Linnæus University
School of Computer Science, Physics and Mathematics

Licentiate Thesis

Towards a Mobile Learning Software Ecosystem

Oskar Pettersson
2011-06-10

Chapter 1

Introduction

The last decades have seen significant changes in computing. Computers have gone from being a device to be used only by experts and occupying several rooms to being a pervasive artifact in many aspects of society (Ceruzzi, 1986). Technological development has put communication devices into our pockets with a computational power that was fantasy some years ago. The popularity of the Internet has changed the way we use communication technology and services in our daily lives (Alexander, 2006). We are now more interconnected than ever before. A device's physical dimensions have decreased while its computational power has increased. This has, in turn, paved the way for powerful mobile devices which increases still further the connectivity of the world (McQueen, 2009).

Some have claimed schools and universities have been very slow in implementing technology in their daily activities, save the occasional PowerPoint presentation (Cross, 2010). Be that as it may, research on how to utilize technology in education has been around for a long time (Suppes, 1966). As computers made their way into the public space, more and more researchers began to show an interest in the field of utilizing technology to support learning. This also prompted efforts in hardware development; Sweden for example, decided to develop its own "school computer" during the 1980s, called the Compis (Kaiserfeld, 2000). The Internet gained much popularity during the 1990s and this prompted distance education via computers to become an attractive notion to pursue (Bates and Bates, 2005). Efforts such as Moodle (Cole and Foster, 2007) and Sakai (Carmichael et al., 2006) were started in universities in order to provide the possibility of giving distance education via computers. Such learning management systems (LMS) now have several millions of users worldwide.

Mobile phones increased more and more in their computational power and gradually became a viable option for use in educational settings (Liu et al., 2003). Early devices lacked a data connection: initial efforts utilized SMS for language education or text-to-speech applications and simply let people call a service to retrieve the spoken information. Better data connections, displays, and memory capacity made context aware learning applications feasible (Ogata, 2008).

There are now plenty of implementations that convey education by utilizing mobile devices (Frohberg et al., 2009). As an extension of technology moving towards ubiquity, there are efforts aiming to bring this to an educational use as well. An example of this is the interactive table for supporting participation balance developed by Bachour et al. (2009). The speed of technological development and the possibilities it brings introduces a large number of challenges. These challenges are thus inherited when trying to implement this technology in educational settings. Using technology in education faces other challenges as well and some of these are elaborated upon in the next section.

1.1 Motivation

Research on technology use in educational settings has seen significant progress since the start of the field. There are still plenty of challenges however, as described by Frohberg et al. (2009), especially in the less mature fields of mobile and ubiquitous learning. Mobile learning (M-learning) refers to education utilizing mobile technologies, preferably using contextualized applications (Vavoula and Sharples, 2009). Computer Supported Ubiquitous Learning (U-learning), on the other hand, refers to blending technology with different aspects of education in a way that makes them almost invisible to the user (Ogata and Yano, 2004). The concepts from M/U-learning have proven hard to integrate with everyday educational practice. A major challenge that M/U-learning face in common is that implementations do not fit well in the organization (Sharples, 2002). And the newer the hardware and the higher the complexity of the implementation, the more disruptive it becomes. This disruptiveness is not the only challenge however. There are other notable areas of concern:

- **Pedagogical aspects** of M/U-learning (Mobile and Ubiquitous learning) are still a challenge (Sharples et al., 2007). How to engage and educate with the use of devices is a complex matter. However, learning design has provided a forum for highlighting these challenges (Conole et al., 2004).
- **Tools and implementations** used in the domain is another area of concern. As these may rely on customization and adaptation to context, switching context might require a significant development effort. (Frohberg et al., 2009)
- **Computer-supported Collaborative Learning** is in many ways related to the two previous items. It is concerned with the challenges related with facilitating collaborative learning. These may be of a practical, pedagogical, or computational nature (Hoppe et al., 2007).
- The **ecosystem** of the domain is another challenge. 'Ecosystem' in this case is the term used to describe all the components required to implement a solution such as software, hardware and roles (Uden et al., 2007). We have previously raised concerns that M/U-learning solutions are difficult to integrate into everyday practice. To better understand

this challenge, processes, life cycles and organizations need to be studied further to enable better integration (Alvarez, 2011). There are other inherent challenges in ecosystems as well, and a some of these are:

- Poor **sustainability** is an observation from many projects (Wingkvist, 2009). This implies that not many implementations survive their projects funding.
- Handling the **organization** inside ecosystems (Dillenbourg and Jermann, 2011). The ability to administer activities, processes and actions both indoors and outdoors is something that is needed in a well functioning ecosystem (Chang and Uden, 2008).
- System and sub-system **coherency** within the ecosystem. (Hoppe et al., 2007) Making systems inside the ecosystem function well together and, hopefully, not appear as separate entities.
- In order to achieve coherency, **middleware** is one of the approaches (Byrne, 2011). Middleware allows for better technological integration between systems but also between clients and different kinds of devices.

These challenges also have one other thing in common: reuse. Recent trends such as learning ecosystems (Chang and Uden, 2008), learning landscapes (Guetl and Chang, 2008) and organization (Dillenbourg and Jermann, 2010) as well as the sheer volume of efforts already conducted (Frohberg, 2006), suggests that the domain is negatively affected from the lack of overview and ability to reuse different artifacts of the projects. One example would be components that should be reusable, independent, and executable units that can be composed into a functioning system (Szyperski et al., 2002). Another reusable asset is processes. Processes are any mechanism used to carry out work or achieve a goal in an orderly way: this is not limited to an executable environment (Osterweil, 1987). Documentation that support these processes, as well as other documentation such as requirements, diagrams, manuals or user feedback are also a reusable aspect.

The observed lack of reuse is a concern that especially needs attention due to the possible benefits it may offer (Jones, 2000) . Processes in the field are also limited to runtime, which leaves developers and practitioners with an incomplete view of the entire “landscape.” Thus, efforts that emphasize the entire process of software creation for the field would have the possibility to positively affect the development projects in the field. Having a coherent formalization of the life cycle and processes in some of these implementations would form the ground for a common understanding of the anatomy of these systems.

1.2 Problem definition

A challenge facing the utilization of mobile and ubiquitous technologies in educational settings is the sheer amount of different configurations available. Considering the amount of software and hardware configurations that are in use today, the customization needed for delivery to all platforms is a significant challenge (Calvary et al., 2003). Figure 1.1 represents the facets of the problem. Part of this challenge lies in generalizing applications to many platforms, but also in utilizing the variabilities of every platform. Developing applications in this environment while maintaining functionality and quality over all of these instantiations is time consuming. In the context of mobile/ubiquitous learning implementations it is even more complex, as applications need special functionality to meet educational requirements (Traxler, 2008).

Development for the mobile domain have always been affected by the paradox visualized in Figure 1.1. This has sometimes led to either fragmentation, where companies simply ignore platforms, or more recently to the use of cross-platform solutions. These problems are inherited in the development of M/U-learning applications. Even if the mobile market is at this point stabilizing between the largest actors, releasing for these platforms simultaneously is difficult (Butler, 2011).

Thus, the need to utilize methods to shorten development time is crucial for this domain. It would seem that it is not a sustainable approach to software development in this field not to prioritize the reuse issue. Even more so, when considering the implications introduced by U-learning where the integration between software systems becomes a challenge as they are expected to work as a transparent and unified entity (Yahya et al., 2010). The modus operandi of the community is seemingly not a sustainable one as to productivity. As the mobile industry is showing no signs of approaching a less diverse state, the problem is likely to persist or even become more complicated still (International Telecommunication Union, 2010).

1.3 Scope

The research conducted in this thesis aims at exploring the mobile learning domain, focusing on component and process reuse, with the long term goal of constructing a foundation for a more rigid approach to development in the field. Mobile learning is a part of the field of Technology Enhanced Learning (TEL). TEL is concerned with the technological support of any pedagogical approach that utilizes technology (Goodman, 2002). It can be abstracted into three areas of domain challenges: learning, social and cognitive; design and interaction; and technology and engineering (Kurti, 2009). This thesis is concerned with technology and engineering.

That opportunistic reuse is not an option for any reasonably sized project has been known for long (Jacobson et al., 1997). In order to make any significant change, a systematic approach towards reuse is needed in order

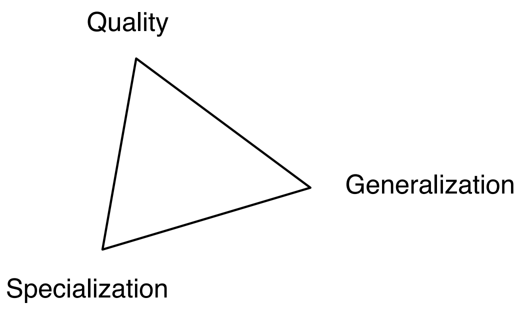

Figure 1.1: The development requirements paradox

to allow the systems and applications to scale beyond limited tests. The claim made in this thesis is that systematic reuse in some cases enables the community to build systems faster, with higher quality, and for multiple platforms. This thesis thus investigates how software development in the domain can approach the issue of software reuse and reap the benefits.

1.4 Overview

This thesis is organized as follows (Figure 1.2). The coming section gives the theoretical foundations that guided the work conducted in this thesis. It discusses software reuse in general and gives an overview of notable efforts and practices in the field. This leads up to a focus on software product lines, software ecosystems and an overview of the practice in the field of TEL. With the current practice discussed, Section 3 elaborates on the research goals and efforts of the thesis. The scope, goals and research questions are discussed as well as the research methods utilized. This section also provides an enumeration of the results, discusses validation, and puts forward a research map that outlines the research activities conducted. Section 4 provides an overview of the publications appended to this thesis. Section 5 analyzes the research efforts, providing a summarized view of the findings presented in the appended papers. Section 6 concludes the thesis by answering the questions posed in Section 3, as well as by providing possible directions for future work.

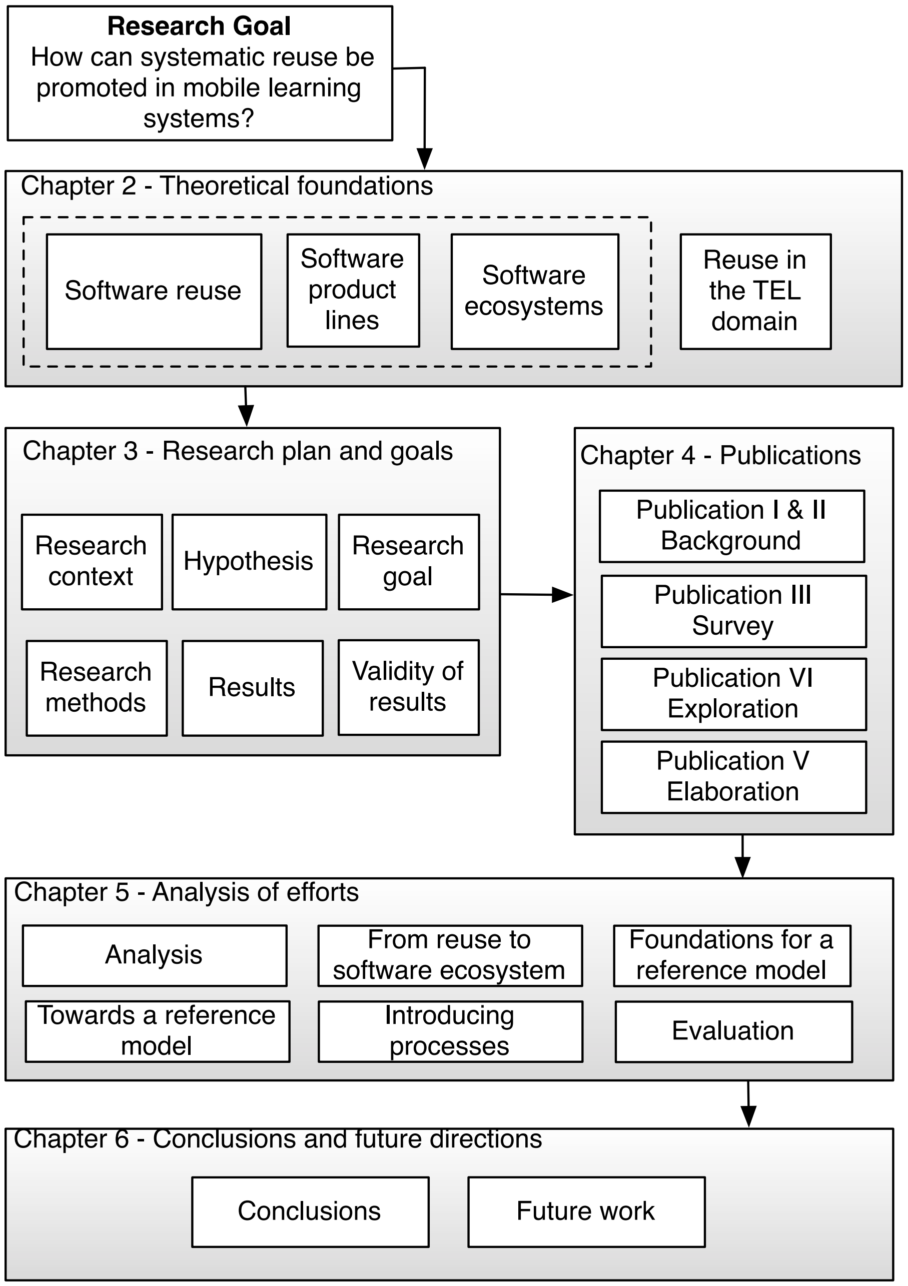
Research Goal
How can systematic reuse be promoted in mobile learning systems?
Chapter 2 - Theoretical foundations
Software reuse
Software product lines
Software ecosystems
Reuse in the TEL domain
Chapter 3 - Research plan and goals
Research context
Hypothesis
Research goal
Research methods
Results
Validity of results
Chapter 4 - Publications
Publication I & II Background
Publication III Survey
Publication VI Exploration
Publication V Elaboration
Chapter 5 - Analysis of efforts
Analysis
From reuse to software ecosystem
Foundations for a reference model
Towards a reference model
Introducing processes
Evaluation
Chapter 6 - Conclusions and future directions
Conclusions
Future work

Figure 1.2: Disposition of the thesis

Chapter 2

Theoretical Foundations

This section presents the theoretical foundations that serve as the basis of this thesis, which explore some of the challenges related to reuse in the field of Mobile Learning. This section thus elaborates on fields of research that are relevant to the work presented in this thesis. These fields are software reuse, its different sub-practices (software product lines and software ecosystems), and reuse in TEL, which will be elaborated upon in that order.

2.1 Software reuse

In order to be able to discuss what efforts are conducted in terms of reuse in TEL, we must first define what reuse is and how it is being practiced. Software reuse is one of the oldest practices in Software Engineering. Described by McIlroy et al., (1968) as:

> "Develop systems of components of a reasonable size and reuse them. Then extend the idea of 'component systems' beyond code alone to requirements, analysis models, design and test." McIlroy et al. (1968)

Constructing artifacts and systems from previously developed, proven and high-quality components decreases development time and improves overall system quality (Frakes and Isoda, 1994, Jacobson et al., 1997). On its most basic level, ad-hoc reuse has been successfully applied by individual developers for a long time. There are several drawbacks with this practice however. One of these being that it implies that reuse is up to the individual programmer and thus is hard to scale and even harder to control by the organization. However, there are special cases where ad-hoc reuse is a viable approach (Jansen et al., 2008).

2.1.1 Reuse: Past to present

A field as wide as software reuse is not isolated and is therefore affected by what is happening in the general discipline of software engineering, since reuse efforts logically depend on what (and how) people are currently developing software. Explaining software reuse is to some extent also explaining the field of software engineering. This section elaborates on some of the notable efforts in the field of software reuse but in order to put these into context it also mentions events outside of the domain of reuse. The development of the field is represented in Figure 2.1. The time and occurrences should, however, not be regarded as absolute, but as more general pointers between important notions and decades.

