UNIVERSITY OF OSLO
Department of informatics

Challenges of Open Source Software Capacity Building among DHIS2 developers in India

Master thesis
60 credits

Ole Kristian Hustad

16. of May 2008

Abstract

This master thesis is based on the author's participation in an open source software based project named DHIS2. The project is developed for the organisation named Health Information System Program (HISP). In the project I have acted as developer within integration development with proprietary software from the Indian health sector. My main effort in this project is capacity building in key tools and frameworks of DHIS2; Subversion, Maven, Spring, Hibernate, JUnit and WebWork in the HISP India node.

The HISP coordinators in Oslo wish to outsource the main development activity of DHIS2 to India, which is the main reason for the capacity building effort in India. The main goal was to empower the HISP developers in India with the necessary skills and knowledge to become the main developer node in the HISP network regarding DHIS2 development.

This master thesis describes the capacity building effort in the HISP India Node during the spring 2007, with special focus on the use of best practices for development purposes. Some of the more interesting findings from the capacity building effort is how the project strive to follow and use best practices, yet in many ways fail to achieve its distributed open source - software development strategy due to path dependencies inherited from local contexts in the Global HISP network.

Acknowledgements

I want to thank Knut Starring and Gianluca Miscione for guiding me through the long and treacherous path of master thesis writing. Furthermore I want to thank them for the opportunity to go to India and experience some thing totally different with the fantastic people of the HISP India team. In particular I would like to thank John Lewis, Anil Kumar, Usha Srinath, *Bharat* Kumar, Suneel Kumar Chevvu, Bhujji, Randir, Ravi Babu, Pandu, Jeswin Jacob and Vivec from the HISP India team for making the stay very memorable. Furthermore I would like to thank the HISP India team for a terrific out-of-Europe experience. I also would like to thank Stian Strandli for keeping me company those many days in India. And at last I would like to thank my many friends and family, for keeping up with and helping this grumpy, bitter, old, sod for the last year.

It has been a wild ride which soon is at an end!

Ole Kristian Hustad

Contents

1 Introduction

This thesis is based on my experiences from capacity building efforts within the Health Information System Program (HISP). The capacity building effort was centred around the District Health Information System 2.0 (DHIS2) in the Indian Node (HISP India) of the HISP network. The author has been involved in the project as developer of customised exporting solutions and integration of Indian modules in fall 2006 to spring 2007. The focus of the thesis however is on our capacity building effort among DHIS2 developers in India, in the spring 2007. In order to understand the nature of HISP and DHIS1 and DHIS2 I have added a brief summation of key historical elements. The latter leads to the thesis research question, which is defined later in this chapter.

1.1 HISP and DHIS history

The HISP initiative is an indirect offspring of the «The WHO Alma Ata declaration» of 1978 (Braa et al, 2004). The declaration states that by the end of the millennium everyone should have access to primary health care services, regardless off country and location.

After sometime it became apparent that better management of resources was necessary to reach the millennium goal. The Idea was to use a Health Information System (HIS) for data collection and analysis, in order to increase precision, efficiency and transparency in the healthcare sector.

As part of South Africa's Reconstruction and Development Program (ANC 1994b, cited by Braa et Al,2004), strategic management teams were established to develop plans for reconstruction of all health sectors, in all provinces in South Africa, including the creation of a unified HIS (ANC 1994a, cited by Braa et al, 2004). Early in 1995 the Health Information System Program (HISP) was initiated after a HIS sub-committee in Western Cape proposed to develop a district-based HIS as part of their final report in 1994. The committee sent in a proposal based on earlier action research (Braa et. al, 2004). The proposal was approved and received funding from the Norwegian Agency for Development Co-operation (NORAD) for the period 1996 - 1998. To begin with HISP was based at two Cape Town universities and implemented in three districts in Western Cape (ibid).

At the time most of the HIS was fragmented on a vertical level. Many different health improvement programs were present in South Africa and each of them had their own

management and data collection systems. An important part of the work in South Africa was the abstraction of the Essential Data Sets (EDS) in order to avoid duplicate data and a lot of unnecessary work. An EDS is the minimum number of data elements needed to create indicators[1] (ibid).

The rapid success of the EDS was enhanced by the prototyping of District Health Information System (DHIS) application, which was used for capturing data and provides analysis for management and health workers. During implementation and deployment phase a similar project in Eastern Cape experienced serious problems with their HIS. The health authorities started looking for other options. They noticed HISP's success in the neighbouring state and asked if it was possible to adapt and deploy it in Eastern Cape with minor adjustments. HISP spread to Eastern Cape and ensured funding from the EQUITY[2] project from late 1998 to 2003. The project's relative success in both Western and Eastern Cape led other provinces to initiate a similar process, and from 2000 it was endorsed as a national roll out scheme supported by EQUITY (Braa et al 2004).

A part of the HISP strategy for sustainable system development is to spread the development vertical and horizontal. This means to forge alliances on every level in the governmental system and create a Health Information (HIS) system with the healthcare workers to inspire ownership, a sense of responsibility for the system. In addition it is important to ally with the Education facilities in order to spread the word and interest as well as to educate a workforce.

In 1999 HISP spread to sites in other locations. The first new node in the network was Mozambique. This was accomplished with financial support from Norwegian Council of Universities' Committee for Development Research and Education (NUFU) and with University Eduardo Mondlane (UEM) and MISAU (Ministry of Health) as partners. This second national node is very different for South Africa on the political as well as infrastructural level. Mozambique has a centralized government style which means that any

[1] Indicators are analysed data, which usually describe a rate based on positive occurrences divided by the number of samples, and multiplied with a scalar like 1 000, 10 000, 1 000000 etc.
[2] EQUITY is a South African Ministry of Health project financed by the United States Agency for International Development (USAID) and a supporter of HISP. Key proponents behind the EQUITY project also were active in the Malawi case presented later.

initiative must be approved from the central government before any pilot program can be started, unlike South Africa which has a stronger provincial government, and therefore allow such decisions to be taken by the province (Braa et al).

Mozambique suffers from some of the same problems as other South African countries since much of the money in the health care sector comes from donors. The donor organisations often have their own management and software. Also, the fragmented databases demand a lot of additional work, since data often have to be collected and reported multiple times. This puts an extra strain on the healthcare workers as well as blurring the picture of the nation health situation (ibid).

The old HIS system consisting of database applications was graded obsolete by MISAU and therefore a need for a new and better system in all government levels became evident.

HISP initial aim was to implement a pilot project in three districts, but the need of a central government made it apparent that another approach was necessary. To show the true value of a HIS with analytic capabilities, it was necessary to collect data from all districts to show the relative situation. In order to achieve a full scale demonstration, data from the old (pre-HISP) application was transferred to the DHIS application. The 5 doctoral students who had been the driving force behind the implementation have adapted it to the different district needs. Through a joint effort they made a hard coded translated version in Portuguese for use in Mozambique.

The progress for HISP was slow in the early stages due to lack of political support. HISP had government approval, but was one of several competing organisations and agencies. A major organisation wanted to develop a master plan for computerization of the entire health sector. The plan had a budgeted cost on $55 million, and even though MISAU thought the chance of getting that kind of funding from donors was unlikely, they did not want to exclude the possibility at once. This led to a deadlock from 2001 to 2003. After much back and forth MISAU decided late in 2003 to give the green light for making DHIS the new national HIS and thus counting on support from HISP (Braa et al, 2004).

One of the professors in HISP is from India, and he had a connection in the government in Andrea Pradesh. With the help of a senior officer in the chief minister office HISP received

permission to do a pilot study in the constituency of the chief minister in Chittoor district. The opportunity which had emerged was not without a drawback. The political situation in Indian states can be rather unstable, hence changes in government policy posses a high risk. This turned out to be a problem in Chittoor[3] in 2004 (Braa et al, 2004) and later in Gujarat[4] in 2007 (Field notes, 13.04.2007)

The software was named District Health Information System (DHIS, also known as DHIS 1.x, where the "x" represent the different versions of the system). It was based on an Access database and Visual Basic. Later the work of a 2.0 version was started in order to change the development platform to something more modern which enabled a web interface. As well as removing the dependency of proprietary software in order to make it more interesting to Open Source Software developers. One of the main reasons however, was that the former DHIS version had become very complex and difficult to develop due to high coupling between different parts of the program. The latter made further development of the program difficult.

The new DHIS2 was therefore supposed to use more frameworks as well as a three tier architecture[5] (Nordal, 2006, p 48), in order to lower the level of coupling, get a cleaner architecture and thus reduce the complexity(). The new system named DHIS 2.0 is supposed to eventually replace the former DHIS 1.X versions. However, there are a lot of problems which have to be solved. Currently DHIS 1.X support more functionality and has fewer bugs, as well as being significantly simpler and faster to use.

During the last 6 months the choice of technology platform for DHIS2 has been criticized by the developers in India (Appendix F, 17.1). The number of frameworks makes the participation bar high since a significant amount of knowledge is needed in order to start working on the project. Furthermore some of the solutions made with frameworks are

---

[3] The commissioner supporting the HISP India implementation had to leave because his political party lost an election.

[4] Both the public administration officers supporting the DHIS2 solution left their job in the Public health administration. Since DHIS2 lost both their advocates in the administration. The officers supporting other HIS software threw DHIS2 out of the competition.

[5] A 3 tier architecture is a simple variant of the multi-tier architecture. The architecture is similar to MVC, but has a strict demand of how the tiers communicate. The tiers; Presentation Tier, Application Tier and Data tier, represents 3 conceptual layers which are very loosely coupled. The latter should in theory enable you to replace any of the layers with a new layer whenever technological evolution calls for it.

significantly slower to build and run, in addition to being more complex to implement. The latter will be central in the discussion chapter.

The HISP strategy has been to develop the DHIS2 application in Norway, India and Vietnam, where the last two are of the newer nodes in the HISP network and also pilot (pioneer) areas of implementation. A main part of the strategy has been to relocate the core development done by master students in Oslo to the developers in India and Vietnam. HISP try to use a global distributed development strategy in the DHIS2 project. The strategy is central to the thesis, and the most significant reason for the capacity building effort in India.

During a 3.5 month field study in India, a fellow master student and I spent some of the time training developers in use of DHIS2 technology, in addition to working on development tasks related to the DHIS2 software. During our stay we discovered several unexpected issues related to the HISP India context and the main strategy for relocate core development of DHIS2.

1.2 Personal Background

At the age of 24 I started on my master degree at the University of Oslo (UIO) in January 2006. The first semester I did not choose any master assignment, since the current options of assignments did not appeal to me. At first I thought of doing a master assignment in network communication, but later I thought about doing a thesis in Software engineering. Even later "I discovered" the wonderful world of open source software (OSS), and I changed my mind again. Luckily I have a flexible background from UIO, which makes the transition between the different domains rather easy.

From my earlier background I had solid knowledge of Java Standard Edition 5.0, unit testing, the Integrated Development Environments (IDE) Eclipse, and also an understanding of agile methods like Test Driven Development (Beck, 2002) which in many ways reflects the preferred development approach of HISP and other open source communities. I also did a summer job with similar technologies in PHP the summer of 2006. The latter also introduced me to several OSS technologies. This later influenced me to apply for and choose the current thesis topic.

1.2.1 Motivation

I first approached HISP because I wanted to do a Master degree in Open Source Software (OSS). I directed my initial request to a professor which later redirected it to the HISP. I first had a meeting with my later to be master assignment supervisor, where he presented the application I was suppose to work on as part of my assignment (DHIS 2.0). He furthermore gave a small introduction to what a master thesis about HISP implies.

My reason for doing a master in Opens Source Software is first and foremost because I am quite fond of the idea of communities working together for a greater good (voluntary communal work). I have earlier participated in social and scientific students unions with just this goal in mind.

I also have a personal interest in Open Source Software (OSS) from a user aspect from earlier experiences. Some applications exist exclusively in this domain because they have not been caught up by commercial cooperation, or they have simply been forgotten. I am thinking of computer games which are released under an OSS license, such as Settlers and FreeCol (2007). There is also a number of interesting other mainstream applications, programming tools and languages released under OSS license as well; Firefox, Thunderbird, Open Office, Eclipse, JUnit, MYSQL, PHP and Java just to mention a few.

I agreed to do this master assignment at the end of August 2006. Shortly thereafter I attended 2 courses – INF5750[6] and INF5210[7] – which broadened my horizon with regards to; HISP, OSS, action research, cultural differences and challenges, and introduction to Information Infrastructure (II) strategies and challenges. These courses also introduced me to the technical domain of the HISP project DHIS2.

During my introduction and first exposure to HISP, I met with two other master students who wanted to do their master in an OSS domain. One of them later joined me on the field trip to

---

[6] INF5750 is the course code, of the Global software development course also known as the OSS course among the students at UIO. In this course students are introduced to OSS technology and ideology as well as DHIS.
[7] INF5210 is the Information Infrastructure course. In this course the students study a particular class of information systems, known as Information Infrastructure. Some of the syllabus is articles regarding the HISP organisation and the implementation and realisation of DHIS 1. X

India, and discussions between the two of us greatly coloured my HISP experience. For later reference I will call him my "fellow master student".

At the fall 2007 I function as teacher assistant in the INF5750 course which I mentioned earlier. Initially the idea was to use this as a part of my thesis, perhaps as a comparative view between capacity building at school versus capacity building in the field. But the latter was later disregarded due to narrowing the scope of the thesis.

1.3 The capacity issue

At the time of writing the distributed development strategy mentioned in chapter 1.1, is failing due to lack of local capacity[8] among the other DHIS2 nodes. For one thing, most higher education facilities in India do not educate students in JAVA[9]. They teach older programming languages such as C, C++, COBOL and other programming languages we barely use at the University of Oslo. The latter was probably not considered during the technology review during the initial phases of the DHIS2 project, and it is a major obstacle for relocating the core development to India. The founders of the project were aware of the lack of java knowledge among the HISP India during the initial stages of the project (Chapter 17.3, capacity building effort in India 2004). However they greatly underestimated the difficulties of moving the development to India, which is an unrealistic alternative even now (4 years later).

As an effect of too little competence in the necessary tools and frameworks, all the DHIS2 development in India in the period from May 2005 (Nordal, 2006, pp 71) to April-May 2007(Hustad, 2007a) was carried out by developing external applications. The developers were using different technologies, and connected directly to the DHIS2 database (the latter approach is very vulnerable to changes of the core database).

The reason for the resistance of such quick fix designs is that it creates an unnecessary dependency against particular versions of the DHIS2. A feasible scenario is that changes to the database will cause the external applications to fail (which already have happened once).

---

[8] Capacity in this context refers to the developers knowledge of the DHIS2 tools and frameworks, in addition to work practises.

[9] Java is a Object- oriented programming language for a little more information see appendix A, or http://www.sun.com/java/

In the worst case scenario an upgrade of the external applications are very expensive, or may even be impossible due to lack of human resources and necessary knowledge. In addition the newer versions may not utilize version dependent functions and thus makes the upgrade to future and better release less favourable. The latter was a huge problem with the 1.x versions, hence the great concern with the creation of version-dependent external applications.

1.4 Research Objectives

At the time of writing HISP has existed as an organisation for 13 years. During this time action research based capacity building has been conducted on many administrative levels in many countries. In my thesis I will study and conduct capacity building, in the HISP India node, among the DHIS2 developers. Earlier attempts on capacity building based on the DHIS2 technologies, have been conducted with different degree of success The two former training effort regarding DHIS2 technologies in India known to me, was Patrick's effort in India summer 2004 (chapter 16, capacity building effort in India 2004) and Nordal's effort in May 2005 (Nordal, 2006, pp 71).

I believe that in order to make the capacity building successful, we need to achieve some fundamental goals. I assume it is impossible to teach everything during my short field study. Thus the main goal must be to start a sustainable learning process, where the participants learn enough to keep exploring and learning on their own. During the capacity building effort, several events indicate the amount of knowledge needed to develop on the DHIS2 in the manner intended by the team in Oslo, is mismatching with the organisations local context. The latter leads me to the following research questions.

