management
    - Name
    - Access Level
  - Audio
    - Configurable Quality
    - Mute / Unmute Participants
  - Video
    - Configurable Quality
    - Selected or All Participants
    - Camera Resolution
  - Chat
  - Whiteboard
    - Upload Slides
    - View Uploaded Files
    - Multiple Pages
    - Drawing Tools
    - Typing Tools
    - Laser Pen
    - Annotation Capabilities
    - Export as Image or File
    - Multiple Instances
  - Screen Sharing
    -
      - Full Screen
      - Selected Area
      - Application
    - Remote Control
  - File Depot
    - Upload Files
    - Browse Uploaded Files
    - Share Uploaded Files
    - Download Uploaded Files
  - History / Archive
    - Record Sessions
    - Browse Recorded Sessions
    - Play Recorded Sessions
    - Download Recorded Sessions
    - Save Chats
    - Browse Saved Chats
    - Download Saved Chats
    - Attendance Report
    - Meeting Notes
  - Scheduler
    - Select Participants
    - Reminders
  - Polls and Votes
    - Configure Questions
    - View Answers
  - Raise Hand
  - Customizable Layout
  - Encryption
  - Backup

	Root feature
F	Solitary feature *F* with feature cardinality [1..1] (i.e. mandatory feature)
G	Solitary feature *G* with feature cardinality [0..1] (i.e. optional feature)
K	Grouped feature *K* with feature group cardinality [0..1]
	Feature group with group cardinality <1-k>, *k* = group size (i.e. inclusive-or group)

Figure 5 Initial Feature Model, expanded view

4.2 Evolution of Feature Model

After the initial feature model was built, we started adding additional applications to observe what type of changes we need to make to the feature model to support new applications. First three applications, WizIQ, ISL Groop and FuzeBox Meeting produced changes to the feature model that are described below, but the other three applications eLecta Live Easy Meetings, iLinc Meeting, Teleskill Live did not introduce any changes to the feature model.

4.2.1 WizIQ

The WizIQ Platform includes everything you need to take your teaching online, from a virtual classroom, to functionality to create and deliver courses with assessment tools and content sharing feature. WizIQ provides features that save time and enhance collaboration between students and teachers. WizIQ can be integrated with websites through well-documented API, allowing anyone, whether a middle school teacher, a private tutor, a test prep company, a university, or anything in between to start teaching online. Information gathered from the website is in Appendix A.4.

Table 9 shows extracted features that are not covered by the initial feature model.

Table 9 WizIQ features that are not covered by feature model

Feature	Description
Private Chat	WizIQ supports public chat that is visible to all participant and private chat between two participants
Whiteboard • Math Symbols	WizIQ allows to draw mathematical symbols on Whiteboard
Breakout sessions	This feature allows participants to form small groups within

	the environment to discuss different topics/questions.
Course Structure • Wizard • Courseware	This feature allows creating a course structure and attaching documents to the course.

From Table 9 we can observer that we need to change our current feature model in the following way:

1. Add an attribute "Access Level" to the "Chat" feature to distinguish between "Public Chat" and "Private Chat".
2. Add an optional feature "Math Symbols" into "Whiteboard"
3. Add a new optional sub-feature "Breakout sessions" into "Session Management"
4. Add a new optional feature "Course Structure" with two sub features "Wizard" and "Courseware'.

Changes to the feature model were measured in the following way:

- Number of new top level features: 1
- Number of new sub-features: 3
- Number of new attributes: 1

4.2.2 ISL Groop

ISL Groop is a small to mid-size business oriented web conferencing software. It does not claim to provide all the bells and whistles of a web conferencing environment. Instead, it provides core basics that meet the immediate needs of their clients and help them conduct meetings and webinars online. The features include: screen share, VoIP and

video. ISL Groop provides well-documented API. Information gathered from the website is in Appendix A.5.

Table 10 shows extracted features that are not covered by the current feature model.

Table 10 ISL Groop features that are not covered by feature model

Record Sessions • Manual Start and Stop • Saved on local PC only	From our understanding of the functionality of ISL Groop. It supports recording of sessions only manually and the recorded content is stored on the local machine.

From Table 10 we can observe that we need to change the current feature model in the following way:

1. Add an optional feature "Manual Start and Stop" and an attribute "Location" into "Record Sessions"

Changes to the feature model were measured in the following way:

- Number of new top level features: 0
- Number of new sub-features: 1
- Number of new attributes: 1

4.2.3 FuzeBox Meeting

FuzeBox Meeting is aimed to appeal to the hip and trendy generation of internet and tech savvy users. It appeals to the ultramodern and simple aesthetic. The platform claims to be easy to use. The product is available via any device. FuzeBox Meeting provides well-documented API.

Information gathered from the website is in Appendix A.6. Below (Table 11) are extracted features that are not covered by the current feature model.

Table 11 FuzeBox Meeting features that are not covered by feature model

Audio • Calls from landline	FuzeBox Meeting is the first application in our analysis that supports calls from landline phones.

From here we can see that we need to change our current model in the following way:

1. Add two sub-features in the "Audio" feature: "VOIP" and "Landline"

Changes to the feature model were measured in the following way:

- Number of new top level features: 0
- Number of new sub-features: 2
- Number of new attributes: 0

4.2.4 eLecta Live Easy Meetings

eLecta Live Easy Meetings is a personal web conferencing solution best suited for a single host environment. The software provides an environment for meetings where the host can share their desktop and documents, as well as a wealth of additional features for high quality online presentations. The environment is complemented by unlimited video and audio conferencing. This solution provides a well-documented API.

Features provided by the applications are described in the Appendix A.7. Based on their description and the current state of the feature model no additional changes to the feature model were necessary.

4.2.5 iLinc Meeting

iLinc Meeting aims enhance meeting efficiency by integrating a wide range of capabilities. iLinc's goal is to save time and money for their customer by allowing them to stay in touch with remote staff and clients. iLinc claims to provide an extensible API, however we were not able to find description of API online.

Features provided by the applications are described in the Appendix A.8. Based on their description and the current state of the feature model no additional changes to the feature model were necessary.

4.2.6 Teleskill Live

Teleskill Live Meeting is a live videoconferencing solution that claims to meet all business communications needs. The software enables online meetings for all business interactions that happen across geographical distances, from managing meetings, to offering remote consulting and assistance.

Features provided by the applications are described in the Appendix A.9. Based on their description and the current state of the feature model no additional changes to the feature model were necessary.