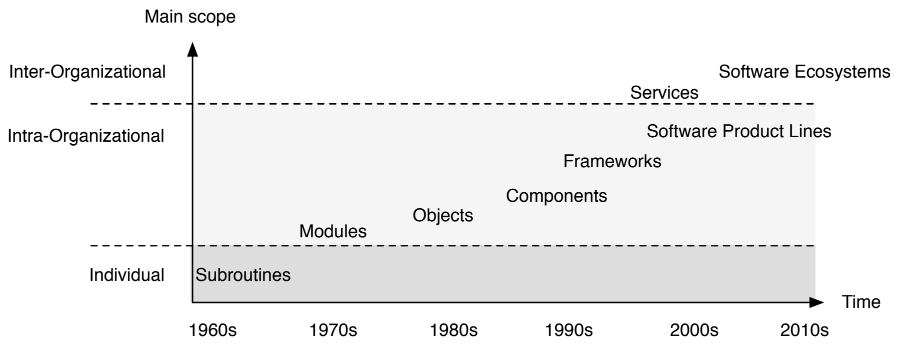

Figure 2.1: Reuse-trends over time

McIlroy is often quoted as a pioneer in the field of software reuse (McIlroy et al., 1968). While this paper has been accredited to be the first to present a vision of a software component industry, reuse existed before that. Engineers' desire to avoid repetitive tasks resulted in the use of a library of subroutines at EDSAC at Cambridge (Wilkes and Renwick, 1949). McIlroy envisioned a software factory where components from a catalog were composed into products, much like electronic systems. At the time however, employment of opportunistic reuse was the only widespread type of reuse. During the end of the 1960s, the phrase *software crisis* was uttered. Software engineering simply was not on a par with technical development and the complexity of software and hardware had gone out of control (Dijkstra, 1972).

During the 1970s, the System Development Corp. trademarked the term *software factory*. This was done in conjunction with an effort that aimed at realizing the vision McIlroy had brought forward years earlier. However, reuse was, at best, implicit in this effort (Lim, 1998). In the same period, the critique against the waterfall model was put forward (Royce, 1970). He argued that the way systems were currently developed was flawed and an iterative process was needed in order to ensure quality artifacts. The fact that black-box reuse was not working as theorized was realized during the 1970s and other solutions were pursued. The first effort that formalized reuse at an organizational level was also conducted by Lanergan and Poynton (1979). During this time, efforts towards the modularization of software also emerged. These, among other things, sought to segregate design decisions prone to change from the more stable parts of an implementation and thus enabled changes to be contained within the affected part of the system. By providing only interfaces and hiding the implementation from other components, it is possible to protect modules from extensive changes in other modules (Parnas, 1972).

Research projects like PCTE (Portable Common Tools and Environment) and REBOOT (Reused Based on Object-Oriented Techniques) were founded in Europe. With the goal of researching the reuse of tools (Boudier et al., 1989) and "providing methods and tools to support creation and use of domain-oriented components" (Sindre et al., 1995), respectively, these projects helped to establish tools and methods for a more practical approach to reuse during the 1980s. A software process model called the spiral model was developed by Boehm (1988). This model presented a move towards an iterative development process with several incremental prototypes. Every iteration goes through four stages: analysis, evaluation, development, and planning.

During the 1980s, Object-Oriented Programming (OOP) gained popularity and would grow to become a significant programming paradigm (Rumbaugh et al., 1991). Analysis gained further favor due to the rise of Object Orientation (OO) and James Neighbours was the first to put forward the role of "domain analyst" (Neighbors, 1984). This practice is one of the key components to realizing systematic software reuse. Frakes wrote about the paradigm shift to systematic reuse and described it as "domain focused, based on repeatable processes, and concerned primarily with the reuse of higher level lifecycle artifacts such as requirements, designs, subsystems etc." (Frakes, 1994). Domain analysis is also an important part of object orientation, enabling models such as the Unified Modeling Language (UML). The notion that in order to have success in reuse, a holistic approach is needed, gained further acceptance (Griss, 1993).

These holistic approaches continued in the 1990s and a focus on processes and maturity models of software reuse became a recognized practice, the first published maturity model being the one by (Koltun and Hudson, 1991). This model puts forward five different levels of maturity: Initial/Chaotic, Monitored, Coordinated, Planned, and Ingrained. Each level of maturity has ten characteristics, providing a mechanism to assess and evaluate reuse maturity in projects. To this decade can also be attributed the Rapid Application Development (RAD) methodology as well as Scrum, which came a few years after. RAD is a software development process that focuses on short iterations and rapid development. This has made it popular for the development of, for example, frameworks. Just as does RAD, Scrum focuses on short concentrated iterations but also contains practices and roles.

Components as an extension to OOP also grew in popularity during the 1990s. The ability to encapsulate logic in functions has existed for long. However, components have important characteristics that make them particularly good in the scope of OOP. All data and functions inside a component are semantically related. They are substitutable so that a component can replace another at both design and run-time. Microsoft Windows utilized both statically and dynamically linked components in the form of Dynamic-Link Library (DLL) and library (LIB) files. Frameworks as an extension of software libraries also gained momentum during this period.

Frameworks are reusable abstractions of code wrapped in a well-defined Application Programming Interface (API). Four characteristics differentiate them from libraries and general user applications: inversion of control, default behaviors, extensibility, and non-modifiable code (Johnson, 1997). Components and frameworks are still an important aspect of both reuse and general software engineering but with the rise of the Internet, requirements changed and software engineering changed with it. In the following decade, the Internet became pervasive. This made software development processes have to deal with even shorter iterations due to the nature of web development. Methods like extreme programming brought about a further emphasis on short iterations. The unified process also gained in popularity, as well as Services.

Services are used as a way to encapsulate and logically group code. Commonly components, or modules into comprehensive and self contained entities performing tasks for other applications or components on request (Krafzig et al., 2005). A service does not specify any technology used inside it. This level of abstraction enables architects to incorporate legacy systems into new designs by simply regarding everything as services (Bell, 2008). Bringing the concept of services to the web has resulted in popular implementations such as REST (Representational State Transfer) and SOAP (Simple Object Access Protocol). Many companies with a large web presence are offering Web Services alongside their web pages. Services can also be used as an architectural approach for entire systems and is then referred to as Service Oriented Architecture (SOA). Software product lines (SPLs) gained in popularity as success stories were published that promised vast improvements in maintainability, scalability and development time (Clements and Northrop, 2001). SPLs will be further discussed in the next section but in short, they mimic the assembly line principle and enables the construction of a product family from a portfolio of components. They focus on internal processes and composition in order to achieve this factory-like behavior. One of the major drawbacks of SPLs is the fact that it is not feasible for smaller projects, as the cost and complexity of implementing a SPL will be higher than merely implementing the project. Weiss and Lai write that with a spartan SPL, the payoff can be achieved after around three projects (Weiss and Lai, 1999).

Software product lines are largely an internal affair inside the company, but with the possibilities that the Internet brings in the form of communication and collaboration opportunities, outsourcing some component manufacturing is not far-fetched. A way of doing this is by creating a software ecosystem (SECO). They do however not imply the existence of SPLs. The notion of SECOs will be further elaborated upon in a later section. SECOs have existed for a long time, but with the collaborative possibilities the Internet brings, SECOs are now a seemingly attractive way of developing software.

As visualized in Figure 2.1, the general trend in software reuse has gone from individual ad-hoc reuse to a inter-organizational, systematic and formalized reuse. This does not imply that the other forms have disappeared.

Garlan et al. (1995) put forward a challenge they referred to as *architectural mismatch*: cases where reuse is difficult due to sub-parts making conflicting assumptions about the environment in which they were designed to operate. In 2009, a follow-up article was published where the authors claimed that even if some of the challenges had been addressed, the notion still stands. Todays' computing landscape also introduce new challenges such as trust, dynamism, architectural evolution, and architectural lock-in (Garlan et al., 2009). The inclusion of externally developed components is an issue of trust, but in SECOs the notion of trust is more complex. Trust must be placed in the chosen platform's success and technical solutions, in the policies, in the security, in the fact that you will get compensation for your work, etc.

This overview puts forward a few emergent contemporary trends. Systematic reuse is one of those. Focus on things that are not code (methods, processes, documentation, etc.), which is one of the prerequisites for systematic reuse, is another (Humphrey, 1995, Jacobson and Bylund, 2000). The effort to minimize development time, increase quality, and the paradox of specialization/contextualization versus the need to be on many platforms, are some of the most outstanding problems. As far as systematic reuse goes, SPLs are one of the most prominent practices today. Another prominent practice is, as mentioned earlier, software ecosystems, which although still in its infancy as a field of research, shows promise in being a part of the solution to the paradox of specialization, generalization and quality previously discussed. This, as it allows organizations to bring in outside stakeholders to specialize or generalize their software platform for them in a formalized manner. The following section will further examine SPLs.

2.2 Software product lines

The field of software reuse has far from stagnated despite its relative age. That opportunistic reuse is not a viable option in large development projects was discussed in the previous section. The size of development projects is growing, and even though the community has developed numerous methods and techniques to combat this, this expansion gives rise to a constant need for improvement and refinement. One of the more prominent alternatives for large-scale reuse is SPLs (Clements and Northrop, 2001).

The term software product line draws, as the name implies, from the practice of product lines and simplified aims to enable a product line where components are put together from a platform or repository and turned into a product ready for release (Pohl et al., 2005). A product line approach aims to make gains in productivity by constructing a portfolio of products with many parts in common, thus enabling reuse of these common parts (McGregor, 2004).

Software product lines exist in many varieties but in their most basic form

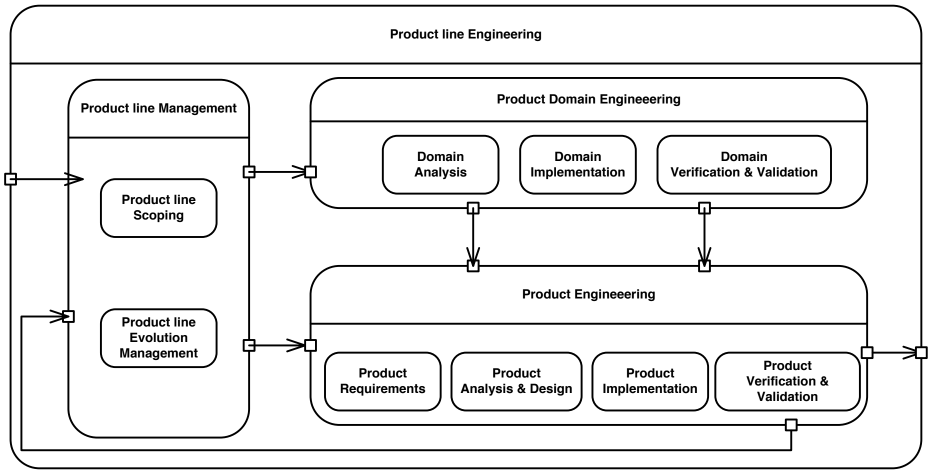

Figure 2.2: A Software Product Line Engineering Process, adapted from Hallsteinsen et al. (2008)

they consists of a software platform shared by a set of products. Each product then has the ability to select and configure components for the product as well as to extend the functionality of the platform itself (Bosch, 2009). A software product line can be seen as divided into three parts: domain engineering, product engineering, and product line management (Figure 2.2). Product domain engineering handles the analysis of the domain, common (reusable) components and validation/verification of the product lines. Product engineering refers to the handling of individual products, creating product-specific parts and integrating all aspects of the individual products. Product line management handles scoping and evolution management for the entire product line.

There are various examples from industry where SPLs have been successfully implemented. One of the early examples is Hewlett-Packard, which obtained a significant decrease in defects (Toft et al., 2000). Nokia is another example (Bosch, 2002). They implemented a large-scale program of product lines where products are built of components that are themselves product lines. There are various other examples, such as Philips, Siemens, and DNV (Van Der Linden et al., 2007). These cases all show that a formalized and systematic approach (in this case using SPLs) towards reuse is beneficial and scales well inside of a company.

2.3 Software ecosystems

A recent focus in the field of software reuse is the notion of SECOs. The foundations of SECOs have been around since software first allowed the inclusion of externally developed components. Operating systems for personal computers have for example commonly been in the center for SECOs.

Reviewing Figure 2.2 and considering the notion of a software ecosystem, all the individual parts would be candidates for externalization to outside actors. For example, externalizing the creation of some of the customized components for products would be consistent with the way in which many services on the web allow developers to access data via APIs. Externalizing domain analysis could be likened to asking the community for comments and feature suggestions. A lone developer with little connection to the outside world will rarely make applications that are usable or relevant for many others. Developers thus need to take a lot of other things into consideration and a vivid feedback loop from users and testers is a requirement (Messerschmitt and Szyperski, 2003). There have been several attempts to define SECOs. One such is the following one,

> "A software ecosystem consists of the set of software solutions that enable, support and automate the activities and transactions by the actors in the associated social or business ecosystem and the organizations that provide these solutions." (Bosch, 2009)

This attempt defines the SECO as a collection of software solutions (and organizations) and thus, something that can exist in conjunction with other dimensions of a general business environment, for example one or more business ecosystems. Jansen et al. (2009) provide another definition of SECO defined as:

> " a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them. These relationships are frequently underpinned by a common technological platform or market and operate through the exchange of information, resources and artifacts. "

This definition is similar to the one given by Bosch (2009) but goes so far as to say that a common platform is frequently a requirement for this kind of phenomenon. Both definitions imply externalization of certain parts of the business of the companies involved with the intent of benefiting everyone involved in the SECO. This is also the basis of the argument that SECOs are to some extent the natural evolution of SPLs. SPLs enable to build products from components and SECOs in addition allow some of these components (and compositions) to be made by external actors. The current definitions thus do not differ greatly from each other.

SECOs can also be described using terminology from the field of business ecosystems (Berk et al., 2010). The formalization of business ecosystems can largely be attributed to (Moore, 1996). It describes ecosystems in the terms of a central hub, a platform and niche players (Iansiti and Levien, 2004). The central hub is the driving force behind the ecosystem (such as Microsoft, Apple, Oracle or SAP), which is largely involved in the platform, and niche players are invited to perform business in these environments.

Considering recent trends such as cloud computing and the many "app stores" launched, this area of research would seem to be far from a decline in researcher interest. But, and in contrast to SPLs, the field of SECOs does lack substantial efforts, for example in the area of processes, methods or general case studies. More research is also needed in the direction of a general theory for software ecosystems (Hanssen, 2011).

The recent popularity of "app stores" as a method for keystone players to provide a uniform way of distributing software inside of an ecosystem is an indicator that for consumer devices, the ecosystem is now a strong selling point (Gueguen, 2009). Reconnecting with the problem definition presented in Section 1.2, SECOs are enabling keystone players to focus on platform and core services while peripheral developers provide software diversification. This does hint at a way to avoid the development requirements paradox by simply allowing others to take responsibility for one or more of the facets.

2.4 Reuse in the TEL domain

Having elaborated on the field of software reuse in the previous section, this section focuses on efforts related to reuse in the chosen domain of TEL. Formalized reuse in TEL can be placed in three distinct categories: content, component and knowledge reuse. Content reuse refers to the reuse of entities such as learning objects and learning designs but also of processes and the documentation related to software projects. Components have already been defined in Section 1.1 and the definition still stands. This section thus describes each category in turn, starting with content reuse. A notion that is hard to classify are patterns, as they can be applied to both categories, but are also in themselves a reusable asset. These will thus be separately discussed in the last subsection.

2.4.1 Content reuse

This section discusses several efforts and standards that have been directed at content reuse in the field of TEL. The standards for content reuse in this domain deal with learning objects, orchestration, and processes, all put in the context of learning.

The first of the standards of interest is IEEE-LOM (Learning Object Metadata) which is concerned with learning object descriptions (metadata), simplifying searching and filtering of learning objects (LOs), for example (Neven and Duval, 2002). SCORM (Sharable Content Object Reference Model) is a standard that profiles many other standards such as IEEE-LOM and IMS-CP (Instructional Management Systems-Content Packaging). SCORM exists in a few versions and encapsulates different standards, depending on the version. It also specifies how packages are meant to interact with one another (SCORM, 2009). However, SCORM largely neglects group activities (Karampiperis and Sampson, 2005). There are also related efforts from IMS including the previously mentioned CP, CC (Common Cartridge), IMS-SS (Simple Sequencing), ePortfolio, and QTI (Question and

Test Interoperability). The last standard is IMS-LD (Learning Design) and which focuses on the orchestration of content and the modeling of learning processes, but differs from SCORM in that it is not an aggregated standard and allows for groups, among other things (IMS, 2005).