How to conduct sustainable capacity building for system developers in OSS based HIS projects in a Non Governmental Organisation in India?

In order to achieve the main goal it can be enlightening to look at what I assume to be related issues. Below I have listed the assumed related issues as secondary research questions.

What is the current knowledge base of the developers?

It is important to understand what people already know before we try to teach them something new. Only then can we identify and fill the necessary gaps and move forward.

How can infrastructure limitations affect capacity building for system developers in OSS based HIS projects in a Non Governmental Organisation in India?

Capacity building is not independent of physical constraints, like poor infrastructure, long distances, and living conditions and so on. Later in the thesis I will discuss how poor infrastructure affected our capacity building efforts.

How does the working environment affect the capacity building effort in India?

From my own experience I have discovered the difference in working capacity of simple things like having a good chair and a desk, instead of working in the bed or on the floor. Other work environment issues like disruptions (noise) and heat also affect my own working capability. When programming and exploring tools and issues. A stable internet connection and power supply are very important for continuity in work and learning. There are several elements regarding working environment which greatly affect the capacity building effort, this elements will be explored later in the thesis.

Initially I was going aiming at answering another research question, however the way things turned out I was forced to change the target (see chapter 6.1). I will still seek to answer the former research questions, but new primary research question is needed. The new primary research question is built upon a paradox and will be the main focus of the thesis.

How can use of multiple open frameworks, design patterns, architectures, and tools create a lock-in situation for a flexible and open project?

I the DHIS2 project we rely on use of an open development strategy, all the tools and frameworks are open source software. The DHIS2 source code is available online for anyone who wishes to contribute to the project. All our communication channels are open for external users. The documentation and email archives are available online. Then how come we in the end still have trouble integrating new developers into the project? In order to answer this question I will look into how the reliance of advanced frameworks affects the openness of the project.

1.5 Structure of the thesis

The reminder of the thesis is coarsely structured in the following order organized in 3 parts;

- Part 1 Theory and Methodology
- Part 2 Empirical study
- Part 3 Discussion, concluding remarks and future work

The three parts is organised in 6 chapters; theory, methodology, empirics, discussion, concluding remarks, future work. In addition the acronym and reference lists have their own chapters.

To supply a broader view of several topics regarding the DHIS2, a more thorough description is found in the Appendixes; DHIS (appendix A), HISP India (appendix B), Graphical Analyser (appendix c), Training exercises (appendix D), IDSP and RIMS (appendix e), Important Emails (appendix F), User Training (appendix G). Interview with *Cordinater1 from Oslo* (Appendix H)

1.6 Notation used in the thesis

Upon doing a direct quote from document types such as email, and articles the text is outlined using a indented italic format followed by a dash '-' and the reference '(Hustad, 2007)' at the end of the quote.

> *This is an example of the a direct quote formatting – (Hustad, 2007)*

Some of the data for this assignment is collected from a field diary. The diary is written and categorised in chronological order with months as main chapters, and days as sub chapters. - Since I have worked with the diary in many different document types (which change the layout of the document) during the analysis period, references based on page numbers has turned out to be rather useless. Thus I have used date's (sub chapters) when specifying references since a date reference is relevant regardless of document type and layout.
Further more, direct quotation to from my field notes (diary), is displayed in text boxes. Example below:

> They were supposed to compare DHIS with an other HIS and discuss strengths and weaknesses in the two systems
> (My FieldNotes, 07.03.2007)

I have also made some changes to conserve the anonymity of the project participators. Instead of using their real names, I have defined new nick names based on their role (coordinator, developer) and status in the project (former, new). In order to avoid developer #1, #2, #3 names on a massive scale I have added location (Norway, Oslo, Vietnam, India, etc.)

As a helpful note to the reader I have included a dictionary of acronym and abbreviations at the end of thesis (Chapter 8) before the reference list and appendixes.

2 Theory

For this thesis I have chosen to rely on the Information Infrastructure (II) conceptual framework as described in the "Information Infrastructure kernel theory" by Hanseth and Lyytinen (2006) for analytical purposes, in addition I include the theory of Knowledge as Infrastructure[10] (KI) in order to apply the theory to capacity building. Further more, best practices and OSS are central components in HISP and the capacity building effort, thus a limited set of theoretical aspect from the two latter domains are included.

In the Action research theory (see Chapter 3), there is a close relation between theory and practice, thus I will outline two perspectives. Firstly the DHIS2 software viewed as an Information Infrastructure. Instead of introducing another framework and concepts I will use II to describe the 'capacity base'(knowledge base) in light of the "Knowledge as infrastructure" theory by Hanseth (2004).

The DHIS2 project is managed and developed using many strategies regarded as "best practices". The best practices can be linked to choice of tools, framework, and architectural decisions. The "best practices" each have one or more clear properties which seems very useful for the project. The reliance on best practices plays a key role in the DHIS2 project. In the 'best practices' subchapter I focus on the use of frameworks, to inscribe best different practices. Later in the same subchapter chapter I present the Framework as both an II and KI.

The project design is also drawing on concepts, ideas, tools, components, framework, methodologies and best practices from the open source software (OSS) community. In fact all the tools and technologies used in the project is OSS. Further more OSS was an important part of the initial part of the project development and design. Thus the latter also deserve a brief introduction.

[10] Knowledge as Infrastructure is also called "knowledge infrastructure", though extension or change of definition is intended.

The theory presented in this chapter is deliberately presented without tight connection to the empirical data and vice versa. In the discussion chapter the theory and empirics will be discussed in light of each other, thus providing the link between theory and practice (empiric).

2.1 Information Infrastructures

Information Infrastructures (II) is a theory about building shared, heterogeneous, and evolving open ended information systems. In many ways the II match the description of DHIS 2.0 and HISP. The Information System Research has over the years provided us with definitions of particular "classes" of systems. If we are able to identify general classes of problems and systems, and furthermore map them to approaches for best solutions, then we can simplify the development and obtain a higher degree of predictable and successful projects.

One of the weaknesses of earlier models is the use of a "snapshot view" of the world when modelling the domain, and proposing a solution based on the static model. This approach is adequate and might even be the best solution in some cases, but for a particular class of information systems it is of higher importance to acknowledge properties such as high longevity and necessity of integration against unknown systems and domains. In some literature the latter systems properties describes a class of information systems known as Information Infrastructures (II).

> *We formulate fragments of a design theory of information infrastructures (II). An II is a shared, evolving, heterogeneous installed base of IT capabilities developed on open and standardized interfaces. It forms complex inter-organizational socio-technical ensembles like the Internet - Hanseth & Lyytinen (2005).*

In the quote above we first address the five properties of an II; open, shared, evolving, heterogeneous, installed base. Of these properties the evolving and installed base is the most important for this thesis, since it regards the past as dependency to the current development, and the evolving property states that this is a continuous effort. This means that the development of the application is *dependent* of what technical and social networks which already exist. The five properties is described and discussed in more detail later in this chapter.

> *IIs are large, complex and heterogeneous; they evolve over long periods and adapt to unknown needs (Hanseth, Lyytinen, 2005).*

> *IIs are designed as extensions to or improvements on the installed base which influences how new IIs can be designed. The proposed theory addresses the design challenge of tackling the inherent complexity, scale and functional uncertainty of IIs.*
>
> *Our theory suggests that II design needs to focus on installed base growth as a means of reinforcing growth and increase II flexibility to avoid technological traps (Hanseth, Lyytinen, 2005).*

In the latter quote I want to emphasize "*to avoid technological traps*" as an important reason for our capacity building effort. By enforcing strict rules for connecting to the data source[11] we avoid database dependency as well as a high probability for updating the database without breaking existing system components[12]. The latter design choices enable us to make low coupled, highly modularised and configurable solution.

> *We show how these requirements can be achieved by enacting design principles that promote early immediate usefulness of II, follow simple design and use, harness the existing installed base, and use extensive modularization. We use two case studies - the design of the Internet and the design for health care II in Norway- to validate these principles and illustrate their application. (Hanseth, Lyytinen, 2005)*

The II theory is designed with intention to theorize about and discuss issues related to the information infrastructures (II) class of information systems (IS). Above the design of the theory is outlined, it focuses on using the installed base as initial point of departure and cultivate already exiting socio-technical systems towards the desired solution. The II kernel theory attempts to describe a methodology for successfully implement new infrastructures.

---

[11] Data source in this case refer to the database, which we use persistence (storage) purposes.
[12] System Components refers to the different modules (parts) of the system.

The theory proves rather useful for description and discussion purposes, and to my knowledge is mostly used as a perspective in which to view the IS development effort. The theory align with the action research method (as an iterative process) and with the "Networks of action" article by suggesting a broad bottom up approach. For my thesis I will use some terms from this theory to describe and discuss the phenomena's and our capacity building effort.

To start with, we can look on the five base properties from the mentioned in the II definition, which separate the IIs from other IS. Below I have listed the five distinct properties of an II with an explanation of what's meant by each one of them.

Installed Base

> *When a part of an infrastructure is changed, each new feature, or each new component has to fit with the as-is infrastructure. This as-is infrastructure, i.e. the installed base, heavily influences how a new infrastructure can be designed (Hanseth, Lyytinen, 2005)*

The "installed base" property represents a very important aspect, both in terms of computer tools, human knowledge, physical infrastructure and socio-technical infrastructure. The acknowledging of value and interest in the existing system artefacts is crucial, since its domain often contains all the future users as and competitors.

Heterogeneous

> *IIs are heterogeneous: they contain a large number of components of multiple sorts, including diverse technological components and non-technological elements. These components are connected in complex ways and change constantly (ibid).*

The "heterogeneous" property acknowledges the possible coexistence of multiple systems, collaborating on one or several tasks. The heterogeneous property includes both technical and non-technical. The non-technical components can be users, administrators, knowledge, paper forms, etc.

Open

> *Our definition highlights two critical features that enable II design. They must be open and as a result of this they must rely on shared standards. Openness in design signifies the lack of borders in terms of the II's scale, scope or function. (ibid)*

"Openness" with regard to II is about keeping the design open, in order to let others connect to or integrate with your system, or perhaps develop new applications, functions and features to the infrastructure. The open property might also refer to different use and user areas than it was initially designed for.

Shared

> *Such an II, when appropriated by a community of users, offers a shared resource for delivering and using information services in a (set of) community. The definition articulates a different concept of an IT artefact from the traditional definition of an IS. (ibid)*

The easiest way to explain the shared property is to picture how the development of internet was conducted or user content driven web communities. The latter is perhaps the best example as the value and usefulness of the technology increases as the community grows with regard to number of users and the content they provide. The increasing value attracts more users, and so fort. The latter phenomenon is called building the "momentum" and is a part of evolving the system.

Evolving

The II theory acknowledges the fact that the problem domain of an Infrastructure will change over time and so will the use and users of the Information Infrastructure (II) (ibid). The infrastructure is presumed to build on the installed base and grow on top of already existing infrastructures and components. The evolving property outlines that an infrastructure keep growing in scale, scope and function, over time.

Table 1: Three classes of information infrastructures (Hanseth & Lythinen)

Class of infrastructure Feature	Universal Service Infrastructure (e.g. Internet)	Business sector infrastructure (e.g. EDI services)	Corporate infrastructure (e.g. ERP capability)
Shared (by)	Potentially any application, service or user on earth	Primarily companies within the sector (including their employees), but also customer and suppliers	Primarily units and employees within the corporation, but also suppliers, customers and partners
Evolving	By adding services and computers to the network since the first packet switching network linking a couplet of computers were established	By exchanging new types of information among the users and by involving more organizations	By integrating more applications with each other, by introducing new applications
Heterogeneous	Many sub infrastructures, different version of standards, service providers, etc.	Multiplicity of competing and overlapping sub-infrastructures, standards, service providers, etc.	Multiplicity of applications and sub-infra structures, users, services etc.
Installed base	The current Internet)	All current integrated services)	Corporate infrastructure (e.g. ERP capability)

In table one you can see the properties described in infrastructures of different scales and what the term Information Infrastructure may be regarded in the different contexts.

2.2 Designing Theory for Information Infrastructure

In order to see the Information Infrastructure in action, I have include a table describing the different parts of the design theory and as well as Design principles for building IIs. The kernel theory includes many key strategies for success.

Table 2: Components of a IS design theory for IIs (Hanseth & Lythinen, 2005)

Requirements/ goals	Grow the installed base as to obtain momentum (section 5) Manage flexibility and offer openness for evolution
A set of system features	Evolving, shared, heterogeneous set of an installed base IT capability among a community of users (section 3.1.)
Kernel Theory	Complexity theory, evolutionary economics (section 4) • Enable organic growth and new combinations • Gain momentum • Recognize path dependency • Create lock-in through positive network externalities • Use modularity to offer organic growth and evolution
Design principles	A codification of five design principles which when applied will increase the likelihood of achieving a desired set of system features i.e. managed complexity, openness and growth in the installed base (section 5) • Design initially for usefulness • Draw upon existing installed bases • Expand installed base by persuasive tactics • Make if simple • Modularize by building separately key functions of each infrastructure, use layering, and gateway

2.2.1 Gain momentum

Gaining momentum is explicitly mentioned in theory as a key to success. Gaining momentum is about enrolling new users, expanding the infrastructure in scale and scope. An infrastructure's value is dependent on the size of it user base because of sharing and standardisation.

The first challenge is to find the first users (Gladwell, 2002), the persons or organisation willing to risk failure and try something unknown. There is also a matter of expenses, the value of the infrastructure increases with scale, thus the first users experience little of the infrastructure's true value while paying a lot of the bill. Hence drawing upon the install base for users as well as useful components is an important part of creating a cheap useful solution. The latter lessen the risk of becoming a 'first' user.

2.2.2 Critical mass and lock-in

An infrastructure may start with only a few users, and grow slowly until they at some point in time may reach a tipping point as described by Gladwell (2002). The 'tipping point' term is used in epidemiology, to describe the point in time when the number of "infected individuals" suddenly increases with exponential growth. The same idea is described in II using the term critical mass. When the infrastructure growth reaches critical mass, then its own momentum is high enough to continue to enrol new users. When the user base has reached the critical mass then, the changes are considered irreversible which is what we refer to as a lock-in situation.

A lock-in is a term describing the situation of when a technology has reached such a high value for the users, so it simply can not be removed. The implementation is irreversible. Creating a standard is based on the intention to create lock-in or vice versa.

2.2.3 Path Dependency

The path dependency describes an important (often chronological) aspect of IIs and is related to the installed base. In our particular case the installed base of DHIS2 have a path dependency to the java platform and the frameworks we have chosen to use. Thus we need to seek java developers in order to proceed with the implementation. The choice of Java and accompanying frameworks are again linked to the knowledge base and interest of the Norwegian master students at the University of Oslo.

2.3 Knowledge as infrastructure

In the former subchapter, I described Information Infrastructure theory as a way to describe the HISP network and the DHIS2 application. In "Knowledge as Infrastructure" Hanseth (2004) argues that there are great similarities between network and infrastructure theory and knowledge theories. There exist several paradigms for how knowledge is created, stored and transferred between individuals. Hanseth (2004) argues that actor network theory(ANT) and information infrastructure is not limited to only one of the paradigms but draw on all of them(ibid).

> *Hanseth find the notion of infrastructure appropriate as a way of underscoring the systemic character of knowledge, and the fact that this also makes knowledge big, heavy, and rigid - and not light and flexible as is often wrongly assumed.*

In order to use II theory as a way to understand knowledge it will be useful to reframe the meaning of IIs properties to the domain of knowledge.

2.3.1 Installed base

The installed base refers to the concepts you already know. In the same way new physical infrastructure is built upon old infrastructure. The new gained knowledge is built upon knowledge. What you can learn depends on what you already know, your networks, access to information, books, websites, other resources, your expectations, believes and so on.

