UiO : Department of Informatics
University of Oslo

Enabling Participatory Design in low resource contexts

A study of distributed software development of DHIS2

Sindre Nicolaysen Berntsen

Master's Thesis Spring 2015

Abstract

The purpose of this thesis was to investigate the development and implementation of health information systems (HIS) and mobile health (M-Health) for low resource contexts in a globally distributed setting (GSD) where participatory design (PD) is central. This thesis shed light on limitations and solutions in existing theories on distributed participatory design (DPD) and it seeks to answer the research question "How can participatory design be enabled in distributed software development of health information systems for low resource contexts?".

Theoretically, the thesis builds on perspectives from literature on HIS and M-Health, strategies for DSD, strategies for PD and theories on DPD.

Qualitative research methods have been employed, in a combination of a case study and an action case. The case study was of a global organization that practices distributed software development (DSD) of a HIS. The action case was of a case where a mobile client for reporting in a HIS was implemented in a low resource context.

The main findings of this thesis indicate that although many efforts have been made, the degree of participation is low in the development of the system investigated. The main reason for the low participation has been the enormous scaling of deployments and spread of the system to a lot of contexts. The findings also indicate that existing literature and theories on DPD have limitations. Therefore, based on aspects from existing theories on DPD, a new approach is suggested to face an increasing number of contexts and use cases.

Acknowledgments

First of all, I would like to thank my supervisors Lars Kristian Roland and Knut Staring for all the guidance, input, sharing of ideas and feedback during the last year. I would like to thank the MUJHU team in Uganda for their support, help, and warm welcome to Kampala, Uganda. A special thanks goes to Ellen Fruijtier for proofreading and giving valuable input in the finalizing stage of this thesis. I also want to thank all my fellow students in the 9th floor for awesome card games and way too long coffee breaks. Lastly, I want to give a special thanks to Lene, for her support, encouragement and patience throughout the entire process.

Contents

List of Figures

List of Tables

Abbreviations

DHIS	District Health Information System
DPD	Distributed Participatory Design
DSD	Distributed Software Development
EMR	Electronic Medical Records
FOSS	Free and Open Source Software
GSD	Global Software Development
HIS	Health Information Systems
HMIS	Health Management Information Systems
IHIA	Integrated Health Information Architecture
IS	Information Systems
J2ME	Java 2, Micro Edition
MUJHU	Makere University Johns Hopkins University
NGO	Non Governmental Organizations
PD	Participatory Design
WEMR	Web Electronic Medical Records

Chapter 1

Introduction

This thesis examines the development and implementation of health information systems (HIS) and mobile health (M-Health) for low resource contexts in a globally distributed setting (GSD) where participatory design (PD) is central. More precisely the thesis looks at limitations and solutions in existing theories on distributed participatory design (DPD).

The empirical findings are based on a combination of a case study and an action case. The case study was of a global organization that practices distributed software development (DSD) of a HIS. The action case was of a case where a mobile device for reporting in a HIS was implemented in a low resource context.

This thesis tries to look at limitations and solutions in the literature and theory reviewed in light of the empirical findings.

1.1 Motivation

Good health information systems are crucial for improving health in developing countries and the impact HIS has on improvement of decision-making in health care is huge (AbouZahr and Boerma 2005). Mobile devices are important for the development of health in low resource countries and the use of mobile devices for web-based data entry can improve the timeliness and completeness of disease surveillance. (Rashid and Elder 2009; Kahn et al. 2010).

The concept of the design-reality gap describes the deviation between local requirements and global design (Heeks 2006). When a HIS spans several contexts globally, it is necessary to strike a balance between the need for information infrastructure that adapts to various local contexts, and simultaneously standardizing toward a universal solution across contexts (Rolland and Monteiro 2002). This balance affects the degree of user participation in the design. Globalization makes software development more and more distributed and there is a need of mutual understanding between users and developers, thus PD approaches become important (Gumm 2006).

Titlestad et al. (2009) present an approach to find a balance between local requirements and global standards by looking at the development as an evolutionary process towards a universal solution. The problem with their approach is that literature reveals that it does not seem to scale very well (Shidende et al. 2014). Based on the empirical work in this thesis, Titlestad et al.'s (2009) approach seems to have limitations.

Thus the motivation for this thesis is to examine Titlestad et al.'s (2009) approach to distributed participatory design in order to investigate if the balance between local requirements and global standards can be found by other approaches.

1.2 Context of the thesis

Within the HISP network, the software development is very distributed, with developers, end-users and managers spread out across the world. With the increasing number of actors in HISP, the network becomes more and more distributed. A key process for the HISP network is to develop software through participatory design (Braa and Sahay 2013, p. 236).

1.2.1 HISP

The Health Information Systems Programme (HISP) is a global Network established in South Africa in 1994. It is managed by the Department of Informatics at the University of Oslo (UiO). The network consists of people, entities and organizations that over 20 years have designed, implemented and sustained decentralized health information systems with an approach where they participate to support the local management of health care delivery and information flows in health facilities, districts and provinces. That has resulted in DHIS2 (*HISP* 2015; *HISP at UiO* 2015; Braa et al. 2004).

1.2.2 The initial plan

'Our plan is to follow the track mentioned to work on 'feedback' and data use for the health worker. Which reports, graphs, charts, tables etc. can we include in the system that makes the tracker more useful for the health workers. '

— Thesis supervisor

To give health workers feedback is important for their motivation and Franco et al. (2002) suggest that feedback is one of the determinants that affect a health worker's motivation. In addition Asiimwe et al. (2011) suggest that with good feedback, health care can be improved.

When the project first started, the plan was to go to Uganda, talk with the health workers, and develop something useful for them in their low resource context. However, it turned out that this would be easier said than done. The health workers didn't use the DHIS2 Patient Tracker themselves. Instead, IT interns took care of the data reporting. This made it impossible to investigate if there was a need for making the DHIS2 Patient Tracker more useful for health workers at all.

1.2.3 The new plan

The field trip then became more focused on solving situated problems. The field study was spent on assisting the project in Uganda with their implementation of mobile devices for patient tracking in DHIS2. The focus was mainly on making the J2ME client work, but the opportunities for the implementation of Android clients were investigated as well.

The difficulties met in Uganda raised the question of what had happened. What was the underlying issue that made the original plan impossible to conduct? Upon returning from the field trip, the investigation started. The main reason was a partly unsuccessful implementation of the DHIS2 Patient Tracker on J2ME clients.

The project then changed to examine how the development and implementation of the J2ME client and DHIS2 in general are done in addition to assisting Uganda with the implementation of the J2ME Patient Tracker.

The focus of this thesis is on development and implementation of HIS for low resource contexts in a distributed setting where PD is central. This is done by looking at a subset of the actors in HISP. The actors are listed below and illustrated in figure 1.1.

- Users in Uganda as a local context, where the use of the low-end mobile DHIS2 Patient Tracker and implementation of mobile devices have been looked at.
- Core developers in Oslo, as a part of the global team where their work processes have been studied. Especially how the communication with the local context work.
- Developers in Philippines, as outsourced developers who developed the low-end mobile client for DHIS2 Patient Tracker. Here the communication with the global and local context has been investigated.
- Implementers and a project coordinator in Oslo, as a part of the global team where communication and coordination have been looked at.

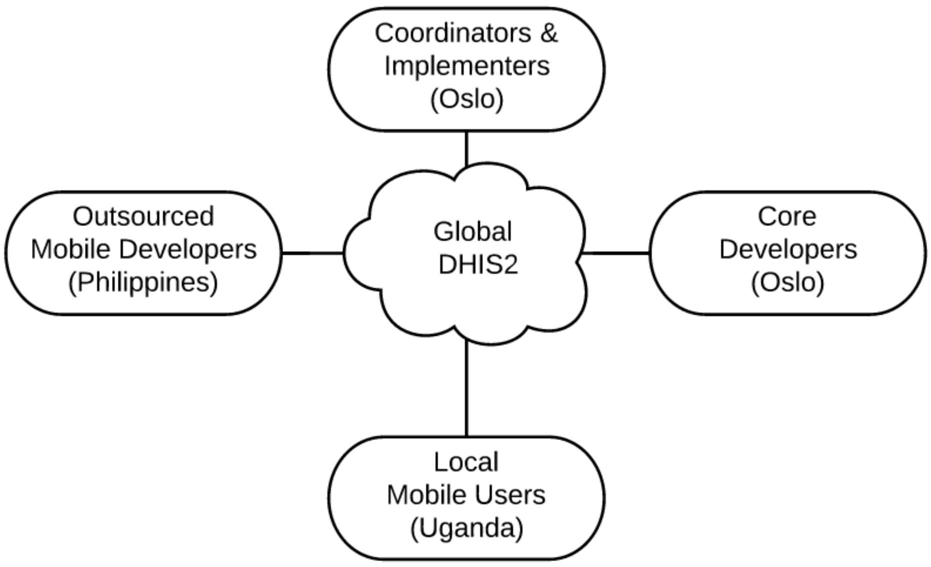

Coordinators & Implementers (Oslo)
Outsourced Mobile Developers (Philippines)
Global DHIS2
Core Developers (Oslo)
Local Mobile Users (Uganda)

Figure 1.1: A subset of actors within the HISP network.

1.3 Research objectives

The objective of this thesis is to explore the degree of participatory design in distributed software development of health information systems for low resource contexts and exploring limitations and solutions in existing theories of distributed participatory design. To concretize this research objective, a research question is shaped.

Research Question: *How can participatory design be enabled in distributed software development of health information systems for low resource contexts?*

Four tasks will be central to answer this research question:

- Being a mediator, assisting the MUJHU project in Uganda in their implementation of the J2ME Patient Tracker for DHIS2.
- Interviewing different actors in HISP Oslo.
- Reviewing others' literature, theories and studies concerning participatory design and distributed software development.
- Analyzing and comparing the literature reviewed with the findings from the field study and the interviews.

1.4 Structure of thesis

Chapter 2 - Background
Explains concepts relevant for understanding of the domain of this thesis. These concepts are HIS and M-health and are exemplified by the District Health Information System 2 (DHIS2) and mobile devices supporting this system, also referred to as DHIS2 Mobile.

Chapter 3 - Literature review
Reviews relevant literature and research on the problem domain. The concepts presented are distributed software development (DSD), participatory design (PD) and distributed participatory design (DPD).

Chapter 4 - Theoretical framework
Presents existing theoretical contributions where the concepts described in previous chapters are central. These concepts and theories provide a theoretical framework for discussing findings of this thesis with others' experiences.

Chapter 5 - Methods
Describes the methodologies used to collect and analyze data in this project: action case and case study. Followed by the approach of gathering data and the techniques used.

Chapter 6 - Uganda
Presents empirical findings in the case followed throughout this thesis, namely the MUJHU project in Uganda.

Chapter 7 - Mediator
Presents empirical findings of the work done as a mediator in the development and implementation of the mobile client in Uganda.

Chapter 8 - HISP Oslo
Presents empirical findings from interviews of actors in HISP Oslo and from a comparison of best practices of DSD and DPD with the development and implementation of DHIS2.

Chapter 9 - Discussion
A discussion of the results and findings from the research in light of the literature and theories reviewed.

Chapter 10 - Conclusion
A concluding chapter with an answer to the research question, a description of findings and contributions, and suggestions for further work.

Chapter 2

Background

The domain of this thesis is the development and implementation of health information systems (HIS) and mobile health (M-health) for low resource contexts. To better understand the domain, it is important to understand what HIS and M-health is. These two concepts are exemplified by the District Health Information System 2 (DHIS2) and mobile devices supporting this system, also referred to as DHIS2 Mobile.

2.1 Health Information Systems

This section defines what HIS is and explains why it is important. It also looks into implementation of HIS, as it is a difficult and complex task and tries to give awareness of challenges that could occur when implementing in low resource contexts.

2.1.1 Definition of HIS

The concept of HIS is no longer straightforward; it can mean different things to different interests. For example systems dealing with aggregated data often are referred to as health management information systems (HMIS) and the systems that handles patient level data are often called electronic medical records (EMR). Integrated health information architecture (IHIA) is a concept trying to cover all the different systems. IHIA is explained as a framework for use of data and information at all levels of health services to support decision-making and it includes components of the HIS such as EMR and HMIS (Braa and Sahay 2013).

A health information system or a health management information system is a system that collect, process, report and use health information for better decision-making in the health sector. A HIS focus is on quantitative data and tries to describe the health status of a population and is used to make better decisions in health (AbouZahr and Boerma 2005). Areas a HIS should include:

- Health factors and the environments within which the health system operates.

- Inputs to the HIS such as policy, organization, health infrastructure, facilities, equipment, costs, human resources and financial resources.
- Outputs from the HIS such as availability, quality and use of health information and services.
- Health outcomes such as health status, disease outbreaks and inequities in coverage and use of services.

HIS aim to contribute to high-quality and efficient patient care and points out important changes in the development of HIS to achieving that. For example two important changes are shift from institution-centered HIS towards regional and global HIS and shift from paper to computer processing and storage. Without a good HIS, high-level quality of care is almost impossible. Without access to relevant data, it will be hard to make decisions for example about diagnosis, with fatal consequences for patients (Haux 2006).

It is important to be aware of the fact that life expectancy worldwide is constantly growing, and this growth has a big impact on health care and the development of information systems. A consequence of the growth are certain needs for HIS in the future, first there is a need for institutional and national HIS-strategies, second there is a need to explore new HIS architectural styles, third there is a need for education in health informatics, including knowledge and skills on HIS (Haux 2006).

2.1.2 Importance of HIS

The impact HIS has on improvement of decision-making in health care is huge. Decision-making is dependent on timely available data and good HIS is supposed to provide that kind of data as its role is to generate, analyze and communicate data. There are several reasons why HIS is so important (AbouZahr and Boerma 2005). Information gathered in HIS is needed on several levels:

- The level of individuals and communities: needed for clinical management and to evaluate if the services meet the needs of communities.
- The level of districts: needed for managers to make decisions on the efficiency of health facilities or the health system as a whole.
- Higher levels: needed for strategic policy-making and resource allocation.

There are potential savings and costs in health care with the introduction of EMR systems. A comparison of health care with other industries proved that the use of EMR systems will lead to great benefits such as economic savings, reduced medical errors and improved health. Health benefits an EMR system could provide are (Hillestad et al. 2005):

- It can provide health staff with reminders to offer service during visits to patients.

- It can improve patients' compliance with preventive care recommendations by giving the patients reminders.
- It can monitor patients and identify risks and give them services based on their risk levels.
- It can identify patient's need of services through predictive algorithms.
- It can lead to better decision-making by recording clinical results.

Other studies have looked at the importance of a good HIS in health care and found possible benefits: With the use of HIS the medication errors made were reduced by more than half (Bates et al. 1998).

By testing a system of HIV-positive patients, it was found that HIS has the potential of improving a patient's quality of life. Even though the focus was on personal HIS for patients to have at home, the findings are relevant for HIS meant for health personnel as well, because the HIS used, provided patients with information and decision support that also are needed for health workers (Gustafson et al. 1999).

Health care does benefit from health information technology in quality and efficiency. It improves quality by increasing compliance to guideline- and protocol based care, by enhancing clinical monitoring and disease surveillance, and decreasing medical errors. It improves efficiency by decreasing utilization of care (Chaudhry et al. 2006).

Making good decisions on strategic planning, quality improvement and control of diseases in health care require timely, accurate and relevant health information. To collect, analyze and communicate this information effectively a HIS is required (Stansfield et al. 2006).

2.1.3 Problems of HIS

If the benefits and importance of HIS are so great, why don't all countries and especially the low-resource countries implement HIS in their health care? To succeed with a HIS is not straightforward. Some of the problems are that few developing countries have resources to implement HIS themselves and establishing sustainable systems is an expensive, long-term task. The health sector lacks staff with statistical skills to generate and analyze data, and health care workers are reluctant to focus more on data recording than patient care. The decision-making is made at the district levels, but they don't have the capacities to make good decisions. Decisions are taken without reliable information, which leads to inefficient use of resources (AbouZahr and Boerma 2005).

2.1.4 Implementation of information systems

When implementing HIS in a new context, it is important to be aware of challenges and difficulties that could occur. Therefore one should look at other implementation projects and learn from their experiences. In the implementation of information systems in developing countries, one needs to consider the organizational context to understand the causes of problems when developing information systems (Gladwin et al. 2003).

For a long time, the usual data flow in most countries was reports going upwards in the hierarchy. But in the mid 90s the focus changed to look at information use at the level it was collected. The health sector started a reform changing from a centralized reporting HIS to a HIS supporting district and facility management and constraints regarding organizational context was discovered. Such constraints could for example be organizational changes that were supposed to be finished before the implementation of the HIS were incomplete and donors having a lot of influence on what kind of data was collected, and they were often interested in reports from national levels, not district levels (ibid).

The organizational context is the root to many challenges in implementing information systems (IS) in health care. Implementation of IS are a changing process; both the technology and organization changes each other and good management and user-involvement is vital for a successful implementation. For example technology can give the users new possibilities, but if the users aren't involved in the design process, the user interface may become useless, showing that technical problems can have organizational roots (Berg 2001).

A sign of a successful implementation is when users of the system, both managers and health workers, appreciate the system. Other signs could be that the system is widely used or has resulted in saved resources. Common for them all is that successful implementation requires careful attention to what success parameters are used, and that all involved interests agree on these parameters. One remark though, is that development of IS in complex organizations have a certain degree of unpredictability; the technology is complex, and there are many interests having something to say about the IS (ibid).

When describing failed HIS in developing countries two topics are central. The first is sustainability, described as the challenge to make IS work over time in a local setting. The second is scalability, described as the challenge of how to make one solution spread to other sites and adapt there (Braa et al. 2004).

Three concerns about IS implementation one should be aware of (Berg 2001):

- The fact that IS implementations will affect the structure and processes of the organization. Changes in workflows, tasks and relationships between staff and managers should be expected. For example EMR changes health staff's recording practices.

- Implementation of IS can't be left to the IT department alone. End-users and managers need to be included from the beginning of the project to adjust the implementation to fit the workflow.

- The fact that not every aspect of the implementation process can be planned has to be faced.

Scaling of IS are a central task. Several HIS-pilots have died since they couldn't scale to a useful level; a district manager needs reports from all clinics in the area, not just a handful that has been piloted. Scaling is defined as enlarging and spreading the heterogeneous networks around technology. They also point to the complexity of health care and IS with all interests making it hard to plan the implementation, and states that you should rather adapt from unexpected situations instead of avoiding them (Sahay and Walsham 2006).

When scaling up HIS, there are some things one should consider (ibid):

- To cover the whole state is important, unless a HIS would most likely become useless for the health department. One remark though, when Sahay and Walsham (2006) wrote this, they focused on HMIS. With EMR-HIS, implementations in few facilities can be useful too.
- Political issues will occur, as the stakeholders most likely would have different interests.
- The health sector is in continuous change, making the environment unstable, such changes could for example come from Ministries of Health.
- As human resources capacity also need to be scaled, training of users are very helpful as it makes it easier for users to change their existing workflow to use new technologies.
- One should inform the users well, so they are aware of the needs for scaling and that scaling not only is about software architecture, but also about escalation of complexity.

Berg's (2001) and Sahay and Walsham's (2006) concerns are important to keep in mind if one want to succeed with the implementation of IS. These concerns also highlight the importance of involving users from the beginning of the implementation process. Since this thesis is about user involvement in development and implementation of IS, it is key to be aware of these concerns.

2.2 DHIS2

Since the District Health Information System 2 (DHIS2) is the HIS looked at in this thesis. It is important to understand what it is.

DHIS2 is free and open source software developed by HISP. It is a tool for collection, validation, analysis, and presentation of aggregate and transactional data, tailored (but not limited) to integrated health information management activities (*HISP at UiO* 2015). DHIS2 is currently used in over 47 countries worldwide (*DHIS2 in Action* 2015). It is a free open-source web-based tool for helping governments in developing countries to manage their health (*DHIS2 in Action* 2015).

DHIS2 is described as a data warehouse designed to address local and national needs. It is a generic tool with a flexible user interface that allows users to specify their own content without the need of programing. DHIS2 makes it possible to manage multiple datasets from different sources in one application, it also gives the opportunity to analyze the datasets from these sources, and present them to the user. It can also present data even if they are from different sources (Braa and Sahay 2012). A key advantage of DHIS2 is that it is flexible and can quickly be changed and adapted to typical health information systems needs based on aggregated data (Braa et al. 2004).

The two components of DHIS2 most relevant for this thesis are Tracker and Mobile.

2.2.1 DHIS2 Tracker

The DHIS2 Tracker lets you store information about individuals and track them over time. It also has the ability to store information about anonymous cases. The Tracker extension has the same design as the overall DHIS2: a generic data model with flexible metadata configuration through allowing customization to meet a wide range of use cases. Another feature of the Tracker is to enable sharing of critical clinical health data across health facilities (*DHIS2 Tracker* 2015).

A use case where the Tracker applies is tracking of women through pregnancy. With DHIS2 Tracker you can register the women every time they visit a health facility. ANC visits, Deliveries and PNC visits are examples of visits you can register. You will for each woman have a dashboard where the progress can be seen, new information can be registered, and new visits can be scheduled.

2.2.2 DHIS2 Mobile

Central in this project is the DHIS2 Mobile component. The mobile project started in February 2009 as an SMS based reporting system in India, and the study showed that SMS codes were difficult for the users. In May 2009 a java-application for DHIS reporting simulating the paper forms were tested out in India. It was a success and the district decided to introduce the mobile for reporting in the spring of 2010 (Sanner et al. 2012). Now there are pilots going on in several places, where they test out the java-application and the browser based mobile client. For example in Malawi they are piloting both clients for health workers to report information (Manda and Sanner 2014). Also an Android SDK is on the way, and expected to be released with the DHIS2 version 2.22 in January 2016.

The focus of DHIS Mobile is to expand the reach of HIS by using mobile technology. The mobile clients can be used together with a web-based HIS to reach all levels of a health service, including community health workers. For example the Java ME based application for low-end Java phones can communicate with an online DHIS2 instance, but it also works offline, enabling reporting when there is poor connectivity (*DHIS2 Mobile* 2015).

2.3 Mobile Health

Central in this thesis is the implementation of a mobile device for reporting information in a HIS. This section will give a brief introduction to what Mobile Health (M-health) is and why it is needed.

The usage of mobile devices worldwide and in developing countries in particular is constantly growing, and new areas of use are developed. Donner (2008) has reviewed several studies and found that mobiles can contribute a lot in developing countries. Because of a mobile phone's simplicity, portability and affordability it can offer connectivity to people in rural areas where PCs and fixed-line Internet doesn't exist. An area of use that is becoming more popular is mobile health.

2.3.1 Definition of M-health

M-health is defined as usage of mobile phones to improve quality of care efficiency of service delivery within health care systems (Lemaire 2011). The introduction of M-health has led to improved capacity, cost efficiency and access to health care adding up to improve quality of health care in developing countries.

2.3.2 Why M-health?

The recent and ongoing development of mobile technology and networks has made M-health possible. When talking about M-health development, four factors are considered important: increased number of mobile users, increased expansion of mobile networks, decreased mobile costs and innovation in mobile technology (Lemaire 2011).