2.4.2 Component reuse

Component reuse has so far mostly been implemented in the form of web services (Vossen and Westerkamp, 2003). The subsection following this one will discuss a few efforts implementing component reuse via plug-in components. It is difficult to evaluate how formalized these plug-in features are as they are only mentioned briefly. However, there have been several efforts implementing web services in learning systems by using SOA, which implies formalization, as services need to have a more or less formalized interface. As SOA implies services and in those services are commonly web services, it contributes to the prospect of reuse. As it also defines processes, it goes one step further in terms of possible reuse (Erl, 2005). The validity of approaching services for learning was discussed thoroughly in (Dagger et al., 2007). In that work it is claimed that defining rigid frameworks and building monoliths goes against the nature of the Internet. Some of the efforts in this direction include those described by González et al. (2009). They present a SOA adaptation of Moodle that enables support of both externalizing internal services and vice versa, thus easing possible reuse efforts. Another effort in this direction is (Dietze et al., 2007). They present an architecture that enables semantic abstraction of existing learning content as well as processes.

The main argument for approaching TEL systems from an ecosystem perspective is the holistic view that it provides. The fact that TEL systems are no longer able to operate effectively as monolithic entities has forced this in the latest years with the need for integration with other systems (Chang and Uden, 2008). Another effort to summarize existing models of ecosystems for learning is presented by Guetl and Chang (2008). They, however, conclude that no models exist to date for sufficiently depicting learning ecosystems. In order to establish the state of Mobile Learning Systems, a brief survey of architectural approaches and formalized reuse in those systems is presented in the next section.

2.4.3 Knowledge reuse

The notion of working with patterns is derived from Alexander (1977), who defines the term as;

> "A pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." (Alexander et al., 1977)

Patterns can thus be applicable for describing many different notions in TEL, both of a pedagogical and technological nature (Goodyear and Retalis, 2010). This section will thus give a brief overview of both, pedagogical and technological patterns. It, however, emphasizes the technological part in the form of architectural patterns in order to establish an overview of the state of M-learning systems and their approaches to formalized reuse.

A pedagogical pattern can denote many different aspects of pedagogy. An example of a pedagogical pattern can be found in the work of Garzotto and Poggi (2010) that presents a pattern for collective role-playing as a learning opportunity. One combining this with social networking is Cloudworks (Conole et al., 2008). This platform enables practitioners to discuss different approaches to pedagogy as well as incorporating technology. It also enables visual modeling of these via an external tool. An effort that utilizes patterns is the Beehive framework, which combines educational design patterns and software components in order to allow practitioners to execute learning experiences (Calvo and Turani, 2010). There are also efforts detailing experiences with utilizing patterns in implementations with greater detail such as those described by Derntl and Motschnig-Pitrik (2010).

A software architecture is made up of components, connectors, and the properties of both of these (Albin, 2003). An architectural style or pattern is thus a specific combination of these components, connectors and properties that solves a particular common problem in software architectures. Patterns thus enable solutions to common challenges in software architectures to be reusable. Examples of patterns are model-view-controller architectures, commonly used in web applications or peer-to-peer architectures commonly used in content distribution systems.

A software architecture may use many patterns. In order to provide a representational view of the which patterns are popular in M-learning system, a limited survey was conducted. This survey draws its data from two editions (2008 & 2010) of the IEEE International Conference on Wireless, Mobile and Ubiquitous Technologies in Education (WMUTE). This particular venue was chosen as it attracts a fair amount of publications with a technical orientation. The overview in Table 2.1 shows the different kinds of architectural styles that the projects claim as well as eventual formalized efforts in both content and component reuse. Absence of such efforts are denoted with a "-".

The overview shows that M-learning systems in almost all cases are Client–Server based. Client–Server in its most basic form means that the server provides a service or a function to one or many clients, which make requests for it to the server. There are exceptions, however, with some blackboard distributed systems and a monolithic application. The Blackboard pattern mandates a solution space with several application-specific hierarchies with domain knowledge partitioned into independent modules (Garlan and Shaw, 1993). Some also use commercial off-the-shelf (COTS) software, which in this survey refer to the architectural style of facilitating the integration of those. The N-tier pattern mandates that the system sep-

Project	**Content reuse**	**Component reuse**	**Architectural approaches**
C-Writing (Chen et al., 2008)	-	-	Components, Three-tier
Butterfly Ecology (Wu et al., 2008)	-	-	Client–Server
MySportsPulse (Metcalf et al., 2008)	-	-	Components, Client–Server
MC-Supporter (Zurita et al., 2008)	-	-	Peer-to-Peer
Participate (Woodgate et al., 2008)	Proprietary	-	Monolithic, COTS
Dog Detective (Hsu et al., 2008)	-	-	Client–Server
Teacher Monitoring System (Ku et al., 2008)	-	-	Client–Server
SMILE (Sampson and Zervas, 2008)	IMS-LD	-	Monolithic
Groupware for Mathematics (Tao et al., 2008)	-	-	Client–Server
Mobiledu (Liu et al., 2008)	-	-	Components, Client–Server
COLLAGE (Lohr and Wallinger, 2008)	-	-	Client–Server
RBPL System (Shih, 2008)	-	-	Client–Server
ULEER (Yin and Yang, 2008)	-	Components	Components, Client–Server
LEMONADE (Giemza et al., 2010)	-	Component plug-ins	Blackboard, Components, Distributed
Mobltz (Lewis et al., 2010)	Mobltz Markup Language	-	Client–Server
SceDer (Niramitranon et al., 2010)	Classroom Orchestration Modeling Language	-	Components, Client–Server
Go Math! (Alexander et al., 2010)	-	-	Distributed
MESLL (Li et al., 2010)	-	-	N-tier, Client-server
DLASH (Liu et al., 2010)	Semantic Markup	-	COTS, Client–Server
CLEQ (Hwang et al., 2010)	-	-	Client–Server
Group Scribbles(Lin et al., 2010a)	-	-	Blackboard, Components, Distributed
P.M.L.A.B.O.M.L.P. (Yau and Joy, 2010)	Codewitz learning objects	-	Client–Server
Microblogging System (Lin et al., 2010b)	-	-	N-tier, Client-server
MoMAt (Di Bitonto et al., 2010)	IMS LIP	-	Components, Client–Server
SIGMA (Kayama et al., 2010)	-	-	Components, COTS, Client–Server
Statecraft X (Chee et al., 2010)	-	-	Client–Server
WeTangram (Jimenez and Lyons, 2010)	-	-	Client–Server
Let's Go (Vogel et al., 2010)	-	-	N-tier, Client–Server

Table 2.1: Overview of formalized reuse and architectural approaches.

arates concerns into separate processes. The use of formalized content reuse approaches varied. However, the use of IMS or SCORM standards was not the norm. Some publications mentioned learning objects but did not further specify the term and were therefore omitted from this category, as the term does not imply a standard. Formalized component reuse was largely non-existent with the exception of some plug-in component approaches. Thus, M-learning system architectures are mostly homogeneous in their nature, being Client–Server systems.

2.5 Summary of findings

These theoretical foundations have elaborated upon software reuse from the early efforts to contemporary formalized efforts such as SPLs and SECOs. These efforts towards formalized reuse have been accompanied by numerous successful cases that validate the efforts. SPLs as an approach towards formalized reuse are mainly intended for use internally within an organization, which makes it a less than ideal candidate for smaller systems in a domain that changes rapidly. SECOs on the other hand might allow small groups to rapidly develop applications by utilizing components already developed by others and by focusing on the novel part of the application.

Formalized reuse in the TEL domain was here split into three types, content, component and knowledge reuse. Content reuse has a number of standards in place, and is a mature field. We also discussed patterns in order to classify implementations in the domain. The overview presented in Table 2.1 shows that implementations are usually homogeneous in their anatomy as Client–Server systems. It also shows some attempts at content reuse, but very few adopt the assumed standards from IMS, SCORM or IEEE. Component reuse has been largely neglected in the surveyed publications.

To Note that none of the presented efforts sufficiently deal with the problem of component reuse in respect to variability or integration (as seen in Table 2.1) is to summarize these TEL reuse approaches. There also is a lack of depth in the discussions around learning ecosystems where aspects such as processes are ignored. The paradox of specialization, compatibility and quality is another matter that is not addressed in any of the surveyed systems. The next chapter will elaborate on the research plan and what goals this effort aims to achieve.

Chapter 3

Research Perspective and Methodological Considerations

The previous sections described the scope of the research. Its motivation was discussed and its theoretical foundations elaborated upon. Based on this, we derive research questions and devise a research plan. We first elaborate on the context of the research in order to provide a description of the settings in which the research was conducted. After that, a hypothesis is formulated based on the concepts and ideas discussed in Section 2. To pursue this hypothesis, the research goal is formulated in the following section which contains the main research question of this thesis. Thereafter, a number of supporting questions that further clarify the research focus of the thesis are presented. Subsequently, we discuss the research methods, present an overview of the results, and briefly discuss their validity. In the final section we provide a research map that describes a schematic view of the research plan.

3.1 Research context

To better understand the work presented in this thesis, some background, to provide a context of this work, is required. All of the efforts depicted in this thesis have their roots in projects conducted at the Center for Learning and Knowledge Technologies (CeLeKT), a research center located at Linnaeus University. The research conducted at CeLeKT is multidisciplinary, focusing on innovative utilization of technology in educational settings emphasizing mobility and collaboration. The projects are geared towards exploring technological and pedagogical aspects of learning activities and the validation of learning outcomes and learning experiences.

The domain specific knowledge is derived from several of these projects from CeLeKT, where the author actively participated as a designer and developer. One of these was the AMULETS project, where we were exploring how teachers can develop and implement novel educational scenarios combining outdoors and indoors activities that use ubiquitous computing technologies together with stationary computers. The AMULETS project consisted of three different activities conducted by students between middle school and university levels (Spikol et al., 2008). In the later iterations, the students were split into teams with indoor and outdoor sub-groups that had to collaborate to solve a series of challenges. In order to achieve this, the students were invited to utilize several different tool utilities in their provided mobile phones.

While analyzing the architecture and technological aspects of these projects there are plenty of different components and platforms needed for successfully executing an activity of this type. Desktop computers were utilized for the indoor groups, who interacted with the system and other peers via an interactive web page. The outdoor counterpart interacted with the system and their colleagues via a chain of mobile applications, started by the triggering of a 2D bar code by a proprietary application that then opened a web browser. Another utility device per group was added to enable documentation and communication via photos, text and audio.

This required tailored components for special platforms. A major issue arose when the phones used were upgraded from one device (used in the first trial) to a more recent device. Most of the components had to be rewritten as APIs had changed. Hardware features such as screen resolution and methods of input had shifted, which also caused the components to break. The lack of component re-useability was in itself not surprising, as the situation was equally unfavorable in several other projects that ran in parallel at the time. Several reasons were given in evaluations and reflections afterwards. Most prominent among the reasons were issues related to interoperability, changing hardware conditions, and changing requirements.

This motivated a literature survey to identify possible remedies for these issues. The survey uncovered similar issues in related work in the field of mobile learning. This prompted the survey to broaden the scope to include the domain of software reuse to uncover what had been done to address similar challenges in other fields. This survey uncovered a favor towards a systematic approach to software reuse that has been proven to be effective in several domains. This led to the forming of a hypothesis.

3.2 Hypothesis

Considering the motivation, problem definition, and the notions and ideas presented in the theoretical foundation, a hypothesis has been formulated. Theory and practice in the research field of software reuse has made progress since McIlroy and it is now established that a systematic approach to reuse has a beneficial influence on software projects. Applying this practice to mobile learning systems thus is likely to have a positive effect, especially considering that the field already has made progress with the reuse of content. The hypothesis of this thesis is that *introducing systematic reuse in the TEL domain, and especially in the mobile learning sub-domain, would result in systems with better quality, shorter development time, and of lower cost.* This would in turn benefit the adaptability of systems to new technology and devices, in turn enabling the domain to focus on domain issues instead of re-implementation. Increasing the quality of systems is not one single parameter however. Quality in the domain of TEL can refer to many things but in this thesis we primarily regard factors such as correctness, reliability, useability and re-useability, as pointed out in Fresen (2007).

3.3 Research goal

Software development in the domain of mobile learning has been argued in this thesis to exhibit some gaps in respect to reuse. In the theoretical foundations section, alternatives from the general field of software reuse that could be introduced into the domain of mobile learning have been discussed. Formulating the main line of research into a concrete question is common practice and provides a focused thread that should go throughout the thesis. Pursuing the hypothesis, the main research question explored in this thesis is formulated as follows:

How can systematic reuse be promoted in mobile learning systems?

Decomposing this question, there are several sub-questions that need to be investigated before an answer to the main question can be given, the main point being that the current approaches to reuse in the domain are in some aspects inadequate. The result is unnecessarily time-consuming development processes and a culture of "use once, throw away."

3.3.1 Research questions

In order to further clarify, to increase precision, and to better explain the main research question of this thesis, a set of supporting questions are formulated in this section. The questions are split into two types: the initial questions Q1 and Q2 are aimed towards addressing the foundation of the research effort. Question Q3 is aimed at gathering what parts of current practice can be refined. As a result of the analysis and empirical work performed to answer the first three questions, the research led to the formulation of an additional question, Q4.

3.3.2 Foundation

The first category of questions is aimed towards establishing a foundation for addressing the main problem of the thesis. Dividing the main question into two areas of concern, software reuse and mobile learning, there is a natural emergence of two questions, each addressing, in part, the current state of the domain of software reuse (Q1) and approaches and standards for promoting reuse in the domain of mobile learning (Q2). The outcomes will bring an understanding of the chosen domains and provide reports on which to base Q3 and Q4.

Q1 What is the current state of software reuse?

Q2 What is the current state of approaches and standards promoting component and content reuse in the domain of mobile learning?

3.3.3 Requirements and refinement

The second category of questions is concerned with how the current practice of reuse could be improved and the refinement of these contributions. The outcomes of these in conjunction with the main research question will be a set of descriptive models.

Q3 What aspects of the current practice of general reuse in mobile learning can be improved upon?

Q4 How may the proposed descriptive model impose consistency and structure on all activities?

The following section will elaborate on the methods chosen to address and develop an answer to these questions.

3.4 Research Methods

This section describes the research methods used. Software engineering has a variety of established research methods, but for this effort which is of both theoretical and exploratory nature, some of these are more suitable than others (Sjoberg et al., 2007). Thus, the selected methods are survey and case study, which will be further elaborated here.

3.4.1 Survey

Surveys are a common method for collecting information and conducting exploratory research in many fields and the field of software engineering is no exception (Wohlin et al., 2000). The method collects input from various sources and then produces tangible results by the analysis of that input. It excels when answering questions such as what, how, how much, how many and why (Pinsonneault and Kraemer, 1993). In pursuit of the research questions formulated in the previous section, two brief surveys were conducted: one to establish the current state of software reuse and the other to establish the state of TEL in respect to standards and practices of reuse. The first survey is mainly concerned with theoretical work and looks at journals, books and papers published in the field of software reuse. The main objective of this survey is to identify notions that could provide some insight into the challenges of reuse in mobile learning systems. The data used for analysis in this survey was filtered by a few conditions. To establish a ground, several renowned articles and books were used. To isolate possible directions of recent trends in software reuse a selection of recent journal and conference publications were used.

The second survey is of a slightly more practical nature and looks both at implementations and theoretical work in the field of TEL in respect to reuse and software ecosystems. The survey studies different functionalities and standards used in popular systems, chosen by user-base size and license model (open source vs. proprietary).

The outcome of these surveys serves as a basis for an analysis that resulted in both a domain analysis of the chosen domain and a descriptive model depicting the domain.

3.4.2 Case study

A case study is a qualitative method that can be used to study a particular phenomenon that is little or not at all controllable by the researcher in its context (Yin, 2003). In this thesis, a set of different cases are investigated in order to establish a view of what features do the systems in Mobile Learning have and how they are generally deployed. The cases were chosen for several reasons. The author has intimate knowledge about the systems, they follow a line of chronological progression and are developed by the same group (with the exception of the aggregated case). A limited study was conducted in order to evaluate the descriptive model to ensure that it supported the things that we claimed it supported. This study is further elaborated upon in Section 5.4.

The outcome of this study subsequently showed that the initial approach to a descriptive model was incomplete and did not cover all the areas. It thus initiated research question 4 and another iteration of analysis.

3.5 Results

This section enumerates the results claimed by this thesis, in the order of the research questions. The results will be further elaborated in Section 5 but to be able to consistently refer to the individual contributions, we enumerate them below.