4.2.7 Convergence of Feature Model

The process of analyzing applications was stopped because we did not observe any changes to the feature model for the last three applications. Table 12 represents summary of changes by application.

Table 12 Summary of changes

Number of new top level features	1	0	0	0	0	0
Number of new sub-features	3	1	2	0	0	0
Number of new attributes	1	1	0	0	0	0
Total number of changes	**5**	**2**	**2**	**0**	**0**	**0**

Total number of changes by application (step) is represented in Figure 6.

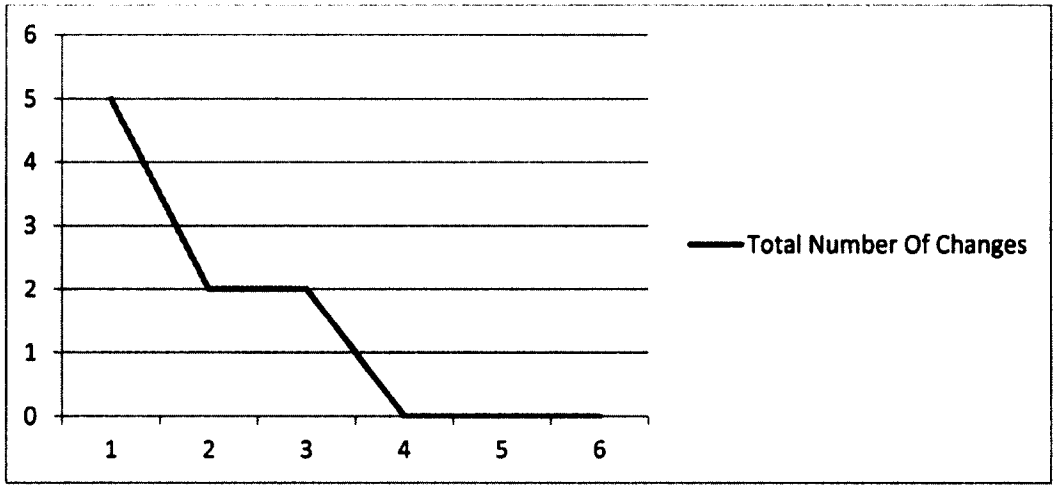

Figure 6 Total number of changes by application

Feature model changes that we observed during our analysis we classify by the following types:

- Introduction of an attribute
- Introduction of a new feature: top level or sub-feature
- Introduction of a group feature to a feature (VOIP vs Landline added to Audio)

These types of changes are in line with the research done by Botterweck et al. (2010).

Finally, Figure 7 represents the final feature model.

- WebCollabApp
  - Session Management
    - Start Session
    - Finish Session
    - Join Session
    - Breakout Sessions
  - Participant Management
    - Name
    - Role
  - Participants List
    - View Participants
    - Add Participants
    - Remove Participants
    - Participant Permission
  - Room Management
    - Name
    - Type
  - User Management
    - Name
    - Access Level
  - Audio
    - Configurable Quality
    - Mute / Unmute Participants
    - VOIP
    - Landline
  - Video
    - Configurable Quality
    - Selected or All Participants
    - Camera Resolution
  - Chat
    - Public
    - Private
  - Whiteboard
    - Upload Slides
    - View Uploaded Files
    - Multiple Pages
    - Drawing Tools
    - Typing Tools
    - Laser Pen
    - Annotation Capabilities
    - Export as Image or File
    - Multiple Instances
  - Screen Sharing
    - Full Screen
    - Selected Area
    - Application
    - Remote Control
  - File Depot
    - Upload Files
    - Browse Uploaded Files
    - Share Uploaded Files
    - Download Uploaded Files
  - History / Archive
    - Record Sessions
      - Manual Start and Stop
      - Saved on Server
      - Saved on Local PC
    - Browse Recorded Sessions
    - Play Recorded Sessions
    - Download Recorded Sessions
    - Save Chats
    - Browse Saved Chats
    - Download Saved Chats
    - Attendance Report
    - Meeting Notes
  - Scheduler
    - Select Participants
    - Reminders
  - Polls and Votes
    - Configure Questions
    - View Answers
  - Raise Hand
  - Customizable Layout
  - Encryption
  - Backup
  - Course structure
    - Wizard
    - Courseware

	Root feature
F	Solitary feature *F* with feature cardinality [1..1] (i.e. mandatory feature)
G	Solitary feature *G* with feature cardinality [0..1] (i.e. optional feature)
K	Grouped feature *K* with feature group cardinality [0..1]
	Feature group with group cardinality <1-1> (i.e. exclusive-or group)
	Feature group with group cardinality <1-k>, *k* = group size (i.e. inclusive-or group)

Figure 7 Final Feature Model, expanded view

5 Chapter: Results

This chapter describes the process that was generalized from data analysis section.

5.1 Process

The overall process of creating a feature model consists of 3 major steps:

1. Definition and Narrowing Down Domain
2. Building Initial Feature Model
3. Evolution of Feature Model and Its convergence

These steps are discussed below.

5.1.1 Definition and Narrowing Down Domain

The basic idea for this step is coming from competitive market analysis. We want to classify/categorize applications available on the market and select a set of applications that are of our interest.

1. Select type of product. Example, Web Collaborative Applications
2. Select 2 dimensions for application classification. For example, one of the dimensions can be extensibility/configurability/customizability of an application, the second dimension can be which customers an application targets - small/mid-sized companies or enterprises.
3. Identify applications of particular product type (selected in step 1) using internet, available literature and other sources.

4. Collect information available on each application via website, manuals, user guides, etc. For this step it's necessary to collect information in order to classify/categorize applications using dimensions selected in step 2.
5. Classify applications, i.e. input information into a two dimensional matrix.
6. Select one quadrant for analysis. The subsequent analysis will focus on the chosen quadrant.

5.1.2 Building Initial Feature Model

The basic idea for this step is to build an initial feature model. Feature models capture customer requirements, i.e. features visible to customers, not requirements that an architect might add to support the backend of the system.

1. Select a group of 3-4 applications. (Eisenhardt, 1989)
2. For each application analyze available documentation, websites. Based on this analysis, create a table of provided features with their descriptions.
3. For each application extract features and create a table with extracted features and their description
4. Based on extracted features, create a cross table of features: For each feature add a row into this cross table.