Especially in developing countries this development is very important since a lot of them skipped landlines and went directly into mobile technology making it the dominant way of communication (Rashid and Elder 2009). The use of mobile devices can provide advantages in scalability, coverage, timeliness, and transparency, since those devices function in remote locations, and are easy to bring and use at any time (Freifeld et al. 2010). M-health can enhance traditional hierarchical health systems where the data flow is slow, by gathering information quickly and improve coverage and accessibility.

Other studies suggest reasons why M-health is beneficial. Kaplan (2006) states mobile networks can be built faster and mobiles are easier to use than computers. Kahn et al. (2010) suggest the use of mobile devices for web-based data entry can improve the timeliness and completeness of disease surveillance, by allowing the health workers to have immediate access to the databases. Curioso and Mechael (2010) suggest that use of mobile phones can strengthen health systems in developing countries since mobile phones make collecting and reporting of health data possible by the use of applications. Mobile phones also make it easier for rural communities to communicate with more central areas in real-time. The cost of implementing M-health compared to other e-health systems is also lower as it requires less infrastructure than fixed lines.

Mobile devices are important for the development of health for several reasons. For example they are portable and works by using the radio specter, so there is no need for physical infrastructure such as lines, making them usable in rural areas. Mobile devices also allow data transfer, making it easier to send information, instead of physically deliver paper reports (Rashid and Elder 2009).

To summarize, the flexibility, usability, robustness and financial cost can be seen as dimensions for characterizing M-health (Sanner et al. 2012):

Flexibility ability to adapt to change as organizational challenges and needs lead to improvements and innovations.

Usability ability to be effective and satisfactory when used to achieve specified goals in specified contexts.

Robustness ability to scale up even though there is variations in quality and coverage of wireless communication networks and limited access to stable power supply.

Cost covers the different kinds of financial costs, such as up-front investments in technology as well as well as running costs.

2.4 Mobile clients

Now that the concept of M-health and the importance of it are described, the mobile clients looked at in this thesis are presented. The presented devices are a part of DHIS2 Mobile.

The mobile clients evaluated in this thesis are tablets, J2ME[1] clients and Android clients. In table 2.1 is an overview of the benefits and drawbacks of using mobile devices.

Pros	Will eventually support offline reporting. Mobile phones are more portable. Almost everyone has a phone. Mobile phone technology is technology most people know from before.
Cons	Fear of new technology that comes with new mobiles. Smaller in size, easier to loose. Smaller screen, requires more scrolling. It takes time to get used to a new device.

Table 2.1: Pros and Cons of mobile devices.

[1]Java 2 Platform Micro Edition is an environment for applications running on mobile and embedded devices

2.4.1 J2ME clients

The J2ME version of the DHIS2 Tracker is a version that can run on low-end phones. An example of the DHIS2 Tracker on a J2ME client can be seen in figure 2.1. In table 2.2 is an overview of the benefits and drawbacks.

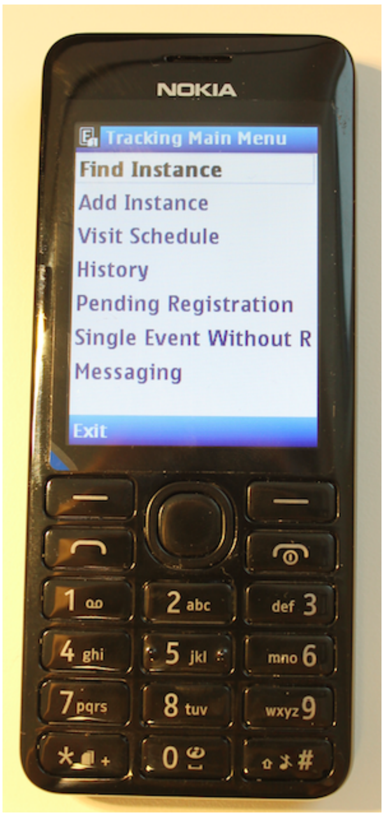


Figure 2.1: DHIS2 Tracker on a J2ME client.

Pros	It has limited functionality, avoiding misuse. They are cheap. They are less likely to be stolen. Number entry on keypad is faster. They have long battery lives.
Cons	Started to become old technology. They are slow. They have small screens and joysticks for maneuvering.

Table 2.2: Pros and Cons of J2ME clients.

Looking at the pros and cons, one can say a little about how well the J2ME scores on the four dimensions characterizing M-health (Sanner et al. 2012). The J2ME client scores high on *cost* and *robustness* since it is a low-priced client that has a long-lasting battery life, and when the offline functionality is done, it will be even more robust. The *usability* of the J2ME client is somehow limited, as it is a slow device. Also they have small screens and often joysticks, making it more time-consuming to maneuver in the application. The *flexibility* of the J2ME is not too great, as it is complex to update an application if there has been an improvement or new features in it.

2.4.2 Android clients

The Android version of the DHIS2 Tracker is soon to be released, and a beta-version of it can be seen in figure 2.2. In table 2.3 is an overview of the benefits and drawbacks.

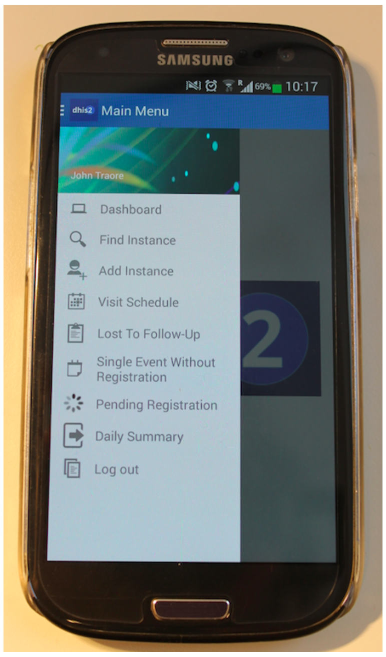


Figure 2.2: Example of Tracker on an Android client.

Pros	The popularity of Android clients is growing. They are fast and has more memory. They have touch screens. The workflow is smoother. They are faster and easier to use.
Cons	The lives of batteries are short. They are expensive. They are easier to break. They have a lot of functionality, misuse is easy.

Table 2.3: Pros and Cons of Android clients.

Looking at the M-health dimensions given by Sanner et al. (2012) with respect to the Android client, it scores lower on *cost* and *robustness* as it is more expensive and has lower battery capacity than the J2ME client and is easier to break, but the Android client will also eventually have offline functionality, which will make it more robust. The *usability* of the Android is higher, as it is faster and easier to use due to the touch screen and network connectivity. The *flexibility* of the Android client is high, as it is easy to update and install applications for improvements and new features.

2.4.3 Tablets

Tablets are also possible mobile devices for DHIS2 Patient Tracker, but then you have to use the web-based Tracker for reporting. Using the web-based Tracker on a tablet is not ideal, as it requires a lot of scaling and navigation, which affects the *usability* of the device as it is more challenging to have a clear overview of the whole screen. Looking at the other dimensions of M-health given by Sanner et al. (2012), tablets *cost* a lot more than Android and J2ME phones. In this setting the tablet was used for reporting with the web based Tracker. This reduces the *robustness* of it, but if an app for tablet is made, offline functionality can be made, making the tablet more robust. The *flexibility* is high, as it is easy to update and install applications on a tablet.

Chapter 3

Literature review

This thesis concerns the implementation and development of software in a globally distributed setting. It is therefore important to look at what experiences previous studies have yielded about distributed software development (DSD). The first section will therefore give a definition, followed by a presentation of claimed benefits of DSD. Then a set of strategies for successful DSD is described.

This thesis also concerns development of software where. participatory design (PD) is central. The second section will therefore give a definition of it. followed by a description of PD in a distributed setting and how it differs from traditional PD. Then a set of approaches for successful distributed participatory design is described.

3.1 Distributed software development

Global software development (GSD) is defined as '*software work undertaken at geographically separated locations across national boundaries in a coordinated fashion involving real time and asynchronous interaction* (Sahay et al. 2003, p. 1)'

DSD is defined as activities like software development, deployment and maintenance where the actors aren't collocated, but have a distance between them (Lings et al. 2006). Three types of distances are defined to describe the distribution: *Temporal distance:* measures dislocation in time by two actors who want to interact. Caused by time zones or work time patterns. *Geographical distance:* measures the effort required for one actor to visit another actor. *Socio-cultural distance:* measures an actor's understanding of another actor's values and practices. The distribution is considered global when all three distances are high (ibid).

3.1.1 Benefits of DSD

Conchuir et al (2009) explains why organizations choose to distribute their software development by describing claimed benefits of GSD. The potential benefits are as follows:

Reduced costs When moving parts of the development to low wage countries, the same work can be done with a reduced cost. But moving the development makes communication harder, and requires more control and coordination.

Leverage time zone effectiveness With developers in different time zones, organizations can increase the number of daily working hours. However this increased effectiveness is hard to achieve, as it reduces the overlapping window for collaboration between time zones and causes unusual work hours for the developers.

Cross-site modularization of development work by dividing task into modules, the development can happen in parallel and reduce cycle time. On the other hand, reduced communication between sites can lead to integration problems and loss of a sense of cross-site team spirit among the developers.

Access to large skilled labor pool Being distributed increases the access to talented developers, which have a great impact on development productivity and quality. But cultural distance then becomes a challenge, and communication problems due to differences in culture is present.

Innovation and shared best practices Diverse backgrounds of developers can lead to increased innovation and shared best practices amongst the team. But due to lack of face-to-face and informal communication the opportunities for sharing are small. Also high-wage developers feel threatened by low-wage colleagues, and will only share necessary information to get the job done.

Closer proximity to market and customer By establishing subsidiaries close to customers' location, a more direct interaction becomes possible. But socio-cultural problems can arise, due to cultural divide amongst the team.

To what degree these benefits are realizable varies. Don't assume overall costs will be reduced with lower wages, as management becomes more complex. Follow-the-sun software development isn't always ideal, as companies may prefer to modularize work instead of taking advantage of developers situated in various time zones. Sharing of best practices between cultures and taking advantage of closer proximity to foreign contexts can be problematic due to socio-cultural distance (ibid).

Herbsleb and Mockus (2003) found that distributed development takes about 2.5 times as long to complete as collocated development. The reason for this increase is that in distributed work, more people are involved than in collocated work. Suggested strategies to reduce the time are use of practices to facilitate communication, coordination and control.

3.1.2 Strategies for successful DSD

As the processes of communication, coordination and control are affected by distance, they have direct consequences on how software is developed and thus there is a need for techniques to improve GSD (Damian et al. 2003). *Communication* involves transfer of information between actors and the tools used to achieve a common understanding, *coordination* involves splitting tasks to each unit, so they contribute to an overall goal, *control* involves management of projects so standards are followed and goals are achieved (Carmel and Agarwal 2001, p. 23).

A set of strategies for successful distributed development, where the three processes of coordination, control and communication are central, is suggested by Lings et al. (2006) as follows:

Have a clear distribution rationale Not all projects and contexts fit to DSD. Only consider it for well structured, understood and stable projects, decomposable into tasks. Also choose teams with a common language and at least a few hours overlap in working time. This strategy affects temporal distance by reducing the need for communication, which also simplifies control and coordination. It also reduces socio-cultural problems such as cultural misunderstandings.

Clarify all understandings Agree on and communicate goals and targets and make sure they are understood. Define teams and what tasks will be done in the different locations. This strategy minimizes misunderstandings in communication, which is a risk in teams with different socio-cultural backgrounds. Good documentation also reduces communication problems.

Use cultural mediation Use a cultural mediator, a person from one context spending time in another context, working as a link between contexts. This strategy reduces geographical and socio-cultural distance by suggesting that it's worth spending resources on face-to-face communication.

Facilitate human communication Communication is best face-to-face, and informal face-to-face communication can improve communication in formal meetings. This strategy focuses on communication across all three dimensions of distance.

Manage processes Have a project leader, with full responsibility and supplemented with local project leaders and teams. This strategy reduces the temporal distance of control and coordination in a project by handling tasks between nodes.

Develop a sense of teamness By ensuring timely feedback about progress, deliverables and ideas, team members can get a feeling of being a part of a team. This strategy facilitates communication and coordination.[1]

[1]Teamness is a word used by Lings et al. (2006) to describe team building and team spirit.

Encourage temporary collocation Investing in periods of collocation can reduce problems in future processes. This strategy takes the cultural mediation further by saying developers should spend time in remote areas. This strategy reduces geographical and socio-cultural distance in communication and coordination.

Develop an effective tool base A common software configuration management tool is recommended for coordination. This strategy aims to facilitate coordination and control through the use of standardized tool support for configuration and change management.

These strategies are supported by Noll et al. (2010), who examined 26 papers for barriers in GSD. The barriers found all were related to the three distances defined and the solutions to these barriers are thus also the same as the ones given above. The mentioned strategies are all efforts to make the distributed team to function as if they were collocated. Edwards and Sridhar's (2003) study, investigating 24 distributed teams, indicated that ease of use of technology, trust between teams and a well-defined task structure influence positively the efficiency, effectiveness and satisfaction level of global teams.

Bird et al. (2009) showed that globally distributed software not necessarily leads to more failures by comparing and analyzing distributed and collocated development. The result was that distribution had very little, to no effect on failures and it is argued that the low impact distribution had on failures was due to use of the mentioned strategies to overcome barriers. For example the employers took responsibility to work late and arrive early on turn to reduce the communication barrier.

3.1.3 Generification

Generification is an important strategy when developing global software systems, for example when adapting a local innovation to a global toolbox. Generification is defined as designing a generic artifact, usable for several contexts (Pollock and Williams 2009). The concept of Generification can be explained as the process of adjusting local requirements to become global standards that can be used by other local contexts. In other words taking something specific and make it generic. This thesis will go into processes that can be seen as process of Generification. But Generification as a concept is not a part of this thesis.

3.2 Participatory design

Central in the development of health information systems is to involve the end-users from the beginning (Berg 2001). A way to achieve user involvement is by participatory design. As a HIS can be developed in a distributed setting, PD also has to take into account the distributed setting. To understand how PD works in a distributed setting, it is important to also be aware of how traditional PD works. The central idea is that those who will use a software, play a critically role in the design of it and is defined as:

> *A process of investigating, understanding, reflecting upon, establishing, developing, and supporting mutual learning between multiple participants in collective 'reflection-in-action'. The participants typically undertake the two principal roles of users and designers where the designers strive to learn the realities of the users' situation while the users strive to articulate their desired aims and learn appropriate technological means to obtain them.* (Robertson and Simonsen 2012, p. 2)

These two principal roles reflect two important aspects of PD: The first tries to enable the users of the software to have a voice in its design, without being technological experts. The second is that users might not be able to define what they want as they don't know what is possible (Robertson and Simonsen 2012, Ch. 1).

The core of PD is genuine participation, and it refers to the transformation of the users' role from being informants to legitimate participants and decision-makers in the design process. The participation becomes genuine when a user goes from answering what he thinks of an issue to step up and draw a sketch of the process (Bodker et al. 2004). Mutual learning is also a central concept in PD, it is an ongoing process where users need knowledge of potential technological possibilities and designers are the source of this knowledge. On the other hand designers need knowledge about users and their practices, and users are the source of this knowledge. During PD all participants increase their knowledge through mutual learning. PD is driven by social interaction as users and designers learn together to create, develop, express and evaluate their ideas and visions. In PD what is being designed is both the product and the process that enables different participants to engage in designing this product (Robertson and Simonsen 2012, Ch. 1). Vital for mutual learning is a close relationship between designers and users (Gumm et al. 2006).

Participatory design is an approach to software development attempting to involve the users of the system in designing it. In a distributed setting the involvement of users can be challenging and mediators are crucial stakeholders in facilitating the PD in the software development (Shidende et al. 2014). It is emphasized that whether the design is participatory or not, depends on an implementer's ability to mediate requirements between users and developers (Braa and Sahay 2013, p. 247).

3.2.1 Distributed participatory design

Due to the facts that globalization makes software development more and more distributed and there is a need of mutual understanding between users and developers, PD approaches become important. Central in PD is a good relationship between developers and users, which relies on good communication and knowledge sharing. In a distributed setting the traditional PD approaches become more challenging. The term DPD tries to deal with the PD approaches of developing software in a DSD setting where stakeholders are distributed geographically, temporally and socio-culturally. The focus of PD has changed from customization of software to empowering users to assess their own practice and technology use (Gumm 2006; Gumm et al. 2006; Obendorf et al. 2009).

A challenge of successful DPD is the combination of real participation and geographical distance. The distance makes informal communication difficult, as it is hard to bring users and developers together face-to-face. A suggested solution was to distribute tasks geographically so that the communication between sites could be as small as possible. Gumm (2006) claims this solution is bad as it reduces opportunities for synchronous work, cooperation and mutual learning between sites. Also cooperation through mail, mobile or Internet decreases involvement of users, as issues can't be discussed in the same way it would be in face-to-face meetings. The distance can also cause contradicting requirements between contexts (Gumm et al. 2006).

New ways of enabling communication between users and designers in a distributed setting are suggested as traditionally PD approaches tend to focus on single case contexts that has little relevance to other contexts (Gumm et al. 2006; Obendorf et al. 2009):

Mediated two-directional feedback Having persons acting as mediators facilitate participation. Mediators collect feedback from users and pass it on to developers, and feedback from developers back to users. The mediators bridge the gap between users and developers.

Inter-contextual user workshops These are held with developers in close cooperation with users to analyze and discuss requirements for future developments. These also enable informal communication by bringing users from different contexts together to exchange experiences of usage in face-to-face interaction. The focus is on reflecting and enhancing use practices within the software design, rather than designing new features.

Commented case studies Are small descriptions of cases that enable exchange of experiences between users from different contexts and developers. By describing cases, users from other contexts have access to other users' experiences. By commenting on case descriptions, developers can document design decisions and enables users to get insights of why the software is developed the way it is. This method gives an opportunity for users to participate even if they can't participate in workshops.

Surveys Can be used to evaluate users' satisfaction of software and how and for what it is used. Here participation is enabled by giving users the opportunity to tell about problems they have with the software or new features they want in further development of it.

User support Providing documentation and personal user support to questions or bug reports can be a means for participation, as it has shown to be highly appreciated by users.

To succeed with PD in settings like low resource countries is more demanding as users often have limited computer skills because of lack of computers, infrastructure and ICT education. In such settings training of intended users is vital for them to contribute in the PD processes. A suggested approach is participatory customization, described as a process whereby users in collaboration with developers adapt an already developed system to meet their needs. It was experienced that participation increased when the users were trained and got more knowledge about the software (Kimaro and Titlestad 2008).

Chapter 4

Theoretical framework

In the previous chapters, the underlying concepts of this theoretical framework were described. This chapter will go through existing theoretical contributions where these concepts are central. These concepts and theories will together provide a theoretical framework for discussing findings from this thesis with others' experiences.

4.1 Tensions between local and global

When one looks at a small local innovation within particular organizational settings, one should not ignore the whole picture. It is important to look at other settings and sites to see how they have shaped the development. A suggestion is to look at the development of software longitudinally as a biography of an artifact. It is emphasized that in an innovation one needs to understand how the artifact is shaped by the context and history (Williams and Pollock 2009).

When IS span several contexts globally, it is necessary to strike a balance between the need for information infrastructure that adapts to various local contexts, and simultaneously standardizing toward a universal solution across contexts. Rolland and Monteiro (2002) suggest two reasons for tensions. First, to establish global information infrastructure, one needs uniform and standardized solutions. Second, information systems need to adapt to local contexts to succeed. So the problem is, how to make IS that adapts to local contexts and also cut across contexts. The accuracy of information must be weighed against the costs of collecting, registering and managing such information. High-level accuracy reduces flexibility. Low-level accuracy increases the flexibility (ibid).

The next section presents a model that tries to handle the tensions between local and global.

4.2 Global design through local use

A global approach to DPD of HIS in the global south is suggested as DPD becomes extra important in the global south for two reasons (Titlestad et al.

2009). Firstly, there is a lack of collaboration when donors introduce new systems; they are often bespoke and don't fit well into existing reporting routines. For new health systems to succeed, they have to fit into existing workflows (Asiimwe et al. 2011). Secondly there are often lack of funds and skilled people.

These two challenges can be met by DPD, since through collaboration it is easier to build reusable tools that follow global standards such as those recommended by the Health Metrics Network (HMN) by listening to user and developer communities in different countries.

Titlestad et al. (2009) describe an approach for how to benefit from local innovations to make global standards through DPD. Their case describes the development of a HIS, namely DHIS2, within the HISP network. The goal is to make DHIS2 a toolbox that easily can be used by anyone.

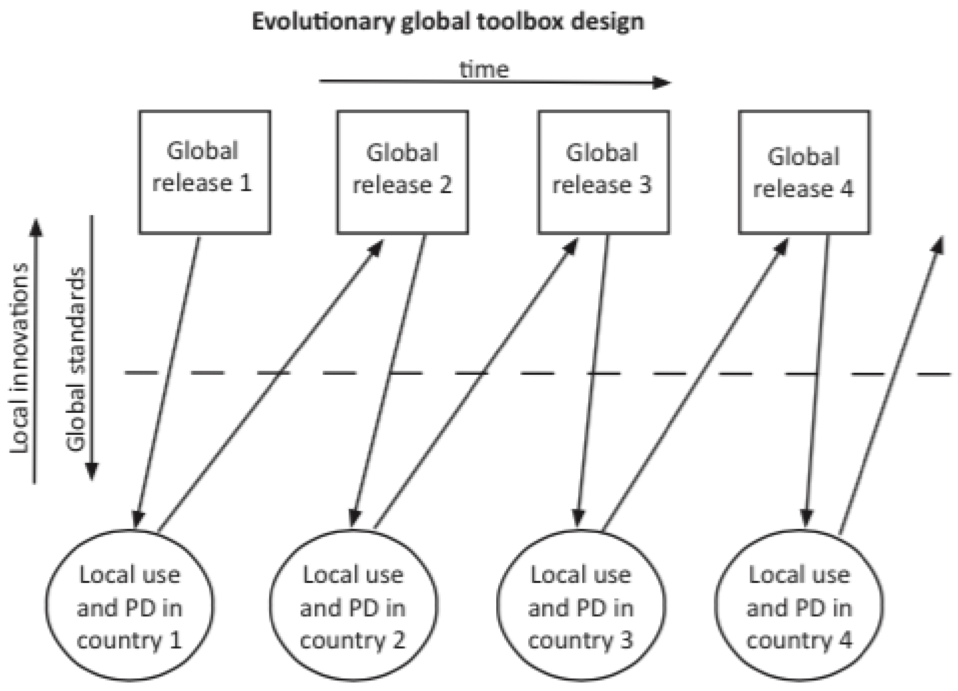

Evolutionary global toolbox design
time
Global release 1
Global release 2
Global release 3
Global release 4
Local innovations
Global standards
Local use and PD in country 1
Local use and PD in country 2
Local use and PD in country 3
Local use and PD in country 4

Figure 4.1: Evolutionary global toolbox design (Titlestad et al. 2009).

Illustrated in figure 4.1 is the interaction between the local and global level through the distributed design activities. The design consists of two processes that are clearly distinct, by taking place at different levels, but still related. The two processes are defined as in-country design and across-country design. The first being design of health information systems with an emphasis on the system rather than the software and the latter being design of globally distributed toolbox of software and best practices for in-country HIS design. Those two processes are vital in the development, as globally distributed solutions arise from local design and use, and the global toolbox is exploited in local solutions (ibid).