R1 Report

- I Depicting the current state of standards and approaches promoting reuse in mobile learning.

R2 Domain analysis

- I A descriptive model depicting the joining of the domains mobile, learning and ecosystem.
- II A life-cycle model of a learning activity
- III A description of the three domains mentioned in R2 I.

R3 Descriptive model

- I A conceptual model improved to incorporate support for Q4.

3.6 Validity of results

Validation of models is a challenge for the scientific community. Descriptive and conceptual models are not exceptions. Conceptual models are subjective in nature as they are human interpretations of concepts or systems

(Moody, 2005). Conceptual models can be seen as a part of requirements engineering as they to some extent reflect the requirements put on the entity it describes. There have been efforts to automate comparisons between models but completely excluding human involvement in the tasks of model interpretation has so far proven futile (Pfeiffer and Gehlert, 2005).

Validation of conceptual models is conducted by; (1) example and (2) evaluation. Validation by example means the model needs to show its applicability to the things it claims to depict in real development projects. This applicability can be shown by utilizing the model in real projects, or develop prototypes that show that the model can be implemented. These prototypes use the packaged knowledge of the model and if the prototype functions satisfactorily, it gives validity to the model.

Validation by evaluation is another method of assessment where theory is used. Due to the exploratory nature of a licentiate thesis, the models presented here are validated by evaluation. The models put forward as results in this thesis are R2 (I and II) and R3. R2-I is an overview of the three domains and their joining. Thus, it is based upon result R2-III that is a textual depiction of the three domains. R3 is based upon a combination of R1 and R2 in conjunction with the intermediary result of the initial reference model. With the validation discussed, the next section shows an overview of the research activities that where carried out in this thesis.

3.7 Research map

In order to provide a comprehensive overview of the research activities, a research map is depicted in Figure 3.1. The circles are representations of the research questions. At the top, the three initial questions can be seen as the start of the research process. These questions are used as inputs, as indicated by the filled arrows in the flow, for various activities, which are symbolized by rounded rectangles. Q1 and Q2 both warranted surveys in order to be answered. There is, however, a notable difference between Q1 and Q2. The survey warranted by Q2 resulted in a result in the form of a distributed report, represented by a bold square. The survey for Q1 was used as an input for the general analysis in order to answer Q3.

The analysis activity resulted in two different results. The first result consists of a domain analysis, consisting of several descriptive models and textual descriptions of the domain. The second intermediary result, which uses the first result as an input, is a descriptive model depicting the domain of Mobile Learning Software Ecosystems.

The outcome of this descriptive model is grounded in the theoretical work presented before it. But in order to be claimed as a result, a brief evaluation was conducted in respect to the case study projects. This evaluation however hinted at several flaws in the initial model. These flaws triggered the formulation of a complementary research question, Q4. In order to answer this question, another analysis was performed on the initial model with the flaws in mind. The outcome of this analysis suggested that paying closer attention to processes hinted at a a solution to these flaws. This enables the final result, the final descriptive model that will be presented in more detail in Section 5.

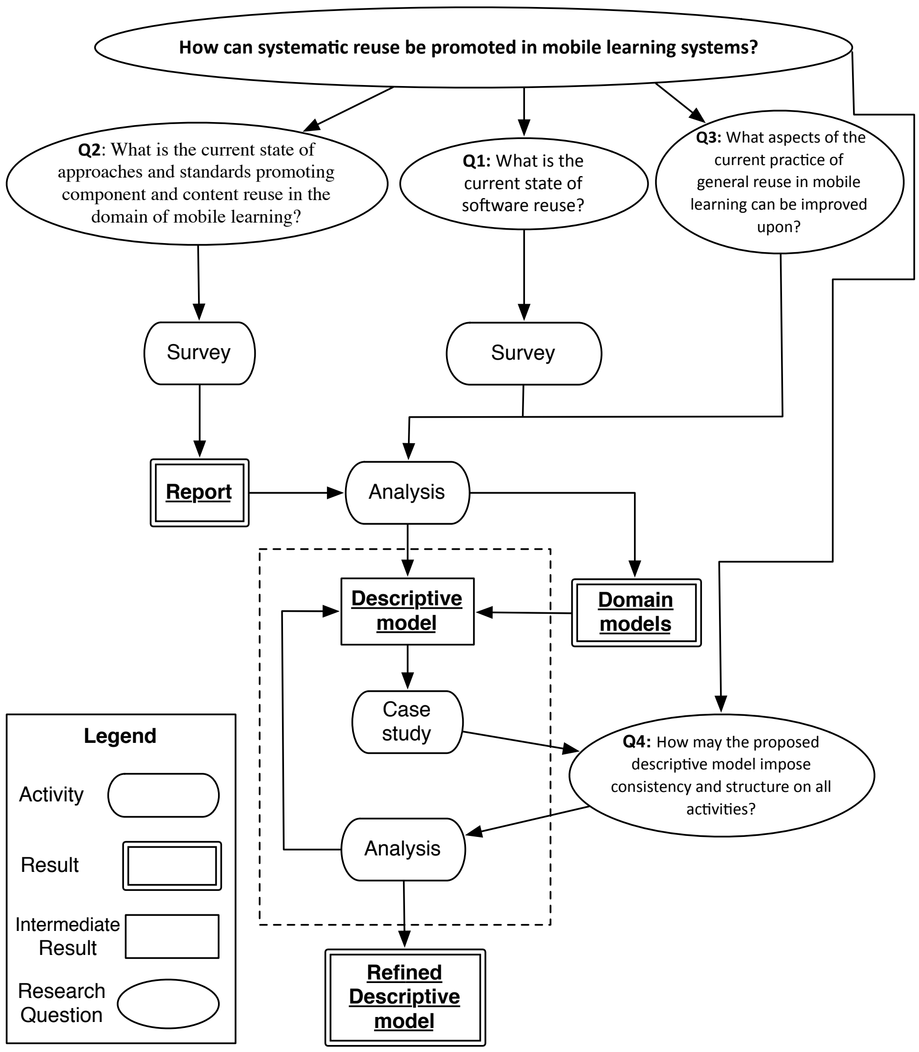
How can systematic reuse be promoted in mobile learning systems?
Q2: What is the current state of approaches and standards promoting component and content reuse in the domain of mobile learning?
Q1: What is the current state of software reuse?
Q3: What aspects of the current practice of general reuse in mobile learning can be improved upon?
Survey
Survey
Report
Analysis
Descriptive model
Domain models
Case study
Q4: How may the proposed descriptive model impose consistency and structure on all activities?
Analysis
Refined Descriptive model
Legend
Activity
Result
Intermediate Result
Research Question

Figure 3.1: Research map

Chapter 4

Publications

This section provides an overview of the five appended publications, together with a summary of their respective contributions. Figure 4.1 illustrates the connections and flow between the publications and the relation to the research question and sub-questions posed in Section 3.3. Publication I elaborates on the background of the thesis and the problems with the lack of reuse in this domain. It also illustrates one of the activities performed in the form of a part of the AMULETS project. Publication II continues on this path and further elaborates on AMULETS. This publication elaborates on two additional scenarios and shows that the lack of reuse is not limited to the one activity presented in Publication I. Publication III follows by exploring reuse approaches in TEL in order to identify possible parts that can be improved upon. Publication IV further explores the notion of reuse in the domain and formulates a conceptual architecture depicting the concepts involved in a mobile learning system. Publications III and IV together explore the research question formulated in Q3. Publication V then goes on to make a domain study of the field of mobile learning and presents several descriptive models depicting the field of mobile learning. It concludes with a final reference model that is aimed at providing a stepping stone for further discussion and comparison between efforts implementing software ecosystems for the field of mobile learning.

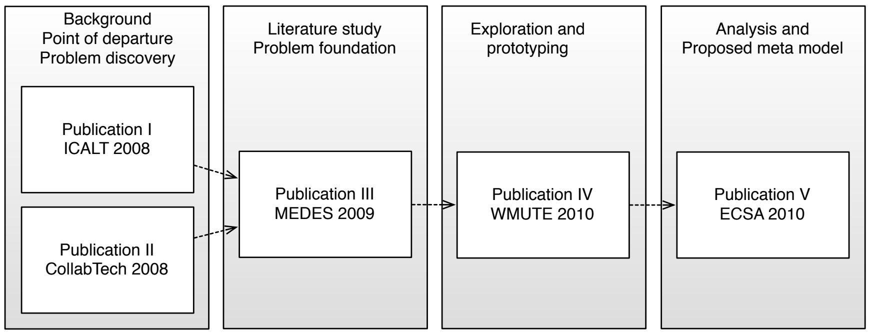

Figure 4.1: Graphic representation of appended papers

4.1 Publication I

Svensson, M. and Pettersson, O. (2008). Making Use of User-generated Content and Contextual Metadata Collected during Ubiquitous Learning Activities. In Proceedings of *the Eight IEEE International Conference on Advanced Learning Technologies* (ICALT, 2008), Santander, Cantabria, Spain, July 1st–July 5th, 2008.

This paper provides an overview of a mobile learning activity with the name AMULETS Biology, which is a sub-activity of the AMULETS project. It elaborates on the implementation of the ACS (Activity Control System), which is a system for orchestrating mobile learning activities. The ACS is used to orchestrate a mobile learning activity that invites the participants to learn more about various trees in the neighboring area. The paper to some extent focuses on interoperability at the data level which was seen as a main priority at the time. The paper describes the architecture behind the project and notes that ACS is a fairly generic architecture for developing these kinds of activities. This paper is the point of departure for the research goal of this thesis as even though the data interoperability was not fully addressed, the even greater challenge of software reuse became apparent as this was the second project in the AMULETS series.

4.2 Publication II

Spikol, D., Milrad, M. Svensson, M., Pettersson, O., and Persson, M. (2008). Mobile Collaboration Tools and Systems to Support Ubiquitous Learning. Proceedings of the *Fourth Annual Conference on Collaboration Technologies,* (COLLABTECH, 2008), Wakayama, Japan, August 30th–31st, 2008.

The second publication depicts a number of projects in the domain of mobile learning we have carried out as a part of the efforts that went into this thesis: two different activities from the AMULETS project and another one called Skattjakt. The common thing with these three activities is that they were all implemented on a system called LAS (Learning Activity System). The LAS included the ACS described in the previous section as well as other components to enable the implementation of diverse learning activities. The LAS evolved during these activities, however, and had to be extensively modified and customized between each of the activities. The client applications used to access the LAS where also different in each of the iterations, running on different platforms and using different technologies and sensors. This publication does not explicitly state it, but in reality, it hints at the hypothesis of this thesis, that the domain would benefit from reuse. The publication only briefly outlines the architecture utilized in the different projects but even from the textual descriptions, it is possible to deduce that the reuse between the projects was not excellent. The future work described in this publication was to evaluate the technological approaches taken in the presented cases, which is one of the major factors for the motivation of the next publication.

4.3 Publication III

Pettersson, O. (2009). Software Ecosystems and E-learning: Recent Developments and Future Prospects. In Proceedings of *The International ACM conference on Management of Emergent Digital EcoSystems (MEDES09),* Lyon, France, October 27th–29th.

This publication is an initial attempt to address the concerns raised in the previous two publications about the lack of reasonable software reuse in these mobile learning activities. The paper thus makes a survey of different standards for general reuse in the broader domain of TEL. This survey enabled the identification of a potential base for interoperability between systems. What came out of this study was that there are several standards for both content and composition interoperability. A brief study of the larger systems (as in a big deployment base) for TEL was conducted in conjunction with the previous study. This study surveyed the actual use of the standards that the first study put forward. The study revealed several notable results but the most evident was that there is a widespread adaptation of standards for content reuse. The study also revealed that the adaptation of standards for composition was nonexistent.

The paper provides an overview of the term Software Ecosystem and notable efforts in the area. It goes on and briefly explores different aspects of the systems approach towards their developers, users and Software Ecosystem in general. The systems all have the ability to be extended by external developers by the means of plug-ins. This is supported a bit differently but the open source systems generally have a more open approach towards their external developers. This ability to reuse components was a major motivation for Publication IV. The state of some of the larger Software Ecosystems in TEL is declared promising but capable of improvement. The survey turned up no notable efforts towards a mobile learning software ecosystem however, which is the major motivation for Publication V.

4.4 Publication IV

Pettersson, O. and Gil, D. (2010). On the Issue of Reusability and Adaptability in M-learning Systems. *Proceedings of the Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education*, (WMUTE 2010), Kaohsiung, Taiwan, April 12th–16th 2010.

This publication builds upon the challenge of the lack of reuse that was found in Publication I and II and then further investigated in Publication III. The result of the survey in the previous paper being that there are no significant efforts addressing the lack of software reuse in mobile learning, the publication gives a brief overview of other efforts of partial reuse in the domain. Examples of that are standards such as IMS-LD, SCORM. The publication also elaborates on the nature of mobile learning as a domain with a plethora of problems due to variable configurations. Due to this, the domain is arguably a prime candidate for efforts related to reuse in general.

The publication argues for a conceptual architecture and a common ground for discussing these issues. By separating logic, resources and agents, a conceptual architecture for emphasizing reuse is put forward. This architecture is then briefly validated as to concepts and orchestration by a prototype implementation. This prototype was successfully used in two activities, one called GEM (GEometry Mobile) and one called MULLE. GEM and MULLE were learning activities in the field of mathematics. In GEM students were invited to calculate the heights of buildings using different methods and to measure the areas of large fields supported by mobile technologies and applications. In MULLE, basic geometry was explored. In order to generalize the findings and lessons learned from these efforts, an envisioned architecture was also provided for use in future work.

4.5 Publication V

Pettersson, O., Svensson, M., Gil, D., Andersson, J., and Milrad, M. (2010). On the Role of Software Process Modeling in Software Ecosystem Design. *Proceedings of the 2nd International Workshop on Software Ecosystems,* Copenhagen, Denmark, August 23rd.

The last of the publications builds further upon the line of argumentation from Publication III and Publication IV. This publication continues on the line of reuse but puts forward a further focus on Software Ecosystems as a component for solving the issues of reuse in the domain. In order to examine this, a classification of the concepts involved in a mobile learning activity ecosystem was proposed. This classification outlines outstanding concepts in the fields of Mobile, Learning and Ecosystems. Based on this explorative work of the different fields, a few descriptive models were developed. The models depict different parts of the domain chosen such as the life-cycle of an activity. It also includes an initial reference model for Mobile Learning Activity Ecosystems.

This reference model is evaluated against several use cases. These use cases are based on real world examples of mobile learning activities. This evaluation hinted on several shortcomings in the reference model. These were related to expressivity, adaptivity and adaptability of concepts among other things. To that end, the model was re-engineered and changed in order to facilitate the needed changes. This resulted in the adaptation of SPEM in the model. This adaptation had several benefits, due to SPEM being a proven and well-used model for modeling and describing processes.

4.6 Summary

To summarize, the common thread throughout the publications is the notion of reuse. As it has been argued so far in this thesis, putting further emphasis on reuse in the development of mobile learning systems could prove beneficial to several aspects of the domain. In pursuit of this, the present thesis has included five publications that show a line of research efforts pursuing the notion of reuse in mobile learning.

The first two publications (Publication I & Publication II) revealed the lack of reuse in the domain. The third publication (Publication III) followed up on this gap and provided an exploration of both the available standards for reuse and the approaches taken to SECOs by some of the larger systems in the domain. Based on the outcome of this, Publication IV elaborated on the different aspects of reuse in the domain of mobile learning. These two efforts combined can be seen in Figure 3.1 as the result Report and is enumerated as R1 in the result list. Drawing on the outcomes of Publication III and Publication IV, the final publication (Publication V) summarized the different sub-parts of the domain of Mobile Learning Activity Software Ecosystems (R2 III). It also elaborated on several descriptive models including a descriptive model of the joining of the different domains (R2 I in the result list and Descriptive model in the research map), a life-cycle model of a learning activity (R2 II). This paper culminated in a reference model deriving concepts from both the explorative study and the adaptation of a software process engineering metamodeling language which is enumerated as R3 and known as the Refined Descriptive model in the research map.