Feature 1 • Sub Feature 1	Corresponding feature if any	Corresponding feature if any

5. Create an initial feature model for this set of applications.

5.1.3 Evolution of Feature Model and Its Convergence

1. Add an application to analysis
2. Analyze available documentation, websites. Based on this analysis, create a table of provided features, with their descriptions.
3. Create a list of features that are not covered by the existing feature model
4. Refine the existing feature model to include new features.
5. Document changes and their types. Changes can be documented in the following table

Product	**Feature**	**Type of Change**

6. To measure changes a metric should be introduced. In our example, we used 3 metrics:
    - Number of new top level features
    - Number of new sub-features
    - Number of new attributes
7. Repeat steps 1-6 for other 6-9 applications as suggested by literature available on multi-case study approach. Observe changes to the feature model and metrics. Typically the feature model should converge as new cases do not add new information to the model.

6 Chapter: Discussion

To engage organizations in creating a collective the leader organization should demonstrate that an idea for a product is viable (Weiss, 2011). Attracting member to a collective is critical for the success of efficient product development. This research develops a process of building a feature model for an initial version of a shared platform. The process is built on the traditional software product line engineering approach to product development. The process that is discussed in this research provides a way for leader organizations to demonstrate potential members that a potential platform is viable. The developed process is modeled after the traditional software product line engineering approach and extends it in the following ways:

- Serves the core asset needs of multiple organizations (not owned by one, but by many organizations)
- Includes surveying existing applications/products
- Takes into account future evolution of the product

In traditional product lines, "a central unit is responsible for integrating and testing reusable assets and making them available to product groups (Bosch & Bosch-Sjitsema, 2010). The central unit is usually able to control requirements, product roadmaps, features, and variability. However, in a software ecosystem, a product line is shared with external partners; the "central unit" function is shared among the ecosystem members. While the leader organization may carry out a heavy coordination role, it does not solely hold the decision-making and control power of the "central unit". Weiss (Weiss, 2011) introduces the concept of collectives where core asset ownership is shared by members of

a collective. Members of the collective share in the risks and the benefits of having access to the basic code and the increased quality that results from collaborative development.

The final product is assembled by external developers, who are the customer of the product line platform (Bosch, 2009). Opening a product line to external developers allows the external developers to participate in the creation of products that best suit their needs and allows external developers to claim ownership to a portion of the product (Bosch, 2009). This research extends the basic principle of one platform owner and speaks to a mechanism where the ownership of the core assets is not expected to be by the platform owner organization. This research is more in line with Hanssen (2010) who claims that external developers can, not only customize the product to their needs, but contribute core assets to the product line.

Kang et al (2002) describes importance of marketing and product plans in developing high value product line assets. Lee et al (2009) also speak to the importance of market analysis of competitor's applications in identifying current trends and understanding market needs. Neither Kang et al (2002) nor Lee et al (2009) describe the actual steps of how to do market analysis when product requirements are not clearly identified. This research complements the works that came before it and proposes a process of how to survey the market and identify requirements.

To design a product line with evolution in mind is important. There are a couple of approaches that deal with unstable or immature domains: Vranic & Marko (2006) and Voget & Becker (2002). However, approach that Vranic & Marko introduced this research finds subjective: "The most critical step of the approach—identification of archetypal entities and their interactions—is principally highly dependent on the insight

of developers" Vranic & Marko (2006). Becker on the other hand, disregards the unstable subdomains and uses only stable components to build the feature model. This research took specific interest in future evolution of the created feature model. It looked at changes that the feature model underwent when additional products were added to the analysis that built the feature model. Changes in the feature model showed that evolution is probable and must be a major consideration in shared platform development.

7 Chapter: Conclusion, Limitations, and Future Research

7.1 Conclusion

Success of the Eclipse project was the leading motivator for research described in this paper. As the initial owner of the Eclipse software, IBM had vast internal resources to support and evolve the externalized platform. For start-up companies to take example from the Eclipse project is to answer the question of how to successfully pool resources to build a shared platform without an existing technology.

For start-up companies the path of externalizing existing software, the path that IBM took, is not feasible. Instead it is more realistic to form a community that would develop a shared platform.

The goal of creating a shared platform is that members of the collective need to contribute only a small portion of their resources to the development of core assets before they can dedicate majority of resources to their differentiators. In terms of the life cycle of the technology, the development of the shared platform is not the early stage of innovation and different ways of solving the same problem, but the stage when a dominant design is created. The shared platform contains that dominant design.

In this research we defined a conceptual process for developing an initial version of a shared platform. An initial version of a shared platform plays very important role in forming a collective. It demonstrates that a platform is viable and helps attract potential members. The goal of creating an initial shared platform is to gain collective member commitment to the platform by demonstrating that the platform's features meet their basic needs.

The main idea behind the process is to synthesize features of a shared platform from features of multiple existing applications.

A feature model allows for capturing of the requirements of a shared platform. The proposed process was generalized from the steps of building a feature model for domain of web-collaborative applications.

The novelty of the process lies in a combination of market analysis, traditional product line approach and planning for evolution. This research suggests surveying existing market offers to build an initial version of a shared platform because this is the only easily accessible source of information. Involving collective members at a very early stage can have negative consequences for the project (for example, potential members can lose interest in the project).

Because market and customer requirements continuously change, products that are built with evolution in mind will have a higher success rate of being and staying competitive in the market. In this research we demonstrated how the feature model of a platform can be iteratively revised, and that as market and customer requirements change the initial feature model created by this process will need to change. Our research identified three types of feature model evolution: (1) introduction of an attribute; (2) introduction of a new feature: top level or sub-feature; (3) introduction of a group feature to a feature.

By following the defined process, entrepreneurs and platform leaders will be able to create a feature model for the initial version of the shared platform by: (1) defining and narrowing down domain of existing applications using 2 dimensional market segmentation; (2) creating an initial feature model; (3) applying updates to the feature model and creating feature model for the initial version of the shared platform. By

following these steps platform leaders will be able to use the initial version of the shared platform to attract potential collective members, lower development costs and achieve higher product quality.

For the academia this research provides an initial look at developing a shared platform without known requirements and proposes a list of steps that should be taken in jump-starting a shared platform. Members of the academia will benefit from this research when they seek to study this area more in depth, when they seek to define additional methods of jump-starting a shared platform, or when they want to tailor the process to the needs of specific industries.

While the process developed during this research does not touch on architecture modularity, it is a very important characteristic of shared platforms.

7.2 Limitation

There are several limitations of this research. The process described here is conceptual and still requires validation. When the case studies were analyzed the steps of extracting features had a subjective element, as such, there is a question of whether another research body would come up with the same features.