2.3.2 Standards

Standards are a useful and necessary tool to share knowledge. An example of knowledge standards, are languages, alphabets, terminologies and concepts. Hence some common understanding is necessary for us to understand the meaning concepts and ideas. "Translation" can act as a gateway between different standards and domains.

> *Thus, if network externalities and increasing returns are seen to apply to knowledge, then the standardization of knowledge becomes beneficial, although - just as in the case of technologies - standardizing knowledge will, of course, also have its drawbacks. We can say that the value of a standard is primarily determined by externalities. A standard cannot be used for anything in itself, as it enables interaction only with other adopters of the same standard. Accordingly, the value of a standard is the number of adopters in its installed base (Hanseth, 2004).*

As Hanseth points out, the value standards are based on the numbers of users. If a standard has high diffusion, then the value of adopting the standard is higher, thus favourable.

2.3.3 Evolving

Knowledge is evolving. A common learning technique is the metaphor (Sfard, 1998, pp2). The student tries to picture a new concept using the properties and boundaries of something familiar in order of gain understanding.

> *Because metaphors bring with them certain well-defined expectation as to the possible features of target concepts, the choice of a metaphor is a highly consequential decision. Different metaphors may lead to different ways of thinking and to different activities (Sfard, 1998, pp2).*

Hence an idea of possible properties of the new concept may easily emerge, thus what you learn is in some ways dependent on what you know (ibid).

> *It is also noteworthy that metaphors are a double-edged sword: On one hand, as a basic mechanism behind any conceptualization, they are what make our abstract (and scientific) thinking possible on the other hand, they keep human imagination within the confines of our former experience and conceptions.*

The knowledge of a concept is growing as new properties emerge in different contexts, where the flaws or inaccuracy of the metaphor can be uncovered.

Hanseth argues in "Knowledge as an infrastructure" (2004) that there are great similarities between infrastructure in regard to standard and paradigm shifts. The way paradigms shifts may emerge can be separated in three categories:

1. Extensions; build out the existing infrastructure/standards/paradigms
2. Moderate; shift out parts of the infrastructure/standards/paradigms
3. Radical; total revolution, throw out everything, and replace it with something new.

The two first options underline the evolving property, the last one however only occur rarely and may also actually build on a path dependency which ultimately leads the to the paradigm shift, thus the "revolution" is also apart of the evolving property (ibid).

2.3.4 Heterogeneity

In our heritage from the natural science we are often seeking the abstract and universal knowledge, under the belief that there is such a thing. From this heritage we have adapted the idea of using simple models with limited coupling between components. The models are very useful for giving a shallow but often adequate description of reality. But a model is not "reality" and knowledge is not simple.

> *Knowledge in its purest form is viewed as isolated and independent from other parts and aspects of reality. Of course, this is not how things are. By adopting the perspective of communities-of-practice, we can see that knowledge is inseparable from our working (and other) practices; more strongly, we can see that knowledge is practice (Hanseth, 2004).*

As knowledge in its purest form is what we call “abstract” which represents universal truths, reality however is closer to tacit or “first hand” knowledge. With other words “knowledge is practise”.

> *Further, practices are based on specific tools and technologies, organizing structures, various kinds of institutions, and other important factors. Knowledge is also dependent upon, and tied to, the context within which its epistemic culture and machinery is produced (Hanseth, 2004).*

Knowledge is built up of many different components, and particularly in this case study the knowledge is partially thought through some general purpose training with a rigid point of origin and destination, while other parts of the capacity building is of a more experimental manner where the “target” knowledge is distributed between the different Indian developers. The knowledge is available through several different mediums; written, spoken, hidden (unspoken, passive, in side a developers mind), pair programming (practice), and in different languages; Norwegian, English, Hindu.

2.4 Best Practices

The experience gained during my time in the DHIS2 project thought me to use many different (developer) best practices[13], in addition to domain specific HISP best practices. Each one of them seemed to add nice features which made different part of the implementation job much simpler. I really don’t see a single practice, which don’t add great value to the project or in other ways seem redundant. In other words, all the best practices seem to be beneficial. So how come we are still struggling with simple implementation issues?

The situation today is that we are failing to train developers in India. In order to discover why we are failing, it is probably useful to look into the technology we are using and known problems with this technology.

[13] Definition of best practice: A superior method or innovative practice that contributes to the improved performance of an organization, usually recognized as "best" by other peer organizations. www.asq.org/glossary/b.html

To the best of my knowledge there exist no articles on the subject of using multiple software frameworks[14] and difficulties at all. The frameworks we use are rather new, and the appliance of multiple frameworks in development is also a rather new endeavour. The latter might explain the lack articles on the subject.

A software framework (including the frameworks used in our case study) can be described as predefined skeletons (Soundarajan, 1999) which are designed in order to simplify implementation of solutions to a particular problem. In our case the frameworks are manifestations of different design patterns[15] and architectures (Chapter 5). So in order to examine difficulties with the framework, I will look upon studies done on frameworks and design patterns, and assume that the known weaknesses and problems in design patterns is also reflected in their framework implementations, thus related to our capacity building effort.

Design patterns are best design practices to which is used to cope with known design issues. A design pattern applies a general solution to a recurring problem. There exists many different design patterns, but what is more interesting for us is how the frameworks used in the DHIS 2.0 project in many ways are a manifestation of different design patterns and architectures.

Though there is little theory on this subject, in this thesis will look into what happens when an organisation uses many frameworks (design patterns + components[16]) in order to simplify their code base.

The relevance of frameworks with regard to capacity building can be viewed in some interesting quotes form a case study on frameworks. “You can often tell that a class library is

---

[14] Definition of framework: A skeletal software component that performs functions required by a system and which is incorporated into the design of such systems. javaworkshop.sourceforge.net/glossary.html

[15] Definition of Design Pattern: A design pattern is simply a defined and repeatable design solution to a common programming problem. Design patterns aren’t code per se, but they define a proven recipe for creating the code to solve a given problem. I’ll go over these in more detail in a later article. (en.wikipedia.org/wiki/Design pattern (computer science)

[16] Definiton of Component: A piece of software with a clear function that can be isolated and replaced by another component with equivalent functionality. javaworkshop.sourceforge.net/glossary.html

a framework if there are dependencies among its components and if programmers who are learning it complain about its complexity (Johnson, 1997, pp. 39-42)"

"Frameworks are more customizable than most components, and have more complex interfaces. Programmers must learn these interfaces before they can use the framework, and, consequentially, learning a new framework is hard (ibid)".

During my experience with the frameworks we use in DHIS2, I would claim that the difficulty of understanding and learning to use a framework efficiently is greatly underestimated. In order to use a framework the developer needs to spend a lot of time learning how to use the basic functions of the framework, but there may take even more time before the developer reach a proficient level of understanding of the framework and can work at optimal speed.

If we look on a framework in view of the Information Infrastructure (II) and knowledge infrastructure (KI) theory. Then we can recognise a number of properties related to both theories, thus I choose to se the 'frameworks' as both an II and a KI.

The framework has an II installed base consisting of code examples, active projects, the framework implementation code, earlier versions, user base, developers, competing frameworks and technologies, complementary frameworks and technology. In the KI's installed base, we can identify, design patterns, architectures, user experience, developer experience, terminology, methodology, explicit knowledge such as documentation and articles and last but not least tacit knowledge.

It is common that the frameworks are evolving over time, both in scope and scale. The frameworks mature (evolves) over time as the different bugs are removed and new features are added. While the knowledge base may grow and change over time in terms of user knowledge and available documentation articles, books, presentations, videos, communities, education facilities, etc.

The most interesting II & KI property is how the frameworks may act as a standard if it reaches the critical momentum. The framework as an application can be viewed as an explicit implementation of design patterns which outline how the patterns and frameworks should be

used combined, with a large user base this may acts as a ‘de facto’ standard. On the KI side the developers knowledge of the framework can be moved from project to project thus the knowledge of a framework can act as a knowledge standard.

The framework is Open by definition, since it only acts as a skeleton for other applications. We can replace or extend parts of the frameworks with similar features from other tools or frameworks. From a KI view, the knowledge of the framework may be combined with knowledge about other frameworks and tools. This has in our case manifested itself in use of multiple frameworks and templates.

In our case Hibernate, Spring and WebWork are also heterogeneous since they are linked together with each other in addition to support other technologies (like velocity). From a KI perspective we may look upon the many different design patterns and architectures making up the frameworks as a heterogeneous knowledge infrastructure, I could also argue that technologies like hibernate bridge between two different domains with object orientation on one side and relation databases on the other.

Of the former mentioned properties, the installed base, and the standard property is of special interest to us. In particular how these still are bound to local contexts even in the global world of IT development. Because of the popularity of the frameworks in Norwegian IT industry, we thought we should be able to benefit from using this standard. We recognise a great number of benefits using the framework may yield, but as a part of the discussion we will focus on some ways the frameworks are holding us back with regard to capacity building.

2.5 Open Source Software (OSS)

Open Source Software started out with the open standards of the Personal Computers (PC), this enabled anyone to write a piece of software, and distribute it to others. The development of proprietary software and their restrictive licences challenged a number of ideologists who strongly believed that all software should be *free*. Free in this context regards to four defined properties of *freedom* listed below the quote.

> *The word "free" in our name does not refer to price; it refers to freedom. First, the freedom to copy a program and redistribute*

> *it to your neighbours, so that they can use it as well as you. Second, the freedom to change a program, so that you can control it instead of it controlling you; for this, the source code must be made available to you. (Stallman, 1986, p8)*

The definition was later rewritten to the list posted at the Free Software Foundations webpage

The freedom to run the program, for any purpose (freedom 0).
The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
The freedom to redistribute copies so you can help your neighbour (freedom 2).
The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.

As the definition clearly states, there is a significant difference between giving away the software for free, and preserving the freedom rights of the users. Though expressed clearly in the definitions, many people "unfamiliar" with the OSS definition presume the software cost nothing either (which is often the case), thus making it hard to sell free software solutions. The free software had less success than the potential summon. One of the key issues for the lukewarm response to free software is the common assumptions of free products. Free products are often associated we poor quality and trickery (old habits die hard). In order to change the general opinion of free software in the market, a new business marketing strategy (Michael Tiemann, 2006) was attempted in 1998.

> *The immediate chain of events that was to lead to the formation of OSI began with the publication of Eric Raymond's paper The Cathedral and the Bazaar in 1997 (Tiemann, 2006).*

> *The 'open source' label was invented at a strategy session held on February 3rd, 1998 in Palo Alto, California. The people present included Todd Anderson, Chris Peterson (of the Foresight Institute), John "maddog" Hall and Larry Augustin (both of Linux International), Sam Ockman (of the Silicon Valley Linux User's Group), Michael Tiemann, and Eric Raymond (Tiemann, 2006).*

The new label was made in order to market the products without facing the prejudice associated with the "free" label. In addition the Open Source Definition (OSD) is more pragmatic and therefore suits business needs better than the ideological perspectives of FSF. The Open Source Initiative (OSI) was jointly founded by Eric Raymond and Bruce Perens in late February 1998.

OSS is in many ways presented as a philosophy, but if we are going to compare it to proprietary software, we must first acknowledge that OSS also functions as a business model.

2.6 Open Source Software Development model

The open source software development model is driven by user interaction. The users of open source software are referred to as the user community. The community is playing a significant role in the OSS development process. Through the interaction with the community new requirements are discovered.

The common strategy is to start making a simple prototype, release it to the public, get feedback from the community, change the product, release it again, and so forth.
Since all open source projects give away the source code, the users may track and fix bugs themselves. The ideal thought is that the bug and possible bug fix is sent back to the community and distributed among the other users. In the best case scenario users decides to participate as developers in the development process and thus speed up the development and secure human resources to continued development of the software (Raymond, 2000).

The number of users of an open source software product grows fast beyond the number of quality testers a proprietary developer can afford to hire. The valuable feedback from the users is the key for further requirements, and the unrestricted numbers of code reviewers are likely to discover most mistakes and security issues. The latter originate from form one of the most famous informal comparison studies of two OSS projects by Eric Raymond in the article "The cathedral and the Bazaar".

According to Raymond (2000, pp 8) Linus Thorvaldsen followed the philosophy "Given enough eyeballs, all bugs are shallow" thus "many eyeballs tame complexity" (ibid). The latter is dubbed Linus Law by Raymond and summarise the main philosophy behind the success.

The project was driven by a simple iterative strategy based on user interaction, where Thorvaldsen was exploiting his own "constructive laziness" and apparently always looking

for the minimum effort path from point A to point B (Raymond, 2000, pp8). Thorvaldsen was directly aiming to maximize the number of person-hours thrown at debugging and development, even at the possible cost of instability in the code and user-base burnout if any serious bug proved intractable (ibid).

Raymond's finding are interesting for us, though we aim DHIS2 to have its own OSS community, we have in many ways designed the software following a more rigours approach. The key to success according to Raymond is following the minimum effort path. If you are aiming at drawing developer aid from the OSS community, then it must be possible to check out the code, understand it and contribute within a short time span. Our extravagant design limits the possibility of finding motivated people with tool, framework and domain knowledge, who have the time and will to contribute. A former student and DHIS2 participator said:

> *"Learning the tools and frameworks was quite interesting, but working on the project demanded so much effort, I could not pick it up during an evening and just program a little and then check it in again, but I guess this is how it is when you want to contribute these days (Former INF5750 student, Jan 2008)".*

We have tried to address this issue of complexity by using frameworks, modularisation, three tier architecture and code conventions so the code it self is clean and simple, but the development environment has become complex.

2.7 Stronghold of Open Source

Traditional Open Source Software communities are dependent on a large knowledgeable user community to thrive. The communities with this the latter property is most likely to develop where the knowledgeable users and the developers originate from the same domain. Thus the domain of software development is a prime domain for OSS development. The latter statement is proven by the great number of excellent OSS software produced by developers for developers.

So far, there has been written few articles which suggest there is a link between the different domains and development of OSS tools, but the complexity of domain, tools and frameworks has not been mentioned with regard to active OSS domains. From my data I can presume that there is also a link between the initial knowledge necessary to participate in a project and, the appeal the project have on OSS developers. The OSS community originated from the early

hacker community (Lin, 2003), and the possibility to pick up some code, and manipulate and tweak the code is fundamental for hackers will to join an OSS community (Raymond, 2000, p 6).

3 Methodology

In this chapter I describe the research methods used. The research methods were chosen after suggestion from my supervisor. I have not at any earlier point studied the different research methods available in social science, so my approach turns in a direction of a very simple data-oriented approach. Later I have learned that this approach is known as Grounded theory (Silverman, 2004).

In fact I have used 2 different approaches during the scope of the master thesis. For the purpose of data collection and participation in the DHIS2 the author have relied on the Action Research method, but in the part of analysis and discussion of the data a grounded approach have been used.

3.1 Research methods

Most of the data collection was done in India in the spring semester of 2007. Initially I was working on an integration solution between the DHIS 2.0 software and two other HIS software solutions IDSP & RIMS (see appendix), which are used by different health promotion programs in India. During this work and taking the Course INF5750 I have gained some experience about the architecture and technology used in DHIS 2.0.

We tried to achieve this using a process known as action research. The reminder of this chapter will contain a short description of qualitative research, case study and action research. Furthermore I will describe in detail the different data collection methods which we used, and try to identify possible bias and validity of the data.

3.2 Qualitative Research methods

I have discovered that there are numerous different qualitative research methods. It is difficult to categorise and to identify the boundaries between the different methods, since there exist different views of which categories we have and what category each method is a part of, even within the same book. In order to investigate this topic I started with the *philosophical perspectives* as described by Myers and Avison (2002). Furthermore Myers and Avison

(2002, p. 5) suggest three distinct epistemological categories; *positivist*[17]*, interpretive*[18] *and critical*[19]. Note that this is an ideal distinction and not always so clear cut (ibid). Myers and Avison (2002) further argue that *qualitative search* do not necessary imply *interpretative* research. Whether interpretative research is used depends on the philosophical view of the researcher. A qualitative researcher may be a positivist, interpretive or critical. It follows that the choice of research methodology is independent of philosophical views.