Chapter 5

Discussion

This section provides an analysis of the different stakeholders involved in mobile learning projects and the types of formalized reuse present in these efforts. The section proceeds by describing the challenges associated with formalized reuse and its effects on the different stakeholders. We propose a path from ad-hoc reuse to a software ecosystem that consists of four steps, starting with presenting the domains that constitutes the area of interest: mobile, learning and ecosystem. With the domains outlined, a life-cycle model of a mobile learning activity is presented. The domains are then elaborated upon in greater detail separately, in order to identify the concepts that are candidates for inclusion in one cohesive model of the domains. Based on these elaborations, a reference model is constructed and evaluated. This evaluation results in another iteration of analysis presented in Section 5.5. Here the issues of the first reference model are reviewed and addressed, resulting in an improved one.

5.1 Analysis

In this thesis, we have identified problems related to reuse in the field of TEL with a focus on M-learning. Software reuse spans the range from ad-hoc reuse, via different practices to systematic reuse approaches such as SPLs and SECOs. Approaches to systematic reuse in the field of TEL were also surveyed and several of the current efforts have been discussed in the previous chapters. This subsection identifies the types of reuse possible in the TEL domain together with a stakeholder analysis and maps stakeholders to specific types of reuse. The ideas discussed in this section elaborate on the roles of different stakeholders that have been identified and then continues with a discussion on the influence reuse may have on each one of these stakeholders. At the end of the section, an overview regarding the drawbacks and advantages related to reuse is discussed. We made a distinction between three types of reuse as previously discussed in Section 2.4; (1) content reuse, (2) component reuse, and (3) knowledge reuse. We observed that the focus of the TEL domain has been on content reuse, but in order to promote systematic reuse the scope must be widened and include all types. The next section maps these to what reuseable assets they include.

5.1.1 Reusable assets in TEL projects

In a software development project many different artifacts are created. Each artifact is a candidate for reuse. In TEL projects additional artifacts of different kinds are created. Jones lists ten types of reusable assets, as described in Table 5.1 (Jones, 1993). Greenfield and Short (2003) claims that knowledge could be considered as the eleventh reusable asset, formalized as patterns.

(1) Architectures	(2) Source code
(3) Data	(4) Designs
(5) Documentation	(6) Estimates
(7) Human interfaces	(8) Plans
(9) Requirements	(10) Test cases

Table 5.1: The different types of reusable assets (Jones, 1993)

Prieto-Diaz (1991) demonstrates a method for making faceted classification schemes of software reuse. This classification scheme categorizes reuse into entities, which are categories of reuse, and activities in these different categories. Following that line of reasoning, this thesis classifies reuse in the domain of TEL into three facets (entities) that include: (1) content, (2) component and (3) knowledge. The concepts elaborated upon in this section are denoted with what types of reusable assets, described in Jones's list (Table 5.1). Learning design as a concept appear in several places, as it is a wide notion and overlaps several areas of the proposed classification. When placed under content reuse, it refers to composition, sequencing and other aspects of content handling, while under knowledge it refers to learning design as a way to reuse pedagogical ideas, logistical tips or any other domain-specific insights. This list denotes the different kinds of reuse and how they relate to the reuseable assets presented in Table 5.1.

- Content
  - Learning Objects (3, 5, 9)
  - Learning Design (3, 1, 5, 9)
- Component
  - Subroutines (2, 9, 10)
  - Modules (2, 9, 10)
  - Components (2, 9, 10)
  - Frameworks (2, 3, 4, 5, 9, 10)
- Knowledge
  - Design Patterns (4, 11)
  - Architectural Styles (1, 4, 7, 11)
  - Learning Design (4, 6, 8, 9, 11)

We elaborate on the stakeholders involved in m-learning projects below and then will map the stakes in reuse to the respective stakeholder. Based on that knowledge we continue and discuss the challenges involved implementing formalized reuse.

5.1.2 Stakeholders

The different types of reuse mentioned previously provide benefits to different system aspects. However, in order to understand a domain there is also a need to analyze its stakeholders and their roles in the system. Freeman (1984) proposes a seven step method for systematic stakeholder analysis.

(1) Develop a stakeholder map of the project
(2) Prepare a chart of the specific stakeholders
(3) Identify the stakes of stakeholders
(4) Prepare a power versus stake grid
(5) Conduct a process level stakeholder analysis
(6) Conduct a transactional level stakeholder analysis
(7) Determine the stakeholder management capability

This subsection focuses on defining the stakeholders found in a generic m-learning project. This definition is conducted in a fashion inspired by the method proposed by Freeman (1984). At this point we are only interested in the stakeholders and the stakes that these have. Thus, we make a two step approach consisting of (1) and a combination of (2) and (3) from the proposed method. We let two projects, GEM (Pettersson and Gil, 2010) and AMULETS (Kurti et al., 2008), represent a generic m-learning project. Following Freeman's method the first step is to develop a stakeholder map of the project. This map is presented in Figure 5.1 where the circles denotes stakeholders, the square the project, and the arrows show that there is a relation between the stakeholder and the project.

The second step in our approach, which combines steps (2) and (3) from Freeman's method, aims to map the stakes of the domain to the stakeholders identified in step one. This is first done by describing the stakeholders and their stakes in reuse. We then conclude by clarifying the mapping between stakes and stakeholders in Table 5.2.

End users of the system are learners and instructors. Learners are end-users and utilize the system to support learning processes. A learner in this context can not be seen as a passive receiver. They may produce a wide range of digital content that can be part of learning objects. Learners can benefit from the reuse of learning objects as learning objects may be reused in several activities. By providing communities for learning objects, learners may benefit from consuming libraries of LOs on multiple devices such as their mobile phone, in a best case scenario.

Instructor represents a wide range of individuals, a teacher that wants to employ Mobile Learning in his/her school class, or a parent who wants

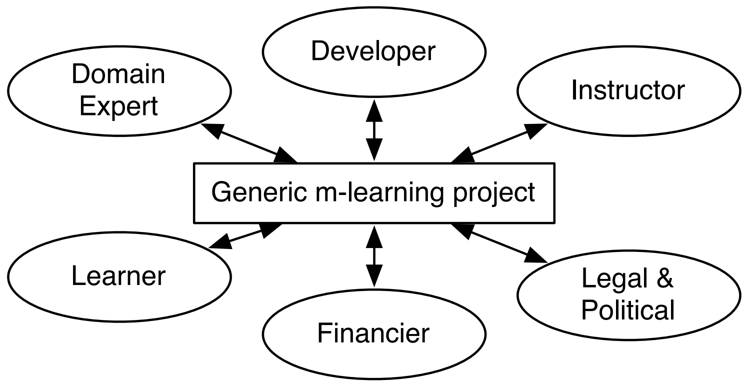

Figure 5.1: Map of stakeholders

to extend learning activities into the free time of a learner. Even a learner may become an instructor. Instructors are essentially gatekeepers and moderators as well as conductors in a specific learning activity. If a learner is unable to finish a task for example, instructors are able to redirect the learner to a new task. Instructors do not benefit in the same sense from learning objects, as this role does not create new content. They do however benefit from learning design as it provides structure and information about the flow of the activity and it supports instructors in their role.

A core group of stakeholders is composed by those developing the activity: developers and domain experts. A developer in this context is anyone who is involved in the development of the system that does not generate any learning content. This stakeholder may be defined with higher granularity but in the review of the publications reported in previous sections, it is rare that there is any mention of a differentiation between, for example, architects and testers. Developers are thus grouped together. Developers may benefit from formalized reuse in many different ways. Formalized reuse approaches such as components, plug-in abilities, frameworks, services and even utilizing COTS all shorten development time and increase quality if correctly implemented.

Domain experts are a different type of development stakeholder. They create content, as well as handle the pedagogical flow and orchestration of the learning activity. They decide what can be in a lesson or activity and in what order. This stakeholder benefits from learning objects, which may provide reusable content of quality. They also may benefit from the orchestration and pedagogical support in learning design. Domain experts may also benefit from Learning objects as they can see how the content generated by students makes sense in relation to the learning activity. This specific area is central for the field of contextual metadata.

Financiers and administration is the last distinguished stakeholder category found with this approach. This is a kind of customer, which might be a research funding foundation or a government, that pays for the development and deployment of the system. This may well be a regulatory body, which can for example be the headmaster of a school, or a government, which has rules for what can and cannot be done in the context of a learning activity. These stakeholders benefit from learning objects as they provide an easily monitored metric of how a project is doing. They also benefit from services.

Legal & political represent a group of stakeholders that have no direct part in the activity (and thus are not end users) but are still interested in the results of the activity. These stakeholders mainly benefit from knowledge reuse, as if praxis for how activities are performed is established, their work of regulating and inspecting will take less time.

Reuse aspect	**Stakeholders**
Learning Objects	Learners, Domain expert, Financiers
Learning Design (Content)	Instructors, Domain experts, Developers
Subroutines	Developers
Modules	Developers
Components	Developers
Frameworks	Developers
Patterns	Domain experts, Developers
Styles	Domain experts, Developers
Learning Design (Knowledge)	Domain experts, Developers, Instructors, Financiers, Legal & Political

Table 5.2: Stakes and stakeholders

As this analysis looks at aspects related to reuse, the stakes will map to the facets of reuse presented in the previous section. This mapping of stakes and stakeholders is based upon the findings of the previous steps. Table 5.2 shows that domain experts and developers seemingly have the most to gain from reuse even though all stakeholders are involved. Noteworthy is that financiers and legal & political have stakes in reuse as well. This is due to the financial benefit of creating reusable assets that can be used in other projects. It is also beneficial for conveying methods and activities which ease inspections and conformity to regulations.

5.1.3 Challenges

Having suggested a classification of stakeholders and how they may benefit from reuse, there still exist plenty of challenges in this domain. As noted in the beginning of this thesis, there is a challenge with keeping up with the development requirements paradox depicted in Figure 1.1. Extensive reuse would enable developers to implement across platforms and standards, needing less effort. The results presented in Table 2.1 however indicate that formalized reuse has not been widely adopted. Formalized reuse is a double-edged sword. Standards for learning objects and learning design need to be implemented and integrated, which takes time. Developing reusable components also takes time, as they need to be generalized to fit a wide variety of learning scenarios. The hypothesis of this thesis is that introducing systematic reuse to this domain would result in systems with better quality, shorter development times, and of lower cost. The elaboration of ideas presented in this section, as well as those elaborated in Section 2, indicate that SPLs are not an ideal fit for solving some of these problems, as projects might never live long enough to benefit from this approach. A Software Ecosystem approach does not present this problem, provided the ecosystem is already available.

Software Ecosystems allows developers to share and compose applications from reusable components and make their own customization. This speeds up development and allows domain experts to share orchestration, learning objects, and compositions as well as patterns, which enables others to improve upon them and reuse. It allows instructors to choose and execute activities without involving developers and domain experts. This approach would also enable researchers access to a large dataset from all the activities in a given ecosystem. The motivation for moving from the ad-hoc reuse that is currently conducted in most of the surveyed systems towards a software ecosystem has been elaborated upon in these sections. The next sections aim to present the proposed path towards a software ecosystem for m-learning.

5.2 From ad-hoc reuse to software ecosystems

The first step on the path towards a software ecosystem is to perform a domain analysis. A graphical representation of these domains can be found in Figure 5.2. This representation is the baseline of this effort and the conceptual view of the combined domain and it is inspired by the work conducted by (Kool, 2009) where a model "that frames mobile learning," FRAME, is presented. FRAME focuses on device/individual/social interactions however and does not provide a complete picture for arguing about developments in the field. In short, the model highlights the three individual domains that are put together to create novel opportunities and possibilities. We see two distinct types in the domain analysis: entities and activities. Entities are the different concepts present in the domain whereas activities are the interactions and tasks performed.

5.2.1 Entities

For the mobile learning ecosystems which we consider, the domains presented in Figure 5.2 are Mobile, Learning and Ecosystem. Some of the benefits that combining these concepts brings can be seen in the intersections. The concept of Mobile represents facilitation for distributed settings managing a diverse set of mobile and fixed device resources and context-awareness. Learning represents everything from activities performed in the context of the domain such as resource creation, authoring and planning, to performing formal learning and teaching activities. The first intersection,

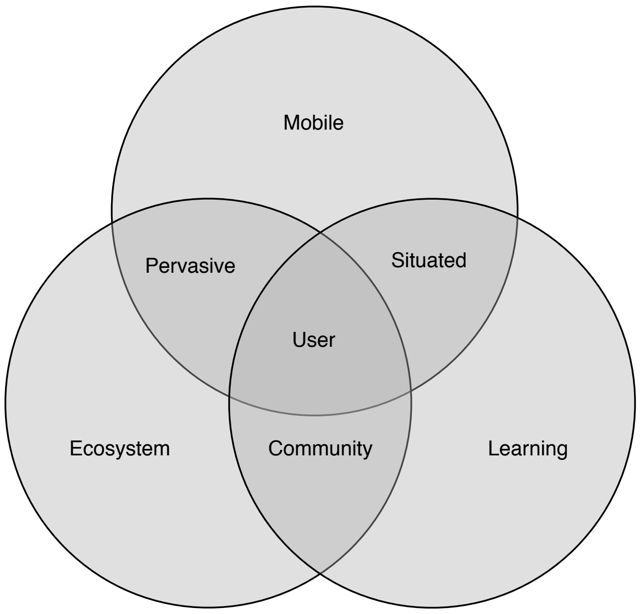

Figure 5.2: Joining the three domains of concern

the one between the concepts of mobile and learning (thus, representing the area of mobile learning), enables *situated* activities. "Situated" in this context meaning that things can be performed on a location in an authentic environment, for example learning about trees in a forest.

By introducing the third domain, Ecosystems, other properties emerge in the respective intersection. The broader meaning of ecosystem is further elaborated in the theoretical foundations but in short it refers to platforms, tools, processes, developers, domain experts, and community aspects, among other things. One of the emerging properties is *pervasive*, meaning that giving Ecosystem a Mobile aspect enables the system to avoid being tied to desktop systems. The last emergent property is the intersection between ecosystem and learning that results in a community. Community formation is one of the major benefits of ecosystems. One of the prime examples of the benefits associated with this is the F/OSS (Free and Open-Source Software) community (Scacchi et al., 2006).

5.2.2 Activities

Based upon the cases described in the AMULETS and GEM projects, a life cycle for mobile learning activities was constructed. A life cycle model provides understanding of how and why some assets are created, maintained and modified (Rodriguez-Martinez et al., 2010). Examining the AMULETS and GEM projects, we identify several patterns of activities. Two interconnected main areas of concern were identified, authoring and learning activity, as seen in Figure 5.3.

These two cycles contain sub-activities. The first cycle, Authoring, con-

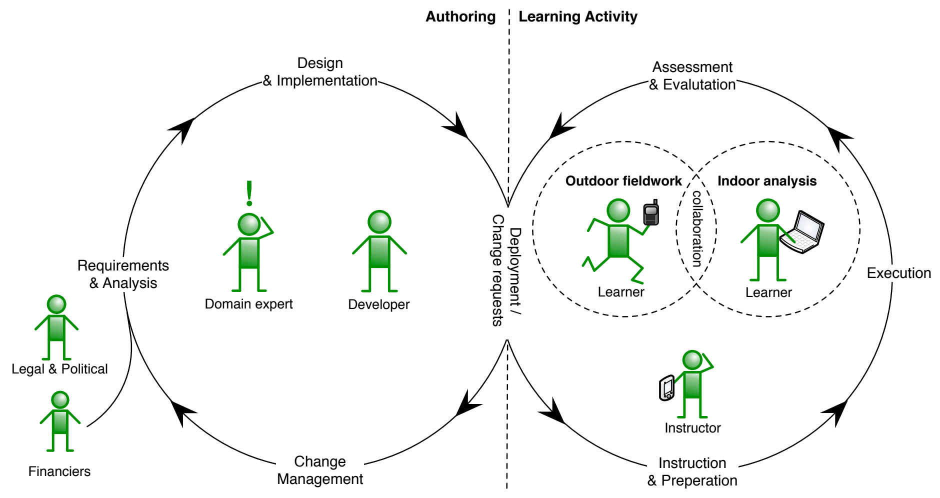

Figure 5.3: The lifecyle of a learning activity

tains the following: (1) Requirements and analysis, (2) Design and implementation, (3) Deployment and (4) Change management. Requirements and analysis elicits, specifies, and prioritizes the requirements imposed on learning resources and activities. In subsequent activities, resources and activities are designed and implemented, possibly through reuse that incorporates and integrates existing resources and activities. When the resources and activities are ready and tested, they are deployed in the system, thus making them available for both execution as learning activities and possible reuse in other settings. Change management analyzes change requests that already executed learning activities have generated and takes appropriate measures. The second cycle, the learning activity, represents the various stages of a deployed learning activity being performed. The sub-activities identified in this cycle are: (5) Instruction and preparation, (6) Execution, (7) Assessment and evaluation, and (8) Change requests. The instruction and preparation stage includes briefing participants in the activity on the different practical and technological aspects of the activity. This might include a theoretical foundation, intended to enable participants to better understand and follow the activity. Execution needs no further explanation. Assessment and evaluation focuses on post activities and reflection for active participants and other actors. This stage also refers to the activities surrounding the evaluation of the activity conducted and to enable the assessment of positive and negative aspects of the activity. Change requests are products of a cycle and imply a process of outlining and formalizing the desired and required features for coming iterations. These cycles are, as the intersection hints, not at all isolated. The lack of an end-state is very much intentional, to further emphasize the evolutionary aspects of a learning activity life cycle. Elaborating on the life cycle model has allowed for further understanding of the development process that takes place in a learning activity. This in turn allows reasoning around the sub-parts of the process of developing software in this domain. The next step in expanding a theoretical foundation for software development in the domain is to propose a reference model.