7.3 Future Research

Several opportunities for future research are identified. First, the process that presented in here is conceptual and needs to be validated. Second, the defined process for creating an initial version of a shared platform still creates a platform that is owned by one organization, the next step would be to study the following question: how to attract

potential members of collectives?; how to give up control of the shared platform without losing competitive advantage? Third, the defined process was applied for web collaborative applications, it would be beneficial to apply the proposed process for a different domain and observe what type of changes the feature model undergoes.

Appendices

Appendix A

This appendix provides tables with information collected from applications websites.

A.1 Banckle Meeting

Table 13 Banckle Meeting Features

Feature	Description
Audio and Video	The video and audio conference mode allows you to have a very interactive and 'old-school'-feeling online meeting. See and hear each other, to effectively add a personalized touch to your eLearning sessions, webinars and web conferences. Banckle Meeting supports FullHD video streaming. It also allows you to mute/unmute participants and show video of all or selected participants.
Presentation, Whiteboard Annotation and Drawing Tools	It is true that now most of the users prefer the video conferencing or audio conferencing sessions. But nevertheless, the importance of presentations, whiteboard annotation and drawing tools stays the same. Banckle Meeting champions every mode of communication. This functionality lets you use many usual file formats as presentation views. It is converted to the PDF and presented in the Whiteboard mode. This helps you and the other participants to annotate the slides as required. Banckle Meeting allows you to upload multiple presentations at the same time. The whiteboard in Banckle Meeting offers a powerful and fun drawing tool. You can choose from 22 different tools to convey your ideas to the participants. You can also search the images from Flickr and add those to your Banckle Meeting whiteboard. You may also add as many different whiteboards as you require. The whiteboard navigation is pretty easy and you are presented with a virtual laser pen to draw attention of the meeting attendees. If you are to go away you can securely lock your whiteboard so no one can make amendments.
Screen and Application sharing and Remote Control	For any online conferencing solution the screen as well as application sharing, both are highly important. Many times you have to look at the attendee's screen to get the whole message. Sometimes you want to see what is going on at attendee's end to understand the issue s/he is facing in the

	application. Banckle Meeting enables you to do both. The specific application share feature allows presenters to maintain their privacy and only share a specific window across the cloud. The desktop share feature provides the enhanced control to the presenter to access the attendee's screen. The attendee can revoke back the right to access the screen or application at anytime. As an attendee there are scenarios in which you want to be more proactive and share your screen/specific application to the presenter. Screen and application remote control feature in Banckle Meeting allows you to do exactly that. You can select a particular participant to whom you wish to delegate the authority and/or duty of controlling your screen / shared application.
Public Meeting Room Chat	Banckle Meeting allows you to chat with all meeting participants and the host using the public meeting room chat functionality. You may also specify your chat font color to allow easy readability and understanding. By default all chat messages are sent to all participants.
Searchable Chat	During the course of heavy attendance and long chat sessions, it becomes difficult for anyone to track if a specific topic has been already discussed. Searchable chat allows you to hunt a specific word/term from the chat session. Banckle Meeting allows you to search for words in a specific chat session. This allows you to quickly locate all instances of a particular word, and helps you track your chat communications.
Saved Chat Transcripts	Saved chat transcripts are a great way to double check the notes that you jotted down during the course of your web conferencing session. Banckle Meeting saves all chat transcripts and also makes a downloadable copy available for your download at all times. Once downloaded you can also send out the chat transcript in email or print it for your later reference.
Public Meeting Notes	Public meeting notes can be entered by any attendee with rights for Banckle Meeting sessions. These notes serve as your personal meeting charter or notebook, in which you can record all important details from the meeting. The presenter can also toggle the permission to edit meeting notes on and off for participants, if that is required, through a single click.
File Depot	Be it a web conferencing or eLearning session, the business meeting and or your video interview, you will probably need to share files with the participants. Through its File Depot feature, Banckle Meeting, makes it quick and fun for you to share the required files securely. These files can quickly be accessed by all

	participants. Files shared in a Banckle Meeting session are also stored for later access through the recasting or meeting history feature.
Raise Hand Feature	The Raise Hand feature in Banckle Meeting ensures that no question goes unasked and hence unanswered, and allows everyone to draw presenter's attention. This ability ensures higher comprehension ratios, and brings an interactive feeling to the session. Without a doubt, you can use this feature as per a scheme devised by you and your colleagues as well.
Built-in Voting Functionality	Through its built-in voting functionality, Banckle Meeting allows each participant to quickly cast a screen vote or poll. This streamlines all decision making activities during a session. This handy feature can allow the presenter to easily request feedback or opinions without disrupting the flow of a meeting. Without a doubt, you can use this feature as per a scheme devised by you and your colleagues as well.
Scheduled meetings	Banckle Meeting empowers you to schedule your web conferences and webinars in many ways. Scheduling meetings has never been easier and more fun. You can schedule the meeting months in advance or at one day notice. You can also schedule one-time or recurring meetings. Provide the time zone of the meeting session to make sure everyone's availability. Select prospective attendees from the list of existing contacts or manually enter their email addresses. It also allows you to send reminders.
Participants List	While in a meeting, the most important thing is to know your audience and participants list. Knowing to whom are you presenting is as important as what are you presenting. Banckle Meeting conveys this important information to you through its Participants List pod in the on-going meeting session. It lists all the attendees of the on-going session. The role of each participant is denoted with a unique icon. Same is true about the online status of each attendee. Moreover, you can perform various actions on the individual participants such as Screen Share etc.
Participant Roles	With your growing business, you are supposed to attract larger audiences to attend your eLearning sessions and webinars. This makes it difficult for you to manage the meeting as the only host and moderator. Banckle Meeting allows you to change the role of meeting participants to be fellow presenters. This way you as the meeting host can focus on the meeting itself, perform some other activity or just get some break.
Control Participants	Hosting a meeting is all about using every attendee to its full