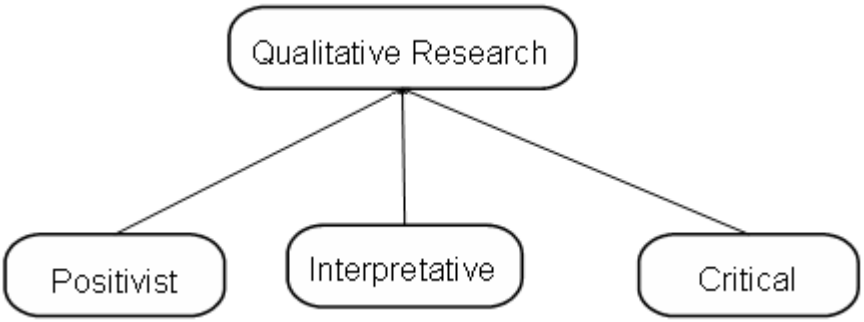


Figure 1 Underlying philosophical assumptions (Meyer, Avison, 2002, p. 6)

3.3 Case study

According to Myers and Avison (2002) there is no standard definition of a case study. However the case study described by Benbasat et al (1987) examines a phenomenon in its natural setting, and it employs multiple methods for data collection to gather information from one or a few entities. Furthermore Benbasat et al (1987) have identified three kinds of case studies; Application description, Action research and Case study research. The action research (AR) methodology, used in this thesis, will be described in the rest of the chapter.

3.4 Research Approach

I and my fellow master student had a loose agreement with our supervisor that we were supposed to teach our Indian colleagues the necessary tools and architecture, and empower them to be able to develop on the DHIS 2.0 software in a proper manner. We tried to achieve the capacity building with "one to one" sessions with some of the developers over a period of

[17] A positivist studies generally try to test theory, in an attempt to increase the predictive understanding of a phenomena.
[18] The interpretative researcher assumes that access to reality is only given through social constructions such as language, consciousness and shared meanings.
[19] The critical researcher assumes that social reality is historically constituted and acknowledges that people's choices are influenced by their social, cultural, economical and political situation.

about 1.5 months, as well as holding a 7 day workshop for the HISP-India developers from all the states, which at the that time numbered 7 developers (See Timeline).

At the 1st of March the HISP India govern board member approached us with the suggestion to hold a workshop when we were in India. Initially it was suppose to take place in Kerela in the beginning of May. In the mean time we were supposed to work with some of the members in the HISP India team, and during collaboration exchange knowledge in a kind pair programming fashion (Beck, 2002). The latter would secure us the necessary domain knowledge and enable us to explore the India specific requirements and solutions.

3.4.1 Timeline

Below is an illustration of my work by the use of a timeline. On the timeline I have noted down some key events. Among these is my first meeting with HISP, as well as the field trip and the continued capacity building as a teacher assistant in the INF5750 course at the University of Oslo.

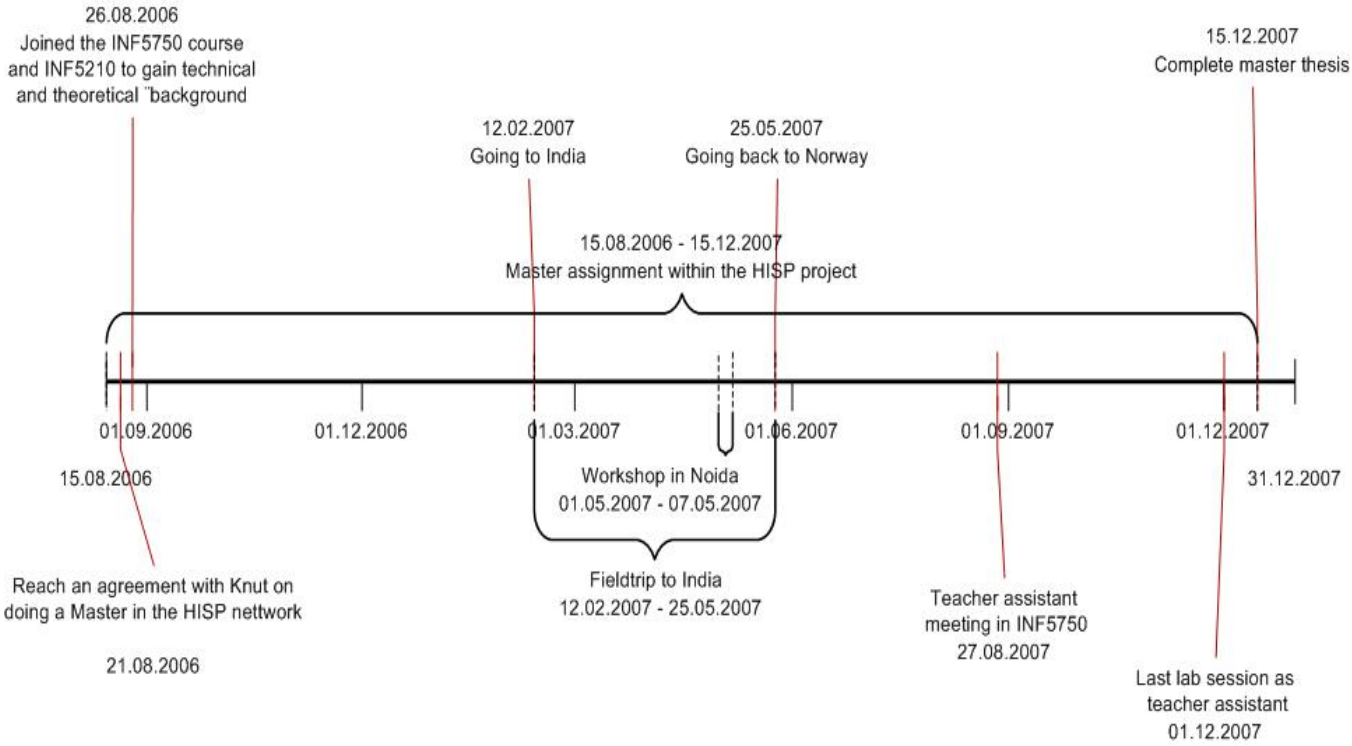


Figure 2 – The action research time line

As you may haven noticed the original plan was extended with a few more months of writing, however no further 'action' with regard to the project has been initiated by the author at a later after the original deadline.

3.5 Action research

Action research is a very wide definition, and one of the main reasons for this lies in the nature of action research itself. The action researcher is not only a professional academic researcher, since the researcher may also be the subject of the research. The latter implies that action researchers may come from very different backgrounds. As a result we have a wide range of research methodologies within the AR sphere (Herr & Anderson, 2005, pp. 1-7). The Action Research (AR) term serves as an umbrella for many methodologies which has action as a central step in their research cycle.

As opposed to traditionally research methods, the AR advocates action and change. The typical action researcher may have an academic background (outsider) or be a member of the community under investigation (insider). In both cases there are major pros and cons. The position of the researcher influences the view of the situation, and thus which power relations and strengths that is visible to the researcher (Herr, Anderson, 2005, pp 1-7).

Argyris and Schön (1991) describe the goal and methods of action research from an organisational and professional point of view.

> *Action research takes it cues-it's questions, puzzles, and problems – from the perceptions of the practitioners within a particular, local practice context. It bounds episodes of research according to the boundaries of the local context. It builds descriptions and theories within the practice context itself and tests them there. Through intervention experiments – that is, through experiments that bear the double burden of testing hypotheses and effection some (putatively) desired change in the situation. (ibid)*

The specifics of action research methodology in the information system sphere will be discussed later in this chapter.

3.5.1 Action research history

Action research is a new discipline within dissertations. Historically the action researchers were academics which involved research participants in their studies to a much greater extent than what was typical in traditional research (Herr & Anderson, 2005, pp. 1-7). Some view the work of Kurt Lewin and the group dynamics movement of the 1940's as the beginning of the Action Research methodology. Lewin was not the first to conduct this kind of research, but he was the first who made it a respectable form of research in the social sciences (Herr & Anderson, 2005, p. 11). Until the 1940's industrialist believed that workers, just as machines, could be manipulated by simple mechanics, such as adjusting the light in the workplace and so forth.

The 1940's human relations movement's critique of Taylorism was widely known and many industrialists were experimenting with these more worker centred participatory techniques (Herr & Anderson, 2005, pp.12). This was the birth of Action research. During the decades AR have been known by many names, such as Action science (Chris Argyris), Participatory

Research (Paulo Freire), Participatory Evaluation, Action Research in Education, The Teacher-Researcher Movement in Britain and The Practitioner Research Movement in North America (Herr & Anderson, 2005).

3.5.2 Action research Cycle

According to Herr and Anderson (2005) the action cycle consists of four phases[20].

1. to develop a *plan* of action to improve what is already happening;

2. to *act* to implement the plan;

3. to *observe* the effects of the action in the context in which others occurs;

4. to reflect on these effects as a basis for further planning, subsequent action and so on, through a succession of cycles. (Kemmis, 1982 p.7)

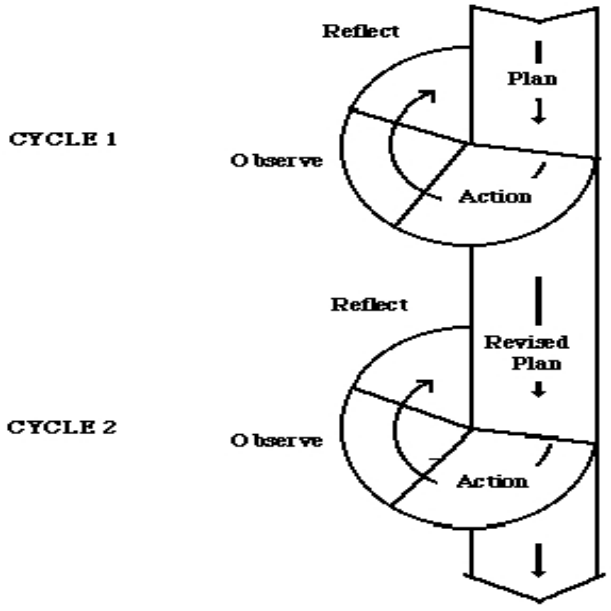


Figure 3 : Action Research Protocol after Kemmis cited in Hopkins (Gable, 1995)

[20] There also exits other definitions of the action research cycle, which includes 5 and six steps.

Below I have included a more detailed illustration of the action research cycle as viewed by Elliot (cited in Hopkins, 1985). The figure below reflects on a more refined, but also a more constrained model of the action research cycle.

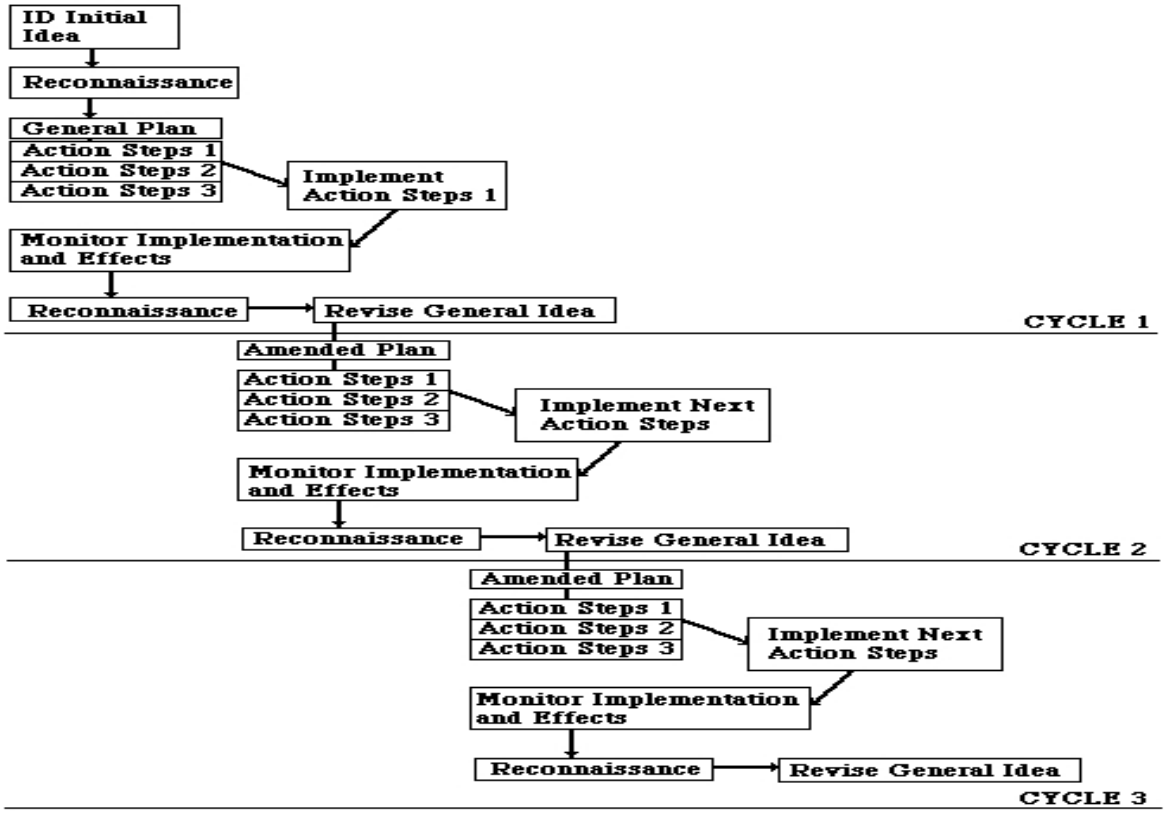


Figure 4 : Action Research after Elliott cited in Hopkins (Gable, 1995)

In the next chapter we will look further into the use of Action research in an Information Systems (IS) context.

3.5.3 Action research in Information HISP

The discipline of IS seems to be a very appropriate field for the use of action research methods. IS is a highly applied field almost vocational in nature (Baskerville & Wood-Haper, 1987).

Action research in Information Systems differ a little from the use of action research in other fields of science (psychology, education, health care), but this is natural since the contexts are different. The action research in the sphere of Information systems focus on human and (computer) system interaction. One key issue is how knowledge is generated and transferred and sustained within an organisation context with regard to development of new and old

systems. Another issue is how and what Information Technology (IT) and Information Communication Technology (ICT) can support older existing systems as well as change them.

> *We can not study a newly created methodology or technique without intervening in some way to inject the newly invented technique into the practitioner environment, that is go to the world an try them out (Land cited in Baskerville and Wood-harper, 1987).*

> *This lead us to conclude the that action research is one of the few valid research approaches that researchers can legitimately employ to study the effects of specific alterations in system development methodologies. It is both rigorous and relevant. (Baskerville, Wood-harper, 1987)*

In HISP one of the cornerstones of former success is the ability to bring immediate change and improvement to the situation. In this particular case type, action research is not only the best available option, but the only research method used for causing intervention and change to the subject of the research. In the HISP cases changes of political support caused by changes in the government is a relevant issue which happened in Cuba and India, (Braa et al, 2004). Thus we can deduce that the time span necessary to perform the research and applying the changes is a key factor for the success, not only because of the visible link between the research and the improvement, but more importantly that the action has less risk of being blocked by external interruptions caused by political changes in the domain.