The model tries to show that a module will be designed through an evolutionary process where global standards are used to make new local innovations, which again are adapted into new global standards in several iterations. But Titlestad et al. (2009) underscore that the global toolbox doesn't work well across boundaries without circulation of people. Implementers should work as boundary spanners. Implementers are described as persons doing activities to put the software developed into use. Boundary spanners are described as persons who act as mediators between organizations and teams. Implementers act as boundary spanners to balance the global standards with the local needs by mediating between global core developers and local end users (ibid).

4.3 Free and Open Source Software and the role of DPD

Free and open source software (FOSS) enables and facilitates low resource countries to make use of technology and software since it is perceived as free of charge. This again enables participation and contributes to the software's adaption to local conditions. The FOSS approach is suitable for development of software where participation in the design is important, as it encourages more participation from the users. The focus when developing FOSS is on general use, and therefore the software only fulfills some user requirements, as it only tries to cover the universal aspects of the local requirements (Camara and Fonseca 2007).

The concept of a design-reality gap tries to explain the deviation between local requirements and global design (Heeks 2006). Managers of health programs worldwide face great challenges in adapting to FOSS HIS when there are conflicts between local requirements and global design.

Hewapathirana and Rodrigo (2013) have explored managers' awareness of the design-reality gap when taking decisions on implementation of FOSS HIS. Also managers' acceptance of the gap and adjustment of their local requirements to fit the restrictions coming from the global software have been looked at. This adjustment is phrased as actuality improvisation over design improvisation. Design improvisation is described as customizing software including change of code and actuality improvisation is described as changing the local requirements to fit the existing software.

It was found that managers were not dejected when realizing there was a design-reality gap, but they were doubtful to change their requirements to fit existing features of HIS. The managers preferred design improvisation over actuality improvisation as it was more successful in minimizing the tension over the design-reality gap. But actuality improvisation was accepted in the presence of the gap. As long as the most important requirements were met, the managers were willing to sacrifice some less important requirements (ibid).

4.4 Participation in a scaling environment

PD is focused on designing a system to fit specific software to one use case. When a single-site software system scales to multi-case systems, there is less space for single cases driving the feature set of the platform. Traditional PD struggles to meet the challenges of scaling to multiple locations. If the process of making a system generic to fit multiple use cases and participatory design are to be combined, the system's architecture needs to support a high level of flexibility simultaneously as it has a generic core. A strategy is to separate the system and its processes into layers, where different design methods can be used (Roland et al. Forthcoming).

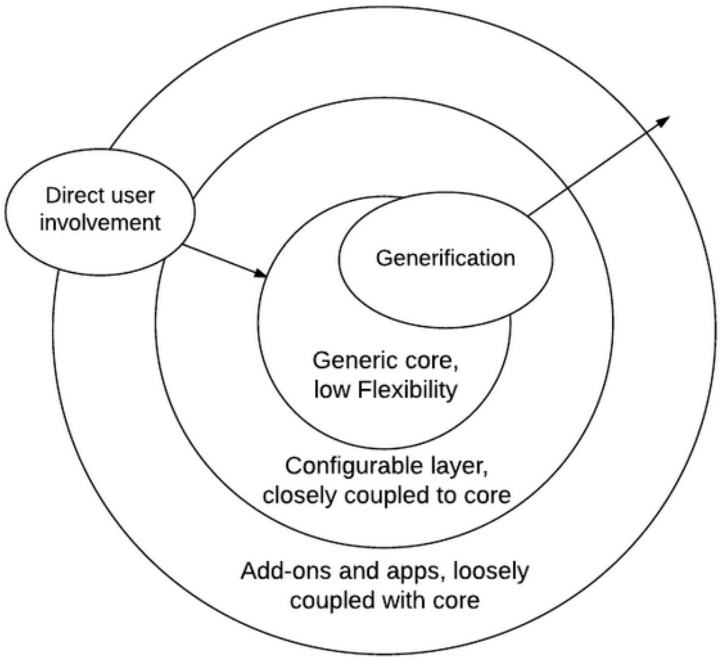

Direct user involvement
Generification
Generic core, low Flexibility
Configurable layer, closely coupled to core
Add-ons and apps, loosely coupled with core

Figure 4.2: A layered system (Roland et al. Forthcoming).

Figure 4.2 describes the different layers in a system. In the centre is the generic core that is difficult to change without affecting all implementations. In the middle is a layer that offers implementers to configure the system's behavior. The outer layer enables developers to make applications that are detached from the core and able to use PD principles (ibid).

4.5 How participatory is the participation?

Several studies have looked at the PD in HISP and discussed the involvement and challenges of the different stakeholders through the process.

4.5.1 History of PD in HISP

HISP is a heterogeneous global distributed network that attempts to apply participatory approaches when designing HIS in cooperation with various users in a variety of contexts. Over the years HISP has changed from a software development project with PD in a single context, to a distributed network with PD processes in a multiplicity of contexts (Braa and Sahay 2013).

PD in HISP can be divided into three phases. The first (1995-2000) started with traditional PD in a single context as collaboration between health researchers (users) and informatics researchers (designers/developers). In HISP they applied evolutionary PD, which is PD where software never is finished, as there always will be changes of environments and room for improvements in the health sector (ibid).

In phase two (2000-2006) HISP was expanding across countries and contexts, becoming a network of PD projects. A challenge then was to keep traditional PD in multi-country contexts. It was realized that context matters in PD, as it depends on time, space and cultural conditions in which it evolves. The solution was the creation of a network of action, which refers to linking people or institutions together for collaboration around the actions of developing, implementing and scaling a HIS (ibid).

In the third phase (2006-2012) the software changed from stand-alone pilots to networked applications, which resulted in a challenge to locally focused PD approaches. The development of DHIS2 aims at distributed development in multiple countries to bring the software development closer to the contexts of use. Even though the DSD of DHIS2 was an effort to bring the software closer to users, the new architecture made PD approaches difficult (ibid).

When the development of DHIS2 started in 2008 and until 2011 the process of making the software generic was very much serial. Developers and implementers were in one country at a time and the users were very involved in the process (Roland et al. Forthcoming).

From 2012 more countries joined and the distance between the users increased. The implementations happened in parallel and the amount of requirements given to the core developers was enormous and requirements were sometimes conflicting. It happened that one feature was changed back and forth several times as two different countries changed the same feature to fit their needs. The DHIS2 development changed from a case-driven process to take into account parallel cases and made the traditionally PD approaches more challenging. This has led to decreased user participation in the development (ibid).

The PD in HISP involves a diversity of contexts, technologies, levels and interventions, so one could say that the process over time has changed from traditional to distributed participatory design (Braa and Sahay 2013).

4.5.2 Recent trends of PD in HISP

There is need for a change of DHIS2 to become more platform-based, allowing local communities to have more influence on the software to fulfill their requirements to suit their needs, simultaneously as they can benefit from the global software. The need for this platform is great in contexts where users at the local level need to fulfill requirements, which aren't fulfilled by the software. Without the platform it is the core developers who decide on the openness of the software to extend with new features (Manda et al. 2014).

Manda et al. found that the decision power on design in very distributed projects, like the development of DHIS2 is difficult. It can be that two stakeholders have very specific requirements that are in conflict, thus the requirement for one of the stakeholders can't be supported at a global level. The majority of stakeholders will end up having little influence on the design. In the end, the core software developers end up with the power when it comes to decisions on design. In the cases they looked at, it was found that there never were any direct communication between developers and the end-users, and all the requirements on the design were gathered through other tasks, such as training. Some of these requirements required change at the global level, but due to the end-users' lack of influence on the design globally, the software ended up with reduced functionality locally, which in the end made it hard to participate in the global development of the software. Manda et al. suggested that a platform-based solution could solve these problems.

The interaction between implementers acting as mediators, users and developers in DPD processes in a low-resource context have been investigated, and some challenges have been identified (Shidende et al. 2014).

A great challenge encountered was for the mediators to communicate and follow up on the local requirements to the global developers, as the connection between them was poor. As a consequence of this, specific requirements failed to be implemented, and further participation on the design for the users became limited. If the global developers improve in providing better documentation of the software, this could ease the communication for the implementing mediators, as they would obtain more knowledge about the software.

Another challenge to address was deciding on which users should participate. This becomes hard as some end users can be involved in later stages and others lack knowledge of health and/or technology. Thus they aren't able to participate properly in the design phase. Training of users can resolve this challenge by making sure they understand both the technology and health domain. If the training also includes training for self-learning, the challenge of deciding on doing upgrades of the system or not decreases. Without users' ability of self-learning, the mediator has to evaluate the value of the new features against the cost of more training. The development of new core versions could lead to incompatibility with the local users, which requires more skills (ibid).

In networks such as HISP, where the number of actors is constantly on the increase, the role mediators have in participatory design becomes more and more important. In these networks it is impossible to involve all stakeholders that will be affected by system changes. Thus the mediators are the link between the global and local, and need to make the global system fit the local context. To succeed with that, the mediators need to understand the context well, as they have to interact with local and global stakeholders who are distributed, and possess different skills and work experiences. The different stakeholders are health workers, local leaders and global core developers (Shidende and Mörtberg 2014).

Shidende and Mörtberg (2014) focus on the role the implementation mediators have within the HISP network. Mediators need to understand the local context to be able to see if their work practices deviate from the global design. If a deviation is the case, the local requirements need to be shared with the global developers for them to become new features. By sharing requirements with the global developers, they can reflect on the design and add new features.

4.6 Summary

Chapter 2, 3 and 4 have introduced and explained different concepts and theories from relevant studies that are related to each other. This section describes the relation between these concepts and why participatory design is central, in order to compare other studies with the experiences and findings of this thesis.

Health information systems are vital for better health in low resource countries; health care does benefit from health information technology in quality and efficiency (Chaudhry et al. 2006), especially when it comes to decision-making (Stansfield et al. 2006). The growth in usage of mobile devices worldwide has a central role in HIS for low resource contexts (Donner 2008). M-health is important for low resource contexts, because the use of mobile devices can provide advantages in scalability, coverage, timeliness, and transparency, since those devices function in remote locations and are easy to bring and use at any time (Freifeld et al. 2010). It can also be important for low resource countries that the HIS is FOSS, as it is perceived as free of charge and thus facilitates use of it.

Berg (2001), Sahay and Walsham (2006) argue that in the development and implementation of health information systems for low resource countries, users and managers need to be included from the beginning of the project to adjust the implementation to fit the workflow. The capacity of users needs to be strengthened; training of users makes it easier for them to change their existing workflow to use new technologies. Users should be informed, so they are aware of the needs and changes that come with HIS. These arguments indicate that participation is important. Thus, Participatory Design can be central in development and implementation of Health Information Systems for low resource countries. As FOSS encourage user participation, development of it is suitable for PD approaches(Camara and Fonseca 2007).

Globally distributed software development is software development where the temporal, geographical and socio-cultural distances are high (Lings et al. 2006). Organizations choose to distribute software development in order to achieve benefits such as being closer to users and time zone effectiveness (Conchuir et al. 2009). Communication, coordination and control are affected by distance, and have direct consequences on how software is developed (Damian et al. 2003). Lings et al. (2006) have therefore suggested strategies to succeed with DSD. These suggested strategies are quite similar to methods of participatory design, and due to the fact that globalization makes software development more and more distributed and there is a need of mutual understanding between users and developers, PD approaches become important (Gumm 2006).

Heeks' (2006) concept of a design-reality gap describes the deviation between local requirements and global design. A source of this gap can be when IS span several contexts globally, because it is necessary to balance between software that adapts to various local contexts, and simultaneously standardizing toward a universal solution (Rolland and Monteiro 2002). This balance affects the degree of participation in the design.

Titlestad et al. (2009) and Roland et al. (Forthcoming) suggest two quite different approaches for coping with this design-reality gap with the use of PD approaches.

Titlestad et al.'s (2009) approach is an evolutionary approach where local innovations and global standards in turn affect each other on the road to become a generic toolbox. The idea is that users participate in the design in the local innovations, which will become global standards. Roland et al.'s (Forthcoming) approach is a platform-based approach where local innovations happen at an outer layer where users participate in the design and global standards are made in a generic core without much influence from users.

Both of these approaches give perspectives to understand how one can enable participatory design. It could be interesting to explore whether it is possible to combine these two approaches, as they enable participatory design in very different ways.

Chapter 5

Methods

This chapter discusses the methods used to collect and analyze data in this project. The first section will describe the theory behind the methods chosen. The next section describes the research approach, followed by a section presenting the data collection techniques used.

5.1 Research methods

The research method chosen in this project is based on the framework for IS research presented by Braa and Vidgen (1999). The theoretical foundations of the framework are described and research methods relevant for this project are explained. Finally, this project's placement in the IS research framework will be described.

5.1.1 The IS research framework

The framework suggested by Braa and Vidgen (1999) is illustrated with a triangle seen in figure 5.1. The three points represent intended research outcome, namely prediction, understanding and change. The desired research outcome will affect the choice of research method. To make reliable predictions, the researcher is concerned with reducing the area of investigation with methods such as Field Experiment. To gain understanding, the researcher is concerned with interpreting a situation with methods such as Case Study. To make a change, the researcher is concerned about gaining knowledge through making a change in an organizational context with methods such as Action Research.

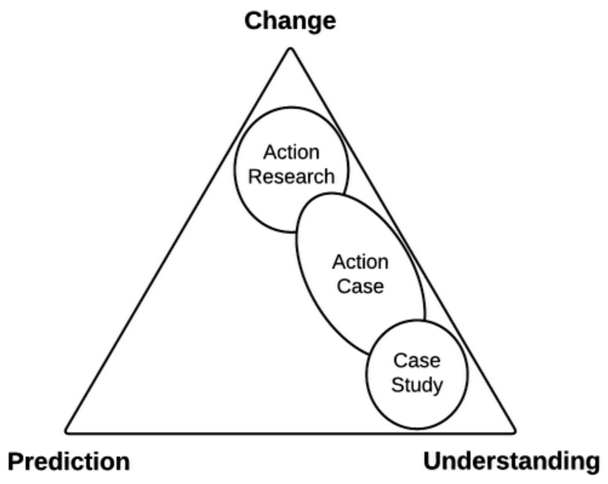


Figure 5.1: An IS research framework (Braa and Vidgen 1999).

5.1.2 Research methods

This section will describe the research methods action research, case study and action case. These three research methods are suggested in the IS research framework and are the ones relevant for this project. More about why these are the relevant ones and methods chosen is explained in section 5.1.3.

Action research

Action research aims to solve current practical problems while expanding scientific knowledge. The action researcher is concerned about creating organizational change and simultaneously studying the process. Action research is a way of building theories that are tested through experiments that affect a change in a situation (Braa and Vidgen 1999).

Susman (1983) suggests that an action research cycle consists of five phases:

Diagnosing Identification of situated problems and their underlying causes. During this phase, a hypothesis that will be used in the other phases is formulated.

Action planning Specifying of the actions that can improve the problem situation. That also includes specifications of prototypes based on the problems discovered.

Action taking Implementation of the intervention specified in the action planning.

Evaluating assessment of the intervention. This is done in the problem situation where the diagnosis was conducted.

Specifying learning Documenting the learning outcomes of the action research cycle. These learning outcomes should lead to contributions to theory and practice.

Participative action research is a kind of action research where the people being studied participate with the researcher in the action research cycles (Whyte 1991).

Case study

If you ask questions like "how" or "why" in your research question to explain a phenomenon, a case study will be a good method to choose (Yin 2013, p. 4). This fits well with Iacono and Holtman (2009)'s contention that a case study focuses on understanding a phenomenon in its natural setting. It is well suited for studying IS, when the focus is on organizational rather than technical issues, as attention is paid to contextual conditions when investigating a phenomenon. Researcher bias is mentioned as a concern, as the researcher's own beliefs and influences on participants' behaviors can affect the results, but then again, case studies contain no greater bias than other methods (Flyvbjerg 2006, p. 237).

A definition given on case study defines it as an empirical study, which investigates a contemporary phenomenon in its natural context, where lines between context and phenomenon are unclear. Because of this blur, a case study needs to rely on more than one source of evidence (Yin 2013, pp. 16-17).

Yin says that the first principle of data collection is to use multiple sources of evidence, and three sources emphasized are documentation, interviews and direct observation. Documentation covers all kinds of documents related to the case study such as e-mails, project protocols, minutes of meetings and so on. These documents provide valuable information. Interviews are important, since most case studies are about humans and their actions. It is vital to understand the real world of the case you study, and thus direct observation is great source (Yin 2013, Ch. 4).

Action case

Action case is a trade-off between being a researcher who interprets (understands) and a researcher who makes a change. In a case study the researcher aims to get insight in a situation. In action research the aim is to support change. Action case balances between understanding and change by being a small-scale intervention with a deep contextual understanding (Braa and Vidgen 1999).

The difference between Action Case, Action Research and Case Study is important. Table 5.1 gives an overview of the characteristics of action case and how it differentiates from the two other research methods.

	Case Study	Action Case	Action Research
Change	Unintended	Small scale	Large scale
Prediction	Low	Low	Low
Understanding	High	Medium	Low
Duration	Any	Short to Medium	Long
Participation	Low	Medium	High
Time Orientation	Historic and contemporary	Contemporary and building future	Building future

Table 5.1: Characteristics of Action Case compared to Action Research and Case Study (Braa and Vidgen 1999).

5.1.3 This project

When this project started, the intention was to make a change by finding ways to make the DHIS2 Tracker more useful for health workers through feedback. Thus action research was a suitable research method. The two first steps, diagnosing and action planning, of the action research cycle were started and a part of these steps was to go on a field trip to Uganda to come closer to the users. So in a way, the plan was to follow the method of participative action research.

The project in Uganda hadn't come as far as originally assumed and health workers were not reporting data in the Tracker themselves. Thus there was no need for feedback and the project had to change.

Instead, the focus changed to looking at the opportunities for implementation of mobile devices for reporting in DHIS2 Patient Tracker. In Uganda, they had tried to implement a low-end mobile client for reporting, but not succeeded. Then the objective of this project became to investigate and find an explanation of why the implementation of low-end mobile clients had been so difficult.

With the change of focus, also the research method had to be changed. Instead of resulting in a change, the new focus resulted in understanding. Thus action research wasn't suitable anymore. For this kind of research a case study was more appropriate. Case study as a method is suitable because the research question in this project asks "how" and to answer, it is necessary to see how it works in the natural context. Also the three mentioned sources of evidence were used to better understand the case.

In addition to getting an understanding of what had happened in Uganda, a secondary objective was to assist the project in implementing the low-end mobile client for DHIS2 Patient Tracker. The implementation of mobile clients can be seen as a minor change by enabling health facilities in rural districts with electricity and connectivity problems to report into the Tracker. With this kind of research, action case would be a suitable research method.

Looking at figure 5.1, this project end up somewhere between action case and case study. To answer the research question a case study was necessary, and by implementing the low-end mobile clients a change was done, but the research method was more a case study than an action research, thus not completely an action case. The change was a move towards getting more understanding.

5.2 Research approach

The master project can be seen as a process with three phases: *a*) the preparation before the fieldwork; *b*) the fieldwork in Uganda; and *c*) the period after returning home. An overview of the work done in these three periods will be provided in the following sections.

Preparation for fieldwork (August 2014 - October 2014)

In August 2014, the University of Oslo arranged the DHIS2 Academy in Oslo where several projects and use cases involving DHIS2 were presented. To see this variety of uses gave a good foundation for the understanding of the project. It was also during the DHIS2 Academy the decision of going to Uganda was made.

The plan for the visit was to follow up on the MUJHU project and to investigate the possibility of motivating health workers using the DHIS2 Patient Tracker through feedback. The feedback was supposed to motivate the health workers by making the Tracker more valuable for them. The suggestion was to develop a web-application enabling feedback in real time to the health workers using the Tracker so they could benefit from it.

Preparation for the trip to Uganda consisted of getting a better understanding of the DHIS2 Patient Tracker, which was the component used by health workers in the project, preparing for application development by improving programming skills and making templates, and writing a plan for the trip including aspects such as interview guides and what to observe and who to talk to.

Field study in Uganda (October 2014 - November 2014)

The field trip to Uganda started in mid October and was conducted with two fellow students within the same research area. In the first week of the stay DHIS2 training was arranged, which gave insight in the uses of DHIS2 in Uganda and the network around the software. During the first week a quick sit-down with representatives from the project revealed that the need for timely feedback was not present yet. The health workers weren't actually using the Tracker themselves due to capacity problems. Instead, IT interns were hired to do the reporting of data into the Tracker.

A change of focus then was required and after a talk with the supervisors back in Oslo and the local project representatives, the new focus was on implementing the J2ME client for tracking in Uganda, and testing the Android client. The new focus included several tasks:

1. Testing and implementation of the J2ME client: tested and looked for bugs on a borrowed J2ME client. The bugs found were reported.

2. Testing of the Android client: In parallel with the J2ME client, its replacement, the Android client was also tested. Bugs found were reported.

3. Discussion of the two clients with MUJHU: The two mobile clients were presented, demonstrated and discussed with the MUJHU team in several meetings.

4. Discussion of different interfaces with MUJHU: Other interfaces like tablet and laptop were also discussed. A special focus was on the factors important for the clients to function well in their project.

5. Discussion and observation of the clients in the field: By visiting clinics, observing and interviewing health workers and IT interns.

After returning home (November 2014 - May 2015)

The return to Oslo was in mid November. Until January 2015 the time was used on analyzing and processing the data gathered in Uganda in parallel with further work on the J2ME client such as testing functionality, reporting bugs and discussing features with developers and project leaders. In the start of January the J2ME client was ready for rollout. More about the implementation process is described in chapter 7.

The following period from January to May was spent on further data collection. In this period, interviews of developers, implementers and project coordinators in HISP Oslo were conducted. More about what the actors of HISP Oslo said is presented in section 8.1.

5.3 Data collection techniques

This section will explain the different data collection techniques used in the thesis. A variety of techniques were conducted to get a wider understanding.

Field study

Location	Date	District
Naguru Hospital	22.10.14	Kampala
Kawaala Health Center	23.10.14	Kampala
Komamboga Health Center	28.10.14	Kampala
Ggoli Health Center	30.10.14	Mpigi

Table 5.2: Overview of Locations.

In mid October a one-month study was conducted in Uganda. Most of the time was spent in Kampala, the capital of Uganda. One week was spent on DHIS2 training in Mbarara and a day-trip went to the Mpigi district. During the stay four different health facilities were visited. The facilities are listed in table 5.2 and their positions are illustrated in figure 5.2. The field trip was carried out in cooperation with the MUJHU collaboration project and besides gathering data an effort was made to successfully implement a J2me client for DHIS2 Patient Tracker.