Permissions	potential as well as controlling participant's permissions. The host needs to define which task is to be done by which attendee so that they do not waste the limited meeting time in unnecessary actions. Banckle Meeting allows you to control and fine-tune the permissions of the participants. This way you can ensure that what a participant can access and/or edit and what s/he cannot. This increased level of control ensures that your meetings stay secure and well managed precisely in the way you want and hence use each participant's potential in a productive way.
Evict Unwanted Participants	Not all participants are always having a good day. Sometimes someone tries to ruin your meeting. For such instances Banckle Meeting allows you to evict (kick) such unwanted attendees. This could be any attendee that you may wish to force out of your web meeting, eLearning sessions or webinars. This allows you to ensure that no intruders or undesirable participants are participating in your web conferencing session.
Recording and Playback	Through its meeting recording and recasting feature, Banckle Meeting allows you to record all meeting sessions and save them for later viewing. This is especially useful for eLearning and Podcasts through Banckle Meeting. Recorded meeting sessions can easily be viewed again at a time of your convenience. Banckle Meeting recasting reduces your IT cost by eliminating the need for storing and uploading big movie files and other hosting arrangements. These recordings do not use us any processing or storage resources from your PC. Banckle Meeting session playback protects your intellectual property by providing a secure environment, which does not allow downloads or redistribution. Recast replay quality is identical to the original session and does not have any resolution loss. The thing that makes Banckle Meeting stand out from the crowd is that its meeting recording is not only just a playback but you can also interact with this meeting recording. You can seek and jump to your required time and you can download the files shared during the meeting etc.
Meeting Session History	Through its meeting session history feature, Banckle Meeting keeps a log of each and every meeting that you perform. You can view the chat transcripts, participant attendance report and the files uploaded during the meeting. If you recorded the meeting then you can also view its recording later at your convenient time. You can download the meeting recordings as well as the shared files. Although the column sorting makes it very quick to locate any meeting but with passage of time your meetings would increase

	and it might be difficult to manually locate your required meeting. So for such scenario, a search panel is provided so you can search the desired meeting based on time period or date range. You may also filter the meetings that you hosted as well as the meetings that you attended without being its host
Meeting Attendee Report Log	You are just done conducting a very successful eLearning webinar and now you want to know if everyone who registered for the session actually showed up. Through its comprehensive meeting attendee report log feature, Banckle Meeting provides you detailed log report for each attendee. Every time an attendee joins or leaves the meeting an entry is made in this log mentioning the joining and leaving time. This report is very useful if you are to keep track of the attendance.
Customizable Layout	Layout customization means how effectively you can use the real estate of your screen area. Banckle Meeting, through its highly customizable layouts, enables you to do exactly that. To boost productivity and enhance the ease-of-use, Banckle Meeting enables you to customize the layout of its User Interface (UI) elements according to your preferences and requirements. We have used the concept of panes/pods to group the logically related UI elements. You can then move these pods around to different locations of your screen area as per your requirements. You can even show/hide them. To enhance the productivity even further, we have ensured that frequently used features are more prominently placed, with the less used features appropriately positioned just a few clicks away. Customizable User Interface Elements: Screen customizations such as pane and column resizing are possible when using Banckle Meeting. These customizations allow you to customize the appearance of the web conference, for your user only, to better suit your requirements and preferences. A customizable pane-based (pods) layout ensures that each and every tool is easily available at-hand to all Banckle Meeting participants, with each pod designed to be an effective and easy-to-use meeting aide. You are allowed to show/hide these pods during an on-going meeting session. If you want to keep a pod on your screen but do not want it to distract your attention, just minimize it.
Security	Enabling our customers to conduct totally secure online meeting, webinars and eLearning sessions are the top priority at Banckle. Banckle Meeting, like all other Banckle Collaborative and Social Apps, works over Hypertext Transfer Protocol Secure (HTTPS) protocols. This, with the additional security that the Secure Sockets Layer (SSL) brings, allows you to trust Banckle Meeting

	with your personal and business related data. SSL ensures that your computer and the Banckle Meeting server goes through a 'hand-shake' process when transferring data – every byte of data is transmitted over the industry standard for web security. Banckle Meeting is designed and optimized to allow dependable security, without affecting data transmission speeds.

A.2 OpenMeetings

Table 14 OpenMeetings Features

Feature	**Description**
Audio and Video	There are four options to use OpenMeetings audio/video functions, which you can select during a conference session. • audio + video • audio only • video only • picture only Additional you can change video-/audio-quality, choose multiple camera resolutions (4:3, 16:9 or 3:2) and choose your input devices.
Multi-Whiteboard	Multi-Whiteboard, you can add new whiteboard instances, each whiteboard can have the full range of tools and documents inside. Save whiteboards. You can save each whiteboard instance as a file. The file is located in the File-Explorer and can be drag n' drop'ed to the white board again and organized like any other document, image or folder. Whiteboard with drawing, writing, Drag n' Drop, Resizing, Images (Drag n' Drop from File-Explorer), Symbol(s)/Cliparts. Full-fit does rescale the document on the screen to be 100% visible on all screens no matter what kind of screen resolution different users have. You can import from a wide range of document formats (PDF, DOC, ODT, PPT, et cetera...)
Screen sharing	Provides capabilities of screen sharing. It is possible to select full screen or an area for sharing. It supports different quality steps for screen sharing. It also allows you to do Remote Control.
Chat	You can chat during the conference. The chat can be read by anyone able to log in.

File Explorer	Advanced File-Explorer in every conference room, Drag and Drop interface for managing uploaded files, including the possibility to create a document tree with folders. Private and Public Drive in File-Explorer. The File-Explorer has two different views; one is the Private Drive and the other the Public Drive. The Private Drive always contains the same files. Those files are visible only to the user currently logged in. The Public Drive is not bound to the user, but to the conference room instead. All users in the conference room have access to the Public Drive.
Polls and Votes	You can create a poll with yes/no or 1-10 questions. Llet the user vote and see the voting results. Polls can be stored and results viewed as pie-chart
Plan meetings with integrated calendar	Plan your conferencing and invite attendees from OpenMeetings or External. The invited attendees will receive an email with details to the meetings and a link with a secure hash to the conference room. Share your calendar with your contacts.
Moderating System	During a conference, the moderator can adjust the user permission to every user individually: • Allow/Deny moderation • Allow/Deny to draw on whiteboard • Allow/Deny screen-sharing/record screen • Allow/Deny Remote Control Screen • Give exclusive audio to others or self • (Re-) Start Audio, Video or Device settings
Meeting recording	Capabilities of recording include: • Recorded sessions contain everything including sound recorded from all audio streams in exactly the way you've seen it in the conference. • Recorded sessions can be downloaded as AVI/FLV files. • Watch and organize recordings in an integrated Drag and Drop File-Explorer
User Management	You can manage users and multiple organizations in a single OpenMeetings instance. There are three types of users. The first is the administrator, the person who manages and configures the system (general parameters, the rooms, users, etc.). He has the right to do anything anywhere in the system. The second type is the simple user, he can access different page, he may enter a room and interact according to the rights granted to him and dependent on the type of room in which he is. Finally, there are the moderators. Moderators are power users, they are usually (but not necessarily) the ones who are the organizers of conferences. They