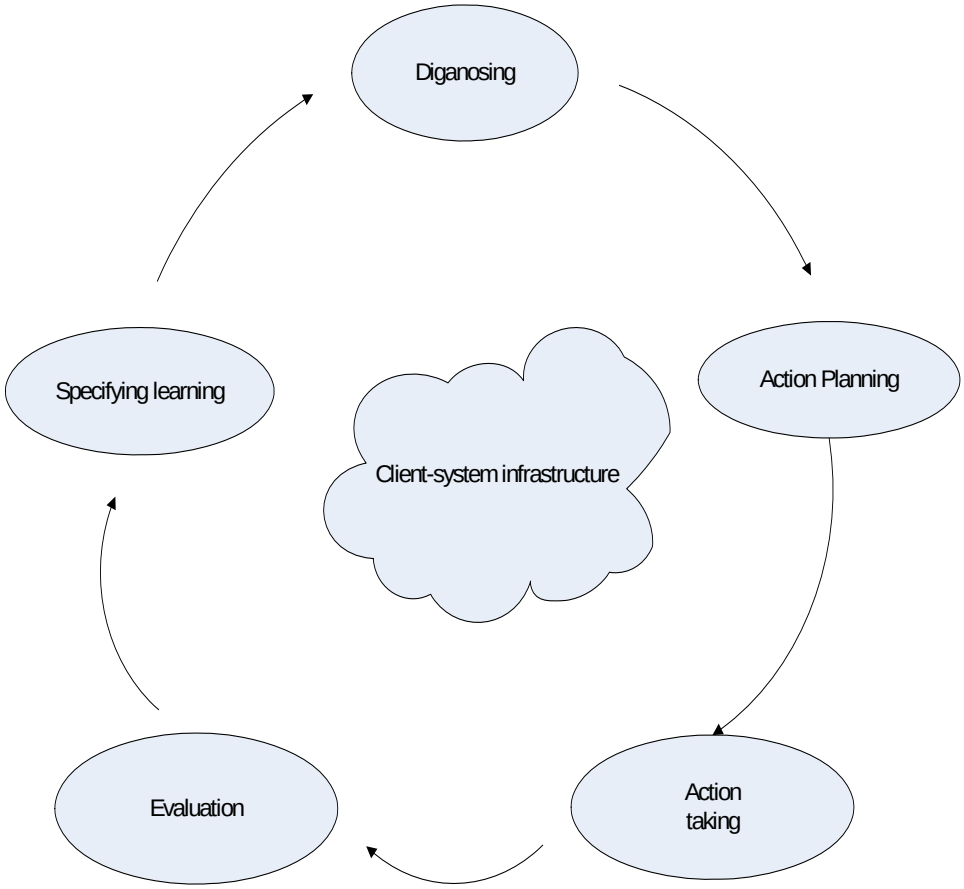


Figure 5

The figure above (figure 5) is based on the cycle described in Susman (1983, cited in Myers and Avsion, 2002, p134). The cycle matches very well with tasks at hand, and it will therefore be the foundation of the research design. There is however a matter of how we should look on the action research conducted.

The capacity building for the Indian developers of DHIS 2.0 was initialized by a former HISP master student. We continued on the work he started and draw upon the earlier experience. Thus we define the action research as part of an existing cycle of action research, which hopefully will continue for many years after I have completed this thesis. This is illustrated in the model below. Where the cycles '*i*' too '*i+2*' have been conducted by the author, while the other iterations are from pre and post researchers.

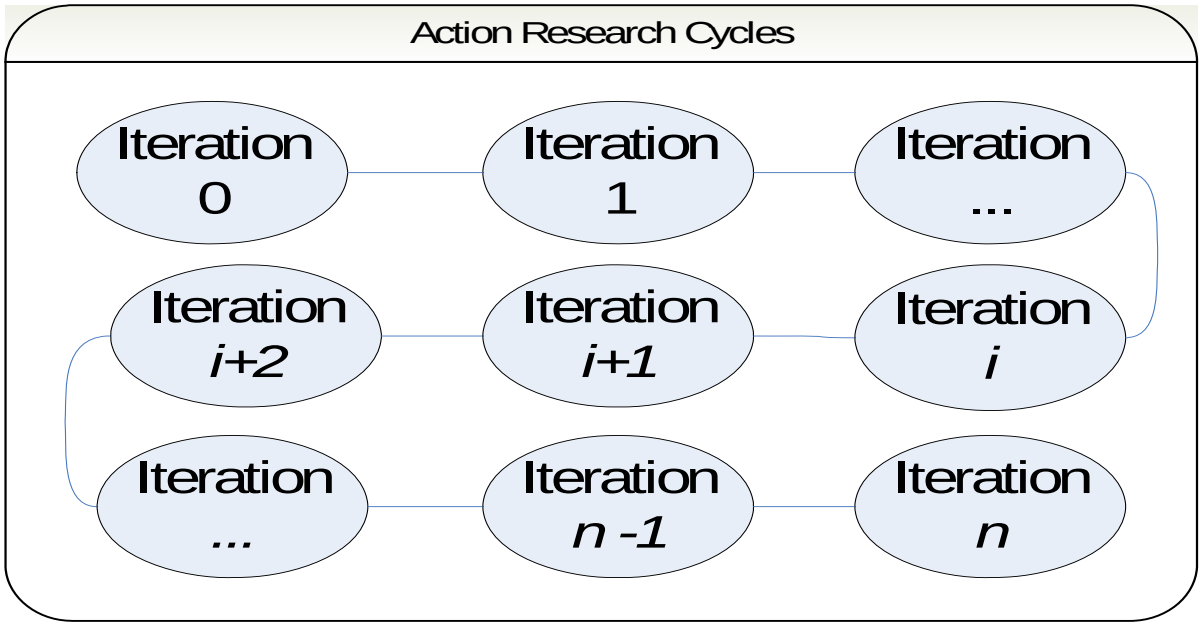


Figure 6 - Research cycles

My particular case contains 3 cycles. The research design is described below.

Iteration I

1. Diagnosing, draw upon earlier experience, and get an overview of the situation. Explore the knowledge base of the participants and define the goals of the capacity building.

2. Action planning – define a useable capacity building framework and define exercises or schemes

3. Action – use the framework to initialise training which include reviewable exercises

4. Evaluation – Review the work done and correct mistakes

5. Specify training – as part of the review inquire about the exercises to discover what knowledge was gained

Iteration II: workshop

1. Diagnosing – explore the knowledge base of their participants and the lesson learned from the first iteration

2. Action planning – Customise the framework to the particular situation.

3. Action – conduct the workshop and make use of existing resources like the participants trained in iteration I.

4. Evaluation – Involve the participators in the evaluation, as well as reviewing the exercises.

5. Specify learning - Review the knowledge gained through an evaluation of the exercises and an oral presentation by the participants.

Iteration III: Graphical Analyser

1. Diagnosing – explore the current interface and the possibilities for integrating the GA in the DHIS 2.0. Exchange enough domain knowledge for the Indian developers to understand the API of DHIS 2.0 and the Norwegian master students to understand the GA.

2. Action planning – explore possible solutions for integrating the GA

3. Action – Integrate the GA

4. Evaluation – Try out the GA

5. Specify learning - Through the work of the integration we explore the domain knowledge of the Indian developer, through the terminology learned in the 2 former iterations. Specify the extension of old knowledge as well as the new one gained.

3.6 Fieldwork

3.6.1 Trips

During my master thesis I did a 3.5 months field trip to India. The trip had four major goals regarding the work on my thesis; to see the Application used in the field, to meet and work with the HISP team in India, to complete the module I was working on as part of my master and to hold a workshop for my fellow HISP members in India.

The trip brought me to 2 of HISP India's state development offices (Gujarat and Bhopal), and the main national office in Noida. During the workshop we met developers from Jarkhand and Kerela as well.

3.6.2 Teacher assistant

During the fall of 2007 I acted as a teacher assistant in the Open Source Software course INF5750 (2007) at the University of Oslo. This course focuses on the HISP organisation, the particular development of DHIS 2.0 and teaching of necessary tools. This job was not very relevant for my case study in India, except the fact that two of my associates from India attend the course this fall. Thus makes it possible for me to follow up on the capacity building and nurture social bonds.

3.7 Data Collection

Data collection is the foundation of any research and in my case study I used field notes (or log). The data in the log is partially based on observations and informal inquiries and few unstructured interviews. The scale of the case study is limited to only one country. I will do a few more interviews with key personnel, in order to strengthen my data base.

I have categorized my data collection activity using the four categories described by Creswell (2002, pp. 185-186).

3.7.1 Observation

Observation is one of the data collection methods available to the qualitative researcher. The researcher may take field notes on the behaviour of individuals at the research site, in an unstructured or semi structured manner. The latter implies use of some prepared questions. The researcher may also have different roles in the project, reaching from just an observer to a participator in the project (Creswell, 2003, pp.185-186).

The observation method was used on a daily basis, and we tried to capture a rich picture of the situation since we were not sure of what data which would prove relevant at the time. I continued to do this kind of data collection when I returned back to Oslo in order to recognize and retrace the origin of influences as well as other information.

3.7.2 Semi structured interviews

I chose to use face to face interviews when that was an option. However, some of the interviews were done by email or instant chat systems. The latter is more robust and cheaper than phone calls. The backside is loss of spontaneous answers as well as some possible misunderstanding of the subject's intended meaning. The interviews were used to scrutinize the key issue which my research question is built upon. I intended to fill some gaps from the

field studies by doing formal semi structured interviews. In addition to filling the gaps, the interviews broaden the understanding of the domain in question. I selected some key personnel, which includes one central coordinator, the main developer and two state developers in India for this particular inquiry.

3.7.3 Document analysis

According to Creswell (2002) the fourth category of data collection in qualitative research is the document analysis. The document analysis includes public documents, news, meeting minutes, reports, e-mails and personal notes. I chose to include books and articles under the category of ”public documents”, though the latter is not explicitly included by Creswell (2002). All of the data used in this thesis is captured in written documents, where the field work log is the main source of this data. In addition I have received several hundred e-mails as well as written a great deal myself during the entire time span of the thesis. I also studied some emails from the initial analyse and design phase of DHIS2 in 2004. The third most important written source of information is books and articles.

3.8 Role as Researcher Creswell (2003)

As part of my research I participated in an OSS development project of a HIS in India. This project is the source of most of my field data. In this matter I have had several different interests and views. From the personal view I would like to experience India, which means travelling around and see some of the tourist sights, meet new friends and have some fun as well’s as collecting data for an interesting master thesis. During the field study I also took a course at the University of Oslo.

From an educational and professional view my efforts was focused on two tasks assigned to me by my supervisor in Norway. One the one hand, I needed to develop a module for HISP-India dealing with Integration with other software, see the modules used in the field, and in addition I was supposed to conduct capacity building in HISP-India in form of developer education/training.

As a member of HISP I was a participator of the global HISP network which in some ways is an external stakeholder in view of HISP-India, whose mainly focus lies aligned with the India government interests. The latter means that my HISP related views and beliefs do not necessary align with the interests of HISP-India, or the particular developers which I was working with.

At last it is worth to mention that my ethnic background (Norwegian) gave me a certain set of assumptions about how the project work was suppose to be conducted, and how to behave in different settings. The cultural and language differences were likely to cause misunderstandings and limit the linguistic vivid narratives. On the other hand the foreigner factor might work as an ice breaker.

3.9 Data Analysis

As part of my data analysis I applied coding to be able to order my data and link different subjects, as well as each subject to specific instances in my data material (Silverman, 2004). For the analysis I used a technique called coding. In my case it was used to consolidate notes in different subgroups in order to get a good overview of the data, and to be able to look at more abstract relations between the topics. For the "coding" I used a tool called Atlas.TI.

3.9.1 Atlas.ti

The Atlas.ti software is designed specific for the purpose of analysing text using the coding technique. The software support Auto-coding which is a useful feature which makes some standardisation of codes possible. The Software has a code and network manger as well, where the latter makes it very easy to make networks based on one code, and then to import related codes (Silverman, 2004, pp.188 - 207).

In the coding process I used the auto-coding for capturing most of the factual data. This means data which can be captured by using a combination of key words for searching. In addition I did some "eyeballing"[21] to capture complementary data manually (the latter include hermeneutic data).

3.10 Data validation

In any qualitative study, the data collected by using a field log is biased by the researchers own expectations. I have background as a student, and thus have a very academic view of the state of the world. The latter means that I believe the textbook examples and the articles I read strongly reflect the domain they describe, but this is not always the case. Or to phrase it

[21] Eyeballing is the ancient technique of using you eyes for searching through data. This "technique" was used in India to manually validating forms and data sheets.

differently; the articles don't lie. I just don't comprehend the context dependent application of what is described because my own experiences are from completely different contexts.

To strengthen the validity of my data, I have discussed the matters (log data) with my fellow master student. He is also biased by the culture differences, but through a dialectic approach I believe we have achieved a more objective understanding of what is going on. I have also discussed my views and concerns with some of my Indian co-workers. This helped me to put things more into perspective, as well as correcting misunderstandings. The semi structured interviews is partially also done to confirm or disprove key issues from my field notes.

4 Part II Empiric

In the Empirical part I will focus on the actual events of the field trip to India, and the capacity building effort. The main focus will be on the capacity building efforts done while we were India. In this chapter, I will outline the different aspects of the issues we experienced during the field trip. The main purpose of the chapter is to give an account of the most relevant activities, events and problems. The account is task oriented and in chronological order in the first part of this chapter, while the last part is topic oriented.

4.1 The fieldwork

As explained earlier the field work was conducted during a three and a half months long stay in India in February-May 2007. During the field trip I and a fellow master student participated in; user training, developer training, trips to two HISP India state offices, meetings with many of the key participants of HISP India and development on HISP India specific DHIS2 modules.

4.1.1 Summery of the key activities of the stay in India

The stay started in Delhi where I and my fellow student spent 9 days, we were supposed to meet someone from HISP India (See appendix B for an overview of HISP India) the first day, but the meeting was postponed due to logistic complications. We met one of the members of HISP India govern board the third day, and the HISP India Cordinator1 came on the fourth and sixth day. There was supposed to be an office in Delhi by the time we got there. However it was not ready when we arrived. Instead we went on a field trip to Gujarat[22], where we met some of the HISP India developers and managers.

4.1.2 Gujarat

We stayed primarily in the vicinity of the Gujarat State Office in Gandhi Nagar, but participated in two DHIS2 user training sessions in two district offices in Ahmedabad and Surendra Nagar.

We spent 3 weeks in Gujarat and stayed at a Guest House, a 10 minutes walk from the Gujarat state healthcare office, where HISP India had access to some office space (this is about as far

[22] Gujarat is a large state on the west side of India.

as we felt comfortable walking in the very hot Gujarati climate). However we did not spend much time at the main HISP office, due to lack of continuous access to computers and internet (we were not allowed to connect our own laptops to the internet). We usually worked at an Internet café about a 10 min drive from the Main office and our guest house. The situation limited our communication with HISP India team. This basically meant that we were not working "with" the HISP India developers the time we were in Gujarat. From my field notes I have listed a common situation which occurred the when we went to the office at the Healthcare administration centre.

> None of the HISP members were present when we arrived at the office, my fellow master student and I sat down and tried to make a plan. We decided that we should implement the changes we discussed yesterday, before we go to the Cyber Planet. (03.03.2007)

> We went to the HISP office, but there was no HISP people there, so we sat down and waited. After a while *HISP India Coordinator3* arrived and told us that there was an important meeting today. They were suppose to compare DHIS with an other HIS and discuss strengths and weaknesses in the two systems. However the meeting was cancelled, and since nothing else was scheduled today, my fellow master student and I went to Info City shopping to get some breakfast.

We achieved very little on these visits in term of work, since we were unable to do any programming or other useful work during the time we spent at the office. So instead we usually went directly to our local internet café. In retrospect I see this as very unfortunate, not only because we lost valuable time with regard to capacity building, but it also limited our ability to share actual working practices. By the latter I refer to the way we use the development tools (design, programming, solutions, communication, etc.) in our work, shortcut keys, and other small but very helpful tips and tricks and tacit knowledge.

At the beginning of March we had a short informal lunch meeting with the managers and developers of HISP India who were present in Gujarat at the time. During the lunch meeting the HISP India Govern Board member suggested that we should hold a workshop in Kerela at the end of April.