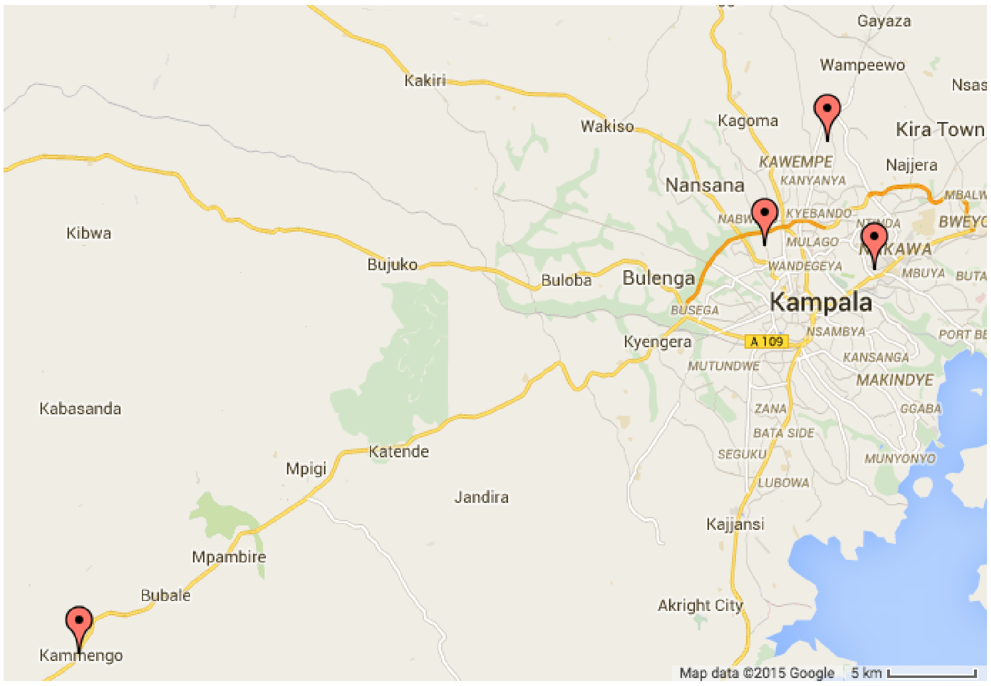


Figure 5.2: A map of the locations.

Observations

Location	Description
Kawaala Health Center	ANC nurses using DHIS2 Patient Tracker on laptop
Naguru Hospital	ANC nurses and IT interns using DHIS2 Patient Tracker on laptop
Komamboga Health Center	ANC nurses using DHIS2 Patient Tracker on laptop
Ggoli health Center	Health workers using DHIS2 Patient Tracker on laptop, tablet and android

Table 5.3: Overview of Observations.

During the visits to the different health facilities, the behavior of the health workers and the IT interns were observed. The observations covered use of laptops when registering information in the DHIS2 Patient Tracker, mother passport, health card and register book. In Ggoli the whole process of treating a patient and the use of tablet and android for registering in the DHIS2 Patient Tracker was observed.

An overview of the observations is listed in table 5.3

Interviews

Location	Description
Kawaala Health Center	Group interview of an ANC nurse
Komamboga Health Center	Group interview of an ANC nurse and a HMIS officer
Ggoli Health Center	Interview of a health worker
MUJHU Collaboration Center	Interviews of an IT intern and a team leader
HISP Oslo	Interviews of a core developer, three implementers and a project coordinator in HISP

Table 5.4: Overview of Interviews.

During the stay in Uganda, interviews were conducted of health workers, IT interns and MUJHU-project members. These interviews covered use of the Tracker and the interviewees understanding of the use of it.

In Oslo a core developer, three implementers and a project coordinator of the HISP organization were interviewed to get an understanding of the global aspect of the distributed participatory design process. The topics of these interviews were their approaches when it comes to development and implementation, communication between actors in the organization, documentation processes, requirements gathering, coordination and management and to what degree development model suggested in section 4.2 still is valid.

An overview the interviews is listed in table 5.4

Informal talks

Location	Description
DHIS2 training in Mbarara	Talk with the project team about the goal of the trip
Kawaala Health Center	Small talk with an IT intern about mobile devices for tracking
MUJHU Collaboration center	Discussion of different interfaces for tracking with project leaders

Table 5.5: Overview of Talks.

While staying in Uganda informal talks were held and covered topics like goals for the trip, discussions of different mobile interfaces for tracking. An overview of the talks is listed in table 5.5.

Meetings

Date	Location	Description
10.10.14	HISP Uganda office	Introduction to HISP Uganda and project team
20.10.14	HISP Uganda office	Skype-meeting about Android Tracker
20.10.14	MUJHU Collaboration center	Introduction to the WEMR project
21.10.14	HISP Uganda office	DHIS Mobile meeting about Android Tracker
22.10.14	MUJHU Collaboration center	Weekly meeting of the WEMR project
28.10.14	MUJHU Collaboration center	Weekly meeting of the WEMR project
15-22.12.14	HISP Oslo	3 Skype-meetings about J2me Tracker

Table 5.6: Overview of Meetings.

Meetings were attended and the topics were varied. In Uganda the topics were weekly meetings of the WEMR project and Skype meetings about mobile devices.

In Oslo the Skype-meetings included project members in Uganda, software developers and global project managers. An overview of the most important meetings is listed in table 5.6. The Skype calls were an important way of communication, as it enabled discussions. During these calls status updates were given and they contained information about what kind of issues were fixed and which were planned to be fixed in near future. Also specific requirements regarding bugs were elaborated.

Thematic Analysis

When the interviews were transcribed, they were analyzed by following the method of thematic analysis, which is a method for identifying, analyzing and reporting patterns within data (Braun and Clarke 2006). The six phases of thematic analysis were followed: *a*) became familiar with the data by transcription, reading and re-reading; *b*) coded the data for interesting topics; *c*) put the codes together in different themes; *d*) reviewed the themes to see how they support the rest of the study and the research question; *e*) a new analysis reshaped the themes capturing what is interesting about the themes; and *f*) produced a report of the analysis making meaningful contributions to answering the research question.

Other material

Other material to help collect more and better data is listed below.

Interview guides
Interview guides were made and refined through the interviewing process. The guides were not strictly followed, but used as support when needed.

Field notes
Taking notes was one way to capture data while doing interviews and observations and participating in meetings. The notes were a great resource when making daily summaries.

Voice memos
Another way of capturing data was with a voice recorder. It was used during interviews, and was later transcribed. The transcriptions contained all information, and eased the task of taking notes while interviewing.

E-mails
E-mails were the most important way of communication as the different interests in this study were distributed around the world. Managers were located in Oslo, developers were located in Manila and the users were located in Uganda. When reporting bugs, e-mail was the best solution, since there is a time difference involved.

Pictures
Pictures were taken at health facilities, to easier describe and illustrate the context and the workflow.

Documents
Project protocols and descriptions were provided. These documents were valuable for a clearer understanding of the projects. Transcriptions of interviews were important documents giving valuable data when analyzed.

Testing
Time was spent on testing and reporting bugs both on the Android and J2me client. This work helped the WEMR project on the way to successfully implement the J2me client and resulted in more understanding of the context.

Chapter 6

The Ugandan case

This chapter presents the case followed throughout this thesis. First the background of the project is presented. Then the situation in Uganda is explained with focus on the opportunities for implementing mobile technology. The last section will describe challenges of implementing mobile clients.

6.1 HISP Uganda

HISP Uganda is an organization located in Uganda that is a part of the global community of the HISP network. HISP Uganda collaborates with the Ugandan Ministry of Health, higher institutions of learning and Non-governmental organizations to strengthen learning and sharing about health information systems including all its implementation phases. An example of such collaboration is the WEMR project having members from MUJHU/UCSF, CDC, MOH and HISP. Their philosophy is to improve health care in developing countries through research and implementation of HIS (*HISP Uganda* 2015).

6.2 Project background

The project followed in this thesis takes place in Uganda and is about tracking patients. The project was initiated by the MUJHU collaboration.

6.2.1 Patient tracking in Uganda

The health care during pregnancy and after birth was traditionally split in Uganda, for example two different health cards were used to capture data about pregnancy and after birth. Seen in figure 6.1 are the health cards for pregnant women. The MoH in Uganda expressed a wish for an information system to improve maternal and child health by a more integrated approach. The result was a program following the mother through the pregnancy and after birth with several visits, and the opportunity to adjust number of visits accordingly. The tool used to capture the data and align the different practices and programs were the DHIS2 tracking module (Roland et al. 2013). There is a hope that in the future, the use of DHIS2 can replace the paper reporting, and implementing J2ME clients is a step towards that goal.

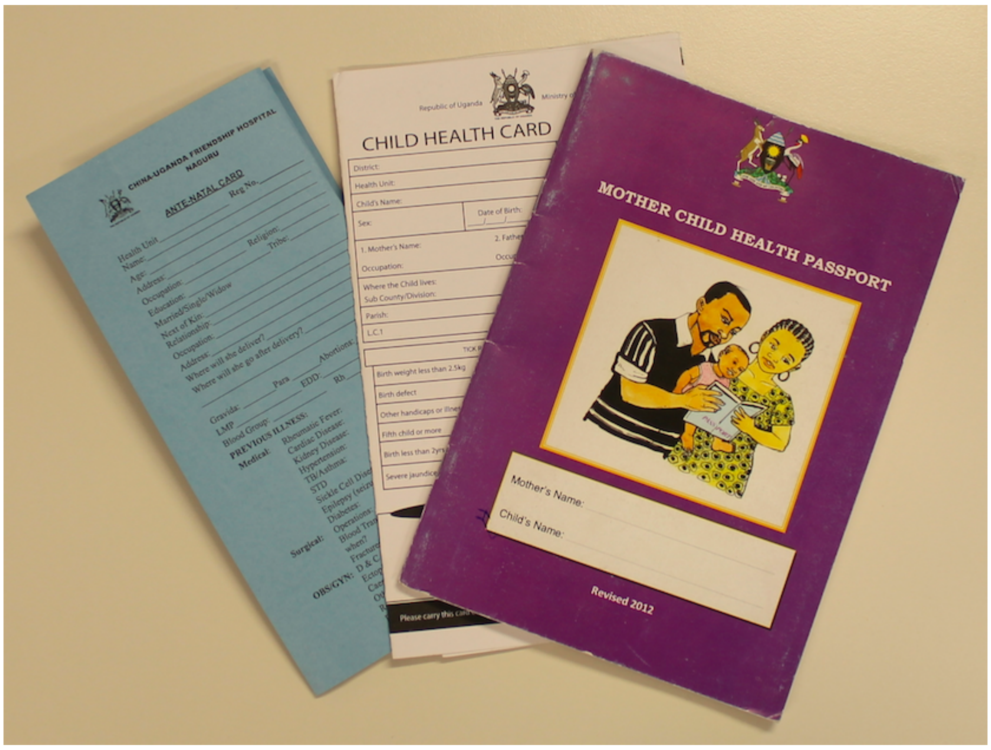


Figure 6.1: Health cards and Mother Passport from Uganda.

6.2.2 The MUJHU collaboration

The Makerere University - Johns Hopkins University Research Collaboration (MUJHU) is a collaboration that started in 1988 between Makerere University and John Hopkins University. The focus of the research is on preventing mother-to-child transmission of HIV virus during pregnancy. The research collaboration has given over 7000 families the opportunity to receive several services like immunization, family planning and medical care and treatment through letting them participate in the studies and programs. The research Centre of MUJHU is located at the Mulago Hospital in Kampala, Uganda and the aim of the Centre is to investigate topics like risk factors for HIV transmission during pregnancy and the effect HIV has on the pregnancy and the child. (*MUJHU Research Collaboration* 2015).

6.2.3 Prevention of Mother-To-Child Transmission

The project followed in this thesis is called PMTCT (Prevention of Mother-To-Child Transmission) and a part of it has been to evaluate the use of web-based EMR/SMS system to improve PMTCT follow-up in Uganda (Namukwaya et al. 2013).

The background for the project is that rate of women delivering in health centers are low, and there is significant migration between health facilities delivery programs. Especially for HIV-infected women and their HIV-exposed infants, these low rates of attendance are a serious problem as poor follow-up can lead to unsafe infant feeding practices, a delayed treatment and high mortality of HIV-infected infants. Another challenge is to follow-up on mothers in long-term care, as the mothers tend to go to different clinics, for example going for ANC visits in one clinic, deliver in another, and going for PNC in a third facility. The problem of using several clinics is that there is no shared data collection system, allowing sharing of data between the clinics so health workers could trace the mothers (Namukwaya et al. 2013).

An attempt to solve this problem is done by the use of the DHIS2 Tracker as the DHIS2 platform is already in use in Uganda for the management and evaluation of data reported monthly by all health facilities in the country. However, the use of EMR[1] to manage the follow-up implies infrastructure and resources that are not yet available in Uganda. There is thus a great need to develop and test simple, affordable, readily accessible and user-friendly tools to track both HIV-infected and uninfected pregnant women and their infants throughout the entire continuum of care. The implementation of mobile devices is therefore important. Making PMTCT records accessible across health care facilities will allow clinicians caring for pregnant women and their newborns to have rapid access to ANC and delivery data in order to provide optimal care and maximize retention of mother-baby pairs in long-term care (Namukwaya et al. 2013).

The overall aim of this project is to improve follow-up of pregnant women, both before and after delivery through the use of an EMR system.

6.3 The context of Uganda

It is important to understand the context where the implementation takes place, as a part of this thesis was to implement mobile devices for reporting in Uganda, a description of the context is now given.

6.3.1 Workflow at the health facilities

It is important to remember that the change of workflow for the health workers cannot be too radical (Asiimwe et al. 2011). When designing for change you need to develop on what the health workers are used to do to have a successful implementation. Therefore you have to look at the current workflow to understand how things are done, and to see if something can be changed or improved. The workflow observed is illustrated in figure 6.2; when a pregnant woman come to an antenatal clinic her health card or health passport are filled, then she waits for

[1]A collection of electronic health information about a patient or a group of patients.

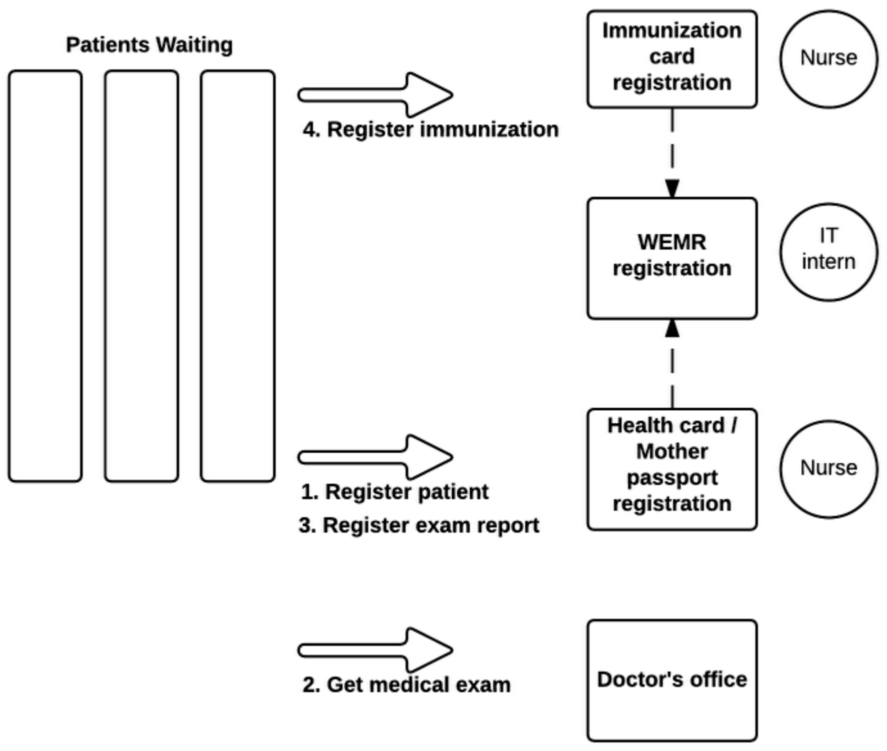


Figure 6.2: An illustration of the workflow at the health facilities.

the examination, where the rest of the information are filled. When the examination is done, the health card/passport are delivered for registration in the register and DHIS2. In figure 6.3 you can see a health worker filling in a card while the pregnant woman answers questions. Afterwards, another health worker registers the information written into the Tracker on laptop.

6.3.2 Willingness to gather data

The MUJHU project team has showed great eagerness to register data and use a lot of resources to do it properly.

The MUJHU project team has done a lot to prepare the health workers for the implementation of mobile devices in their project. The health workers interviewed had experience with the mobile as training sessions had been arranged for the health workers to get to know the devices even before they were deployed. Training of health workers to be familiar with the devices is a good way of reducing the fear of new technology. When the tablets were deployed, an IT intern spent the whole day at the facility to assist the health workers in getting more familiar with the mobile device.

IT interns are also used in the antenatal clinics visited in Kampala, since they are extremely busy. There are long lines of waiting pregnant women, so MUJHU use IT interns to help out registering information, so the health worker could take care of the pregnant women. In the long term, it is envisioned that the health workers will have time to register information in

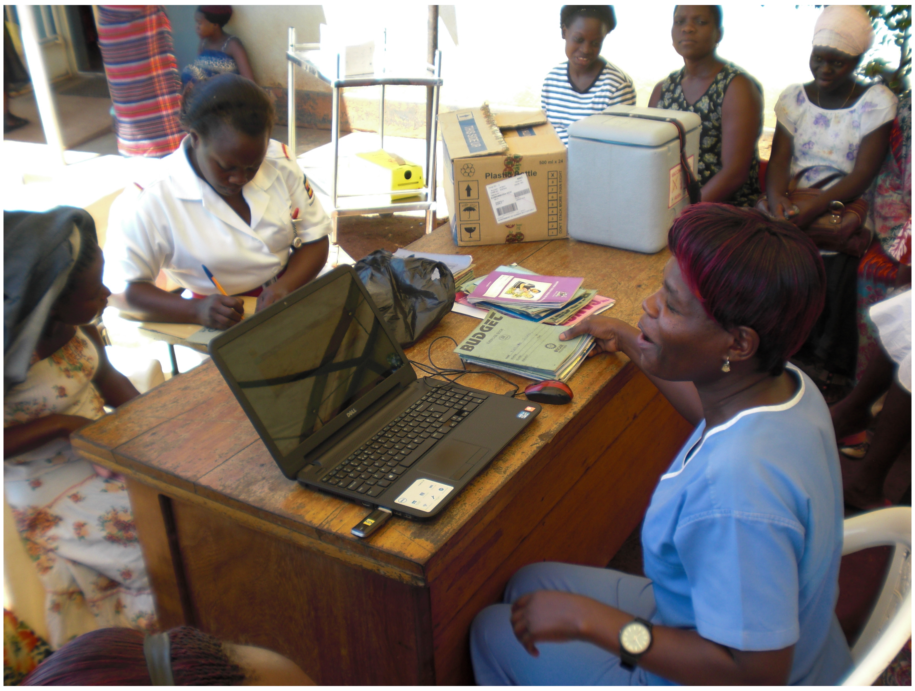


Figure 6.3: A typical situation at the clinics.

DHIS2 themselves, and replace the register book, the mother passport and the health card completely by DHIS2.

Even though the places were very busy, the health facilities always had at least one person registering both in the register and in DHIS2, showing that they are very eager to gather data. But the fact that they are dependent on an IT intern indicates that they have capacity problems, because without the interns, usually no data would have been gathered in DHIS2.

As seen in Figure 6.4 the clinics today are busy, and there are long lines to see a midwife. All the women seen here are waiting for a examination.

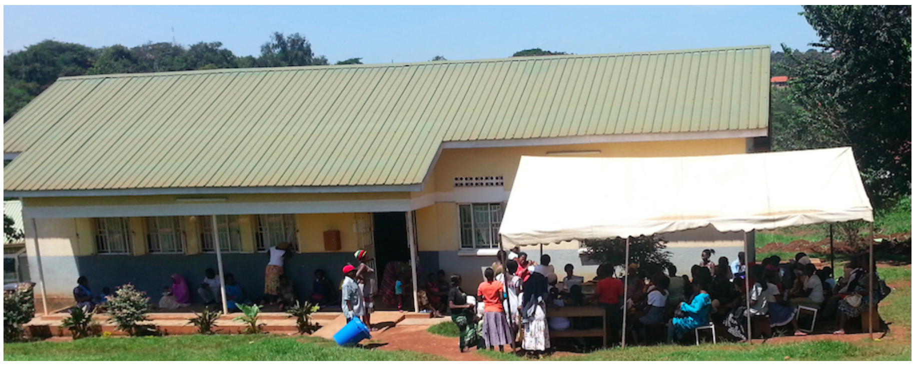


Figure 6.4: A typical clinic.

The use of IT interns to let health workers focus on health care is just a temporary solution. The health workers eventually need to take over the registration themselves.

6.3.3 Mobile devices

The project leader in MUJHU said in an interview that in the long run, they want to replace registering on paper completely with EMR. Implementation of mobile phones for reporting is a step on the way to shift from paper to computer, which enables instant access to relevant data for decision-making (Haux 2006).

The main devices for reporting in DHIS in Uganda have been laptops connected to Internet with the use of USB modems. Because of the benefits of M-health and the context of Uganda, the project followed was interested in trying to replace the laptops by reporting in DHIS2 on mobile devices.

The potential of going for M-health in Uganda is great. The simplicity, portability and affordability of mobiles can offer connectivity to people in rural areas where PCs doesn't exist (Donner 2008). Mobile devices have specifications that make it better than laptops, such as scalability, coverage, timeliness and transparency since they function in remote locations and are easy to bring any time (Freifeld et al. 2010).

The mobile device the Ugandan project team was interested in was primarily the J2ME client, as it was the most suitable device for the given context; J2ME clients are low-end phones, which is very common in Uganda. Since an Android client for reporting soon will be released, naturally it was also considered and both the project team and the health workers showed curiosity about the Android client. Tablets are already rolled out in this project, but due to the constrains of cost, usability and robustness, the project team still would prefer to implement the J2ME clients.

Prices

A vital part of implementing mobile devices for reporting in a low resource country is their cost. It is therefore important to consider the prices of the mobile devices. The prices of the mobile devices varies a lot in Uganda:

- J2ME phones cost about $70.
- Decent Android phones (Android 4.4.2 or better) cost from $130.
- Tablets cost about $450. In Uganda they had to buy from Amazon since they wanted Tab 3. In Uganda they can only get Tab 4, which goes for about $500.

6.3.4 Health workers' thoughts

Berg (2001) says that successful IS implementation requires that all interests are involved. That also includes the end-users on the floor who are going to use the IS. Thus it is important to hear what the health workers think about the mobile devices.

When talking to health workers and demonstrating the mobile clients, most of the health workers were very positive. They particularly showed big interest in the Android phone version, most of all because of the big touch screen. They would prefer to use an Android phone instead of a J2ME phone because it's easier to navigate back and forth with the touch screen instead of using the joystick of J2ME phones. When asked about mobiles as a replacement to laptops, the health workers doubted that mobiles would work better than laptop, but they thought it could be a good backup solution if they had some problems with the laptop, for example with battery power or network, as both the J2ME and Android phones will support offline functionality. Only the J2ME phone can solve problems with battery power, since the Android phone has poor battery life.

During observations of the health workers use of the tablet, it was discovered that they had to scale and navigate a lot to fill in information. To get a readable size, they had to zoom in so much they weren't able to see the attribute names and the input fields at the same time, making it easy to lose the overview and causing the registration to be a more time-consuming task, i.e. there is a need for a customized tablet client to make registration efficient or larger tablets.