5.3 Foundations for a reference model

The hierarchy put forward by Bass et al. (2003) and depicted in Figure 5.4 mandates that in order to make informed decisions about development for a particular domain, a reference model is needed. A reference model illustrates how different concepts can be combined (Bourque et al., 1999). In addition, it is abstract and technology-agnostic. Such a reference model can be used to develop reference architectures for a domain. Reference architectures are used to derive product specific designs that in turn guide implementation. This flow is presented in Figure 5.4.

In the work carried out by Moody (2005), it is argued that there are several ways of evaluating a reference model. Most are applicable to categorizing reference models, for example: theory-based, experience-based, observation-based, and derivative-based. These methods are further detailed in Section 5.6. This reference model can be classified as a mixture of all the ways previously mentioned as it was established from both publications, real implementations, observations, and existing standards and practices.

A desideratum that was imposed on the reference model was that it should reflect the three main domains that were presented in Figure 5.2. In order to transfer these from broad domains into slightly more workable concepts to use in a reference model, exploration of the domains is required.

5.3.1 Mobile

The first domain in Figure 5.2 is mobile . The term can include many things but is likely to be used as a prefix (such as mobile web, mobile computing and mobile learning activities) and refer to the opposite of static. It can even refer to the users and interaction (Ballard, 2007). Recent advances in technology have pushed even further the boundaries of what is possible to put inside a device. Currently, many mobile devices have a GPS, WiFi, and a camera. High-end models even have light sensors, accelerometers and expansion possibilities to attach additional hardware such as pH or conductivity sensors. This enables developers to contextualize their applications in a whole new way, as they know more about the user context through the numerous sensors in the mobile device without needing the user to input this data manually. In (Kurti et al., 2008) *context* is defined as

> "The information and content in use to support a specific activity

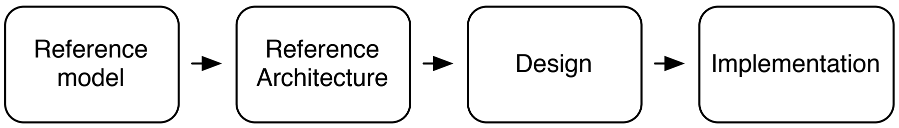

Figure 5.4: Hierarchy of models as proposed by Bass et al. (2003)

> (being individual or collaborative) in a particular physical environment at a specific time. ”

Besides the contextual information provided by sensors, user preferences are relevant to determining how to adapt software and content in a mobile device to function optimally in the current context. For example, when a user has bad network coverage, a textual representation of content should be sent instead of video. User Preferences provides personalization, a desired property due the large variety of existing applications (Barkhuus and Dey, 2003). Personalization further facilitates and enhances user interaction. Moreover, mobile applications also define requirements for mobile devices (Yamin et al., 2005). Furthermore, device features make it possible to acquire context as input and provide mechanisms to devices to adapt specific requirements and preferences.

5.3.2 Learning

The second domain is the domain of learning. Kurti et al. (2008) claim that learning, in the concept of TEL, can be described as a set of tasks performed by learners in the context of activities to achieve a particular educational goal. These tasks can generally be anything from a simple calculation or taking a picture to simulating the life of a lion in the savannah of Africa. What is important is that they are the atomic units of work for the Learning concept. An activity is performed by learners, under the supervision of one or many instructors. The degree of involvement varies depending on the activity being performed, including variables such as activity difficulty, or the skill of the learner. Learning also requires special attention to the material and the composition of Activities, thus there is a need for someone to design the Activity. Any individual may take on multiple roles at any given time, e.g., an Instructor may also have the role of a Designer.

5.3.3 Software Ecosystem

The third domain in Figure 5.2 is Ecosystem. We used the following definition as a basis.

> “A software ecosystem consists of a software platform, a set of internal and external developers and a community of domain experts in service to a community of users that compose relevant solution elements to satisfy their needs.” (Bosch and Bosch-Sijtsema, 2010)

This definition coincides with the earlier work of Bosch and to some extent with the definition by Jansen et al. (2009) presented in Section 2.3. This definition presents several concepts and actors that are required for the formation of a SECO. One of the central concepts in the definition is the interplay between the Producer Community and the Consumer Community where the Producer Community with Developers and Domain Experts develop Solution Elements to fulfill consumer needs.

5.4 Towards a reference model

With these domains elaborated upon separately, we will now join them into one cohesive domain. During the analysis, several common concepts have emerged. The concept of User exists in all three of the domains: thus this concept is used as a starting point when modeling the aggregated domain. One part that stands out in the reference model is the relationship between developers and users in the Mobile Learning Ecosystem as users can be developers and developers most likely are users as well. In Figure 5.5 this can be seen in the division of Community and User. The users are a part of the Community that influences the Software Platform that provisions the Solution Elements. In Figure 5.5, Community is a composition of several actors, these are, External & Internal Developers and Domain Experts from the Learning domain. The community composes Solution Elements which are consumed by the Users.

This model maps the three different domains and combines them, thus elaborating on all the concepts and relations involved. Based on the domains depicted in the model, it could be used as a standard for objects and their relationship. It also fulfills the other requirements set out such as such as enabling comparison between implementations. The model needs to be evaluated against software projects in order to give it validity, i.e., show that the model can describe these systems properly.

A short initial evaluation of this model was conducted using three experiments connected to three of our ongoing research projects that provided use cases (1) AMULETS (Spikol et al., 2008), (2) Let's Go (Vogel et al., 2010) and (3) GEM (Pettersson and Gil, 2010). This was a first step towards validating the model by doing functional walk-throughs of the systems. This was done by instantiating the system and trying to map functionality on to the model. This evaluation, however, found several outstanding issues with the model. These issues are discussed in (Pettersson et al., 2010). Briefly summarizing them, issues were identified related to the logic (design and implementation are separate processes), lack of solution element adaptivity, lack of separation between the development and the deployment use, and difficulties related to the adaptation of roles. This warranted further efforts in refining the model in order to mend these issues.

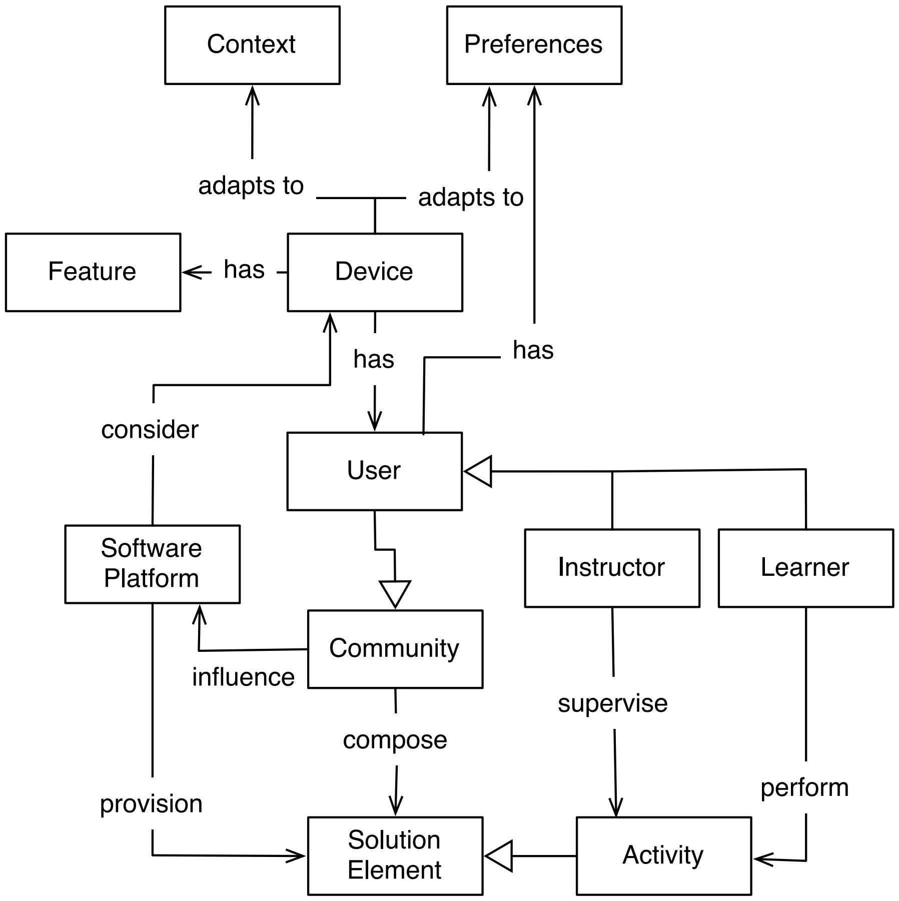

Figure 5.5: Conceptual aggregation of the concepts that are part of the three domains

5.5 Introducing processes

While reviewing the range of issues identified in the proposed conceptual aggregation (Figure 5.5), we investigated how to address these. In particular, a majority of the issues are concerned with specializations of role–artifact interactions as well as role–role interactions. In short, the model is not flexible enough in respect to role–artifact interactions. The issues outlined here are further elaborated in Pettersson et al. (2010). Promoting roles and artifacts to first class elements in the model brings several changes and enables further flexibility, as introducing Process components to the model will provide an abstraction for role–artifact interactions.

In order to realize process elements into the model, theoretical support is needed. Thus, the field of process engineering is consulted. This is discussed in more detail by Pettersson et al. (2010) but in conclusion, software process engineering rests on two pillars. The first is the relationship between the quality of the process and the products produced by the process. The second is that software processes are software too (Osterweil, 1987). Thus, it is necessary to approach process development in the same way as we approach software development in order to produce quality products.

The field of Software Process Engineering has already produced several meta-models, such as the ISO12207. The ISO12207 defines the area, but it is not on a sufficient level of detail for what we would like to do, refine the metamodel. SPEM (Software Process Engineering Meta-model) (OMG, 2008) is one more applicable to this purpose. SPEM allows for a level of detail where life-cycle processes, as well as the components, can be defined. This meta-model is constructed around three basic concepts: Role that performs Activities that produce and use WorkProducts (as seen in Figure 5.6). A WorkProduct is a composite of InformationElement instances. Examples of WorkProducts include models or any other type of artifact. An Activity in SPEM is defined as a number of steps. A Step in SPEM is on the same level of granularity as Tasks in ISO12207. The concept of Role is self-sufficient and is not further decomposable.

A first step towards introducing SPEM in the reference model is by introducing the ProcessPackage and its concepts. This enables a complete end-to-end description of processes, complete with Roles, Activities and WorkProducts. In addition, a process package also contains guidance, which can be anything from clarification notes to templates and guidelines.

Unwrapping SPEM's process package, there are four distinct concepts. The first reference model instance covered, to some extent, the concepts of WorkProduct (Solution elements and the specialization Activity) and ProcessRole (User with the specializations Instructor and Learner). It however completely lacks Activity (development) and Guidance. The concept of Activity can be found in the refined reference model as the Ecosystem Activity.

The top tier of the reference model should be considered as highly extensible, i.e., that these may be further specialized as sub concepts. E.g.,

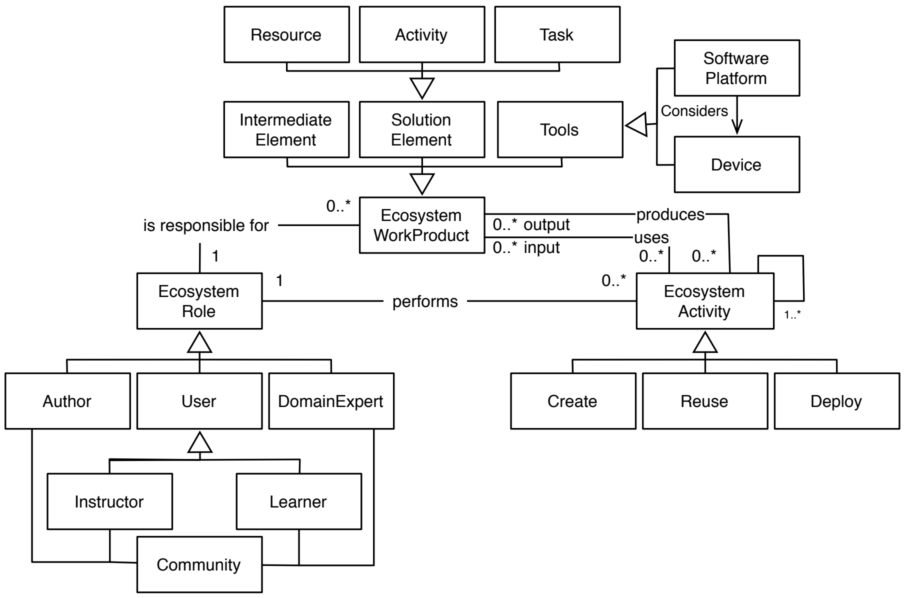

Figure 5.6: A reference model for Mobile Learning Ecosystems

specializations of Solution Element are not limited to Resource, Activity and Task. In the resulting reference model the protocols in role–artifact interactions are captured in EcoSystemActivity specializations, for example, as realizations of specific use cases. The introduction of Software Platform and Device as tools in the EcoSystemWorkproduct means that the additional features that these areas bring are available as integral mechanics in the processes. The Tools concept is just like most other concepts, further extendable, which allows for introduction of additional supporting mechanisms in the model. This allows the reference model to devise processes as solutions to problems exhibited in the previous model. Another aspect is that anyone may now describe any process by defining and introducing new process packages.

The model presented in Figure 5.6 can be seen as a first step towards a common vocabulary and a model to compare functionality between Software Ecosystems within the domain of mobile learning. The introduction of SPEM into the model has several benefits but the main concern that is addressed is the fact that software processes are software too. Highlighting this is something that has not been done before in this particular domain and implementing this realization holds several benefits.

5.6 Evaluation of the reference model

In this subsection we attempt to validate the quality of our reference model. As previously noted, the evaluation of conceptual models is a challenge as they are subjective in nature. The reference model can be evaluated from several perspectives including theory-based, experience-based, observation-based, and derivative-based (Moody, 2005). The theory-based approach implies that the reference model is evaluated against theory from a domain such as software engineering or other practices, i.e., publications, reports, etc. Experience-based evaluation on the other hand emerges from practice where people with experience from the field formalize their knowledge. The drawback of this approach is that this knowledge might not represent the entire domain. Observation-based evaluation constructs a model bottom-up approach based on observed facts. This approach achieves a high validity if many cases are observed but has proved complex in practice (Moody, 2005). Derivative-based evaluation uses established standards, and projects these onto the model. Below we reflect upon the reference model using the perspectives as our prism. Note that we do not claim that this is an unbiased evaluation. However, the point we want to make is that our reference model may be traced back to not one or two but four information sources, a fact that is likely to increase the model's quality (validity) and thus how trustworthy (reliable) it is.