> *The HISP India govern board member proposed that my fellow master student and I should hold a workshop in the DHIS technologies (Maven, Eclipse, Subversion, Spring, Hibernate) in the end of April, beginning of may. After some discussion, The HISP India govern board member proposed that we should take the train down to Kerela and hold the workshop there (Field notes, 2007, 01.03.2007).*

Later that day, I started a discussion with my fellow master student about what we should do during the workshop. At this point in time, there were few boundaries and guidelines on what we were supposed to achieve during this workshop. The discussion boiled down to a question about the installed base, and what the HISP India developer's current knowledge, skills and practices were? After some thinking I made a list of some of the different technologies we use, which is listed in the text box below.

One other question that came forth was what level of competence can we expect them to have?

1 Basic java?
2 Eclipse(IDE)
3 Subversion(cvs)
4 Maven(ant)
5 Spring(ejb)
5 Junit(test framwork)
6 WebWork(tapestry)

As an immediate thought I would like to mention that even though we did not work together, we often socialised in the evenings. The latter does not directly yield any useful gains in terms of the technical capacity, but the socialisation process is important with regard to closing the cultural gap between us, thus making conversation and knowledge sharing in a less formal context.

As part of the preparations for the capacity building me and my fellow master student tried to get an overview of our Gujarat Gujarati colleague's knowledge base. During one our social event we asked *the HISP India from developer Gujarat* and *HISP India Coordinator3* about their technical background. *HISP India Coordinator3* for example have prior education in; COBOL, FORTRAN, C, C++, SQL, UML (Field notes, 14.03.2007). Unfortunately I did not

catch *the HISP India from developer Gujarat*'s background. Later in Bhopal we continued the inquiry.

After we had spent three weeks in Gujarat, we were ready to move on, but the office in Delhi was still not ready. So we went to Bhopal to meet the two developers who were working there.

4.1.3 Madhya Pradesh

We arrived at Bhopal[23] in midst of March, and went directly to the office (which also served as accommodations) and met the national Indian developer and the state developer. The office had obtained some extra desks as well as an internet connection, so we were able to work with the Indians at their office.

Apart from a two week vacation with my family I stayed in Bhopal from mid March until end of April. My fellow student stayed there the whole period. We were supposed to work on our modules, and do some day-to-day capacity building. The first weeks we worked on the modules we were assigned. I started the capacity building by trying to explore my colleague's knowledge base (diagnostic, 3.5.2) in order to prepare a suitable capacity building plan (action plan, 3.5.2).

At this point in time we had discussions with our supervisor and other HISP members in Norway about what to do and how to proceed (details about the training are provided later in this chapter and as an anthology in appendix D).

Due to the fact that my only real experience with tutoring is within a class room environment (both as student and teacher assistant) I had some fairly strict ideas about how tutoring should be conducted. My main idea for the capacity building effort derived from my experience as a student from the INF5750 course autumn 2006.

[23] Bhopal is the capital of Madhya Pradesh which is a state in India.

My first suggestion was to use the first 3 or 4 lectures (Subversion, Eclipse, Maven, Spring, Junit, 3-tier architecture) from inf5750 as a basis for the workshop, and the 2-3 mandatory assignments as well. This material is easy to understand, and both my fellow master student and myself are familiar with the content. We can hopefully draw on the experience accumulated to by the former teacher assistants from their last semester, in order to adapt the course material to the Indian context. (11.04.2007)

In the field notes below, I address the latter issue which occurred in the planning phase.

Cordinater2 from Oslo: By focusing on a concrete task you also avoid digging into all the details that are not relevant or that might not be needed at this stage. I also very much agree with *My Supervisor* that this workshop must be followed up by daily/weekly

In a recent email *Cordinater1 from Oslo* suggested to sit together with members of Indian team while developing specific parts of software. It will help them in learning the real practice, not only through a one-shot class (which is important, too). Local capacity building should take place and be sustained day by day. (11.04.2007)

As emphasized by the field notes, the HISP tradition is more pragmatic and advocates a "learning by doing" approach. Besides the task at hand, capacity building is a much broader term which include more than just lectures and exercises.

My Supervisor: Span the learning process through all your stay there. Do tasks with people there, otherwise a course will be largely insufficient to build the needed capacity to carry on development locally. As we wrote to your fellow master student, it's up to you both to make HISP India team able to go ahead with DHIS 2 (11.04.2007).

All advice from The HISP node in Oslo was for us to sit together and work on Indian specific tasks. The idea is okay, but at the time, I did not have the domain knowledge to contribute to the development.

The solution we chose was to begin going through some basic lecture foils and training exercises (Training exercise I & II form appendix D). We also created a third exercise (a description of the exercise is found appendix D and later in this chapter) based on HISP India specific reports. The former capacity building effort in India, resulted in a prototype report design which enabled the developers to develop multiple customized reports for each state, but suffered from being connected directly to the database instead of using the API for getting the data. There was demand to replace the direct queries with use of the DHIS2's API in order to uphold the database independent design. Thus the third exercise was directly linked to requirements from Oslo and based on the necessity to uphold the database independent design of DHIS2 (See assignment III from Appendix D).

However, planning how to conduct the capacity building was not the only issue. I had problems finding time to conduct the planned capacity building, because of other problems we were facing.

We had our own development projects running, and we had to prepare for the workshop. Neither of us had much experience with some of the tools and frameworks (Hibernate, and WebWork), and had too spend a lot of time preparing for each brief introduction lecture. Besides the DHIS2 architecture itself, is also a challenging subject, which we only had limited experience from. The experience we had was based on the development of our own modules, thus limited to knowledge necessary to implement those modules.

It is also worth mentioning that when we started with some basic training for the developers in Bhopal, the capacity building effort was often disrupted by immediate needs for development and maintenance by HISP India. Because of the high level of necessity to complete these tasks, the capacity building effort was temporarily postponed (see field notes below). After a while we did get started with the basic training, I first did a few brief lectures, followed by the two first training assignments, and then continued with two brief lectures and the second training assignment. Below I have listed a few occurrences of the problem from my field notes.

The National Indian developer was later supposed to do the assignment #2 but had to postpone this, since he was supposed to finalize the Jarkhand database, for a presentation in Jarkhand on the 16th of April, but they had big troubles making it run. (14.04.2007)

The National Indian developer had received some more working tasks form *HISP India Coordinator3*. The meeting in Jarkhand had been postponed, since the database still did not work. In addition they needed some more Report generators for Jarkhand (PHC profile). (16.04.2007)

I expressed my concern since, I am kind of running out of time to do the training, I realize that we actually have much less time than I hoped, and therefore explained to him the importance of getting further with the training, and that we were running out of time. (16.04.2007)

I wanted to proceed the training with *theHISP India national developer* but he had to make two reports by the 19. so we postponed the training. (17.04.2007)

Today I was going to work with *the HISP India national developer*, but was caught up in some school work for the most of the day. (20.04.2007)

I had asked the national Indian developer in the morning if it was fine that we did some training today, and he answered that this was okay. In the evening he come back to the office, but said that this was a holyday, and at first refused to do any exercises, I argued that we had and agreement from earlier on, and he accepted my argument (21.04.2007)

The issue I try to address with the extracts above is the difficulty of doing training with key project resources in a production environment, especially during critical faces of project evaluation. Like the just mentioned issues around a presentation of the pilot project in Jarkhand.

One of the developers finished with all the training exercises, the other claimed to finish the first two. The final exercise (training exercise 3) was completed in a pair programming session were I collaborated with the National HISP Indian developer. The other HISP India developer did not do this exercise. The further plan was to draw on the newly gained experience to improve lecture foils and presentation (from the INF5750 course) and adapt them to the Indian context for the workshop (Field notes, 11.04.2007).

After we completed the first cycle of capacity building, I evaluated the "result" and made some adjustments to the assignment text, further more some constraint on the physical infrastructure, like unstable internet access made it apparent that optional distribution of course material is necessary.

> I need to have all this information offline, since we might get trouble with the internet connection. I have already experienced this several times, when I tried to do this training in Bhopal.
> There are still some documents left to complete. The three most important are the WebWork foil, the "navigate in DHIS 2.0 Architecture" foil, and the "How to re-implement the web-reports module" lectures. (29.04.2007)

We had some problems on a social level when we arrived in Bhopal, neither of the developers spoke much with us, and we did not do any much social stuff together. At our first day in Bhopal, the facilitator kept us company during breakfast. Since Bhopal was a new town to us we were wondering what the Indian team usually do during their leisure time.

> An unexpected event of the day, is that none of the HISP-team Indians eat at the places we eat, so we don't get the contact as we did with the developers in Gujarat. (16.03.2007)

> The HISP India developers don't talk much which is kind of strange for Indians. During today's breakfast I asked the facilitator what he did for Fun in Bhopal, and he told me there is nothing to do for fun I Bhopal, it sounded kind of depressing, (16.03.2007)

The answer which I have quoted above was rather depressing and set the boundaries for socialisation. The socialisation problem we experienced in Bhopal yields the same problem as the "office problem" we experienced in Gujarat. We did not get the technical small talk going, nor did we work together so we could share the tacit knowledge (tips and tricks) in a natural way.

At the end of April we travelled to the main office in Noida, Uttar Pradesh (40 min from Delhi).

4.1.4 Uttar Pradesh

The Noida office served as new national HISP India office, and was the chosen site for the earlier mentioned Workshop. We arrived 2 days ahead of the workshop to check that everything was ready for the big event.
The workshop and frameworks are discussed in greater detail below. The rest of the stay in Noida included some nice trips to Delhi. We went to see the Taj Mahal with the other HISP participators. After the workshop we continued training with the remaining participators, this including the main national developer and a student from an Indian IT-school.

The workshop was the single largest capacity building effort both we and the HISP India developers had participated in at that time. The focus of this subchapter is on what we did during workshop, with a special focus on the technologies and concepts we used in the DHIS2 project. One point which I find interesting, is that the design patterns was not part nor focus of any of our lectures or exercises. The only patterns which we mention explicitly are the Inversion of Control (IoC) and the Model-View-Control (MVC). Perhaps the two patterns are the only ones of significance to us? Another more likely reason is that I had not learned about them during my earlier training or as part of my own implementation tasks. The latter is perhaps why I did not view the other design patterns as important "individual" parts of the frameworks.

We started the workshop on the 2nd of May 2007. The 1st day was used as an introduction to basic tools; some of my key issues were to explain how the tools work, but also why we should bother to use them.

> The first lecture was on SVN and Tortoise.
> In the middle of the lecture the power went out. It started raining and thundering. HISP India Cordniator2 asked some questions for the other guys, it doesn't seem like they will say something if they don't understand. For example binary files were something they didn't know. (02.05.2007)

> Ole did maven after this. It went all right. There were some questions we could not answer perfectly. The reason for artefact id and group id i.e.. There were two people who put their hands up when we asked if they had used Eclipse before; *the HISP India national developer* and the HISP India Developer10 from Jarkhand raised them. We also know that *the HISP India developer from Bhopal* has used it in Bhopal. (02.05.2007)

not ask questions directly to me or my fellow master student, but rather asked HISP India Cordniator2 or *the HISP India national developer* when they were wondering about something.

> Something that's worth mentioning, which HISP India Cordniator2 told me on the first day of the workshop, was that many of the participators want ask my fellow master student and myself questions when they wonder about different thing in different subjects. They are however more likely to address these issues to HISP India Cordniator2 or *the HISP India national developer*. So far this turn out quite good, *the HISP India national developer* knows most of the topics already and we believe that he contribute significantly in the learning process by explaining and discussing the respective topics with his fellow Indians in the mother tongue. (03.05.2007)

We experienced many technical issues. The issues were rising from rather trivial matters like a continuous supply of electricity, or enough working network cables. These issues were interfering with our progress but we mostly managed to work around the infrastructural issues.

> After this all their laptops were dead from the power out so we stopped there for now. *The HISP India from developer Gujarat* and *the HISP India national developer* went out to get the inverter and hopefully we can charge it up a bit before the scheduled

> We came back at 1550. Power is back on and inverter is installed. Everyone continued with configuration and the assignment. We found a few things that were confusing because they referred to the course. Since we didn't make them install JSE some of the things were not doable. About half of the participants had completed the first assignment at 20:00, after about 6 hours of actual working time, which is almost on schedule. The rest was done by nine. The reason for the one hour delay was that some network cables were broken, so everyone did not have access to the server at the same time. (02.05.2007)

We also had some unfortunate side effects from using the inf5750 course material; several places I had forgotten to remove references to the inf5750 course.

The next day we continued with a short lecture on the Spring framework and the Inversion of Control pattern.

> We started with a lecture on interfaces and IoC and then we started on the Training Exercise II. (03.05.2007)

In the evening when the participator had completed the second training assignment I continued with a lecture on DHIS2 architecture.

> Most were done with assignment 2 by 16:30, and at 17:30 I held a lecture on DHIS_2.0_-_Absolute_Basics, and my fellow master student elaborated some around the structure of the modules. I particularly stressed the importance of using the interfaces correctly and their by making a flexible architecture. There were many questions about layering (the point, how we do this and simple examples). (03.05.2007)

The next day I started with a lecture on WebWork followed by the last training assignment.

I started with my lecture on WebWork, and my fellow master student elaborated on the part of the interceptors, afterwards, he hold a short lecture about how to retrieve data using the services layer and explained in detail how to use the specification class.
Afterwards we separated the participants in 3 groups and my fellow master student, *the HISP India national developer* and I took responsibility of one group each. My fellow master student and I took the groups we felt had the strongest English skill, and in *the HISP India national developer*'s group everyone spoke *the HISP India national developer*'s mother tongue. (04.05.2007)

During the capacity building effort in Bhopal we were often disrupted by the need to solve problems on running installations of DHIS2. Most of these problems occur when some report are supposed to be presented to the government, thus closely related with the evaluation meetings of the different pilot projects.

In the evening there was some commotion about some problems in Kerela which moved two of the participators attention away from the task at hand, and to the problem in Kerela, I felt this was very unfortunate because until this point, they had shown good progress (04.05.2007)

Earlier in our journey when we were I Gujarat we discovered that the DHIS2 developers had no formal education in Java. Although many of our tools (subversion, maven) are not dependent on knowledge in java, all the frameworks are tightly dependent on the java programming language. What I found more surprising was that the student who arrived from one India's top schools the India Institute of Technology (IIT) neither had any java experience.

The IIT student has not used Java before, and is concentrating on basic java concepts. He is therefore not participating on the same professional level as the rest of the attendants. (04.05.2007)

At the end of the workshop we spent some time reflecting on what we had rushed through in the last few days. Since we have our education from the Norwegian schools, we have inherited a belief in the participation learning metaphor (Sfard, 1998), and wanted the

workshop participators to have a very active part in this session. So we made everyone try to explain some general ideas around each of the different technologies.

> Afterwards I made everyone pick at one subject and explain what this technology is/does and why/when we should use it. This exercise showed us what the different participants have learned. Some was as expected and some was a little surprising. There was a very big difference in learning output between the different participants. It was a strong correlation between the most active, enthusiastic and curios participants and the once who had the most learning output (06.05.2007)

Below I have listed all the subjects, and all marked with an 'X' were explained by one of the participators. The participators choose their own subjects, and some of them explained more than one subject. As mentioned above there were great difference between the participators capability to formulate what they had learned about the different topics, there are of course a number of possible variables to explain this difference, but one interesting observation is the strong correlation between the different participators level of participation and their capability to explain the different topics.

Nr	Name of the topics we had presented during the workshop	Presented during repetition session
1	Spring	**X**
2	WebWork	**X**
3	Subversion	**X**
4	Maven	
5	Hibernate	**X**
6	Tortoise	**X**
7	Interfaces	**X**
8	Inversion of Control (IoC)	**X**
9	Testing	**X**
10	JUnit	**X**
11	DHIS Architecture and Technologies	
12	Eclipse	**X**
13	Web Reports	**X**
14	OSS	**X**
15	Jetty	**X**

It was quite exiting to find that at least on a conceptual level many of the participators had captured much of the concepts and Ideas from the lectures.