6.3.5 Project leaders' thoughts

Interviews with project leaders of MUJHU revealed what is required for them to start using Android for patient tracking and how they see Android compared to other already existing devices.

The most important thing is connectivity, and with new devices like Android phones, the connectivity and coverage are better than with J2ME phones. The ability to connect to networks with weak signals is very important because it is vital that the devices work where the signals are weak. It is positive that offline functionality is being developed for the Android phones, enabling use of it where network doesn't exist. Another concern to keep in mind is the lifetime of batteries in Android phones, which are much worse than with laptops and J2ME phones. In Uganda health workers rely on long lasting batteries in the devices, which is a big weakness of most smartphones (*Smartphone battery life* 2015).

Compared to other devices, tablets still are better than Android phones, because of the already mentioned battery time, but also because of screen size. The importance of a big screen has an explanation, when using tablet, health workers use the built in web-browser to register on DHIS2, which doesn't scale good to small screens. So using the built in web-browser on Android phones is an annoying task, but with the introduction of the DHIS2 Android application the screen will scale better. Tablets are also preferred over J2ME phones, but for the short time J2ME worked, the experiences with it was very positive. If Android phones could work the same way as the J2ME phones with faster speed and less errors, it will definitely be a device to consider for reporting in DHIS2.

In the end when it comes to what devices the project leaders of MUJHU would prefer it is a question of cost and benefits. J2ME phones are clearly the cheapest alternative and Tablets have the most preferred screen size, but if Android phones will have better functionality than the already existing devices, they certainly will be considered.

6.3.6 Data use

When looking at data use in DHIS2 in Uganda, a difference between health workers and health managers was discovered.

Health managers and MoH want health workers to register information in DHIS2 for them to easier get structured data for reporting different statistics. For example health managers' interests are described in the WEMR project protocol: '*Women's and infants' data will be collected by regular health workers (midwives/counselors) as part of the routine HMIS in use by the MOH* (Namukwaya et al. 2013).'

Health workers on the other hand, when asked about their data use, were more interested in unstructured data. Examples of data used by health workers from patients' earlier visits are data used to check if they still need drugs, if they were given any drugs last time, if they have received a service or to update current information like phone number, HIV-status and similar.

> *With the register book you are not able to look at it all the time to see if the mother received this and this at the previous visits, but with the system you can easily check for example if it was not recorded in the passport, but it was recorded in the register, and the register was tracked into the system, you are able to know if the mother received the service. Without the system we might have to check three or four books if a mother received the service last time, but with the system we just have to look at the first antenatal and see if the service was given or not.*
>
> — Health worker

For example, if a pregnant woman complaints about pain in her stomach, they look at the patient record in DHIS2 to see if there are some comments about that from the last visit, to see if something has been done at an earlier stage.

'*If a mother comes with a complaint, we are able to retrieve her patient record, if she complained last time we are able to see that in DHIS2 because it has the ability to make comments about her.*'

— Health worker

This tells us there is a gap between local users. On one side you have the local health managers and actors from MoH interested in statistical data to make reports. On the other side you have the health workers more interested in data that can help them treat patients better.

6.4 Challenges of mobile implementation

The field study done in Uganda showed a need for mobile clients for patient tracking in DHIS2. During and after the field study in Uganda, an effort was made to implement J2ME phones for health workers to use.

This section will explain several challenges of implementing mobile phones for reporting in low resource contexts based on experiences from Uganda. The focus was mainly on making the J2ME phone work, so most of the challenges are derived from it, but many of the challenges are transferable to Android phones as well.

6.4.1 Technical challenges

Connectivity is a big challenge in low resource context such as Uganda, wired Internet is almost absent and in most rural places the wireless network often is poor. The lack of network is actually a problem that eventually will fade away as the mobile network in Uganda and the rest of Africa is constantly being improved, but it will take some time.

When there is lack of network, DHIS2 Tracker on laptop is useless, but a solution given by both J2ME and Android is to offer offline functionality, so health workers are able to register information regardless of connectivity, and when the signal is back, all the information registered is sent to the DHIS2 server.

Electricity is also a major issue in low resource countries and as experienced in some of the health facilities, loss of electricity is common in the project area. Most of the facilities visited had power generators, but they avoided using them as much as possible, as fuel is expensive.

Tablets, laptops and mobile phones can handle a power outage, but if it lasts for longer periods Android phones come up short because of poor battery life, and if there also is no connectivity, laptops and tablets are useless. So in situations without power and connectivity, the J2ME phone is the clear winner, as it has offline functionality and has a great battery capacity.

6.4.2 Challenges of implementing

Implementing new technologies to an ongoing project can be difficult. The MUJHU project started their collaboration with HISP in 2012 when implementing the DHIS2 Patient Tracker. In 2013, implementation of J2ME phones for patient tracking also started. The implementation process has met some problems on the way, stalling the project.

For example, the introduction of tablets took time, as they had to be ordered from Amazon.com, since the model they needed was not at the market in Uganda. When the tablet was handed out, the health facility workers had waited for them for over two months, as they had been trained to use the tablets on an earlier stage and were promised to get one.

In Uganda, the J2ME client was very close to be successfully implemented in July 2014, but due to an enhancement, the process stopped. An Interview with a project assistant in the MUJHU team about why the J2ME client wasn't rolled out yet, revealed that during testing they discovered some bugs and enhancements that needed to be fixed. An example was that they wanted to allow partial attribute search of the keyword. The bugs and enhancements were sent to the DHIS2 J2ME Patient Tracker developers. When the bugs were fixed, the updated version was sent back to the MUJHU team in Uganda.

Now they encountered a problem in Uganda. MUJHU's DHIS2 server was version 2.15, but for the bugs to be fixed, the server needed to be on version 2.16, thus the project of implementing J2ME clients stopped. The server finally was updated in mid October 2014 during the field study and testing of J2ME could start again.

When testing and implementation of the J2ME phone started again, new errors and bugs that wasn't present in the old version occurred. This issue shows that code can have an expiration date; things you might think work, could appear to not work after all when time passes and things changes. Changes in the data model make it difficult to maintain features, and this is also the case with the J2ME Tracker. Changes in the data model require hacking of the database to change the old values. Changes on the data model also required changes on the mobile clients.

The development of DHIS2 is a big project where a lot of changes happen all the time, and if a module, such as J2ME Patient Tracker, stays unused for a while, changes done can affect the unused module. In chapter 7 further details on how the process of implementing the J2ME phones are given.

The greatest problem of implementing the J2ME client has probably been the lack of priority. From the developers' side J2ME clients have had low priority lately due to its old technology. The developers' focus has instead been on the Android clients, since they are the future.

In addition the communication between the developers in Oslo and the users in Uganda was a problem. HISP Oslo took it for granted that the J2ME client worked since they didn't hear anything about it. But the reason was that the newest J2ME client wasn't tested due to the server build problem. This misunderstanding proves lack of communication.

Lack of prioritization and communication can be due to lack of ownership. With a large organization such as HISP, it can be difficult to know who's responsible for what, such as follow-up, bug-fixing, bug-reporting and so on. For HISP it is difficult to follow up on every country and project.

Chapter 7

The mediator role

This chapter describes the mediator role in the development and implementation of the J2ME Patient Tracker. Being a mediator gave valuable experiences of how distributed software development and participatory design takes place.

To implement the J2ME Tracker in Uganda has been a time-consuming process. The implementation has been supported by the DHIS2 mobile development team, which is distributed, with actors in Norway and Philippines. During the stay in Uganda, an effort was made to speed up the implementation by being a mediator. The role of being a mediator is illustrated in figure 7.1. The different actors are Uganda (local), Norway (global) and Philippines (outsourced) and the involvement in this project was to mediate between these three actors.

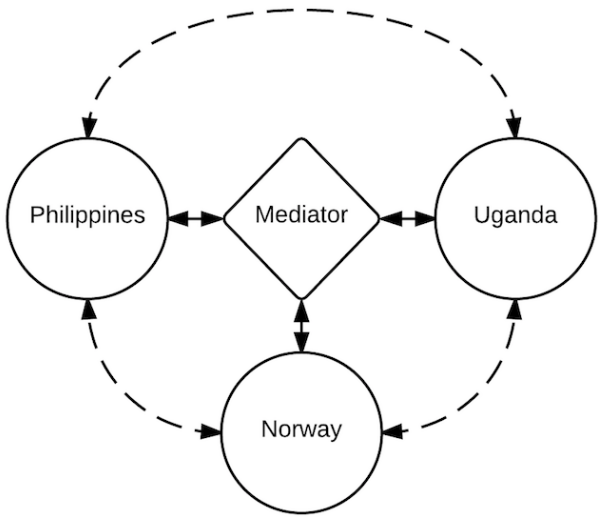


Figure 7.1: The role of being a mediator.

The implementation was done in close collaboration with the developers located in the Philippines, the project team in Uganda and the global team in Norway. This collaboration also resulted in interesting findings on how the communication between the outsourced, global and especially local context works and why it takes time. The collaboration with the Philippine developers was an important part of implementing the J2ME client in Uganda.

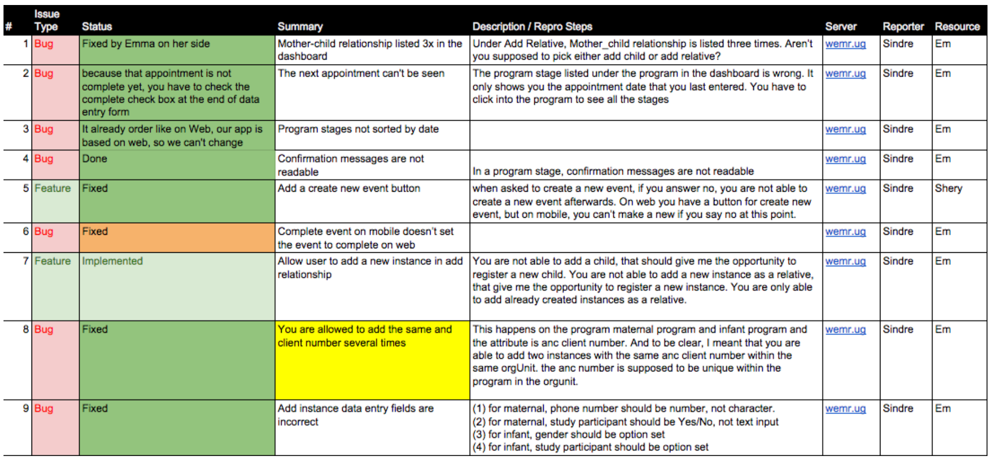
The role as a mediator was very important for the different actors to effectively communicate with each other and included four tasks:

- Testing functionality of the J2ME client to make sure everything worked.
- Reporting bugs found to developers and provide descriptions when needed.
- Discuss with the local team and sometimes the global team on what was required and needed.
- Testing functionality of the Android client.

7.1 Process

This section gives a description of how the process of implementing J2ME clients in Uganda has been.

The communication between the project team in Uganda and the developers in Philippines was mostly through Skype calls, but sometimes e-mails were used. The information flow worked as follows: *a*) a bug or issue was discovered through testing; *b*) the team leader reported the bug to the developers; *c*) and the developers added the bug reported to a sheet as seen in figure 7.2.



#	Issue Type	Status	Summary	Description / Repro Steps	Server	Reporter	Resource
1	Bug	Fixed by Emma on her side	Mother-child relationship listed 3x in the dashboard	Under Add Relative, Mother_child relationship is listed three times. Aren't you supposed to pick either add child or add relative?	wemr.ug	Sindre	Em
2	Bug	because that appointment is not complete yet, you have to check the complete check box at the end of data entry form	The next appointment can't be seen	The program stage listed under the program in the dashboard is wrong. It only shows you the appointment date that you last entered. You have to click into the program to see all the stages	wemr.ug	Sindre	Em
3	Bug	It already order like on Web, our app is based on web, so we can't change	Program stages not sorted by date		wemr.ug	Sindre	Em
4	Bug	Done	Confirmation messages are not readable	In a program stage, confirmation messages are not readable	wemr.ug	Sindre	Em
5	Feature	Fixed	Add a create new event button	when asked to create a new event, if you answer no, you are not able to create a new event afterwards. On web you have a button for create new event, but on mobile, you can't make a new if you say no at this point.	wemr.ug	Sindre	Shery
6	Bug	Fixed	Complete event on mobile doesn't set the event to complete on web		wemr.ug	Sindre	Em
7	Feature	Implemented	Allow user to add a new instance in add relationship	You are not able to add a child, that should give me the opportunity to register a new child. You are not able to add a new instance as a relative, that give me the opportunity to register a new instance. You are only able to add already created instances as a relative.	wemr.ug	Sindre	Em
8	Bug	Fixed	You are allowed to add the same and client number several times	This happens on the program maternal program and infant program and the attribute is anc client number. And to be clear, I meant that you are able to add two instances with the same anc client number within the same orgUnit. the anc number is supposed to be unique within the program in the orgunit.	wemr.ug	Sindre	Em
9	Bug	Fixed	Add instance data entry fields are incorrect	(1) for maternal, phone number should be number, not character. (2) for maternal, study participant should be Yes/No, not text input (3) for infant, gender should be option set (4) for infant, study participant should be option set	wemr.ug	Sindre	Em

Figure 7.2: A part of a bug spreadsheet.

When the bugs were fixed, a new version of the application was sent to the team leader. Since the team leader in Uganda often had a lot of other task to do, it happened that it took some time before the new build was tested and feedback on it was given back to the developers. This also made the process slow and time consuming. Another factor making the process slow was the bugs involving change on server side, which required the server to be built. A build takes around 40 minutes. This again required the local team to update their server to the newest version to see the fixes, and this update also was time-consuming.

7.2 Being a mediator

This section will describe the work done when trying to speed up this implementation process by being a mediator. The process is illustrated in figure 7.3: First the features of the mobile clients were tested. Then bugs and enhancements were reported to the developers. Then, if some bugs or enhancements were unclear, they were clarified. The next step was for developers to fix bugs and enhancement, and when it was fixed, the new version was ready for further testing.

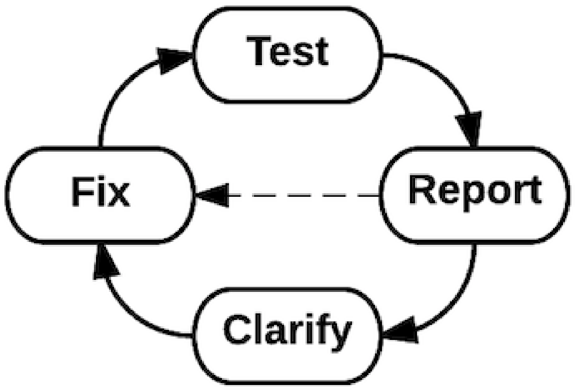


Figure 7.3: The implementation process.

The process was done in two iterations in the period from mid October 2014 to mid January 2015, the first took place in Uganda, the second in Norway. Also, one iteration was done with the Android client while in Uganda. The greatest finding in this process is how the communication between the actors works. The task of implementing the J2ME client in this project has been ongoing since 2013, and since it hasn't been successfully implemented yet, it is clear that something could be improved in the process.

In July 2014, the MUJHU team came very close to successfully implement the J2ME client, but an enhancement was needed: they wanted the ability to search with partial key values and display attributes containing that string. Until now the search key was required to exactly the same as the attribute value. This was a reasonable enhancement and the functionality was added. The new feature required the team in Uganda to upgrade their build version of DHIS2 server. This update wasn't done because they had some issues related to the Tracker module, which led to a major delay and the project stopped.

7.2.1 First iteration (mid October 2014 - November 2014)

The DHIS2 server was finally upgraded from 2.15 to 2.16 on 16th of October and on 20th of October a Nokia 206 was borrowed with the latest version of the J2ME Tracker application installed. Now a new cycle of the implementation process of the J2ME phones could start.

Through *testing*, several issues and bugs were discovered, and they were described and listed in a Word document. In parallel the MUJHU team leader also tested the client, and the bugs found were received on e-mail. Most of the bugs found were the same, but those that were different, were added to the list. A list of 15 issues then was *reported* to the mobile team on e-mail on 21st of October.

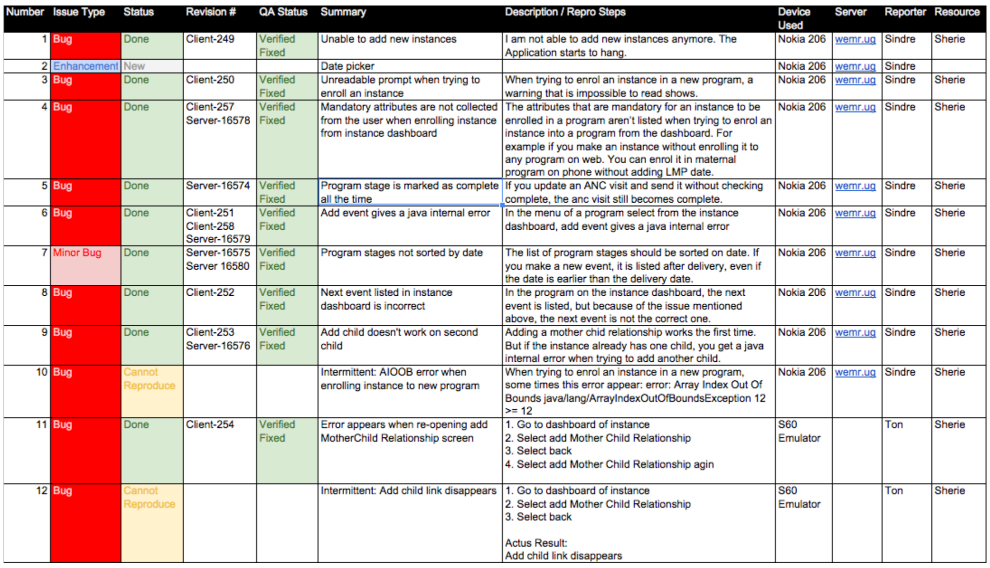
After receiving the bugs, the developers needed *clarifications* and asked for better descriptions and repro steps of some bugs. Also a spreadsheet containing the bugs as seen in figure 7.2 was shared, to ease tracking of bugs. Since the access to the spreadsheet was read-only, the explanations had to be sent by e-mail. The developers also asked for a copy of the database used by MUJHU for faster *fixes* of the bugs. The database was provided to them on e-mail. After some communication back and forth, a new version of the J2ME Tracker was ready on 6th of November. Further *testing* and *reporting* by the MUJHU team leader lead to new *fixes* and yet another release on 18th of November. At this point the stay in Uganda had ended, and the J2ME client borrowed was delivered, so further *testing* was delayed.

7.2.2 Second iteration (December 2014 - mid January 2015)

When back in Norway, a new iteration of the implementation process was ready to be done since a new version of the J2ME application had been developed.

Before testing of the latest version could begin, a J2ME client had to be bought. It arrived on 22nd of November, and *testing* of functionality started again. While testing, new issues occurred and a set of 10 bugs were described and *reported* to the developers. Even though the version had been out for a couple of days, the MUJHU team leader in Uganda hadn't discovered the issues either, as the person was out traveling. Again a database dump was sent to the developers for faster bug *fixes*. A new spreadsheet was shared on 3rd of December, and this time with full access, so bugs could be written directly into the sheet. The new spreadsheet can be seen in figure 7.4.

Now the development had been ongoing for a while and it had been a time-consuming task. Members from the global mobile team saw it necessary to participate.



Number	Issue Type	Status	Revision #	QA Status	Summary	Description / Repro Steps	Device Used	Server	Reporter	Resource
1	Bug	Done	Client-249	Verified Fixed	Unable to add new instances	I am not able to add new instances anymore. The Application starts to hang.	Nokia 206	wemr.ug	Sindre	Sherie
2	Enhancement	New			Date picker		Nokia 206	wemr.ug	Sindre	
3	Bug	Done	Client-250	Verified Fixed	Unreadable prompt when trying to enroll an instance	When trying to enrol an instance in a new program, a warning that is impossible to read shows.	Nokia 206	wemr.ug	Sindre	Sherie
4	Bug	Done	Client-257 Server-16578	Verified Fixed	Mandatory attributes are not collected from the user when enrolling instance from instance dashboard	The attributes that are mandatory for an instance to be enrolled in a program aren't listed when trying to enrol an instance into a program from the dashboard. For example if you make an instance without enrolling it to any program on web. You can enrol it in maternal program on phone without adding LMP date.	Nokia 206	wemr.ug	Sindre	Sherie
5	Bug	Done	Server-16574	Verified Fixed	Program stage is marked as complete all the time	If you update an ANC visit and send it without checking complete, the anc visit still becomes complete.	Nokia 206	wemr.ug	Sindre	Sherie
6	Bug	Done	Client-251 Client-258 Server-16579	Verified Fixed	Add event gives a java internal error	In the menu of a program select from the instance dashboard, add event gives a java internal error	Nokia 206	wemr.ug	Sindre	Sherie
7	Minor Bug	Done	Server-16575 Server 16580	Verified Fixed	Program stages not sorted by date	The list of program stages should be sorted on date. If you make a new event, it is listed after delivery, even if the date is earlier than the delivery date.	Nokia 206	wemr.ug	Sindre	Sherie
8	Bug	Done	Client-252	Verified Fixed	Next event listed in instance dashboard is incorrect	In the program on the instance dashboard, the next event is listed, but because of the issue mentioned above, the next event is not the correct one.	Nokia 206	wemr.ug	Sindre	Sherie
9	Bug	Done	Client-253 Server-16576	Verified Fixed	Add child doesn't work on second child	Adding a mother chid relationship works the first time. But if the instance already has one child, you get a java internal error when trying to add another child.	Nokia 206	wemr.ug	Sindre	Sherie
10	Bug	Cannot Reproduce			Intermittent: AIOOB error when enrolling instance to new program	When trying to enrol an instance in a new program, some times this error appear: error: Array Index Out Of Bounds java/lang/ArrayIndexOutOfBoundsException 12 >= 12	Nokia 206	wemr.ug	Sindre	Sherie
11	Bug	Done	Client-254	Verified Fixed	Error appears when re-opening add MotherChild Relationship screen	1. Go to dashboard of instance 2. Select add Mother Child Relationship 3. Select back 4. Select add Mother Child Relationship agin	S60 Emulator		Ton	Sherie
12	Bug	Cannot Reproduce			Intermittent: Add child link disappears	1. Go to dashboard of instance 2. Select add Mother Child Relationship 3. Select back Actus Result: Add child link disappears	S60 Emulator		Ton	Sherie

Figure 7.4: Another spreadsheet.

'I just saw the J2ME client now and it was kind of depressing. I think [Mobile team leader] needs to be more hands-on on this project. The bugs need to be fixed asap.'

— Mobile coordinator

Their involvement speeded up the process. Skype calls and test sessions were arranged with developers and the MUJHU team where bugs and new features were discussed. In this period, 4th-23rd of December, 3 Skype calls were done, 8 new J2ME Tracker versions were released, and it was finally ready to be rolled out.

On January 8th, the team manager in Uganda discovered four issues and reported them to the developers on e-mail. On January 14th, the newest version was released, the MUJHU team was satisfied and J2ME clients have been distributed to facilities, and training has started.