- From a theory-based perspective, this reference model has been built using relevant pieces of knowledge that emerged from a significant number of publications in the field. These are discussed in Section 2 and analyzed in 5.3. These publications elaborate on the domains of mobile, learning and ecosystem that constitute the domains of interest. Based on this body of theory, we have produced an aggregation of the domains presented in Figure 5.2, which subsequently led to an aggregation of concepts as presented in Figure 5.5.
- From an experience-based perspective, the AMULETS and GEM projects were used as cases. The experience from those projects guided the efforts represented in Figure 5.3, which provides the activities in a mobile learning project. This enables a better understanding of the stakeholders and their activities in the domain as a whole. The findings provided in Section 2 that mobile learning projects are mostly homogenous in their architectural nature further supports this claim.
- From an observation-based perspective we evaluated the initial reference model from Figure 5.5 for flaws with respect to the AMULETS and GEM projects. This evaluation pointed out several weaknesses in the model, which triggered another evaluation round to address those weaknesses.
- From a derivative-based perspective we iterated over the initial model and derived things from the already established standard SPEM. The model was also implicitly inspired by earlier efforts of formalized reuse in the TEL domain such as IMS-LD.

As this licentiate thesis is of theoretical nature, aimed at providing a basis for future research directions, it lacks real world testing of the reference model that future research efforts will bring. This lack of real world testing makes validation of the reference model challenging. However, the different forms of evaluation give us a way of combining different methods to assess its validity. As a mixture of peer-reviewed sources, established standards, use-cases and experience have been used as a basis, the reference model has validity in respect to those approaches. This seemingly has produced a model that is reliable enough with acceptable quality, even if this can not be said for certain. This certainty will come from real world validation in future iterations of this scientific pursuit. The next section will conclude this thesis, further elaborate on the outcomes of this work and discuss where future efforts could be directed.

Chapter 6

Conclusions and Future Directions

This section finalizes the thesis by first summarizing the efforts undertaken for this work and then presenting possible directions for future work.

6.1 Conclusions

The previous section elaborated on the research efforts that have been conducted as a part of this thesis and its appended publications. This section reconnects with the research questions that formed the basis for this work and connects these with the results outlined in Section 3.5 and elaborated in the previous section. We enumerate the questions and highlight their answers, ending with a reflection on the overall research goal for the thesis. The first question was stated as follows:

Q1 What is the current state of software reuse?

This question establishes the theoretical starting point for all other questions as the problem that prompted the thesis was the finding that reuse in mobile learning systems was not a prioritized area. An exploration of software reuse as a field is presented in Section 2, which presents the general state of software reuse. This exploration showed among other things that one of the more promising directions the field is taking is towards a focus on Software Ecosystems as they facilitate a novel approach to utilize formalized reuse in collaborative settings. However, in order to specialize for the domain, the next question was formulated as follows:

Q2 What is the current state of approaches and standards promoting reuse in the domain of mobile learning?

This question narrows the scope of research to standards and approaches promoting reuse in mobile learning. As the previous question pointed out that SECOs are one of the more prominent paths in the field of reuse, is there something along this path in the domain? In R1 it is put forward that there are no prominent efforts in the line of SECOs in the domain of Mobile Learning. As R1 put forward however, there are several standards and approaches promoting reuse in the domain of TEL. There are several larger implementations that even make an effort with their SECOs. All these ideas are further elaborated in R1 and Section 2.4. The next questions build on the findings provided by Q2 to find what can be done to improve the current state of the domain.

Q3 What aspects of the current practice of general reuse in mobile learning can be improved upon?

Q3 is another question of an explorative nature. This question was answered in part by R2, which contains several descriptive models that maps the domain and to some extent its habits of reuse. These descriptive models are used as a basis for reasoning about what can be done in terms of reuse for this domain. This work is also elaborated upon in Section 2.4. The outcome of the work conducted trying to answer this question led to the conclusion that there are already several standards for content reuse in use. There is however, room for improvement in the general reuse approaches in respect to components and their approaches towards a healthy software ecosystem. The final outcome to address this question is the reference model. After evaluation, this reference model was found insufficient and triggered a fourth question.

Q4 How may the proposed descriptive model impose consistency and structure on all activities?

In pursuit of the fourth question, a process oriented approach was chosen. In order to enable the model to facilitate the desired features, the field of software process engineering was looked at. The field already has some established standards and practices. One of these is SPEM, which proved to be able to facilitate the required features. The reference model was changed to fit the ideas behind SPEM and resulted in R3. With the reference model adapted and refined, the main question and goal of the thesis remains.

MQ How can systematic reuse be promoted in mobile learning systems?

Reviewing the four previous questions leading up to this one, a line of argumentation has been drawn from basic software reuse towards SECOs. This thesis argues that software ecosystems are a feasible path to pursue in order to increase the adaptation of systematic reuse in systems for mobile learning. This line has been drawn via the supporting questions that surveyed software reuse, closing in on SPLs and SECOs in general, looking at how reuse was utilized in TEL and then at what could be improved in the current approaches. It then goes on via a set of supportive models that could serve as a basis for the construction of a SECO for mobile learning activities. Ending in the refined model, a basis for future development towards a SECO in the domain of Mobile Learning systems is put forward.

6.2 Future work

With the research questions answered, there remain plenty of opportunities for future research and efforts. Some of the areas where future work could be conducted are presented in the list below.

(1) Validation of the reference model.

(2) Prototype a SECO for Mobile Learning Activities

(3) Exploration of this prototype

  (a) Further explore the different kind of processes that exists inside this Software Ecosystem and possible improvement of these.

  (b) Further explore the general notion of SECOs based on this.

This thesis is of an exploratory nature and there are a number of relevant issues to be explored in this field that are outside the scope of this work. The following two areas of concern provide some of the most interesting challenges and deserve more attention. The first of these is the exploration of the different kind of processes that emerge inside of a Software Ecosystem. To study these in light of the reference model proposed in this thesis might enable a generalization of the model. If the processes are explored and formalized, some opportunities for change or optimization might emerge. The second area of desired exploration is the general area of Software Ecosystems. This might seem trivial at this point but the community has yet to agree on several key points. In respect to business ecosystems there are some questions regarding the boundaries between the neighboring areas. Looking at Software-as-a-Service (SaaS) ecosystems, some argue that these differ from SECOs and how will this affect the field? Further exploring this area is therefore a high priority.

Another area for planned future work is concerning prototyping a SECO for mobile learning activities. The scientific case for making prototypes to try ideas and in some cases even validate approaches is well established. To this end, in order to take the approaches suggested in this thesis forward, prototyping is needed. Prototypes will help identify the feasibility of the approach. They might also identify practical issues in future implementations and provide a test-bed for use testing.

Another point of interest for future efforts is the continued validation of the proposed reference model. Its validation is no simple task and is affected by all of the previous points to some degree, among a plethora of other things. To rigorously validate the model is crucial for its continued adaptation. Future work is naturally not limited to the things suggested in this section, but the ideas presented here are where the most urgent attention is needed.

Bibliography

Albin, S. T. (2003). *The art of software architecture: design methods and techniques*. John Wiley & Sons, Inc., New York, NY, USA.

Alexander, A., Blair, K. P., Goldman, S., Jimenez, O., Nakaue, M., Pea, R., and Russell, A. (2010). Go Math! How Research Anchors New Mobile Learning Environments. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 57–64, Kaohsiung, Taiwan.

Alexander, B. (2006). Web 2.0: A New Wave of Innovation for Teaching and Learning? *EDUCAUSE Review*, page 7.

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S. (1977). *A pattern language*. Oxford University Press.

Alvarez, C. G. (2011). *Implementing constructivist scaffolds based on mobile computers for supporting face- to-face collaborative learning scaffolds based on mobile computers for supporting face-to-face collaborative learning*. PhD thesis, Pontificia Universidad Catolica De Chile.

Bachour, K., Kaplan, F., and Dillenbourg, P. (2009). An Interactive Table for Supporting Participation Balance in Face-to-Face Collaboration. In *Ubicomp*, pages 1–10, Orlando, Florida, USA.

Ballard, B. (2007). *Designing the Mobile User Experience*. John Wiley & Sons, Chichester, England.

Barkhuus, L. and Dey, A. (2003). Is context-aware computing taking control away from the user? Three levels of interactivity examined. In *Ubicomp 2003: Ubiquitous Computing*, pages 149–156. Springer.

Bass, L., Clements, P., and Kazman, R. (2003). *Software architecture in practice*. Addison-Wesley Longman Publishing Company, Inc., Reading, Massachusetts.

Bates, A. W. and Bates, T. (2005). *Technology, e-learning and distance education*. Routledge, London, England.

Bell, M. (2008). *Service-oriented modeling: service analysis, design, and architecture*. John Wiley and Sons, Hoboken, New Jersey.

Berk, I. V. D., Jansen, S., and Luinenburg, L. (2010). Software Ecosystems : A Software Ecosystem Strategy Assessment Model. In *2nd International Workshop on Software Ecosystems*, Copenhagen.

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, 21(5):61–72.

Bosch, J. (2002). Maturity and evolution in software product lines: Approaches, artifacts and organization. In *Proceedings of the Second Conference Software Product Line Conference (SPLC2)*, pages 247–262, San Diego, California.

Bosch, J. (2009). From software product lines to software ecosystems. In *Proceedings of the 13th International Software Product Line Conference*, pages 111–119. Carnegie Mellon University.

Bosch, J. and 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.

Boudier, G., Gallo, F., Minot, R., and Thomas, I. (1989). An overview of PCTE and PCTE+. *ACM SIGPLAN Notices*, 24(2):248–257.

Bourque, P., Dupuis, R., Abran, A., Moore, J. W., and Tripp, L. (1999). The guide to the software engineering body of knowledge. *Software, IEEE*, 16(6):35–44.

Butler, M. (2011). Android: Changing the Mobile Landscape. *IEEE Pervasive Computing*, pages 4–7.

Byrne, P. (2011). *MUSE - Platform For Mobile Computer Supported Collaborative Learning.* PhD thesis, University of Dublin, Trinity College.

Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., and Vanderdonckt, J. (2003). A unifying reference framework for multi-target user interfaces. *Interacting with Computers*, 15(3):289–308.

Calvo, R. and Turani, A. (2010). E-Learning Frameworks = Design Patterns + Software Components. In Goodyear, P. and Retalis, S., editors, *Technology-Enhanched Learning: Design Patterns and Pattern Language.* Sense Publishers, Rotterdam, the Netherlands.

Carmichael, P., Rimpilainen, S., and Procter, R. (2006). Sakai: a collaborative virtual research environment for education. *Imprint*, 84:19.

Ceruzzi, P. (1986). An unforeseen revolution: computers and expectations, 1935–1985. In Corn, J. J., editor, *Imagining Tomorrow. History, Technology, and the American Future.* The MIT Press, Cambridge, MA.

Chang, V. and Uden, L. (2008). Governance for E-learning Ecosystem. In *2nd IEEE International Conference on Digital Ecosystems and Technologies, 2008. DEST 2008*, pages 340–345.

Chee, Y. S., Tan, E. M., and Liu, Q. (2010). Statecraft X: Enacting Citizenship Education Using a Mobile Learning Game Played on Apple iPhones. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 222–224, Kaohsiung, Taiwan.

Chen, T.-S., Chang, C.-S., Lin, J.-S., and Yu, H.-L. (2008). Context-Aware Writing in Ubiquitous Learning Environments. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 67–73, Beijing, China.

Clements, P. and Northrop, L. (2001). *Software Product Lines: Practices and Patterns*. Addison-Wesley, Reading, MA.

Cole, J. and Foster, H. (2007). *Using Moodle: teaching with the popular open source course management system*. O'Reilly Media, Inc.

Conole, G., Culver, J., Williams, P., Cross, S., Clark, P., and Brasher, A. (2008). Cloudworks: social networking for learning design. In *Hello! Where are you in the landscape of educational technology? Proceedings of the Ascilite 2008*, pages 187–196, Melbourne, Australia.

Conole, G., Dyke, M., Oliver, M., and Seale, J. (2004). Mapping pedagogy and tools for effective learning design. *Computers & Education*, 43(1-2):17–33.

Cross, J. (2010). They had People Called Professors...! Changing Worlds of Learning: Strengthening Informal Learning in Formal Institutions? In Ehlers, U.-D. and Schneckenberg, D., editors, *Changing Cultures in Higher Education*, chapter 4, pages 43–54. Springer Berlin Heidelberg.

Dagger, D., O'Connor, A., and Lawless, S. (2007). Service-oriented e-learning platforms: From monolithic systems to flexible services. *Internet Computing*, pages 28–35.

Derntl, M. and Motschnig-Pitrik, R. (2010). The Practicioners perspective on design patterns for technology-enhanced learning. In Goodyear, P. and Retalis, S., editors, *Technology-Enhanced Learning: Design Patterns and Pattern Language*, pages 215–231. Sense Publishers, Rotterdam, the Netherlands.

Di Bitonto, P., Roselli, T., Rossano, V., Monacis, L., and Sinatra, M. (2010). MoMAt: A Mobile Museum Adaptive Tour. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 156–160, Kaohsiung, Taiwan.

Dietze, S., Gugliotta, A., and Domingue, J. (2007). A semantic web service oriented framework for adaptive learning environments. In Franconi, E., Kifer, M., and May, W., editors, *ESWC*, volume 4519 of *Lecture Notes in Computer Science*, pages 701–715. Springer.

Dijkstra, E. W. (1972). The humble programmer. *Communications of the ACM*, 15(10):859–866.

Dillenbourg, P. and Jermann, P. (2010). Technology for Classroom Orchestration. In Khine, M. and Saleh, I., editors, *New Science of Learning: Cognition, Computers and Collaboration in Education*, pages 1–20. Springer New York.

Dillenbourg, P. and Jermann, P. (2011). Technology for classroom orchestration. In Khine, M. S. and Saleh, I. M., editors, *New Science of Learning: Cognition, Computers and Collaboration in Education*, pages 1–20. Springer.

Erl, T. (2005). *Service-oriented architecture: concepts, technology, and design.* Prentice Hall PTR Upper Saddle River, NJ, USA.

Frakes, W. (1994). Systematic software reuse: a paradigm shift. *Proceedings of 1994 3rd International Conference on Software Reuse*, pages 2–3.

Frakes, W. B. and Isoda, S. (1994). Success Factors of Systematic Reuse. *IEEE Software*, 11:14–19.

Freeman, R. E. (1984). *Strategic management: a stakeholder approach.* Harpercollins College Division, Boston, Massachusetts.

Fresen, J. (2007). A taxonomy of factors to promote quality web-supported learning. *International Journal on E-Learning*, 6(3):351.

Frohberg, D. (2006). Mobile Learning is Coming of Age: What we have and what we still miss. In *Proceedings of DeLFI*, pages 327–338, Darmstadt, Germany.

Frohberg, D., Göth, C., and Schwabe, G. (2009). Mobile Learning projects - a critical analysis of the state of the art. *Journal of Computer Assisted Learning*, 25(4):307–331.

Garlan, D., Allen, R., and Ockerbloom, J. (1995). Architectural mismatch: why reuse is so hard. *IEEE Software*, 12(6):17–26.

Garlan, D., Allen, R., and Ockerbloom, J. (2009). Architectural Mismatch: Why Reuse is Still So Hard. *IEEE software*, 26(4):66–69.

Garlan, D. and Shaw, M. (1993). An introduction to software architecture. In Ambriola, V. and Tortora, G., editors, *Advances in software engineering and knowledge engineering*, pages 1–40. JBW Printers and Binders, Singapore.

Garzotto, F. and Poggi, C. (2010). Design Patterns for Collaborative Learning Experiences In Online 3D Worlds. In Goodyear, P. and Retalis, S., editors, *Technology-Enhanched Learning: Design Patterns and Pattern Language*, pages 83 – 106. Sense Publishers, Rotterdam, the Netherlands.

Giemza, A., Bollen, L., Seydel, P., Overhagen, A., and Hoppe, H. U. (2010). LEMONADE: A Flexible Authoring Tool for Integrated Mobile Learning Scenarios. In *The sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE2010)*, pages 73–80, Kaohsiung, Taiwan.

González, M. A. C., Peñalvo, F. J. G., Guerrero, M. J. C., and Forment, M. A. (2009). Adapting LMS Architecture to the SOA: An Architectural Approach. In *Fourth International Conference on Internet and Web Applications and Services*, pages 322–327, Venice/Mestre, Italy.

Goodman, P. S. (2002). *Technology enhanced learning: Opportunities for change.* Lawrence Erlbaum Associates, Mahwah, New Jersey.

Goodyear, P. and Retalis, S. (2010). *Technology-Enhanced Learning: Design Patterns and Pattern Languages.* Sense Publishers, Rotterdam, the Netherlands.