As a last part of our workshop we had a feedback session. We tried to make an evaluation of the effort to get a general impression of the usefulness of the workshop, and find possible improvements. Personally I tried to advocate the benefits which might be harvested from producing more general purpose code, and share this code so other HISP projects in other countries might benefit from the work done in India.

Around 15 - 16 we started a feedback session, Ole Kristian started by explaining a little of process improvement-why I do this-and then hopefully get some feedback on what was good, what was missing … etc The session Lasted for 2-3 hours and we have noted as much as we remembered from the feedback, this notes can be found below. Though everyone was present and Ole Kristian tried to get a good discussion going, most of the feedback is only from a few persons (06.05.2007).

Below I have listed some of the feedback from the workshop. The feedback listed below is gathered from my field notes (Field notes, 2007).

What is missing?

- Database structure, there are a lot of new tables
- Get the lecture slides before the workshop itself.

What was useful?

- The training assignments were very use full, both T1 and T2, as well as the Web report reimplementation.
- Good to use maven, to build the project run the test, and running the jetty container. Use the data entry module in DHIS to enter test data, to check that the report is actually working.
- It is good to use the "Dataprovider[24]" and action class to provide the necessary data instead of direct queries.
- Learnt the architectural layers of DHIS2

[24] The data provider is a module made for getting a collection of values from the database, based on a few condition, and an easy interface.

- Good to learn Hibernate
- Good with examples which were DHIS related.
- They have heard of IoC and Spring, but learnt idea at the workshop, it proves difficult to get this idea just by reading a book.

Some more general information and feedback:

- How to start as developers on wiki. One of the developers printed out all the information he could find, but did not understand everything. It was difficult to understand some of the Terminology
- One of the developers wondered how to understand the way data transfers inside the DHIS. What is the flow from view to database?
- Must have the lecture slides.
- The training was easy to understand
- Good to test only web reports module.
- Auto period creating

Some problems mentioned from the India developers.

- Don't understand feedback from mails he sends out.
- Don't understand feedback from mails.
- Don't understand information from Wiki.
- Too little former training.

After the workshop *the HISP India national developer* stayed with us in Noida, which now was the new national office of HISP. The capacity building effort continued, and one of the main goals was to integrate the Graphical Analyser (GA) (Appendix C) with the DHIS2 application. The capacity building effort was conducted as sessions of pair programming.

. We left for Norway at the end of May.

4.2 HISP India Knowledge base

When we left from Norway, I had the impression, that there was a large active team in India, working on the DHIS2, this latter is also true, but the relevance of most of the developing

efforts might be questioned. After a while it became apparent that there was a general lack of capacity in the technology used in the DHIS2 project.

> Then I asked the HISP India Coordinator1 how come the national Indian developer is the only one developing the reports it is not difficult to do this the Indian way, using direct queries. The HISP India Coordinator1 answered that most of the developers did not know SQL, only C, C++, Visual Basic, Access, etc. And that it was much easier to develop something in these technologies, than using the heavy DHIS 2.0 frameworks (28.04.2007).

In fact the national developer had developed all the HISP India related extensions of DHIS2. The HISP India specific extension was implemented using direct SQL queries to the DHIS2 application database, instead of using the DHIS2 API, the development had mostly been done using JSP (instead of "velocity" which is the standard used in the DHIS2 web view). Further more the solutions was designed during the capacity building effort in India conducted by Nordal (2006, pp. 70 - 77), but had become are problem since it was breaking important design choice (database independence) with regard to making a flexible application.

During the workshop we also noticed that the other HISP India team members had inadequate training in the necessary technologies. This I had suspected earlier from the chat with HISP India Cordinator1. The national developer can be regarded as the most capable in the HISP India team, and had some responsibility for training the other developers in the team.

> *The Hisp India national developer* NOV 2006 taught the HISP India Developer from Jarkand2 Web Reports, Cygwin, create war, tomcat plug-in and how to implement it in DHIS. How the database works. This is all the Prior training given to any of the HISP India participants. (06.05.2007)

As explained in the quote above, the HISP India Developer from Jarkand2 had some basic training in similar technologies, but not the same as we use in Oslo.

One of the developers emphasized that some of this concepts (design patterns) were really difficult to get, from just reading a book.

They have heard of Inversion of Control design pattern and spring, but learned the idea here (*the HISP India developer from Kerela*), it proves difficult to get this idea just by reading a book (06.05.2008).

The Inversion of control pattern advocates a loosely coupled application where contracts between classes (interfaces) are used to define usage and loose relationships. The main benefit is that it becomes very easy to exchange an old component with a new as long as they both implement the same contract (interface), however there is harder to track how the system is wired together since there are few direct links within the code it self.

4.3 Earlier efforts

During my literature review I read two former theses, which regarded initial developer training in the DHIS2 frameworks in India and Vietnam. These are the only two earlier efforts from HISP to conduct training in DHIS2 in these frameworks and thus all the existing context specific knowledge we have on the subject. I had an email discussion (in Norwegian) with one of the former students. The following is the key points from the e-mail discussion.

- Disappointing that the Indian developers still did not use the DHIS2 API or the Frameworks
- OSS values was not highly regarded among the hired developers
- Difficulty to convince the Indian developers of the true value of using "unit testing" and "dependency injection", since the latter concepts have little value in smaller projects.
- Language barrier, heavy accent a complicated topics may make it difficult to understand each other.
- One major issue for success is to get the developers started on something which they can continue on their own after the workshop.
- A second issue is to get the Indian developers more active in the global development, get them to use the mailing lists more actively.
- A five day workshop is to little time to reach the level of fully trained developers, so don't expect too much results from the effort.

Since this is the limited access we have to similar work with the same frameworks in the same context these comments was a contributing factor for choice of further approach. We decided

to drop one of the biggest of the three training exercises (from the INF5750 Autumn 2006) in favour of a customised exercise based on a real life issue from HISP India.

4.4 Training Exercises

The exercise I and II are described in detail in Appendix D, below I provide a conceptual description of the exercises used in the developer training in Bhopal as well as the workshop in Noida.

4.4.1 Conceptual description Exercise I

The first Training assignment is an introduction to Subversion (See Appendix D) and Maven (See Appendix D). The participators are supposed to make a repository on a central server, check out the repository to their local machine (Working Copy), add files to the working copy, and commit them back to the server. The files include a simple java "HelloWorld" program, and a maven "POM.xml" for compiling it and make an executable JAR file. The participator is supposed to commit all source files to the central repository for evaluation.

4.4.2 Conceptual description Exercise II

The second assignment is about the Spring Framework(see Appendix A), the participator is supposed to learn and understand the concept of "Inversion of Control" and how to wire together loosely coupled objects using a simple configuration file (beans.xml). In the former assignment, the participator is supposed to commit the files to a central repository for evaluation.

4.4.3 Conceptual description Exercise III

The final exercise was one we made in Bhopal in April 2007 (before the workshop). In every state in India the public health administration have a great number of specially formatted reports which they want to produce each month. The former solutions made with JSP and direct database connection. The coordinators and developers in Oslo strongly advocated that these forms should use the DHIS2 API for data access thus making all the forms more resilient to database changes. So back in Bhopal I and my fellow student made a prototype report using the DHIS2 API instead of the direct database connection. We found a way to integrate the current solutions which enabled both the new and the old solutions to co-exist at the same time. Afterwards I did re-implementation of a report in collaboration with the national developer. We never wrote an assignment specification for the last assignment, but planed to split the workshop participants in three groups, and then I, my fellow master

student, and the HISP India national developer were supposed to supervise (guide) one group each.

4.4.4 Email discussions regarding the workshop:

A main issue was that I only had a very vague overview of what I was supposed to do during the workshop. Therefore I discussed the matter with my supervisor and coordinators in Norway. In order to draw on earlier experience I also contacted former master students which had done similar work before. Mail correspondence with *Cordinater1 from Oslo*, *My Supervisor*, *and Cordinater2 from Oslo*. Below I have added some extract of the e-mail discussion regarding the workshop, where the discussions cover strategies and content.

> *I Wrote:*
>
> *I appreciate the response. However I would like to emphasize that I am not familiar with this way of doing training, I have only acted as a teacher on the before hand and that give me some presumptions of what order things should be done in, and how they should be done. In general I see the points of your concern but without more concrete schemes of how I can put your Ideas into action then I am kind of stuck.*

As stated in the quote above, one of my main concerns was how to conduct the actual training effort. From earlier experiences I am used to class room lectures assignments. However the HISP tradition is to use the problems at hand, and combine capacity building and developing. The latter was purposed by the supervisors.

> *I agree with Cordinater2 from Oslo on this: Try to avoid getting bogged down with too many specific details, and try to avoid "lectures" which are far removed from everyday needs. In general, people learn best when they are highly motivated and understand the importance of what they must learn, because they see immediate use for it in a context that they know (Cordinater1 from Oslo).*
>
> *I think the solution to this "problem of where to start" is to start with a concrete practical problem that needs to be solved rather than starting with general exercises and lectures (Cordinater2 from Oslo).*
>
> *E.g. adapting the GA to DHIS2 standards (that is maybe already done…), the target module, or some other task that must be solved by HISP India right now (Cordinater2 from Oslo).*

After the e-mail discussion and some discussion with my fellow student (see below), I decided to throw out the last assignment for some more useful real life exercises.

> *My fellow student supported the coordinators on the point of using real life problems for training purposes, but also agreed with some of my concerns regarding the short time span of the workshop, he emphasized the importance of the getting started with the training as fast a possible and use cooperation as a way of capacity building both to get the developers (the HISP India national developer and HISP India developer from Bhopal) into the Tools and frameworks and thereby into the Global context, while I gain local tacit knowledge in return.*

The most important module in DHIS for the Indian healthcare workers is the report module. The module produces complete electronic reports, which look very much like the original paper version. There were also suggestions of working on the integration of the graphical analyser during the workshop, but since we had little documentation on the GA, then we thought it would be much more difficult to work on the GA than working on the reports. The reports also existed in multiple different versions, but the general change we had to do, was similar in each report.

4.4.5 Stakeholders

Another issue was high expectations, and different expectations to output, while I was focusing on theoretical learning, my supervisors were focusing on a participatory approach. The Indian management needed something to "justify" all the resources they spent on arranging this workshop. There was a general consensus that we needed to work on something from the HISP India domain. However there was an obvious time constraint and we could not deliver a good training and do a complex integration. The latter is expressed in the email extract below.

> *I would also like you to consider what you would like to be the primary goal of this workshop, and how this should be achieved. As I have understood it, on one hand you want us to train the basic concepts of DHIS technology, on the second hand you would some return of interest from this effort in form of some work done on the Indian modules.*

My primary concern was to avoid unrealistic expectations, and try to focus on making the best of the short time we at our disposal. I was afraid that the slightly different agenda would cause us to focus too much on output and too little on training.

4.5 Infrastructural issues

I have already mentioned the problems with too little office space and problems using our laptops at the Gujarat office. Mostly we were confined to working in internet cafes for the first month. A couple of important problems who arose from the situation:

1. The first and significant effect was difficult working environments, since I could not "Google" when I was uncertain about a technical issue.
2. Working from the hotel bed often resulted the unfortunate bi-effect of sound sleep
3. We were separated from the HISP India team.
4. The working situation at the Internet cafes was far from optimal which reduced our working capacity greatly

In the second part of the trip, we were living in the office HISP offices in Bhopal and Noida. In Bhopal we experienced frequent power losses which temporary shut down our internet connection for a short periods of time. In Noida there were regular scheduled power losses (black outs) in addition to some arbitrary blackouts.

4.6 Aftermath

After the workshop we started to work on the integration of Graphical Analyser, my fellow student did quite a lot of pair programming with the HISP India main developer before we left for Norway. After the summer the main HISP India developer travelled to Norway to participate in the INF5750 course, and so did HISP India Cordinator1.

Since there have past some time between my capacity building effort and the finalisation of the thesis, I have been able to observe the "fruits" of my labour. Below is an overview of the status they now have in HISP India:

Nr	Role of the Participator	Participators role in HISP 8 months later
1	*HISP India Cordniator2*	**Quit**
2	*HISP India developer from Bhopal*	**Quit in may 2007**
3	*HISP India developer from Kerela*	**Quit**
4	*HISP India from developer Gujarat*	**Working in Kerela**
5	*HISP India developer1 from Jarkhand*	**Working I Jarkhand**

6	*The HISP India developer2 from Jarkhand*	**Quit**
7	*ITT Student*	**Became ill and never delivered code**
8	*HISP India national developer*	**Working on a MM module**

Email from: HISP India Coordinator1

> *Most of the people are left now.*
>
> *Current people are*
> *Developer: HISP India national developer, two new people 'new HISP India developer2' and 'new HISP India developer2', but they are working on PHP for ART system*
> *The HISP India from developer Gujarat and HISP India developer1 from Jarkhand are in Jarkhand and, The HISP India from developer Gujarat has moved to Kerala.*
> *The HISP India developer from Kerela and others left. (18.01.2007)*
>
> *The HISP India from developer Gujarat is taking care of Kerala field level implementation. Like checking the database for errors. Import the database to create district wide database. Testing various reports, giving training to the HISP India System facilitator (19.01.2007).*

Earlier capacity building efforts has greatly suffered from high turnover among the developers we have trained. The initial capacity building in Vietnam suffered for numerous reasons, but the general outcome has been that most developers participating in the training left the project. As described by Nordal (2006) the first Vietnam training effort ended when the entire partnership with an external firm dissolved, due disagreement of contracts.

> In the HISP Vietnam node 3 of 4 developers quit in February 2007 and of the new recruits only one was hired but she also quit. (04.09.3007)

In February 2007 there was another occurrence of high turnover in the HISP Vietnam node,

The exploration of why the developers quit the project is outside the scope of this thesis. However, the complexity of our system in terms of number of technological components, architectures, design pattern, guidelines and distributed development, makes it time-consuming to build local capacity.

The latter mark the high turnover as a significant challenge for the local capacity building effort. Our choice of highly distributed development also put a higher demand on knowledge on the individual developers in both Norway and in India, because the developers are more dependent of having a total overview of most of the components in the system in order to make good and effective implementation choices and be able to reuse existing solutions.

As the results show, the capacity building efforts provided small gains for the DHIS2 development at a global scale for most of the participators. Only the Indian national developer contributed code which later was integrated into the DHIS2 system (However he was the only one contributing code in the first place), so what should we do? Just lie down and give up? Of course not, but we need to rethink our ideas on benefit of using advanced frameworks with regard to the Knowledge Infrastructure and local capacity (installed base). In the discussion I will argue why our project are failing in view of view of the KI theory and capacity building.

5 DHIS 2.0 – Technological Overview

Below follow a more technical description of the DHIS2. For an anthology of the system history and its components see Appendix A. The technologies described here were all part of the topics covered in the workshop, however practical training is limited to the technologies used in exercises described in detail in appendix D.

The DHIS 2.0 is developed using the Open Source Java frameworks and tools: The Spring Framework, Hibernate, WebWork and JUnit are Java frameworks used in the implementation. In addition the source build tool Maven and the source content management system Subversion is used for building the source code, and distribute updates, respectively. The tools and frameworks are described below.

The software development process is a global collaboration between students, researchers and developers in Norway, India, South Africa, Ethiopia and Vietnam, (HISP, 2007c). There are however a large gap between which technologies developers in the different countries are familiar with, thus the challenge of capacity building. Below I have described the different technologies used in DHIS, and our capacity building effort.

5.1 Main Programming Language – Java SE 5.0

The DHIS2 application is developed for the most part in regular Java 5.0 Standard Edition, with additional java based OSS framework such as JUnit, Hibernate, Spring, WebWork to simplify the development and secure different properties. Java is an object oriented programming language, but some of the frameworks we use complement java with new properties.