Thank you so much. I have been doing some testing on the J2ME. The issues that we had shared have been sorted. Today, we distributed the phones to two facilities, giving a one-on-one training. They like it, it just takes a longer time than on the tablet. Fingers crossed they will be using it on their own.

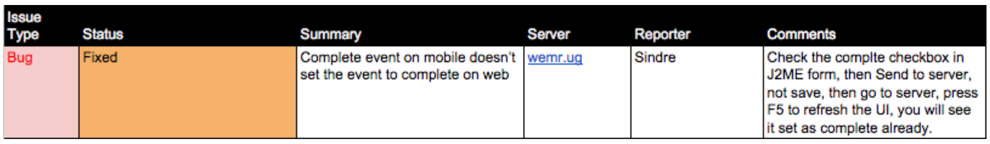
— MUJHU project leader

7.3 Working with Android

One round of the implementation process was also conducted with the Android Patient Tracker. Two of the bugs were listed in a spreadsheet and taken care of in the next version. But as it was more urgent to make the J2ME client work and the Android Patient Tracker still was an early beta, it wasn't prioritized. It is worth noting, the bug reporting process was similar to the J2ME bug reporting. A spreadsheet was used to add, update and explain bugs. Also worth noting is the fact that the server used in Uganda was too old for the Android app so testing had to be done on the DHIS2 test-server. To test the Android Patient Tracker, DHIS2 version 2.17 was required, while the MUJHU server had version 2.16. This requirement was another reason why working on the Android Tracker wasn't prioritized, as MUJHU's DHIS2 server was incompatible with the Android Tracker.

7.4 Tracking a bug

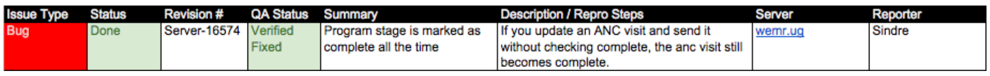
The bug-fixing process was time-consuming, and to illustrate that, the life of a bug found while testing will be followed. The bug described is inconsistency when completing a program stage. When completing the stage on the J2ME client, the stage wasn't complete on the server. This bug was discovered 20th October, and sent to the developers the 21st. As seen in figure 7.5, it was registered in the bug-fix spreadsheet. The developers were unable to reproduce the bug, so it was marked as fixed the 23rd and a new J2ME client version was received by mail 6th November. Unit testing by the developers revealed that the bug actually exists, due to a bug server side. The bug was fixed and a new J2ME version was sent 18th November, and this time the bug was fixed for real.



Issue Type	Status	Summary	Server	Reporter	Comments
Bug	Fixed	Complete event on mobile doesn't set the event to complete on web	wemr.ug	Sindre	Check the complte checkbox in J2ME form, then Send to server, not save, then go to server, press F5 to refresh the UI, you will see it set as complete already.

Figure 7.5: First iteration of bug.

When testing the new version, to see if all bugs were fixed and there weren't any new ones, it was on 27th November discovered that there still was an issue with the completion of a program stage. This time the stage was completed regardless of you deciding to complete it or not when registering information about a patient in a program stage. This bug was sent on e-mail to the developers, and registered in a bug-fix spreadsheet 2nd December, as seen in figure 7.6. The bug was fixed 5th December, and a new J2ME version was sent by e-mail 6th December. This time the completion functionality worked as it is supposed to.



Issue Type	Status	Revision #	QA Status	Summary	Description / Repro Steps	Server	Reporter
Bug	Done	Server-16574	Verified Fixed	Program stage is marked as complete all the time	If you update an ANC visit and send it without checking complete, the anc visit still becomes complete.	wemr.ug	Sindre

Figure 7.6: Second iteration of bug.

7.5 Reflection over the role as a mediator

Being a mediator is a demanding task, and as a master student without prior knowledge working in large distributed projects, it can be difficult. From the experiences gained by taking part in the implementation process of the J2ME Tracker in Uganda, an important question needs to be asked.

Was it necessary to send a master student with very limited experience to Uganda, to work as a mediator? The greatest achievement made, was helping out in making things happen, but could this maybe be done without the mediator? Listed are reasons why the mediator role in this project made a difference, but also reasons why it might not was needed:

Communicating The bugs found when testing had to be reported to the developers, and when the MUJHU team leader found bugs oneself, the descriptions were given to the mediator to be passed on to the developers. Also when the developers needed further explanations, the mediator was asked to get them from the MUJHU team leader. Instead of communicating through the mediator, they could have communicated directly with each other to save time. Adding a third part makes communication more difficult, especially when the parts are located in three different time zones. Further investigation of the project history, revealed that communication between developers and end-users had been done without the involvement of the mediator at earlier stages.

Reporting bugs When reporting bugs it was an inconsistency in tools used for reporting. Tools used were two different spreadsheets, Word documents, e-mails and Skype calls. This was a result of no introduction to how to report bugs when testing. Some times the bugs were described directly in e-mails, others in attachments to e-mails or during Skype calls. All the bugs were registered in the spreadsheets in the end. If a stricter regime for bug reporting had been used, it could be improved. High quality bug reporting also leads to quicker and cheaper fixes.

Providing database When trying to fix the bugs, developers requested a local database dump for quicker fixes. The mediator lacked required rights to make a copy of the database and had to find out who had those rights. If the MUJHU team leader in Uganda had been asked instead, the dump could have been provided faster, as the leader has a better overview of who can make a copy.

Updating server A similar issue was when the local server needed an update for the bug-fixes to work. The mediator lacked rights to upgrade the server, and again had to find the correct people.

Influencing management Back in Oslo the effort made to make the J2ME client work had taken a lot of time. Having a meeting with the mobile project leaders in Oslo made the implementation process gain speed.

> So what if the trip to Uganda never happened, how fast could things have gone, as it seems like influencing the management to keep the J2ME project on the agenda in Oslo was the most effective way to make things done?

Looking at the points mentioned, one can ask if the mediator role actually was needed in this situation. The answer is twofold. First one could say that the mediator was unnecessary as it made things more complex by including a third part, and all the points mentioned had been done without a mediator before.

On the other hand, the mediator role was necessary for making things happen. The issues in Uganda were brought to light because of the field study. Working closely with the mobile team leaders in Oslo had an impact, and being a mediator in only one project is an advantage as no other projects gets attention.

Chapter 8

The global case

In this chapter the focus changed from looking at one use case in Uganda to look at the global development. This is done by interviewing actors in the global HISP team located in Oslo and by comparing best practices of DSD and DPD with the development and implementation of DHIS2. These interviews and comparisons provide valuable findings for the discussion.

8.1 HISP Oslo

Interviews of different roles in HISP Oslo was conducted to get better understanding of how they see the process of implementation and software development in a distributed context. All statements, quotes and contents in this section are derived from the interviews conducted.

8.1.1 Approach

The approach refers to how the different roles approach a new case, context or situation when working with DHIS. In HISP, there is no defined strategy of how to approach a new case, as the countries and contexts often are very different, but the overall goal is about making DHIS2 work in a context or project.

'Why is there no such very defined strategy where we do this and this every time we go to a new place? It depends on what you are faced with.'

— Core developer

In some cases the countries already use DHIS2, in other cases the countries start from scratch. Regardless of the approaches used in the different contexts, two actions are central.

First it is gathering of requirements. When the implementers gather requirements they work as mediators communicating between developers and users of DHIS2 by going to the sites and meet the users face-to-face.

'It's often working in between the developers and the clients.'

— Implementer A

One approach is to explain the users about the possibilities and limitations of DHIS2 in their context and how DHIS2 works.

'Tell them what's possible, what's easy, what's difficult, what's expensive, what's risky.'

— Implementer A

The goal of the development and implementation is to make the local requirements as generic as possible, so they can be used in other contexts. The system is made generic by taking one step back from the initial requirement, while still making sure the end-users can go the last step by themselves.

'In HISP there is a one-to-many relationship, so they can't include each and every requirement from all the different contexts. DHIS is a tool to achieve what they want, not a system to achieve it.'

— Core developer

The second action central in implementation is training of implementation teams on national level and end-users so they are able to handle the system on their own.

'The most important thing for us is that a country is able to do as much as possible by themselves.'

— HISP project coordinator

The approach of building capacity has changed over the years. Earlier implementation support was only from Oslo. Because of the rapid growth of the HISP network they had to think differently. Instead of supporting every country, the Oslo team started supporting nodes[1] that support nearby countries. An important part of supporting the nodes is to look at the correlation between countries so they can learn from and help each other. Being an implementer isn't so much about spending several months in one country, as it used to be, now it is more about helping countries on a higher level; to see what works and what don't.

'The way that I've been working is to some extent based on the HISP approach of building capacity. I don't want to establish long-term dependent relationships.'

— Implementer A

As DHIS now is spread out in a lot of countries and contexts a strategy of training is to have as little contact with end-users as possible, this to avoid becoming too attached to one context. When working with 20-30 countries, you don't have time to be involved with everyone. Instead it is about making sure the country itself is able to build capacity by training of trainers. With so many different countries to work with, it's impossible to spend quality time with the end-users; the focus is instead on enabling the country to do that by themselves.

[1] A node is an organization or a group of countries collaborating. An example is the East African node with countries such as Kenya, Uganda and Rwanda among others.

8.1.2 Requirements

A vital part of the distributed software development is gathering and handling requirements. Thus how new requirements of DHIS2 are handled in HISP has been looked at.

'You can't sit by yourself and assume what users need; you have to be there, with users, participatory design.'

— HISP project coordinator

When you make generic software, you can't sit in a laboratory and assume what people need. HISP has experienced that the only way to make things work is to spend time in the countries together with the users. That is how they made DHIS2 work. You have to go to a country and solve a concrete problem and constantly keep in mind that this should be possible for others to use. A lot of systems deliver exactly what you want and aren't editable. Instead, DHIS2 is a skeleton or a toolbox that supports data gathering, processing, calculations and visualization.

New requirements usually occur by themselves when developers, implementers and users work with the software. There is no active process of looking explicitly for requirements as they occur either way. When you spend a longer period of time in a country, you get a lot of ideas and suggestions for improvements.

A strategy used when developing new features from the requirements into DHIS is to release often. HISP rather release often with something half-done and get users to test it and give them feedback on the feature and continue to work on it. Releasing every quarter is a good strategy, because if something goes wrong, three months aren't too long to wait for the feature to be fixed.

Users often want DHIS2 to function the same way their former bespoke software did, but that is not always possible.

'For example they say: We want this feature, can you do that? Yes, we can do it, but it is going to cost time, and time equals money.'

— Implementer A

Because of the way things are done in HISP, you don't always get exactly what you want. It is important to make sure the users understand that.

'That's the way it works: You get a lot for free, but if you want something specific, you either have to do it yourself or pay.'

— Implementer A

One approach is to convince the users that they don't need what they think they need, that what they need actually already exist in DHIS2, and then it is more about making their requirements fit to the existing DHIS2. Other times users' requirements don't fit the existing data model in DHIS2, then the approach becomes lowering their expectations and trying to give them something similar instead.

Users don't always understand what a requirement is or how to phrase it in a way that will make sense for developers. Therefore a big part of being an implementer is to translate and filter users' requirements to something meaningful for the developers, but also to make sure that users understand that things they believe is easy to get, can be really complex to achieve.

The requirements are usually shared with the developers through mail, or by making a blueprint[2], sometimes they get implemented, sometimes they don't, it all depends on the developers and it is important to be aware of the fact that HISP is not like a corporate organization, where you tell what you need, and then it gets implemented.

'From my experience, sometimes from distance it's difficult or they do not prioritize, because with the DHIS there are so many messages.'

— Implementer B

Mediating requirements on mail isn't always effective and it happens that implementers don't get answers from developers. Having face-to-face meetings with the project coordinator and the developers has proved to be a more effective approach of getting the requirements implemented.

A user's requirement isn't always prioritized. Developers have to decide on what requirements are prioritized. If the requirement is not critical enough, it is likely to be less prioritized. When releasing every quarter, the developers can't be sure that every country gets what it wants. The developers try to use the network for what it is worth to be in sync with what the countries want; it is important that DHIS2 maintains its relevance. But there will always be some that are disappointed, because some requirements are prioritized over others. Developers have their own plans for the builds and can't take in all requirements.

'We can't bring each and every little thing into the global, we have to make sure that it is in the interest of the global community.'

— Core developer

When a requirement is received, it is analyzed and discussed by the developers. Then decisions are made on whether the requirement should be implemented or not. If the decision is to implement the requirement the approach is to take one step back from the requirement.

'If the users say this is my requirements, I want you to do stuff like this, we just go half way and when that halfway is completed we try to make sure that the users can go the last steps themselves.'

— Core developer

By doing that, the developers can assure that the feature becomes generic and can be used in other cases with similar requirements.

[2]A blueprint is a specification of a requirement. It describes the feature and tracks its implementation status and who's involved.

'In the back of your head you do this thinking and crosschecking, but otherwise we don't. We don't even have a requirement document.'

— Core developer

A thing worth mentioning is that there is no structured way of looking at earlier requirements to see how they are affected by the implementation of new requirements other than what developers can remember themselves.

An issue is when larger organizations fund a lot of the development. Then they will have more influence on the requirements and it can almost look like they have taken over the whole development department. As DHIS grows, more organizations will come and pay and balancing between what requirements to prioritize will be difficult.

8.1.3 Communication

For successful participation and development in a distributed setting communication is important. Communication between developers, coordinators and users is a big challenge. To cope with this challenge, HISP has implementers working as mediators between the local and global levels.

It is easier for countries to use DHIS2 today, as most of the problems that could occur are solved in earlier implementations, but new requirements will always occur and today it is more likely that a requirement occur without an implementer present.

'Now we are very much dependent on someone sending us the requirements.'

— HISP project coordinator

The communication channels in HISP vary a lot; in some countries local end-users use the built-in messaging system in DHIS2. In other countries implementation teams send emails directly to the mediators. In other countries the locals get to know the developers and communicate directly with them. Sometimes communication happens through Google docs, where requirements are written and the developers give comments on them. For reporting of requirements HISP also has a dev. list.

The problem with the dev. list is that it can take time to get an answer, which can be frustrating, and eventually one will give up and sends an email directly to the responsible developer instead. Also when sending emails it can take time to get an answer, so occasionally mediators follow up on requirements personally by consulting the developers directly.

In a case followed by one of the implementers interviewed the difficulties of communication are described: After finding a requirement, the mediator communicated it to the project coordinator, but didn't get an answer.

'Looking at [HISP Project Coordinator], he's a human being, he has a lot to do, he's a coordinator, and he receives a lot of mails.'

— Implementer B

Your mail isn't always prioritized, since there are many emails from many places, and the developers and coordinators haven't time to help everyone. Eventually the implementer consulted the project coordinator in person and a meeting was agreed on, where developers, implementers, users, local project teams and the project coordinator could communicate and discuss needs and requirements. Meetings are more effective, but the project coordinator and the developers don't have time to attend all meetings.

Another approach is to look at other DHIS2 cases in the country and communicate and collaborate with them to get the help needed. But to collaborate locally isn't always a solution; you need to share the same interests and practices.

'If you are focusing on patient and I am focusing on patient, it is easier to collaborate and exchange skills and help each other. When you are focusing on something else like aggregation, we can't help each other because of the different use cases.'

— Implementer B

HISP tries to enable communication by arranging DHIS academies (*DHIS Academies* 2015), where the countries in a node can share experiences and help each other out. In these academies feedback-sessions are held to hear what the users want in next releases, what is not working, what is needed and what is critical.

'We try to listen to what people want and need through feedback sessions were countries meet and discuss at DHIS academies.'

— HISP project coordinator

Another way of enabling communication is to have contracts with the nodes, where HISP Oslo pays for local people to help them out. For example in HISP Uganda, people get paid to work with and test out new things in relation to DHIS Tracker and DHIS Mobile. By having these contracts HISP is able to come closer to the users and the users are able to participate in the innovation.

8.1.4 Coordination and management

Large organizations such as HISP should have some kind of management and coordination. Historically HISP has been an extreme agile group, which does what is needed to make things happen. But as the project has grown bigger the recent years, HISP has been better at having clearer roles.

'We are in an academic environment and this is Open Source. We don't really have that kind of team arrangement where decisions go through a manager.'

— Core developer

HISP is an organization with a very flat structure without a clear hierarchy. As a research project this is an advantage as it is easy for people to take part in different projects unlike more strict organizations where employees work on one thing 100% of the time and managers control everything. A disadvantage of this structure is its tendency to become a bit disorganized in the long run when more actors join the organization.

Right now HISP has one person working as a project coordinator, but that's not enough anymore. As more countries join, more coordination is needed. For example if new NGOs or ministries want to do something, they don't necessarily know about other projects that are similar or that a project coordinator who can help them to get in touch with similar projects exist. Another example is when a use case in a country is nothing similar to other uses of DHIS2 nearby. Then they become dependent on the global DHIS2 team to get somewhere.

'It probably happens relatively often that someone does something someone else has already done, but it can be difficult to know about this, since it's only a few that have overview'

— Implementer C

Thus an area of focus now is the improve the local HISP-nodes to facilitate coordination of projects nearby, making it easier to get in touch with other projects that do something similar to what you are looking for. This is also the idea behind DHIS Academies; for different countries to meet and share experiences with each other to avoid situations where a country has done exactly what you are trying to do, so you don't have to reinvent the wheel.

The DHIS2 project has grown big in a very short time. More clear management could be needed.

'It's really much bigger than probably anybody could have ever dreamed about in terms of the impact.'

— Implementer A

Until now this project has worked quite well with the process of working with developers closely with people on the ground, but the demand is big now, so another approach could be needed.

'I think that there is two different paths necessary and there is recognition of it in the HISP network.'

— Implementer A

The project has become something more than a research project so HISP should maybe split in two. One part with a more corporate approach, focusing on development and implementation, since not all clients care about the research aspects and just want things to get done. And the other part focusing on the research, where people write and go to the countries and gather requirements and information, and also publishing their theses and academic papers.

It could help if there were more defined managers or coordinators having certain responsibilities: Person A for mobile, person B for Tracker and so on. Their jobs could then be to link projects together and help out with coordination and communication since it is important to be able to collaborate with other projects across borders. It could be that the other projects in-country are totally different compared to what you are trying to do.

'Someone should have a specific role who could tell you what you are doing now are very similar to what's being done in Country X.'

— Implementer B

The Oslo node tries to be this coordinating role. Oslo is a special node by not having a local implementation of DHIS2.

'Oslo has the role as matchmaker in the network and help people to see the relations.'

— HISP project coordinator

It is Oslo's job to make sure there is communication between the other nodes, but they haven't had the resources to do it well. In HISP an area of focus is to strengthen this matchmaker role. By having more people working with implementation support, people can spend more time with the different nodes and help them communicate with other nodes and report problems to Oslo faster. Until now HISP has been very dependent on master students to give implementation support, because you need to spend a longer period of time in a country and one project coordinator can't spend several months in one country when there are 50 other countries that also need support.

8.1.5 Documentation

Good documentation can help both users and developers when implementing software. Documentation of requirements makes it clearer for developers what they need to develop, and documentation of how software works makes it easier for end-users to find answers to questions about the software.

'We also lack much documentation.'

— HISP project coordinator

The documentation process in HISP isn't too good. There are no strict methods for how documentation is done. It's up to each individual how well things are documented.

The developers try to force users to create blueprints, in that way they can document requirements, but users rarely do that. It also happens that developers make a blueprint after the requirement is implemented to make sure it is documented. When release notes are made, the developers try to go through the blueprints. By doing that, they force themselves to make sure things are really documented. Sometimes after putting features in the system, they go back and create a blueprint. But it's not like a corporate company where everything gets documented in specific patterns.

'There are a lot of bits and pieces here and there, so to have one system for documentation could make it easier to see what's going on in other places.'

— Implementer C

HISP has a lot of resources and knowledge, but sharing of the knowledge is difficult. Proper documentation could make it easier to approach a new case and share experiences, but no one within this group want to spend time documenting.

'I can't imagine people would use time to do proper documentation.'

— Implementer C

HISP is maybe not the best organization for such strict documentation methods. When mediators or users ask for help, the standard answer is to look at the documentation, but the documentation is poor. That's unfortunate because there is a lot of knowledge about many topics spread out in the world.

'The knowledge is there with someone. But maybe not presented in a good way, if presented at all.'

— Implementer B

HISP has a 500-page manual, with information such as where to click. But documentation on a higher level, such as what are clever data sets is also needed; implementer-oriented documentation where you can learn from best practices, document good processes, good solutions, good use, good configurations, how to set up indicators and so on. A lot of these things have been taught in DHIS academies, but there is no repository where you can find these things. Since the nodes are the ones who know which solutions work best, it is important to include them in the documentation process.

In 2010 more proper documentation started and it has been a lot of improvement, but it can still get better.

'There were pockets, there was a confluence site here, some word documents there, but it was no formal documentation you might expect from a big complex piece of software.'

— Implementer A

It is important to keep in mind that HISP is a research organization, not a business, and documentation hasn't been a priority. Because of the nature of the research method, things have been done informally without a strict documentation regime.

8.2 Distributed software development

This section compares Ling's (2006) suggested strategies for DSD with experiences and findings from interviews and the field trip in Uganda to find out to what degree these strategies are followed in the development of the J2ME patient tracker and within HISP in general.

Have a clear distribution rationale

It is suggested that only well-structured projects should consider DSD and as implied by the developers and implementers interviewed, the structure of the HISP project is a bit unclear. The software developed, DHIS2, is relatively suitable for DSD as development of it is decomposable into tasks due to its modularity. To develop a module, the J2ME patient tracker, in Manila was suitable as it was a separate task with stable and well-defined requirements and decomposable into a separate task.

It is also suggested to choose a team with a common language. The common language in this team was English, and as both the implementers located in Uganda and developers located in the Philippines have sufficient skills in English, the communication wasn't a problem. Also most of the communication was by email, removing the issue of different pronunciations of English.

With developers located in Philippines and implementers in Uganda, the temporal distance is large with 5 hours difference. This is a benefit regarding follow-the-sun work. For example if the developers had a question about a bug and sent it at the end of their day, it would give implementers located in Uganda the whole day to respond to it, and the answer would be with the developers the next morning. When the developers wanted quick answers to minor questions the time difference was a disadvantage, as the overlapping working hours were few.

Clarify all understandings

When starting a development project, such as the development of the J2ME patient tracker, it is suggested to clarify all understandings to avoid misunderstandings in communication. Good documentation is suggested to reduce communication problems.

It became clear through interviews that implementers within HISP meet users and explain about possibilities and limitations to make sure they have a common understanding of goals and targets at the start of a project. But the documentation regime within HISP is poor.

Poor documentation is a problem for implementers coming in late in a project. Implementers and developers within HISP had been to Uganda to clarify all understandings, but due to poor documentation it was challenging to know what's already been agreed on, especially regarding the features of the J2ME client, since the field study was conducted at a later stage and the developers who were in Uganda had been replaced with new ones.

Efforts to clarify understandings during the field study was done through Skype meetings with developers, users and implementers on a weekly basis and a spreadsheet containing bugs and issues in the J2ME client was eventually shared, providing more understanding of the patient tracker's features.

Use cultural mediation

Use of a cultural mediator to reduce geographical and socio-cultural distances is suggested to reduce communication problems. Within HISP implementers work as mediators by being a link between users and developers. They communicate users' requirements to developers, and developers' answers to these back to the users.