	have the right to do everything in a room and have the ability to give the rights to interact to other users in the room. They can also give the moderator status to a regular user of the room.
Room management	Each user has by default 2 personal rooms that are always accessible exclusively for that user. There are buttons to enter those rooms from the Dashboard. You can assign conference room to all users, or you can assign them only to specific user-groups. When creating a room, after entering the room name, choose the number of participants allowed. There is no limitation in the software, it will rather depend on your server and bandwidth. Next, define the type of room (conference room, restricted room or interview room) and whether the room is temporary (it will be removed when it is closed) or not. *A conference room* is a room where everyone shares his camera and microphone, has access to the whiteboard and can manage files. The session recording records the whole screen. *A restricted room* is a room in which users must ask permission to share their camera and where they do not have access to files. The session recording records the whole screen. *An interview room* is a room where only two users can share their video and audio (there may be more than two users in the audience and the moderator will then enable or disable the cameras and microphones) and in which there is no whiteboard.
Backup	You can backup all user-generated data including files uploaded by the users in a single ZIP file and import that ZIP into a new installation of OpenMeetings again.

A.3 OpenTok

Table 15 OpenTok Features

Feature	Description
Audio and Video	OpenTok calls audio and video streams. An application can control where to display video controls and which one to subscribe to and provides ability to publish your own video. An application has ability to get a list of available publishers.
Archiving	OpenTok provides archiving capabilities for flash version only.
Session	Every communication/interaction in OpenTalk is built around session. A *session* represents an entire video chat environment. It is a collection of connections publishing and subscribing to streams.
Token	Token basically is an identification of a connecting user.
Encryption	OpenTok supports encryption using https. Please note, OpenTok video streams are not sent on a secure socket.

A.4 WizIQ

Table 16 WizIQ Features

Feature	Description
Text Chat	Give personal attention to your remote students in one-on-one or group chat; even chat in their local language.
Audio and Video	Have face-to-face interactions with your remote students and conversations without echo, even without headsets. Up to 6 live video streams.
Whiteboards	Write in English, Spanish, Portuguese, and Hebrew; solve math equations with math symbols, draw using the drawing tools, and flip among multiple whiteboards.
Content Library	Publish courseware in any format (Presentations, PDFs, Documents,

	Videos or multimedia content) and share it in the virtual classroom or distribute it to students directly. Share PowerPoint slides, documents, web pages, and even your desktop with remote students in real time. Share YouTube videos synchronously.
Breakout sessions	Students can comprehend difficult concepts and collaborate with their peers through discussions in breakout rooms or watch recordings of live classes.
Recordings	Records the class as it happens, including presenters' audio and video inputs.WizIQ can record your entire class or training sessions for later reference, demonstration, or asynchronous learning. The recordings can be viewed online or downloaded and distributed to the students. The best part is that the recording captures everything that happens in the class including the presenters' audio and video inputs. Even better, you get a weekly report about who viewed or downloaded the recordings.
Attendance reporting	Manage your students with reports on class attendance, time spent in the class, and tests taken.
Course Creation Wizard	Promotes an effective course structure and ensures the alignment of courseware and assessments with course objectives.
Courseware	Through an intuitive interface, organize courseware and add to the course map provided by the course creation wizard.
Assessments and Question Bank	Straightforward assessment tools: create your own question bank, make use of a pre-built WizIQ question bank, or use polls in the classroom.

A.5 ISL Groop

Table 17 ISL Groop Features

Feature	**Description**
Integrated Audio and Video	Enjoy live and effective meetings using many-to-many audio and video communication. The audio stream of any attendee can be muted/unmuted, and their video shown, hidden or replaced by a personal image.

Presentation Content	In addition to sharing live content, slides can easily be created by importing an existing PowerPoint file, inserting the clipboard content, clipping your screen, or making a snapshot from a shared live content stream. Create presentation content in no time!
Import/export presentation	It is possible to import previously prepared PowerPoint slides or export the slides prepared in ISL Groop presentation.
Annotations	For enhanced communication, both presenters and participants can annotate and comment on slides as if on a whiteboard. Insert a freehand drawing, use a marker, and add an arrow to point out an important part of the slide content, or add comments using the text tool.
Share Any Application or Desktop	Add live content to your presentation by sharing with other attendees your entire desktop, a selected region or an application. Share any application or web content necessary for your business and enhance the interactivity of the presentation.
Public or private chat	The chat feature is mainly used for background communication, since it can be used for sending private messages. Primarily, this feature is available for posting public questions during the presentation. Users can send text messages by entering them in the input panel. All visible messages can be copied to the clipboard, but only the host can erase chat history.
File transfer	Users can conveniently upload and download files (File transfer is not needed for sharing content).
Raising hand	Users can get the attention of the podium holder or the host by clicking on the 'Raise hand' button. There are two options available: users can either post a question, or enter a waiting list for the podium. Queue category in the Attendee list gets populated by all users with raised hand in descending order. 'Question' or 'Waiting' status indicates the intention of a user in the queue. A hand can be lowered, thus removing the question or exiting the waiting list.
Rights and Promotions	Web meeting attendees can easily ask to be promoted to a meeting role with upgraded rights, e.g. from a presenter to a host, and request to change the application layout of all users, e.g. from the presentation to the conference layout. Web conference hosts can also request other participants to turn on their voice and video streaming.
Participant Roles	A participant can have one of the following roles: **Host** A host is a user who creates a meeting or a participant that gets promoted to host by another host. A host cannot be expelled from a

	meeting and is the only one with the ability to lock or end a meeting. The host of a meeting is also its default initial podium holder, meaning that he/she starts a meeting on a podium. In case of a meeting with a public scope, the host selects a default role for public access. A host can promote or demote any participant to a different role during a meeting, either directly or following that participant's promotion request. **Podium holder** User on the podium becomes an active speaker with superior permissions relative to his initial role. The most important podium feature is public slide navigation (assigning the global focus to a shared slide). Podium holder can pass the podium to any other meeting attendee or leave the podium, thus making the meeting non-moderated. A host can force the podium holder to leave the podium. **Presenter** A presenter is a participant with the ability to add/delete slides or live content. Presenters may also annotate the content and broadcast audio and video. A host can assign the presenter role to multiple participants, depending on the type of the meeting. In sessions with a small number of participants, a host usually plays presenter's role. **Guest** A guest is a person who who has been invited to participate in a meeting as a passive audience member. Guests don't have any permissions except sending public text messages and posting questions.
Meeting Recording	ISL Groop allows you to record your meetings. It allows you to configure what you wish to include in the recording. You can start and stop recording only manually. You can record meetings on your machine only.
Maximum Security	ISL Groop meetings use maximum security available. All data connections are encrypted with SSL using symmetric 256-bit keys. X.509 certificates are used to guarantee the authenticity of transmission.