Greenfield, J. and Short, K. (2003). Software factories: assembling applications with patterns, models, frameworks and tools. In *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 16–27. ACM.

Griss, M. (1993). Software reuse: From library to factory. *IBM Systems Journal*, 32(4):548–566.

Gueguen, G. (2009). Coopetition and business ecosystems in the information technology sector: the example of Intelligent Mobile Terminals. *International journal of entrepreneurship and small business*, 8(1):135–153.

Guetl, C. and Chang, V. (2008). Ecosystem-based Theoretical Models for Learning in Environments of the 21st Century. *International Journal of Emerging Technologies in Learning (iJET)*, 3(1):1–11.

Hallsteinsen, S., Hinchey, M., and Schmid, K. (2008). Dynamic Software Product Lines. *Computer*, 41(4):93–95.

Hanssen, G. (2011). A Longitudinal Case Study of an Emerging Software Ecosystem. *Accepted for publication in Journal of Systems and Software*, pages 1–24.

Hoppe, H., Ogata, H., and Soller, A. (2007). *The Role of Technology in CSCL: Studies in Technology Enhanced Collaborative Learning.* Springer Science+Business Media, Norwell.

Hsu, S.-H., Wu, P.-H., and Huang, Y.-M. (2008). Dog Detective - An Experience of Game-Based Ubiquitous Learning in Elementary School Science Experiment Class. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 231–233, Beijing, China.

Humphrey, W. S. (1995). Introducing the personal software process. *Annals of Software Engineering*, 1(1):311–325.

Hwang, G.-J., Wu, P.-H., Zhuang, Y.-Y., Kuo, W.-L., and Huang, Y.-M. (2010). An Investigation on Students' Cognitive Load and Learning Achievements for Participating in a Local Culture Mobile Learning Activity. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 27–33, Kaohsiung, Taiwan.

Iansiti, M. and Levien, R. (2004). *The keystone advantage: what the new dynamics of business ecosystems mean for strategy, innovation, and sustainability*. Harvard Business School Press, Boston, Mass.

IMS (2005). Information model, best practice and implementation guide, xml binding, schemas. `http://www.imsglobal.org/learningdesign/`. [Online; accessed 20-May-2011].

International Telecommunication Union (2010). Measuring the information society. Technical report, International Telecommunication Union, Geneva, Switzerland.

Jacobson, I. and Bylund, S. (2000). *The road to the unified software development process*. Cambridge University Press.

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

Jansen, S., Brinkkemper, S., Hunink, I., and Demir, C. (2008). Pragmatic and Opportunistic Reuse in Innovative Start-up Companies. *Software, IEEE*, 25(6):42–49.

Jansen, S., Finkelstein, A., and Brinkkemper, S. (2009). A sense of community: A research agenda for software ecosystems. In *31st International Conference on Software Engineering, New and Emerging Research Track*, pages 187–190.

Jimenez, P. F. and Lyons, L. B. (2010). Studying Different Methods of Providing Input to Collaborative Interactive Museum Exhibit Using Mobile Devices. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 225–227, Kaohsiung, Taiwan.

Johnson, R. (1997). Components, frameworks, patterns. In *Proceedings of the 1997 symposium on Software reusability*, pages 10–17, Boston, Mass.

Jones, C. (1993). Software return on investment preliminary analysis. Software Productivity Research. Technical report, Software Productivity Research, Inc.

Jones, C. (2000). *Software assessments, benchmarks, and best practices*. Addison-Wesley, Boston, Mass.

Kaiserfeld, T. (2000). A case study of the swedish public technology procurement project "the computer in the school" (COMPIS), 1981-1988. In *Public technology procurement and innovation*. Kluwer Academic Publishers, Boston, Mass.

Karampiperis, P. and Sampson, D. (2005). Designing learning services: from content-based to activity-based learning systems. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1110–1111, Chiba, Japan.

Kayama, T., Kaneko, K., Miyakoda, H., and Ishikawa, M. (2010). Effective materials for abstract words in foreign vocabulary learning. In *Sixth IEEE International Conference on Wireless, Mobile and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 207–209, Kaohsiung, Taiwan.

Koltun, P. and Hudson, A. (1991). A Reuse Maturity Model. In *4th Annual Workshop on Software Reuse*, Hemdon, Virginia.

Kool, M. L. (2009). A model for framing mobile learning. In Ally, M., editor, *Mobile learning: Transforming the delivery of education and training.* Athabasca University Press.

Krafzig, D., Banke, K., and Slama, D. (2005). *Enterprise SOA: service-oriented architecture best practices.* Prentice Hall Professional Technical Reference, Upper Saddle River, New Jersey.

Ku, O. Y., Huang, O. W., and Chan, T.-W. (2008). Teacher Monitoring System in One-to-One Self-Paced Learning Classroom. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 196–198, Beijing, China.

Kurti, A. (2009). *Exploring the multiple dimensions of context: Implications for the design and development of innovative technology-enhanced learning environments.* PhD thesis, Växjö University.

Kurti, A., Spikol, D., and Milrad, M. (2008). Bridging outdoors and indoors educational activities in schools with the support of mobile and positioning technologies. *International Journal of Mobile Learning and Organisation*, 2(2):166 – 186.

Lanergan, R. G. and Poynton, B. A. (1979). Reusable code: The application development technique of the future. In *Proceedings of the IBM SHARE/GUIDE Software Symposium.*

Lewis, S., Pea, R., and Rosen, J. (2010). Collaboration with Mobile Media: Shifting from 'Participation' to 'Co-creation'. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 112–116, Kaohsiung, Taiwan.

Li, M., Ogata, H., Hou, B., Hashimoto, S., Uosaki, N., Liu, Y., and Yano, Y. (2010). Development of Adaptive Vocabulary Learning via Mobile Phone E-mail. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 34–41, Kaohsiung, Taiwan.

Lim, W. C. (1998). *Managing Software Reuse.* Prentice Hall.

Lin, C.-P., Wong, L.-H., Shao, Y., Niramitranon, J., and Tong, C.-J. (2010a). 1:1 Learning Technology to Support Collaborative Concept Mapping: A Case Study of Social Studies Lesson in Elementary School. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 3–10, Kaohsiung, Taiwan.

Lin, Y.-T., Huang, Y.-M., and Tan, Q. (2010b). Location-Based and Knowledge-Oriented Microblogging for Mobile Learning–Framework, Architecture, and System. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 146–150, Kaohsiung, Taiwan.

Liu, H., Huang, R., Salomaa, J., and Ma, D. (2008). An Activity-Oriented Design Framework for Mobile Learning Experience. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 185–187, Beijing, China.

Liu, M.-C., Wen, D., and Huang, Y.-M. (2010). Learning Animal Concepts with Semantic Hierarchy-Based Location-Aware Image Browsing and Ecology Task Generator. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 42–49, Kaohsiung, Taiwan.

Liu, T. C., Wang, H. Y., Liang, J. K., Chan, T. W., Ko, H. W., and Yang, J. C. (2003). Wireless and mobile technologies to enhance teaching and learning. *Journal of Computer Assisted Learning*, 19(3):371–382.

Lohr, M. and Wallinger, E. (2008). Collage - The Carnuntum Scenario. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 161–163, Beijing, China.

McGregor, J. D. (2004). Software product lines. *Journal of Object Technology*, 3(3):65–74.

McIlroy, M. D., Buxton, J. M., Naur, P., and Randell, B. (1968). Mass-Produced Software Components. In *Software Engineering Concepts and Techniques*, pages 88–98. NATO Science Committee.

McQueen, D. (2009). The momentum behind LTE adoption. *IEEE Communications Magazine*, pages 44–45.

Messerschmitt, D. G. and Szyperski, C. (2003). *Software Ecosystem: Understanding an Indispensable Technology and Industry.* The MIT Press, Cambridge, Mass.

Metcalf, D., Milrad, M., Cheek, D., Raasch, S., and Hamilton, A. (2008). My Sports Pulse: Increasing Student Interest in STEM Disciplines through Sports Themes, Games and Mobile Technologies. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 23–30, Beijing, China.

Moody, D. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data & Knowledge Engineering*, 55(3):243–276.

Moore, J. F. (1996). *The death of competition: leadership and strategy in the age of business ecosystems.* HarperBusiness New York.

Neighbors, J. M. (1984). The Draco approach to constructing software from reusable components. *Software Engineering, IEEE Transactions on*, SE-10(5):564–574.

Neven, F. and Duval, E. (2002). Reusable learning objects: a survey of LOM-based repositories. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 291–294. ACM.

Niramitranon, J., Sharples, M., Greenhalgh, C., and Lin, C.-P. (2010). Orchestrating Learning in a One-to-One Technology Classroom. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 96–103, Kaohsiung, Taiwan.

Ogata, H. (2008). Computer Supported Ubiquitous Learning: Augmenting Learning Experiences in the Real World. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 3–10, Beijing, China.

Ogata, H. and Yano, Y. (2004). Context-Aware Support for Computer-Supported Ubiquitous Learning. In *IEEE International Workshop on Wireless and Mobile Technologies in Education*, pages 27–34, Los Alamitos, CA, USA.

OMG (2008). Software process engineering metamodel 2.0. Technical report, Object Management Group.

Osterweil, L. (1987). Software processes are software too. In *Proceedings of the 9th international conference on Software Engineering*, pages 2–13. IEEE Computer Society Press.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058.

Pettersson, O. and Gil, D. (2010). On the Issue of Reusability and Adaptability in M-learning Systems. In *Proceedings of the Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education*, pages 161–165, Kaohsiung, Taiwan.

Pettersson, O., Svensson, M., Gil, D., Andersson, J., and Milrad, M. (2010). On the role of software process modeling in software ecosystem design. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, ECSA '10, pages 103–110, New York, NY, USA. ACM.

Pfeiffer, D. and Gehlert, A. (2005). A framework for comparing conceptual models. In *Proceedings of the Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2005)*, pages 108–122.

Pinsonneault, A. and Kraemer, K. L. (1993). Survey research methodology in management information systems: an assessment. *Journal of Management Information Systems*, 10(2):75–105.

Pohl, K., Böckle, G., and Van Der Linden, F. (2005). *Software product line engineering: foundations, principles, and techniques*. Springer-Verlag New York Inc.

Prieto-Diaz, R. (1991). Implementing faceted classification for software reuse. *Communications of the ACM*, 34(5):88–97.

Rodriguez-Martinez, L. C., Mora, M., and Alvarez, F. J. (2010). A Descriptive/Comparative Study of the Evolution of Process Models of Software Development Life Cycles (PM-SDLCs). In *Mexican International Conference on Computer Science*, pages 298–303, Mexico City, Mexico.

Royce, W. W. (1970). Managing the development of large software systems. In *proceedings of IEEE WESCON*.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., and Others (1991). *Object-oriented modeling and design*, volume 38. Prentice hall Englewood Cliffs (NJ).

Sampson, D. and Zervas, P. (2008). Enabling Interoperable Mobile Learning: Evaluation Results from the Use of SMILE PDA Learning Design Player. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 188–190, Beijing, China.

Scacchi, W., Feller, J., Fitzgerald, B., Hissam, S., and Lakhani, K. (2006). Understanding free/open source software development processes. *Software Process: Improvement and Practice*, 11(2):95–105.

SCORM (2009). Sharable content object reference model (scorm) 2004 4th edition. Technical report, Advanced Distributed Learning.

Sharples, M. (2002). Disruptive devices: mobile technology for conversational learning. *International Journal of Continuing Engineering Education and Life Long Learning*, 12(5):504–520.

Sharples, M., Taylor, J., and Vavoula, G. (2007). A Theory of Learning for the Mobile Age. In Andrews, R. and Haythornthwaite, C., editors, *The SAGE handbook of e-learning research*. Sage Publications Ltd.

Shih, J.-L. (2008). The Design of an Ubiquitous Learning System with Research Problem-Based Learning (RPBL) Model for Qualitative Studies. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 173–175, Beijing, China.

Sindre, G., Conradi, R., and Karlsson, E. A. (1995). The REBOOT approach to software reuse. *Journal of Systems and Software*, 30(3):201–212.

Sjoberg, D., Dyba, T., and Jorgensen, M. (2007). The future of empirical methods in software engineering research. In *Future of Software Engineering, 2007. (FOSE 07)*, pages 358 –378.

Spikol, D., Kurti, A., and Milrad, M. (2008). Collaboration in Context as a Framework for Designing Innovative Mobile Learning Activities. In Ryu, H. and Parsons, D., editors, *Innovative Mobile Learning: Techniques and Technologies.*, pages 172 – 196. IDEA GROUP INC.

Suppes, P. (1966). The uses of computers in education. *Scientific American*, 215:207–220.

Szyperski, C., Gruntz, D., and Murer, S. (2002). *Component software: beyond object-oriented programming.* Addison-Wesley Professional, London, UK.

Tao, S.-Y., Ho, K.-W., Chung, C.-W., Liu, B.-J., and Liu, C.-C. (2008). Designing a Groupware with Handheld Devices for Learning Mathematics. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 216–218, Beijing, China.

Toft, P., Coleman, D., and Ohta, J. (2000). A Cooperative Model for Cross-Divisional Product Development for a Software Product Line. In *Software Product Lines: Experience and Practice.* Kluwer Academic Academic Publishers.

Traxler, J. (2008). Learning in a Mobile Age. *International Journal of Mobile and Blended Learning*, 1(March):1–12.

Uden, L., Wangsa, I., and Damiani, E. (2007). The future of E-learning: E-learning ecosystem. In *2007 Inaugural IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2007)*, pages 113–117.

Van Der Linden, F., Schmid, K., and Rommes, E. (2007). *Software product lines in action: the best industrial practice in product line engineering.* Springer-Verlag New York Inc.

Vavoula, G. and Sharples, M. (2009). Meeting the Challenges in Evaluating Mobile Learning: A 3-Level Evaluation Framework. *International Journal of Mobile and Blended Learning*, 1(2):54–75.

Vogel, B., Spikol, D., Kurti, A., and Milrad, M. (2010). Integrating Mobile, Web and Sensory Technologies to Support Inquiry-Based Science Learning. In *IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education*, pages 65–72.

Vossen, G. and Westerkamp, P. (2003). E-Learning as a Web Service. In *International Database Engineering and Applications Symposium*, Los Alamitos, CA, USA.

Weiss, D. M. and Lai, C. T. R. (1999). *Software product-line engineering: a family-based software development process.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

Wilkes, M. V. and Renwick, W. (1949). The EDSAC - an Electronic Calculating Machine. *Journal of Scientific Instruments*, 26(12):385.

Wingkvist, A. (2009). *Understanding Scalability and Sustainability in Mobile Learning: A Systems Development Framework.* PhD thesis, Vaxjo University.

Wohlin, C., Runeson, P., and Höst, M. (2000). *Experimentation in software engineering: an introduction.* Springer Netherlands.

Woodgate, D., Fraser, D., Paxton, M., Crellin, D., Woolard, A., and Dillon, T. (2008). Bringing School Science to Life: Personalization, Contextualization and Reflection of Self-Collected Data. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 100–104, Beijing, China.

Wu, T.-T., Yang, T.-C., Hwang, G.-J., and Chu, H.-C. (2008). Conducting Situated Learning in a Context-Aware Ubiquitous Learning Environment. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 82–86, Beijing, China.

Yahya, S., Ahmad, E. A., Jalil, K. A., and Mara, U. T. (2010). The definition and characteristics of ubiquitous learning : A discussion. *Journal of Education*, 6(1).

Yamin, A. C., Barbosa, J. L. V., Augustin, I., and Geyer, C. F. R. (2005). ISAM: A software architecture for pervasive computing. *CLEI Electronic Journal*, 8(1).

Yau, J. Y.-K. and Joy, M. (2010). A Context-Aware Personalized M-learning Application Based on M-learning Preferences. In *Sixth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education (WMUTE 2010)*, pages 11–18, Kaohsiung, Taiwan.

Yin, J. and Yang, X. (2008). RFID-Based Ubiquitous Learning Environment for School Students. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 176–178, Beijing, China.

Yin, R. (2003). *Case study research: Design and methods*. Sage Publications, Inc, Thousand Oaks, third edition edition.

Zurita, G., Baloian, N., and Baytelman, F. (2008). Supporting Rich Interaction in the Classroom with Mobile Devices. In *Fifth IEEE International Conference on Wireless, Mobile, and Ubiquitous Technology in Education (WMUTE 2008)*, pages 115–122, Beijing, China.