For the implementers to be successful mediators, they are dependent on getting answers from the developers. Mediating requirements to developers on mail have proven to be ineffective and it also happens that the implementers don't get any answers at all. Then it becomes difficult to mediate from developers to users.

Facilitate human communication

A strategy suggested is to facilitate communication, and this is best when done face-to-face.

In HISP this is solved as already mentioned by having mediators who meet the users face-to-face to enable communication. Having face-to-face communication makes it easier to have direct communication such as Skype calls or chats at a later point, as the actors have a relation to each other. Also, one of the ideas behind DHIS Academies is to facilitate communication by enabling actors from different contexts to meet and share experiences.

On the other hand, communication of requirements can improve. When a mediator sends requirements by email to a developer or to the dev. list, it takes long time, if the mediator gets an answer at all. It has happened that mediators have consulted developers in person to get an answer; so having face-to-face meetings with the developers has proved to be more effective.

The communication channels in HISP vary a lot. For example in the implementation of the J2ME client alone, Skype-calls, Google spreadsheets, Word docs and emails were used. Communication could be facilitated if there was one standardized way of communicating.

HISP try to facilitate human communication, but they could be better at answering mediators' emails. HISP tries to solve this issue by improving the local DHIS nodes, so they can support and follow-up on countries nearby.

Manage processes

To have a project leader with full responsibility and supplement with local leaders and teams is a strategy partly followed by HISP. There are local leaders and teams, but they have no clear management due to the flat structure within HISP. Lack of management also affects coordination and control.

It was suggested to have more defined managers or coordinators on the global level with certain responsibilities to ease coordination and control of the projects.

There is a lot happening within HISP in this area. They are currently working on improving and establishing more local HISP nodes to coordinate projects nearby, making coordination and control of projects easier.

HISP is also working on strengthening the matchmaker role of HISP Oslo to help the local HISP nodes to communicate with each other.

Due to the big scale of HISP projects the recent years, a more corporate approach could be needed since not all projects are interested in the research, they only want to get things done. An interviewee suggested splitting HISP in two, one part focusing on research and the other having a more corporate approach.

Develop a sense of teamness

A suggested strategy is to develop a sense of teamness to facilitate communication and coordination.

DHIS academies form an effort of team building on the global level across contexts by letting actors from all over the world meet. On the local level teamness is facilitated by the mediators when they meet users and shape requirements together.

There is room for improvement; by ensuring timely feedback about progress, deliverables and ideas, a greater sense of teamness could be achieved, but as already mentioned the HISP network has scaled a lot in recent years and hasn't had enough resources to follow up every context at the moment, but more local DHIS nodes can enable more follow-up.

Encourage temporary collocation

Investing in periods of collocation can reduce problems in future processes by facilitating communication and coordination.

HISP does that well. The interviewees have all been visiting the local contexts they have been mediators and implementers for. Implementers involved in the project in Uganda have been there several times.

DHIS Academies is another way of temporary collocating different actors.

Develop an effective tool base

It is suggested to have a standardized tool support for configuration and change management.

In the development of DHIS, Launchpad[3] is used. Even though Launchpad offer ways of reporting bugs and requirements, implementers seldom use them. If a stricter regime of reporting was introduced and the mediators were trained in how to report correctly, coordination and control of the project could improve.

[3]Launchpad is a tool that includes components such as version control, bug tracking and tracking of new features.

8.3 Distributed participatory design

This section compares Kimaro and Titlestad's (2008), Gumm et al.'s (2006) and Obendorf et al.'s (2009) suggested strategies for DPD with experiences and findings from interviews and the field trip in Uganda to find out to what degree these strategies are followed in the development of the J2ME patient tracker and within HISP in general.

Mediated two-directional feedback

This approach is followed, as the implementers working for HISP indeed are mediators between users and developers. This mediation facilitates participation since implementers bridge the gap between users and developers by communicating with both actors.

But the implementers have experienced difficulties in who to prioritize; the different users tend to focus on different things and thus have different requirements that can be conflicting (Roland et al. Forthcoming). For example, when looking at the DHIS2 patient tracker, the health workers are interested in information to treat a patient, but leaders on higher levels are interested in statistical information. Both kinds of users have to be included in the PD to be satisfied.

There is also room for improvement when it comes to developers giving feedback to mediators. It occurs that implementers are ignored when communicating with developers due to prioritizing. The developers are few and the contexts are many so HISP doesn't have capacity to answer everyone. But as already mentioned, the establishment of local HISP nodes will reduce capacity problems.

Inter-contextual workshops

When implementers meet users to discuss and analyze requirements for future developments, it can be seen as workshops. Also by arranging DHIS Academies users and developers from different contexts can meet and share experiences. Occasionally it happens that developers, implementers and users share and discuss requirements in face-to-face interactions.

In HISP face-to-face interactions has proven to be more effective than for example having a mediator between developers and users when it comes to getting the requirements implemented.

Commented case studies

Commented case studies aren't used in HISP. The interviewees implied that the coordination of projects could be better, for example when it comes to linking similar projects for cooperation. By using commented case studies, the coordination of projects in HISP could improve. Such case studies give descriptions of cases that enable change of experience between users from different contexts. In addition, if developers comment on the cases when they make design decisions, it gives users the opportunity to get insights on why the decisions were made. This also comes in handy for users unable to participate in workshops.

The workload on developers is big enough as it is, so they probably wouldn't take time to comment on decisions in different cases. But an area of focus is to improve the documentation in HISP. That will ease coordination of projects and further make it easier to link similar projects.

Surveys

Having surveys to evaluate user satisfaction could be a good way to gather valuable data on what features are wanted, what features don't work and what features are used. By letting users of DHIS tell about these topics enables participation.

To date, surveys aren't used in HISP. With the current situation they aren't feasible either. One thing is to conduct a survey, but the results need to be analyzed and right now HISP hasn't capacity to do that. But in DHIS Academies they have feedback sessions, where the users can ask and discuss with each other and the developers on already implemented and forthcoming features in DHIS2.

User support

To give users support is an approach HISP follows. When implementers, developers and users meet, personal user support is given, but there is room for improvements when face-to-face user support isn't possible. Improving documentation routines can make it easier for users to find support and further the need for personal support can decrease. Documentation is to some degree done, but there is no strict regime in HISP, so the tools used and the quality of the documentation varies a lot.

The personal user support can improve with the establishment of more local HISP nodes, as these nodes can reduce the workload on HISP Oslo when it comes to user support. The local nodes are created to support nearby countries and use cases. By establishing the local HISP nodes. Reducing the workload on HISP Oslo can also enable them to become a better matchmaker in the HISP network. HISP Oslo is supposed be a coordinator and help the other nodes to communicate and support each other.

Participatory customization

This approach is followed by HISP as implementers train intended users simultaneously as DHIS is adapted to meet their needs. This approach increases users' experiences and skills, which enables them to contribute more in PD processes. Lately, due to the scaling of contexts, an approach has been to train trainers instead of end-users, this to avoid being too attached to one context, as implementers usually are working in several contexts.

By paying local users to work with, test out and giving feedback on new features, HISP enables more PD. When users evaluate new features, they participate more in the development.

Chapter 9

Discussion

This chapter will discuss the results and findings from the research in light of the literature and theories reviewed.

In chapter 4 it has become clear that when you develop software that is spread globally, you need to find a balance between adaption to various contexts and universal solutions (Rolland and Monteiro 2002). Figure 9.1 tries to illustrate the change of DHIS2 discussed in this thesis from a bespoke system to becoming a generic toolbox to cope with these two tensions as more use cases of DHIS2 appear.

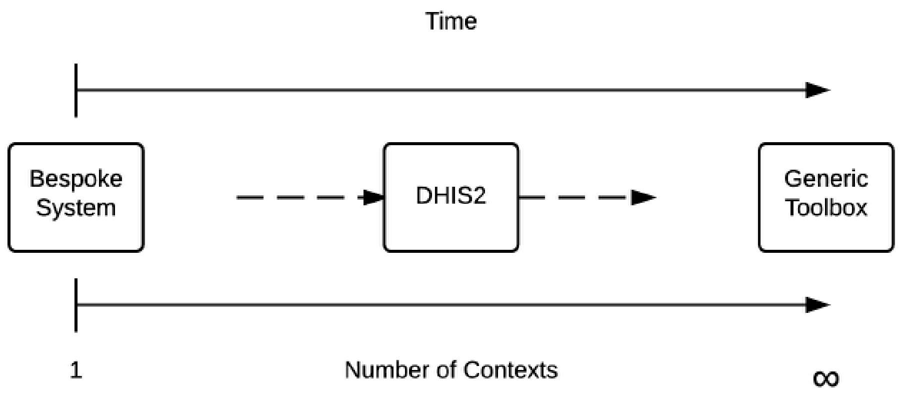


Figure 9.1: The development of DHIS2 the recent years.

'It is important for HISP that DHIS2 keeps it relevance for all users and contexts.'

— HISP project coordinator

As discussed in chapter 3 a way to keep software relevant is to involve users through participatory design. But is PD feasible with a constantly increasing number of countries and use cases? The number of use cases of DHIS2 has scaled more than anyone could have imagined the recent 10 years, and HISP as a research organization has not been prepared for this enormous scaling.

9.1 Successful participation

In chapter 3, a set of strategies for how to succeed with DSD and DPD was described. By following these best practices, one might expect that users to a large extent are able to participate in the design even though the development is distributed. In section 8.2 and 8.3 the development practices of DHIS2[1] was evaluated against these strategies. As it turns out, in the development of DHIS2, most of the suggested strategies are to a varying extent followed, but application of PD is still shallow.

As described in section 4.5, other studies have come to the same conclusion; developers make decisions on which local requirements are implemented and the majority of stakeholders end up with very little influence on the design (Shidende et al. 2014; Manda et al. 2014). This is also coinciding with the empirical findings of this thesis.

As seen in section 8.1, the actors interviewed felt that the degree of participation in the design had decreased. The main reason is due to the enormous scaling of DHIS2 deployments and countries joining HISP in the last five years. Developers receive a lot of requirements, making it difficult to follow up on everyone, and they can't implement each and every requirement into the global and generic software. Related to this is the recent trend of NGOs paying to get their requirements implemented, which can happen at the expense of the users' requirements.

As described in section 8.1, with the recent growth of contexts and use cases of DHIS2, HISP has faced some capacity problems. The developers used to work closely with the people on the ground, but that isn't possible anymore. Developers have distanced themselves more from the users, which affect the degree of PD. It becomes harder to involve the users, and both the users and the developers become very dependent on the implementers being mediators.

9.1.1 The importance of mediators

The author's role as a mediator between Uganda, Norway and the Philippines in the implementation of J2ME clients also gave indications on low PD. As described in section 4.5, Shidende and Mörtberg (2014) argue that the role mediators have in participatory design becomes more and more important. That was also evident in Uganda.

The mediator role was necessary for making things happen. The issues in Uganda were brought to light through the field study. Working closely with the mobile team leaders in Oslo had an impact on prioritization, and being a mediator in only one project was an advantage as no other projects got attention.

As described in chapter 6 and 7, further investigations of the project history, revealed that communication between developers and end-users had been done without the involvement of the mediator at earlier stages, but due to lack of prioritization from both sides, communication stopped.

[1]The evaluation was based on experiences from interviews and the field trip. Not on the development in the entire HISP network.

9.2 Evolutionary global toolbox design

In the previous section we have seen that PD is limited despite many strategies discussed in chapter 3 have been implemented to a great extent. Titlestad et al.'s (2009) approach for how to develop a generic toolbox also refers to these strategies as important. So what does that mean? Could it be that the model of how to develop software in a generic way is somewhat limited?

Titlestad et al.'s (2009) approach for evolutionary global software design in section 3.4 tries to describe how the distributed development of DHIS2 takes place. A part of the interviews discussed to what degree the actors felt that this model still is valid.

Based on these interviews discussed in section 8.1, it has become clear that the model fits better to how the development of DHIS2 was in the beginning. When HISP started developing DHIS2, they were in one country at a time. The first development was in India in 2006. All the developers went there to come close to the users and made the first release. The same thing happened in Sierra Leone in 2008 and in Kenya in 2010.

'Back then it was very much participatory design.'

— HISP project coordinator

In 2012, after the implementation in Kenya, some other African countries started to show interest. Even though there were more countries, HISP still was able to make them participate in the design. But in 2013/14 20-30 other countries joined. Then it became impossible for HISP to be involved in every country and work with the users and follow up on every requirement in the way they were used to.

'Now the model is more chaotic, with maybe forty arrows up and down.'

— HISP project coordinator

The model is valid in sense of what is being implemented, what goes in need to be made generic and usable for other cases.

But it's not only a few in-country innovations anymore, it's more and more common that non-governmental organizations contribute with innovations, but it's the same with them; what is being made has to be useful for others. Also, over time as the project has grown bigger and more and more countries and projects join, the core developers have become less involved in countries' projects; the developers are changing from being local to global.

'It has gone from focusing on maybe three countries to over thirty, and then the developers can't involve themselves as much as they used to in a country.'

— Implementer C

Furthermore, the model lacks unexpected situations that can occur where requirements come from nowhere, such as Ebola. This disease was not anticipated and it demanded changes in the core system and impacted the development in terms of what got in the next scheduled releases.

'This meteorite from outer space that can happen every now and then, which comes down and pushes all other plans away.'

— Implementer A

Due to an increasing number of countries joining HISP the implementations now have started to happen in parallel. Figure 9.2 tries to illustrate the situation of parallel implementations.

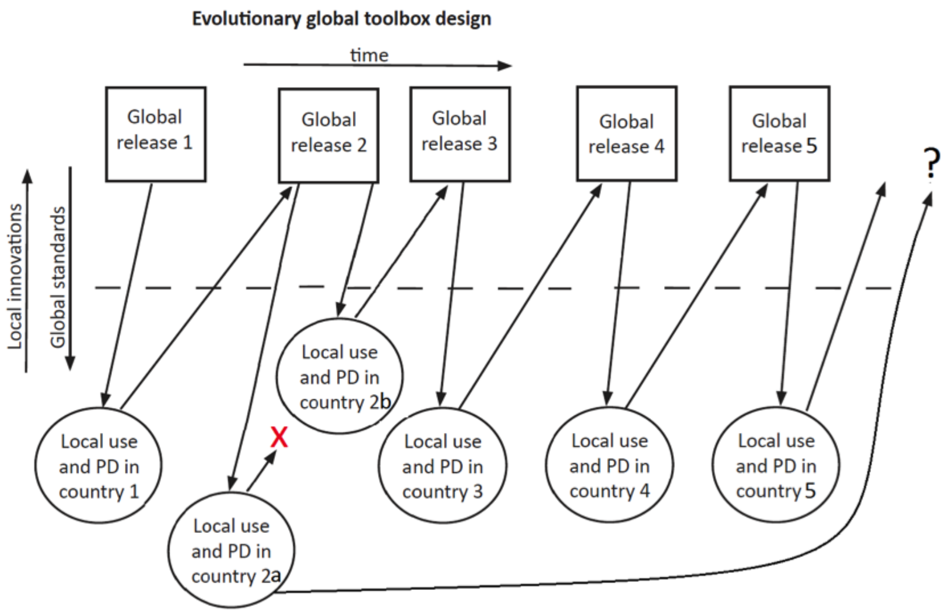

Evolutionary global toolbox design
time
Global release 1
Global release 2
Global release 3
Global release 4
Global release 5
?
Local innovations
Global standards
Local use and PD in country 1
Local use and PD in country 2a
Local use and PD in country 2b
Local use and PD in country 3
Local use and PD in country 4
Local use and PD in country 5
X

Figure 9.2: DHIS2 implementations in parallel.

The parallel implementations are represented with country 2a and 2b. The central issue in this situation is that two countries implement the same generic toolbox at the same time, but their local requirements for new features are conflicting. It could happen that one feature was changed back and forth by different developers to fit the use case they worked with.

Only one country will have its local innovation become a global standard. The situation with country 2a tries to illustrate when the local innovation failed to become a global standard, it has ended up in a difficult situation. The country is stuck with a local version of the generic toolbox, while the rest of the world moves on.

Figure 9.2 is very simplified. Now there are a lot more countries that implement in parallel. A more correct illustration would be with about 30-40 arrows up and down from the countries to the same global release. The developers haven't got capacity to handle all requirements and traditional PD is very challenging with so many countries involved (Roland et al. Forthcoming).

Berg (2001) argues that if users aren't involved in the design process, the software can end up being useless, which also increases the design-reality gap described by Heeks (2006).

This was also seen in Uganda in the implementation process of the J2ME client. The MUJHU project was stuck with a poor version of the patient tracker. With too many countries and contexts the developers hadn't capacity to fix it as they prioritized to make a new better patient tracker, although with less features. By doing a field study and sharing the experiences with the managers in Oslo, the issues were prioritized and soon fixed. From this one can see that what's prioritized by the developers, is very much influenced by the actors in HISP Oslo and not so much the implementers and users.

In sum, looking at how the use of DHIS2 has scaled the recent years, Titlestad et al.'s (2009) model comes up short. It is very suitable for making software generic in serial implementations, but their theory doesn't take into account parallel implementations. In an environment such as the HISP network, where the number of contexts and use cases constantly increase, one is dependent on being able to handle parallel implementations. In addition, in Titlestad et al.'s (2009) theory, both the users and the developers are very dependent on the implementers being good mediators. Without mediators being the link between users and developers, the design-reality gap described by Heeks (2006) increases.

9.3 Platform-based architecture

Manda et al. (2014) express the need of DHIS2 to become more platform-based. This can enable local communities to have more influence on the software to fulfill their requirements to suit their needs, simultaneously as they can benefit from the global software. The need of this platform is evident in contexts where users at the local level need to fulfill requirements, which aren't fulfilled by the software. Without such a platform, it is the developers who decide on the openness of the software being developed.

Roland et al.'s (Forthcoming) suggestion for a platform-based architecture cope with the issues found in Titlestad et al.'s (2009) model for how to develop generic software. With a platform-based architecture with an inner generic core that fits multiple use cases and an outer layer where more bespoke features can be implemented, the software can remain it's relevance for the majority of users and simultaneously enable participation in the design of requirements for single use cases.

An architecture like this is much more scalable as it allows a higher degree of conflicting requirements coming from the users. Because of the outer layer, parallel implementations are possible without leading to only one of the users getting the feature they want. The outer layer can also make it easier to handle unexpected situations that can occur too.

With this kind of architecture, the mediating role becomes less crucial for implementers. With the outer layer, the implementers don't have to communicate the requirements to the core developers for the features to be implemented. Instead, they can involve local developers to implement the features. With this layered architecture, the workload on core developers can decrease, as the requirements don't need to go through them to be implemented.

A challenge of this architecture is to do changes in the generic core without affecting all implementations. When changes in the generic core are needed, it is important that the core developers consider how the change will affect the implementations, but Roland et al. (Forthcoming) don't say much about this consideration.

'In HISP there is no standardized way of looking at how new development can affect the rest of the software other than what developers know of from before.'

— Core developer

By not looking at the history of the software, conflicts can occur, which causes tensions between local and global contexts. A suggestion is to look at the development of software longitudinally as a biography of an artifact. It is emphasized that in an innovation one needs to understand how the artifact is shaped by the context and history (Williams and Pollock 2009).

In sum Roland et al.'s (Forthcoming) strategy facilitates participatory design in a scaling environment more than Titlestad et al.'s (2009), and focus more on handling parallel implementations. But Roland et al.'s (Forthcoming) don't say much about the artifact's history, which is covered by Titelstad et al. (2009). Changes in the generic core can potentially make it difficult for all implementations to adapt.

9.4 Implementation in Uganda

An interesting finding when assisting the MUJHU project in the implementation of the J2ME Patient Tracker was that due to the way DHIS2 is developed; the J2ME client ended up being perishable. When the enhancement of the J2ME client was fixed in July 2014, it worked perfectly well with the server build it was fixed with. But the J2ME client was untouched for almost three months and in the meantime, changes happened server-side, making the J2ME client incompatible with the newest server build. The J2ME client then became outdated.

Perhaps, if another approach had been applied in the implementation and development of the J2ME Patient Tracker in Uganda, the J2ME client could avoid being outdated. By having a platform-based architecture, it might be easier for two implementations to happen in parallel, as conflicting requirements don't necessarily have to exclude each other. In the situation with the J2ME Tracker, implementations happened in parallel, and while the mobile client was untouched for a while, changes on server-side happened elsewhere. These changes affected the J2ME client.

In addition, if Williams and Pollock's (2009) strategy of looking at the biography of an artifact had been followed, the effect the changes had on other modules in DHIS2 could perhaps been considered. Not necessarily by avoiding doing changes server-side, but at least giving the users of the modules a heads up that changes will affect you and then you can adopt to the changes.

9.5 A combined model

Gumm et al. (2006) argue that a mixture of the different practices should be used to achieve user participation in a distributed design process. By looking at a number of practices to succeed with PD in a distributed setting, it was found that HISP does indeed follow several different practices in an attempt of involving the users in the design. In spite of that, the participatory design hasn't been too successful in the development of DHIS2. The reasons have been too many countries and too little capacity on the global level.

Both of the discussed approaches give perspectives to understand how one can enable participatory design. It could be interesting to explore whether it is possible to combine these two approaches, as they enable participatory design in very different ways.

Therefore, a suggestion is to look at distributed participatory design with a platform-based perspective and to combine Titlestad et al. (2009) and Roland et al.'s (Forthcoming) approaches. From Titlestad et al. (2009) one keep the evolutionary process of developing generic features. By doing that Williams and Pollock's (2009) concern about considering the biography of the artifact is taken into account. From Roland et al. (Forthcoming) one keep the layered structure that allow parallel implementations of bespoke and conflicting features, which can reduce the workload on developers and the dependency on mediators.

In that way, both PD approaches and generic development can be enabled and facilitated in the development of software. This can have a decreasing effect on the design-reality gap described by Heeks (2006), by allowing more local features in combination with the global design. Hewapathirana and Rodrigo (2013) described the concepts of design improvisation and actuality improvisation. They found that managers preferred design improvisation over actuality improvisation as it was more successful in minimizing the tension over the design-reality gap, and the suggested model facilitates that.

Due to the layered architecture, the workload on the global team can decrease, which can enable user support and mediated two-directional feedback, which are two of the suggested strategies for PD (Gumm et al. 2006). In addition if the documentation routines in HISP improve, it can make it easier for implementers and users to help themselves and reduce the need for communication. This again might release the workload on the global team even more. Shidende et al. (2014) suggest that by providing better documentation of the software, it would ease the communication for the implementing mediators, as they obtain more knowledge about the software. The reduced workload can enable the global team to spend more time on better user support and mediated two-directional feedback.