A.6 FuzeBox

Table 18 FuzeBox Features

Multi-party HD video	FuzeBox's crystal clear connectivity conveniently links numerous meeting participants and attendees at the speed of light. Connect up

conferencing	to 12 users at a time without sacrificing quality. Geography will not complicate your productivity, nor will it affect the pristine quality of your video conference. Fuze Telepresence extends into conference rooms with HD video and audio quality. Fuze is equipped with full interoperability with H.323 and SIP videoconference technology (Polycom, Cisco/Tandberg, LifeSize, and Teliris), a solution that no other brand offers. This innovative technology allows your company to conduct seamless online meetings, collaborate over HD content, and engage in pixel perfect Telepresence from any device.
Rich Media Sharing and Content Publishing	Upload, stream and share gigantic files without a hitch. Zoom in and out rapidly or pan across your screen at lightning speed without compromising quality or pixilation. FuzeBox Rich Media Content sharing supports all video files including: .mov files, MP4 files and .avi files. "Zip-up" all of your meeting content into a seamless link that can be easily sent to anyone who attended or could not attend your meeting. Once the host has provided a link to the content that was shared or discussed, that content can be viewed, downloaded, saved and reused. There is no limit to how many people can view your content publishing link for easy follow-up and distribution.
Desktop and App Sharing	Give other meeting attendees access to view your desktop during meetings.
Remote Control	Meeting hosts can offer meeting presenters the ability to access and control the host's computer from the presenter's computer. No other video conferencing software offers its users the ability to access another user's computer from a remote location.
Chat	Managed by the meeting host or by other attendees, the Chat feature allows you to make comments and notes privately (or publicly) during a meeting. Chat gives you the ability to alert other presenters and attendees quietly and subtly and is completely unobtrusive within FuzeBox's sleek user interface.
Meeting Recording	Ensure that your meetings live on forever with high-resolution recording and playback. If you need to log and record an entire meeting, or just certain parts of one on the fly, the Meeting Recording feature gives you the power and control to do so - from a PC or a Mac.

A.7 eLecta Live Easy Meetings

Table 19 eLecta Live Easy Meetings Features

Crystal Clear Full Duplex Audio	Integrated Audio Conferencing. Multiple Speakers. High Quality Voice Over IP (VoIP) even on lower bandwidths. No conference calls. No additional charges for audio conferencing.
Live Video Sessions	Enhance your team meetings and group discussions with live video conferencing even on lower bandwidths.
Shared Interactive Whiteboards	Multiple shared whiteboards. Type text, launch images and slides. Full digital tablet support for handwriting or precise drawing.
Rich Markup and Annotation Tools	Use rich markup and annotation tools over images and slides. Use pointers and checkers to focus your attendees' attention.
Share Your Screen and Applications	Enhance your computer based training classes sharing your entire screen or just a single application. Give remote control to students and attendees.
Instant Messaging and Session Comments	Instant messaging / text chat is available for communication along with the audio. The session leaders can enable/disable the text messaging options.
Session Recording and Playback	Record your live sessions and meetings for quality control or archiving. Publish your recordings on your website to attract new students or give individual playback permissions.
Arrange Online Polls and Surveys	In a virtual classroom the online polling feature is a must. Arrange online polls with predefined answers.
BreakOut Rooms for Individual Collaboration	With the eLecta Live BreakOut rooms you can split your class into smaller groups so that your students collaborate and practice individually.
Meeting Schedule	Under your eLecta Live hosted account you will find advanced scheduling tools for managing the access to your live classes and conferences. You can schedule live classes and sessions, start online meetings on the fly and invite attendees and students to join.

A.8 iLinc Meeting

Table 20 iLinc Meeting Features

Included Multi-Person Video and Internet Audio	Stream a vast number of video feeds simultaneously for all online meeting participants. Plus, communicate cost-effectively with included Internet Audio (also known as Voice over IP, or VoIP).
Participation Tools	Get a quick glimpse of the state of your audience through iLinc's Participation Meter and then re-engage them with Hand Raise, Polling, Instant Feedback, online Chat and other features.
Interactive Whiteboard	Drive active participation and focus attention where you want it with pointers, highlighters, text markup and other tools.
Sharing Options	Choose to share your full desktop, a specific application or a region of your desktop.
Reliable Recording	Simple, reliable, server-side recording that captures your meeting properly and doesn't consume processing power or memory.
Content Upload	Upload your meeting materials prior to your online meeting or while you are in your web conferencing session.
Industry-Leading Security	iLinc's rigorous security protections ensure that your data, processes, and network are never compromised during a virtual presentation.

A.9 Teleskill Live

Table 21 Teleskill Live Features

Multipoint full-duplex video/audio	The solution allows up to 8 concurrent remote users to actively participate in the event in full-duplex mode. The Conference Manager selects these users from the list of meeting participants. The selected users can listen and speak to each other at the same

	time, just like in a normal conversation, while the rest of the audience is able to see and hear them live.
List of users	Here we have the list of all the users taking part in the live session. In this section the users make use of the "Show my status" function where they can ask to participate with audio/video in the videoconference. In the same section the Conference Manager makes use of the user management function where he/she can allow or disallow the audio video stream of individual users.
Chat	This function allows each remote user to interact with other users by sending them public or private messages in text format.
Syncronous presentation of slides, documents and images	The solution allows timely sharing of slides, documents, images and video. This function enables the Conference Manger to share his or her contents with the other participants in real time. This includes presentations, slides, images, documents.
Screen Sharing	The function enables the operator of the meeting to show in real time to all participants the content of his PC screen such as: presentations, slides, photos, Internet browsing, documents, software applications, etc. ...
Polls	The tool contains a polling function that enables the Conference Manger to perform a survey among the audience by posing questions, obtaining the answers and graphically displaying the results.
File Sharing	The system allows users participating in the meeting to share files in real-time. Participants can upload and download any file of collective interest to and from the central server at any time during the meeting.
Question! (hand raised)	The participant can ask to take part in the discussion, ask a question or make a comment in audio/video form by changing his or her status.

References

Baldwin, C., & Clark, K. (2006). The architecture of participation: Does code architecture mitigate free riding in the open source development model? *Management Science*, *52*(7), 1116-1127.