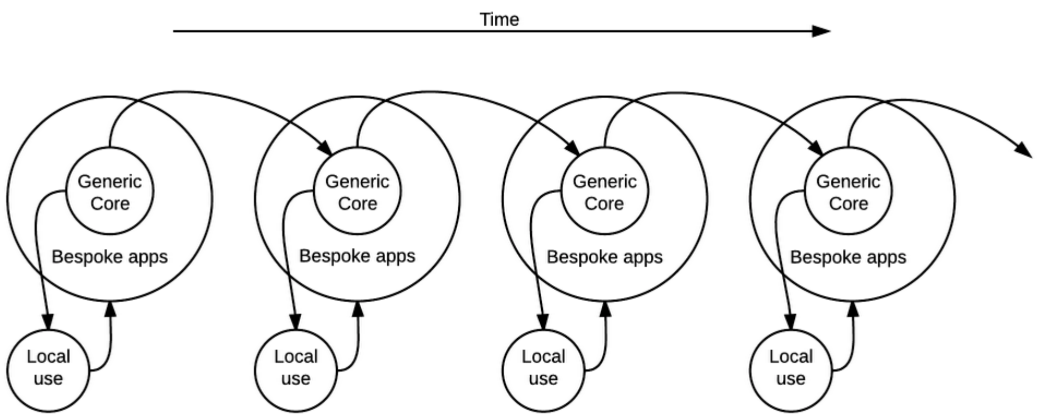


Figure 9.3: A suggested illustration of the development of DHIS2.

Figure 9.3 illustrates how the development of HIS as a platform can look like if the approaches are combined. The model represents an iterative process where each global release affects the generic core and is a step of becoming a more generic toolbox. These releases can be done without much involvement from the users. Local contexts can simultaneous get their requirements fulfilled with the development of more bespoke applications with the use of the generic core and by following participatory design principles. The illustration is simplified; in theory it can be as many local uses as there are countries in HISP.

Chapter 10

Conclusion

This chapter sums up findings in relation to the research objective introduced in chapter 1. The study was conducted in Uganda and Norway and looks at limitations and solutions in existing theory about Participatory Design in distributed contexts.

The first section contains a summary of the findings and discussion and tries to answer the research question. The following sections present contributions of this project, reflections of the research and suggestions for further research.

10.1 Findings and contributions

The objective of this thesis has been to explore the degree of participatory design in distributed software development of health information systems for low resource contexts. To concretize this research objective, a research question was shaped.

Research Question: *How can participatory design be enabled in distributed software development of health information systems for low resource contexts?*

To answer this research question, the following aspects of the research process were essential:

- Being a mediator assisting the MUJHU project in Uganda in their implementation of the J2ME Patient Tracker for DHIS2. The field study gave insights to the mediator-role and the importance of it to achieve PD.

- Interviewing different actors in HISP Oslo. The interviews provided insights in how actors in HISP Oslo perceive participatory design in the development of DHIS2.

- Reviewing others' literature, theories and studies concerning participatory design and distributed software development. This review was used as a foundation for the analysis of the development of DHIS2.

- Analyzing and comparing the literature reviewed with the findings from the field study and the interviews. This comparison made it possible to say something about the participatory design in the development of DHIS2.

It was found that the degree of PD was not high in the development of DHIS2 even though the majority of the suggested strategies for DSD and DPD are followed. The reason for the low degree of PD in the development has been due to the number of use cases of DHIS2 has scaled more than anyone could have imagined the recent 10 years and HISP as a research organization has not been prepared for this enormous scaling.

Furthermore, two models for how to develop software in a distributed setting with focus on PD were analyzed. Looking at how the use of DHIS2 has scaled the recent years, Titlestad et al.'s (2009) model comes up short. It is very suitable for making software generic in serial implementations, but their theory doesn't take into account parallel implementations. Roland et al.'s (Forthcoming) strategy facilitates participatory design in a scaling environment more than Titlestad et al.'s (2009), and focus more on handling parallel implementations. But Roland et al.'s (Forthcoming) don't say much about the artifact's history, which is covered by Titelstad et al. (2009). Changes in the generic core can potentially make it difficult for all implementations to adapt.

The answer to the research question, of how to enable participatory design, is the model suggested in section 9.5. In this model, aspects of Titlestad et al. (2009) and Roland et al. (Forthcoming) are combined. In that way, both PD approaches and generic development can be enabled and facilitated in the development of software.

The contributions of this thesis have been exploring literature and theories on how to develop software in a distributed context with focus on participatory design. The thesis has looked at strengths and weaknesses of Titlestad et al. (2009) and Roland et al.'s (Forthcoming) approaches and suggested how they might work better together. The suggested model is based on their theories and tries to face an increasing number of contexts and use cases better than Titlestad et al. (2009) and Roland et al. (Forthcoming).

10.2 Reflections

This thesis only presents the case of M-health implementation in one local context. It is not given that the findings in that context are transferable to other local contexts within the HISP network. But the answers received during the interviews conducted in Oslo are coinciding with the experiences gained in Uganda. This resemblance can be an indication that it could be similar situations in other contexts. It is also important to keep in mind that finding a recipe on how to enable participatory design in the development and implementation of a HIS is difficult. What succeeded in one context will not necessarily work for other HIS' in other contexts.

Although the data gathered resulted in useful findings the data collection process has limitations.

10.2.1 Personal limitations

The preparation process before going to Uganda could have been better. If the communication with HISP Uganda had been better, the need for changing focus whilst in the field could have been avoided.

Also, if conducting interviews had been practiced beforehand, the situation where interviewees' answers did not seem to make sense could be avoided. It took some time realizing the interviewees were afraid to admit they didn't understand the questions and ask for repetition or answering that they don't know. After realizing the issue, the interviewees were informed that it was perfectly fine to say they don't know or ask for repetition of the question. It is possible the interviewees found it hard to admit lack of knowledge due to their culture.

When conducting interviews, a barrier of language was discovered. During one of the interviews, the question asked was about the health workers' workflow when using the DHIS2 Patient Tracker, but instead of saying Tracker, the word register was used. Later on, when the interview was transcribed it was no doubt that the health worker was referring to the journal, not the Tracker. This is a perfect example of how important it is to be specific when asking questions, there shouldn't be any room for misunderstanding. The language barrier is also relevant for designers gathering requirements. If they are unable to get a deep understanding of the end users' needs and requirements the design-reality gap will increase (Heeks 2006).

10.2.2 Limitations within context

Looking at the number of interviews, observations and meetings conducted, the time spent in Uganda probably could be exploited better, but as Hein once wrote: *"Things take time"* (Hein 1969).

First of all the number of facilities to visit were limited. The DHIS2 software was tested in 10 facilities, in which 5 were in Kampala, the capital of Uganda, and the other 5 in Mpigi, a rural district a one-hour drive from Kampala.

Visiting a facility more than once could have been a possibility to conduct more interviews, but given the circumstances it wasn't appropriate. The facilities in Kampala were extremely busy, and holding of health workers from taking care of patients to answer a student's questions felt wrong. Therefore most of the interviews done were conducted in-group with the fellow students.

Another time-consuming factor was that the local project team had to plan the visits and make appointments with the facilities for the interviews to happen. Also when visiting the facilities the local team often helped out with various tasks, as a return of favor for the interviews, thus when visiting a facility the stay usually took a couple of hours.

As the Kampala district was so busy, it was decided to go to the more rural Mpigi district, to visit health facilities there. Going there also was a time-consuming task; first the driver for the day was 2 hours late, and to Mpigi it was a one-hour drive. At the clinic 3 hours were spent before driving back to Kampala. At the facility visited there was only one health worker so conducting one single interview took about 6 hours. Luckily the facility visited only had three visiting patients on that day so there was plenty of time to conduct a proper interview and even demonstrate the mobile clients and letting the health worker test them out. On that particular day, they were also given tablets, so an observation of their use of it was done.

10.2.3 Limitations of the study

To spend one month in Uganda was a bit short when looking at how the process developed. Spending more time in Uganda could maybe have made the implementation of the mobile client easier due to closer cooperation and communication with users, instead of being located in Oslo and coordinate between users in Uganda and developers in Manila. The reason for the short stay was because the original focus was to develop an application to facilitate feedback to health workers. The plan was to follow the five phases of action research and go to Uganda to gather requirements for the application, go back to Norway and develop it and go to Uganda one more time to test it out. After the change of focus, there was not necessary to return to Uganda, so time was spent gathering data from the HISP Oslo team by conducting interviews instead.

In Oslo, it was difficult to arrange interviews. The actors in the HISP Oslo team are very busy and travel a lot; so finding time for an interview was hard. For example, to get an interview with the core development leader failed, even though it was tried to arrange the interview for over a month. By interviewing the core development leader, the results could have been different as he probably has a different view on PD and development of DHIS2 than the implementers interviewed.

10.2.4 Limitations of the suggested model

The model suggested in section 9.5 is a simplification of how to enable PD in the development of generic software. Based on Titlestad et al. (2009) and Roland et al.'s (Forthcoming) theories, and the experiences from the Ugandan project, the suggested model makes some assumptions. It could be that the Ugandan project was a special case and that these assumptions are too strong.

10.3 Further research

It could be interesting to explore if the suggested model can be applied in distributed software development. It is not given that the suggested approach is feasible outside the context of this thesis. It could also be interesting to explore what effect the suggested approach has on participatory design in low resource contexts and if the suggested architecture facilitates and enables more participatory design in the development of new features.

As this thesis links DPD with platform-based architecture, it could be interesting to compare it with other platforms to see how they develop software and how they cope with participation. Furthermore one could investigate how platforms can interact with each other; platforms do have biographies that affect the devices using them.

A central aspect of the suggested model is parallel implementations, so it could be interesting to look at other countries as well, to see if participatory design works there. Maybe Uganda was a special case. Furthermore, one could look at other industries and sectors. Is distributed participatory design special for the health sector, or is it a widespread phenomenon, and does low resource contexts make a difference.

The HISP project coordinator said that PD is needed when you make generic software. But looking at DHIS2, it seems to be a successful HIS, even though not every recommendation is followed, so one can question if participatory design at all is important and it could be interesting to see if PD is an approach one should strive for.

An overall objective for the future research is further exploration of how organizations can retain a participatory approach in the development of generic software so they don't end up with platforms with too much focus on a small percentage of the users.

10.4 A conclusive remark

When the HISP project coordinator was asked about the future of HISP, he said one of the biggest challenges for HISP is how to keep the participatory approach simultaneously as they scale to more countries. From this, one can see that participatory design is important for HISP and that they want to follow this approach in the future development of DHIS2.

'We want to grow, but to remain relevant for all users. To grow without too much growing pains.'

— HISP project coordinator

Bibliography

AbouZahr, Carla and Ties Boerma (2005). 'Health information systems: the foundations of public health'. In: *Bulletin of the World Health Organization* 83.8, pp. 578–583.

Asiimwe, Caroline, David Gelvin, Evan Lee, Yanis Ben Amor, Ebony Quinto, Charles Katureebe, Lakshmi Sundaram, David Bell and Matt Berg (2011). 'Use of an innovative, affordable, and open-source short message service–based tool to monitor malaria in remote areas of Uganda'. In: *The American journal of tropical medicine and hygiene* 85.1, pp. 26–33.

Bates, David W, Lucian L Leape, David J Cullen, Nan Laird, Laura A Petersen, Jonathan M Teich, Elizabeth Burdick, Mairead Hickey, Sharon Kleefield, Brian Shea et al. (1998). 'Effect of computerized physician order entry and a team intervention on prevention of serious medication errors'. In: *Jama* 280.15, pp. 1311–1316.

Berg, Marc (2001). 'Implementing information systems in health care organizations: myths and challenges'. In: *International journal of medical informatics* 64.2, pp. 143–156.

Bird, Christian, Nachiappan Nagappan, Premkumar Devanbu, Harald Gall and Brendan Murphy (2009). 'Does distributed development affect software quality?: an empirical case study of Windows Vista'. In: *Communications of the ACM* 52.8, pp. 85–93.

Bodker, Kerl, Finn Kensing, Jesper Simonsen and Participatory It Design (2004). *Designing for Business and Workplace Realities*.

Braa, Joern and Sundeep Sahay (2012). *Integrated health information architecture : power to the users : design, development, and use*. Matrix Publishers.

Braa, Jørn, Eric Monteiro and Sundeep Sahay (2004). 'Networks of action: sustainable health information systems across developing countries'. In: *Mis Quarterly*, pp. 337–362.

Braa, Jørn and Sundeep Sahay (2013). 'Participatory Design within the HISP network'. In: *Routledge International Handbook of Participatory Design*, pp. 235–256.

Braa, Jørn, Ola Hodne Titlestad and Johan Sæbø (2004). 'Participatory health information systems development in Cuba: the challenge of addressing multiple levels in a centralized setting'. In: *Proceedings of the eighth conference on Participatory design: Artful integration: interweaving media, materials and practices-Volume 1*. ACM, pp. 53–64.

Braa, Kristin and Richard Vidgen (1999). 'Interpretation, intervention, and reduction in the organizational laboratory: a framework for in-context information system research'. In: *Accounting, Management and Information Technologies* 9.1, pp. 25–47.

Braun, Virginia and Victoria Clarke (2006). 'Using thematic analysis in psychology'. In: *Qualitative research in psychology* 3.2, pp. 77–101.

Camara, Gilberto and Frederico Fonseca (2007). 'Information policies and open source software in developing countries'. In: *Journal of the American Society for Information Science and Technology* 58.1, pp. 121–132.

Carmel, Erran and Ritu Agarwal (2001). 'Tactical approaches for alleviating distance in global software development'. In: *Software, IEEE* 18.2, pp. 22–29.

Chaudhry, Basit, Jerome Wang, Shinyi Wu, Margaret Maglione, Walter Mojica, Elizabeth Roth, Sally C Morton and Paul G Shekelle (2006). 'Systematic review: impact of health information technology on quality, efficiency, and costs of medical care'. In: *Annals of internal medicine* 144.10, pp. 742–752.

Conchúir, Eoin Ó, Pär J Ågerfalk, Helena H Olsson and Brian Fitzgerald (2009). 'Global software development: where are the benefits?' In: *Communications of the ACM* 52.8, pp. 127–131.

Curioso, Walter H and Patricia N Mechael (2010). 'Enhancing 'M-health'with south-to-south collaborations'. In: *Health Affairs* 29.2, pp. 264–267.

Damian, Daniela, Filippo Lanubile and Heather L Oppenheimer (2003). 'Addressing the challenges of software industry globalization: the workshop on global software development'. In: *Proceedings of the 25th International Conference on Software Engineering*. IEEE Computer Society, pp. 793–794.

DHIS Academies (2015). URL: https://www.dhis2.org/academy (visited on 05/05/2015).

DHIS2 in Action (2015). URL: https://www.dhis2.org/inaction (visited on 25/03/2015).

DHIS2 Mobile (2015). URL: https://www.dhis2.org/mobile (visited on 25/03/2015).

DHIS2 Tracker (2015). URL: https://www.dhis2.org/individual-data-records (visited on 25/03/2015).

Donner, Jonathan (2008). 'Research approaches to mobile use in the developing world: A review of the literature'. In: *The information society* 24.3, pp. 140–159.

Edwards, H Keith and Varadharajan Sridhar (2003). 'Analysis of the effectiveness of global virtual teams in software engineering projects'. In: *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE, 9–pp.

Flyvbjerg, Bent (2006). 'Five misunderstandings about case-study research'. In: *Qualitative inquiry* 12.2, pp. 219–245.

Franco, Lynne Miller, Sara Bennett and Ruth Kanfer (2002). 'Health sector reform and public sector health worker motivation: a conceptual framework'. In: *Social science & medicine* 54.8, pp. 1255–1266.

Freifeld, Clark C, Rumi Chunara, Sumiko R Mekaru, Emily H Chan, Taha Kass-Hout, Anahi Ayala Iacucci and John S Brownstein (2010).

'Participatory epidemiology: use of mobile phones for community-based health reporting'. In: *PLoS medicine* 7.12, e1000376.

Gladwin, J, RA Dixon and TD Wilson (2003). 'Implementing a new health management information system in Uganda'. In: *Health Policy and Planning* 18.2, pp. 214–224.

Gumm, Dorina C (2006). 'Distributed participatory design: An inherent paradoxon'. In: *Proc. of IRIS29*.

Gumm, Dorina C, Monique Janneck and Matthias Finck (2006). 'Distributed participatory design–a case study'. In: *Proceedings of NordiCHI 2006*.

Gustafson, David H, Robert Hawkins, Eric Boberg, Susanne Pingree, Ronald E Serlin, Frank Graziano and Chein Lung Chan (1999). 'Impact of a patient-centered, computer-based health information/support system'. In: *American journal of preventive medicine* 16.1, pp. 1–9.

Haux, Reinhold (2006). 'Health information systems–past, present, future'. In: *International journal of medical informatics* 75.3, pp. 268–281.

Heeks, Richard (2006). 'Health information systems: Failure, success and improvisation'. In: *International journal of medical informatics* 75.2, pp. 125–137.

Hein, Piet (1969). *Grooks*. Doubleday & Co.

Herbsleb, James D. and Audris Mockus (2003). 'An empirical study of speed and communication in globally distributed software development'. In: *Software Engineering, IEEE Transactions on* 29.6, pp. 481–494.

Hewapathirana, Roshan and Shan Semuthu Rodrigo (2013). 'FOSS HIS in Public Health Domain: Case Study of Design-Reality Gap and Local Improvisation in Global South'. In: *Journal of Health Informatics in Africa* 1.1.

Hillestad, Richard, James Bigelow, Anthony Bower, Federico Girosi, Robin Meili, Richard Scoville and Roger Taylor (2005). 'Can electronic medical record systems transform health care? Potential health benefits, savings, and costs'. In: *Health Affairs* 24.5, pp. 1103–1117.

HISP (2015). URL: http://hisp.org/ (visited on 25/03/2015).

HISP at UiO (2015). URL: https://www.hisp.uio.no/ (visited on 25/03/2015).

HISP Uganda (2015). URL: http : / / www . hispuganda . org (visited on 25/03/2015).

Iacono, JBA and C Holtham (2009). 'Research methods—A case example of participant observation'. In: *8th European Conference on Research Methodology for Business and Management Studies: University of Malta, Valletta, Malta, 22-23 June 2009:[proceedings]*. Academic Conferences Limited, p. 178.

Kahn, James G, Joshua S Yang and James S Kahn (2010). ''Mobile'health needs and opportunities in developing countries'. In: *Health Affairs* 29.2, pp. 252–258.

Kaplan, Warren A (2006). 'Can the ubiquitous power of mobile phones be used to improve health outcomes in developing countries'. In: *Global Health* 2.9, pp. 1–14.

Kimaro, Honest Christopher and Ola Hodne Titlestad (2008). 'Challenges of user participation in the design of a computer based system: The possibility of participatory customisation in low income countries'. In: *Journal of Health informatics in developing countries* 2.1.

Lemaire, Jeannine (2011). 'Scaling up mobile health: Elements necessary for the successful scale up of mHealth in developing countries'. In: *Geneva: Advanced Development for Africa*.

Lings, Brian, Björn Lundell, Pär J Ågerfalk and Brian Fitzgerald (2006). 'Ten strategies for successful distributed development'. In: *The transfer and diffusion of information technology for organizational resilience*. Springer, pp. 119–137.

Manda, Tiwonge Davis, Marlen Stacey Chawani and Caroline Ngoma (2014). 'Software Architecture as Ground for Exercising Decision-making Power in Distributed Software Development: A Case of DHIS Tracker'. In:

Manda, Tiwonge Davis and Terje Aksel Sanner (2014). 'The Mobile is Part of a Whole: Implementing and Evaluating mHealth from an Information Infrastructure Perspective'. In: *International Journal of User-Driven Healthcare (IJUDH)* 4.1, pp. 1–16.

MUJHU Research Collaboration (2015). URL: http://www.mujhu.org/history.html (visited on 25/03/2015).

Namukwaya, Zikulah, Prosper Behumbiize, Fitti Weissglas and Jaco Homsy (2013). 'Evaluating a web-based electronic medical record system in combination with targeted phone messaging to improve long-term post-natal follow-up of HIV-infected and uninfected pregnant women and their infants in Uganda.' Wemr Project Protocol version 4.0.

Noll, John, Sarah Beecham and Ita Richardson (2010). 'Global software development and collaboration: barriers and solutions'. In: *ACM Inroads* 1.3, pp. 66–78.

Obendorf, Hartmut, Monique Janneck and Matthias Finck (2009). 'Inter-contextual distributed participatory design'. In: *Scandinavian Journal of Information Systems* 21.1, p. 2.

Pollock, Neil and Robin Williams (2009). 'Global software and its provenance: generification work in the production of organisational software packages'. In: *Configuring User-Designer Relations*. Springer, pp. 193–218.

Rashid, Ahmed Tareq and Laurent Elder (2009). 'Mobile phones and development: An Analysis of IDRC-supported projects'. In: *The Electronic Journal of Information Systems in Developing Countries* 36.

Robertson, T and J Simonsen (2012). *Routledge International Handbook of Participatory Design*.

Roland, Lars Kristian, Terje A Sanner, Prosper Behumbiize, Zikulah Namukwaya and Kristin Braa (2013). 'Accommodating Multiple Rationalities in Patient Oriented Health Information System Design'. In: *IRIS*. 36. Akademika forlag.

Roland, Lars, Johan Sæbø, Terje Aksel Sanner and Eric Monteiro (Forthcoming). 'Do Scandinavian methods for user involvement scale'.

Rolland, Knut H and Eric Monteiro (2002). 'Balancing the local and the global in infrastructural information systems'. In: *The Information Society* 18.2, pp. 87–100.

Sahay, Sundeep, Brian Nicholson and Srinivas Krishna (2003). *Global IT outsourcing: software development across borders*. Cambridge University Press.

Sahay, Sundeep and Geoff Walsham (2006). 'Scaling of health information systems in India: Challenges and approaches'. In: *Information Technology for Development* 12.3, pp. 185–200.

Sanner, Terje Aksel, Lars Kristian Roland and Kristin Braa (2012). 'From pilot to scale: Towards an mHealth typology for low-resource contexts'. In: *Health Policy and Technology* 1.3, pp. 155–164.

Shidende, Nima Herman, Marlen Stacey Chawani and Jens Kaasbøll (2014). 'Challenges in Participation and Implications on Human Development: Experiences from Health Information Systems Implementations in Tanzania and Malawi'. In:

Shidende, Nima Herman and Christina Mörtberg (2014). 'Re-visiting design-after-design: reflecting implementation mediators connectedness in distributed participatory design activities'. In: *Proceedings of the 13th Participatory Design Conference: Research Papers-Volume 1*. ACM, pp. 61–70.

Smartphone battery life (2015). URL: http://www.zdnet.com/article/the-secret-behind-poor-smartphone-battery-life/ (visited on 05/05/2015).

Stansfield, Sally K, Julia Walsh, Ndola Prata and Timothy Evans (2006). 'Information to improve decision making for health'. In: *A custom publication of the Disease Control Priorities Project* 157.

Susman, Gerald I (1983). 'Action research: a sociotechnical systems perspective'. In: *Beyond method: Strategies for social research*, pp. 95–113.

Titlestad, Ola Hodne, Knut Staring and Jørn Braa (2009). 'Distributed development to enable user participation: Multilevel design in the HISP network'. In: *Scandinavian Journal of Information Systems* 21.1, p. 3.

Whyte, William Foote Ed (1991). *Participatory action research.* Sage Publications, Inc.

Williams, Robin and Neil Pollock (2009). 'Beyond the ERP implementation study: a new approach to the study of packaged information systems: the biography of artifacts framework'. In: *ICIS 2009 Proceedings*, p. 6.

Yin, Robert K (2013). *Case study research: Design and methods*. Sage publications.