Bosch, J. (2000). Design and Use of Software Architecture: Adopting and evolving a product-line approach. *Addison-Wesley Professional.*

Bosch, J. (2009). From software product lines to software ecosystems. *In Proceedings of the 13th International Software Product Line Conference* (pp. 111-119).

Bosch, J., & Bosch-Sijtsema, P. (2010). From integration to composition: On the impact of software product lines, global development and ecosystems. *Journal of Systems and Software*, 83(1), 67-76.

Botterweck, G., Pleuss, A., Dhungana, D., Polzer, A., & Kowalewski, S. (2010). EvoFM: feature-driven planning of product-line evolution. *In Proceedings of the 2010 ICSE Workshop on Product Line Approaches in Software Engineering*, 24-31.

Boudreau, K., & Hagiu, A. (2009). Platforms rules: multi-sided platforms as regulators. In A. Gawer (ed.), *Platforms, markets and innovation.* Cheltenham, UK and Northampton, US: Edward Elgar, Chapter VII.

Czarnecki, K., Bednasch, T., Unger, P., & Eisenecker, U. (2002, January). Generative programming for embedded software: An industrial experience report. In *Generative Programming and Component Engineering*, 156-172. Springer Berlin Heidelberg.

Czarnecki, K. & Eisenecker, U.W. (2000), Generative Programming: Methods, Tools, and Applications. *Addison Wesley*, Boston

Czarnecki, K., & Kim, C. H. P. (2005). Cardinality-based feature modeling and constraints: A progress report. In *International Workshop on Software Factories.*

des Rivieres, J., & Wiegand, J. (2004). Eclipse: A platform for integrating development tools. *IBM Systems Journal*, 43(2), 371-383.

Eisenhardt, K. (1989). Building theories from case study research. *Academy of Management Review*, 14(4), 532-550.

Eisenmann, T., Parker, G., & van Alstyne, M. (2009). Opening platforms: How, when and why? In A. Gawer (ed.), *Platforms, markets and innovation.* Cheltenham, UK and Northampton, US: Edward Elgar, Chapter VI.

Gawer, A., & Cusumano, M. (2012). Industry Platforms and Ecosystem Innovation. *Presented at the DRUID 2012*, June 19-21, 2012. Copenhagen, Denmark.

Gomaa, H., & Webber, D. L. (2004). Modeling adaptive and evolvable software product lines using the variation point model. *In System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference*. IEEE.

Gruber, O., Hargrave, B.J., McAffer, J., Rapicault, P., & Watson, T. (2005). The Eclipse 3.0 platform: Adopting OSGi technology. *IBM Systems Journal*, 44(2), 289-299.

Hagiu, A. (2009). Two-Sided Platforms: Product Variety and Pricing Structures. *Journal of Economics & Management Strategy*, 18(4), 1011-1043.

Hanssen, G. (2010). Opening up software product line engineering. *Workshop on Product Line Approaches in Software Engineering, International Conference on Software Engineering*, 1-7, ACM

Hwan, C., Kim, C. H. P.., & Czarnecki, K. (2005). Synchronizing cardinality-based feature models and their specializations. *In Model Driven Architecture–Foundations and Applications*, 331-348. Springer Berlin Heidelberg.

Jacobson, I., Griss, M., & Jonsson, P. (1997). Software reuse: architecture, process and organization for business success. *ACM Press/Addison-Wesley Publishing Co.*.

Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., & Peterson, A. S. (1990). Feature-oriented domain analysis (FODA) feasibility study (No. CMU/SEI-90-TR-21). *CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.*

Kang, K. C., Kim, S., Lee, J., Kim, K., Shin, E., & Huh, M. (1998). FORM: A feature-oriented reuse method with domain-specific reference architectures. *Annals of Software Engineering*, *5*(1), 143-168

Kotonya, G & Lee, J. (2010). Combining service orientation with product line engineering, *IEEE Software*, 27(3), 35-41.

Kotonya, G., Lee, J., Robinson, D. (2009). A Consumer-Centered Approach for Service-Oriented Product Line Development, *Proc. Working IEEE/IFIP Conf. Software Architecture (WISCA 09), IEEE Press*, 211-220.

Lee, J., & Muthig, D. (2006). Feature-oriented variability management in product line engineering. *Communications of the ACM*, 49(12), 55-59.

Lee, J., Muthig, D. & Naab, M. (2008). An approach for developing service oriented product lines, *International Software Product Line Conference*, 275-284, IEEE.

Medeiros, F., de Almeida, E., & Meira, S. (2009). Towards an approach for service-oriented product line architectures, *Workshop on Service-Oriented Architectures and Software Product Lines,* 1-7.

Sawhney, M. (1998). Leveraged high-variety strategies: From portfolio thinking to platform thinking, *Journal of the Academy of Marketing Science*, 26 (1), 54-61.

Smith, D., & Milinkovich, M. (2007). Eclipse: A premier open source community. *Open Source Business Resource*, (July 2007).

Spaeth, S., Stuermer, M., & von Krogh, G. (2010). Enabling knowledge creation through outsiders: towards a push model of open innovation. *International Journal of Technology Management*, 52(3), 411-431.

Thomke, S., & von Hippel, E. (2002). Customers as innovators: a new way to create value, *Harvard Business Review*, 80(4), 74-81.

Voget, S., & Becker, M. (2002). Establishing a software product line in an immature domain. *In Software Product Lines*, 60-67. Springer Berlin Heidelberg.

von Hippel, E., & von Krogh, G. (2003). Open source software and the "private-collective" innovation model: Issues for organization science. *Organization science*, *14*(2), 209-223.

Vranic, V., & Marko, V. (2006). Dealing with Unstable Domains in Product-Line Architecture Development. *In Proc. of 9th International Conference on Information Systems Implementation and Modeling (ISIM 2006)*, 57-64.

Weiss, M. (2009). Performance of open source projects. *In European Conference on Pattern Languages of Programs*.

Weiss, M. (2011). Economics of collectives, *International Workshop on Quantitative Methods in Software Product Line Engineering (QMSPLE), International Software Product Line Conference (SPLC)*, 39, ACM.

Weiss, M., Esfandiari, B., & Luo, Y. (2007). Towards a classification of web service feature interactions, *Computer Networks*, 51(2), 359-381.

Weiss, M., & Noori, N. (2012). Managing the quality of platform complements. *Action for Innovation: Innovating from Experience, XXIII ISPIM Conference*, Barcelona 2012

West, J., & Gallagher, S. (2006). Challenges of open innovation: the paradox of firm investment in open-source software. *R&D Management*, 36(3), 319-331.