I will not say much about Java except that is widely used in Norway, it has been the most common Object Oriented programming langue used in courses I have taken through my days at the University of Oslo (2002-2007). Java was in 2007 commonly used as an introduction to programming or introduction to Object orientation in many Norwegian Universities and Colleges; University of Oslo, NTNU, University of Environment and Biological Science (UMB), NITH, to mention a few. Java is generally known as one of the 2 major programming platforms in Norwegian IT industry, along side with the .NET platform.

In November 2006 Sun Microsystems[25] announced that they are going to "open source" java SE and ME. The news was revealed in an interview view with James Gosling[26] , who is one of the "fathers" of java (Ecstein, 2006).

5.2 IDE – Eclipse

Eclipse is a fantastic Integrated Development Environment (IDE). Originally Eclipse was developed by IBM, but was released under Open Source Software license under the Eclipse foundation as a Java IDE. Eclipse has many nice features as code completion (showing a list of legal choices when programming), underline and in other ways to tell you about syntactical mistakes. The JUnit unit testing framework is an integrated with eclipse and is an standard component, further more Eclipse embed an amazing file/package/file/class explorer to navigate in your development projects, packages, files and classes.

In addition you can install plug-ins, and there are many really useful plug-ins. SubEclipse is a Subversion plug-in for eclipse. There also exists a Maven plug-in, but Ant is the default build tool in Eclipse. Further more there also exists plug-ins which enable Eclipse to give similar programming support to other programming languages, such as Python and PHP just to mention a few. The list just goes on and on.

5.3 Source Code Management system - Subversion

Subversion (SVN) is an open source software "source code management system" (also known as a file versioning system), used to distribute source code and updates on source code between developers. The system can be used to distribute any kind of files, but in the DHIS2 project it is primary used for source code. One of the more brilliant properties of SVN is its backup functionality through file versioning, whenever you change a source file, and "commit" the change, SVN might bring back the current and any former committed version of a file.

Subversion uses a repository to keep track of changes, the latter is placed on a central server. In addition every user has a local copy (working copy) on their local machine (client side).

---

[25] Sun is the company that develop the Java programming language.
[26] James Gosling is the vice president of Sun Microsystems

The user (client) first download a “working copy” to their computer, make the necessary changes, and send the change files back (commit) to the repository. The update will now be available whenever another user tries to check out (download for the first time) or update their working copy (check if anyone else has committed something to the repository).

The advanced issues of file versioning systems, distributed source code and parallel development is out of the scope of this thesis.

5.4 Build Tool – Maven

Maven is an open source build tool, made for managing and building source code and managing dependencies to external libraries. The term “building” includes a seven step lifecycle process which includes downloading external libraries, compiling the source code, running unit tests, packing the complied classes in to JAR[27] or WAR[28] files, and install and/or deploy the jar files.

Maven uses a project object model (POM) file to describe a project. The POM file may include links to sub projects as well as dependencies to external projects and libraries. When external references are given, maven downloads the external libraries and copies them to a local maven repository. The local repository is known as “maven home”, and should be placed in a folder which is on the “class path”. The external libraries in “maven home” are then available to all local maven projects. Maven also contains a reporting tool for making reports of the build process.

5.5 Spring Framework

The Spring Framework is an open source application framework for the Java platform. The strategy is not to compete with good existing solutions, but to advocate integration. (For example, JDO, Toplink, and Hibernate are great O/R mapping solutions, thus we don't need to develop another one). The spring framework is to support centralised, automated

---

[27] JAR is the acronym for a java archive. The java archive is a specific folder structure, containing in particular a folder named META-INF which contains the file MANIFEST.MF. The latter file can contain many sort of information, but most common is the path to the “main class” if such a class exists. If the latter path exists the jar file may be executed. The JAR files are compressed using a compressing algorithm similar to ZIP.

[28] WAR also known as Sun WAR is short for **W**eb **Ar**chive may refer to a JAR containing a collection of JavaServerPages, servlets, Java classes, XML files, tag libraries and static Web pages (HTML and related files) that together constitute a Web application.

configuration and wiring of your application objects (SPRINGFRAMEWORK, 2008). Some of the features advocated by SpringSource (Leading developer of the Spring framework) are:

The most complete lightweight container
Integration with Hibernate (ORM)
Aspect oriented Programming (AOP) functionality
A flexible MVC web application framework

In addition spring supports several other Object Relation Mapping (ORM) frameworks. Furthermore Spring provides a common layer for transaction management and JDBC abstraction layer for error handling and reduce the amount of code to write.

As part of discussion on best practices the rich use of architecture and design patterns in the frameworks underline a main point. Thus the Design pattern of Spring is listed below.

Design patterns in Spring (Spring in action)
The Spring framework alone uses many patterns. Though the definition of design patterns neither are focus of the book "spring in action" (at least some of them are mentioned) nor taught in the INF5750 course.

- Singleton
- Prototype
- Model – View – Control (MVC) (architectural pattern)
- Template method
- Object value
- Factory method
- DAO
- Inversion of Control (IoC)

In addition spring is very configurable. It may integrate with several other frameworks, like Hibernate and WebWork, which both are used in our project. Although the existence of some of the common patterns used in Spring was known to me, the significant use of them eluded me until recently.

5.6 Test framework - JUnit

JUnit is an open source unit testing framework for java. The latter is used to test and monitor the behaviour of a particular class or method, and that they provide the correct answer to a

given input. To thoroughly test a unit of code the developer should check response values for each code unit for four different cases of test values; the legal values, the lower limits, the upper limits, and the illegal values.

The unit testing can also be used as regression testing. Once the test is implemented you can guarantee that the unit of code gives a certain output based on a certain input. If changes elsewhere in the application cause your tests to fail, then it will be detected at once. The tests (if written in a correct manner) also provide information of which tests that fail, and thus provide information of what part of your system which is broken.

5.7 Hibernate

The Hibernate is an open source software framework for persistence. There exists Hibernate framework implementation for both the Java and .Net programming environment. Hibernate uses an Object Relation Mapping (ORM) in order to simplify the otherwise cumbersome process saving java objects in a database. The persistence implementation is often error prone because of a large amount of SQL queries which has to be written, and maintained. The ORM alone does not fix all you persistence issues, but aim at relieving the developer 95% of the object persistence job. For this purpose hibernate facilitate standard Create, Read, Update and Delete (CRUD) operations which is based on hibernate mapping files describing the relation between the object and multiple data tables and columns[29]. Since hibernate is in fact a java framework we can use syntax analyse tools (like the java compiler, or syntax check support in an IDE) to check if the persistence operations is correctly implemented an options which is missing when we are using SQL queries directly in the code.

> *Hibernate is a non-intrusive solution. By this we mean you aren't required to follow many Hibernate-specific rules and design patterns when writing your business logic and persistent classes; thus, Hibernate integrates smoothly with most new and existing applications and doesn't require disruptive changes to the rest of the application. (Hibernate in Action).*

One of the perhaps most important features of the hibernate framework it does require to implement the classes with a particular design in order to use hibernate as persistence

[29] From Java se 1.7 annotations can be used to write the mapping direcly in the source file.

middleware, thus any application object may be persisted with the use of the hibernate framework.

As with the other frameworks, I would like to emphasize that hibernate also rely on a number design patterns and architectures.

Design Patterns in Hibernate (hibernate in action)

- ORM
- Data Transfer objects (DTO, also known as Value objects)
- Registry pattern – sharing single Session Factory
- ThreadLocal Session pattern,
- DAO
- Façade
- Type Safe Enum pattern.

5.8 Web development framework - WebWork

WebWork is an open source java web application development framework. The framework is designed with a web customised MVC pattern in mind. In fact WebWork support two different versions of the MVC pattern (Front controller and Page controller). Further more WebWork support internationalisation (i18n).

In addition WebWork include a lightweight servlet container. At the current date there exits many different servlet containers and each have their special features. WebWork is enabled to configure which container to use, thus the possibility to enable containers with different pros and cons.

A powerful property gained by using a lightweight container is the “Inversion of control”. We can define the coupling between different objects through use of an xml configuration file(s) and the lightweight container. The latter makes the links between objects clearer and easy to manage.

For easy webpage programming WebWork supports the user of Velocity, which is a Java-based template engine. One of the major strength is the easy access to objects defined in the java code (APACHE, 2008).

> *When Velocity is used for web development, Web designers can work in parallel with Java programmers to develop web sites*

> *according to the Model-View-Controller (MVC) model, meaning that web page designers can focus solely on creating a site that looks good, and programmers can focus solely on writing top-notch code. Velocity separates Java code from the web pages, making the web site more maintainable over its lifespan and providing a viable alternative to Java Server Pages (JSPs) or PHP (ibid).*

At last it is worth mentioning that WebWork build on a command-pattern framework XWork is a command-pattern framework that is used to power Struts 2 as well. (OPENSYMPHONY, 2008)

> *XWork provides an Inversion of Control container, a powerful expression language, data type conversion, validation, and pluggable configuration. (ibid)*

Xwork was created as part of the development of WebWork 2.0 and later versions as an attempt to separate the non-web related part of WebWork into another framework. The separation makes it possible to use many of WebWorks non-web related features in other types of Software.

The use of design patterns in WebWork is also an important part of the WebWork, the use of architecture (life cycle) and design patterns in WebWork is used to underline a main point in the discussion, thus listed below.

Design Patterns in WebWork

- Design Patterns in struts WebWork (struts 2.0 in action)
- MVC
- declarative architecture , xml based, annotation based
- Xwork – command based – the actions represent commands
- IoC
- Chaining

In addition to the design patterns WebWork rely on a particular order in which events are run as part of http request handling. I call the entire request cycle in WebWork for the "WebWork Lifecycle".

WebWork Lifecycle

The inner workings of the WebWork lifecycle consist of the following key concepts.

- Actions (POJOs, Action Support)
  - Results
  - Interceptors
  - No "form beans": the action is the model
- The value stack allows loose coupling

- Interceptors: “AOP lite”

In short the control flow of the web application is driven by Action results, which determine what to do next. Several actions can be “chained” in a sequence and before and after each action interceptors can be run.

The action is were we implement the necessary algorithm to fulfil the request, while the interceptor can be used to handle small task in front and after an action, like checking if a user is logged inn, setting some attribute values, and so forth. Application security is often handled as an aspect[30], thus the phrase “AOP Lite”.

5.9 Communications

Since HISP is a global network, good, cheap and effective communication is of the essence. I separate between 5 different types of communication, though some of them have overlapping functions. Such as some Instant Messengers (IM) application support Voice over IP (VoIP) and vice versa. The same crossover exists for IM and email clients. However the particular application we use for the task is bound to which protocols the favoured communication applications implement, the latter may vary between the national nodes of the HISP network. Thus the choice of applications and protocols which is used is dependent on the local installed base (see instant messenger).

5.9.1 Documentation – Confluence wiki

HISP has its own wiki page (www.hisp.info), containing information about HISP, and its development projects. HISP uses a wiki application called Confluence. The application is not an Open Source Software application, but it is free to use for Educational purposes. The WIKI allow it’s users to read, add, edit, correct and remove the presented content and even make new sections. The wiki is the main source of documentation of the DHIS 2.0 project which includes conceptual information about the different modules of DHIS2.0 and how to use them.

[30] An ‘aspect’ is a term from Aspect oriented Programming (AOP) which means to take a concern which run through the entire application and extract it to a separate part which is used in all level of the architecture. While an architecture level is suppose to handle a particular task (persistence, business intelligence or view) the aspects is used to handle cross cutting concern running over several levels.

5.9.2 Bug tracker and project management - Trac

Trac is an open source wiki specialised for development projects, it therefore contains project management system, which also can be used for bug tracking.

The system is simple, first part of the project management is to outline a roadmap[31], the roadmap tells us when and where requirements are going to be completed. The road map is organised by milestones, and each milestone contains tickets. The management process is driven by splitting the requirements into tickets (a small atomic task with a specific measurable goal), and distribute the tickets on different milestones. The milestones are in chronological order, and when a milestone deadline occurs, the project participators are supposed to have completed every task linked to that milestone. The latter is not always the case however. Tickets can prove to be irrelevant, useless or too time consuming, and might be discarded or moved to a later milestone.

In the DHIS 2.0 development bugs are handled as tickets. Earlier we used a system known as Jira for this task, but the Trac offer multiple purposes. Some of the developers even argued for moving the content on the confluence wiki to Trac, because the confluence version used by HISP is old, and that Trac may serve multiple purposes. The main rationale was to lessen the burden of maintaining (updating) by using only one piece of software.

5.9.3 Email

The email service is HISP's most common information stream in addition to the word of mouth. Since the DHIS development is distributed into very different time zones the need of an asynchronous messaging service is dire. HISP mainly use four global email lists.

Table 1 - Email lists (HISPWIKI, 2007b)

HISP AT hisp.info	HISP general list	http://www.hisp.info/mailman/listinfo/dhis-dev	
Dhis-users AT hisp.info	User feedback and help list	http://www.hisp.info/mailman/listinfo/dhis-users	
Dhis-dev AT	Developer list	http://www.hisp.info/mailman/listinfo/dhis-	Required for

[31] Roadmap in this context refers to a plan. The roadmap provides a view on the ticket system that helps planning and managing the future development of a project (Trac, 2008).

hisp.info		dev	developers
dhis-scm AT hisp.info	Subversion commit mails	http://www.hisp.info/mailman/listinfo/dhis-scm	Required for developers

There also exits National mailing lists to support the national need of the developers in the different countries, but most technical questions are on the international list (dhis-dev@hisp.info), at least more than on the Indian developer list which I have been member of for the last 10 months. On the Indian mailing list I mostly receive spam mail. The latter was so I annoying at times that I actually had to start using a more aggressive spam filter on my laptop.

5.9.4 Instant messaging

Instant messenger services (like MSN[32]) are one of my personal favourites for communication and problem solving when programming. Due to a significant time zone difference (5.5 hours) between India and Norway this proved less efficient when I am in India since most of my personal network did not logon until 16:00 (Local time). Our Indian friends preferred the Google-talk[33] instead of MSN. The latter is a significant blocking factor for communication.

Internet is also slow, unstable and a little expensive in many parts of India. In addition good internet cafes which provide the necessities for tedious development effort is hard to come by. Many of the HISP India developers don't have continues access to a desktop or laptop computer at home and are confined to limited access at the various HISP state offices or at a local internet café.

5.9.5 VoIP – Skype

During our stay in India we participated in a few online meetings using Skype, or at least we were supposed to. The idea was to have coordinators, Norwegian developers in Oslo, and the Indian coordinators and developers join in. However limitations in the size of chat groups on the free Skype client, as well's as limited access to headsets and computers, made this

[32] MSN is a Instant Messenger (IM) service for the Windows platform, the MSN client application is shipped with the Microsoft Windows operative system, which as you probably already know, pre-installed on approximately 92% of all sold PC's. By common users in Norway, this is the preferred IM, in India however the Google talk is more common.

difficult. In the end only a few (the most important decision takers) participated in the discussion and a summary was distributed to the rest us.

5.10 SQL Server – MySQL

The Hibernate Framework makes the choice of database irrelevant for the most part from the developer's point of view. Since Hibernate is suppose to do all the communication with the database. In the DHIS2 project we are open to using other tools than the most common ones. So far MySQL is most used, but PostgreSQL has also been used. Hibernate configurations for both database systems are available at the HISP website (HISP, 2008d).

HISP strive for keeping the DHIS2 database independent but at a few occurrences there has been implemented directly against the database to optimise performance. The latter was the case for on implementation effort for the Datamart[34] module. The pre aggregated data is used both for DHIS2 internal data presentations, but also represents an easy way to integrate against the DHIS2 database (though integration of such features into the DHIS2 application is our preferred choice). The Indian module "Graphical Analyser" was built using the latter approach before it was integrated into the DHIS2 Spring 2007.

5.11 Web server – Jetty (maven plug-in)

Jetty is an open-source, standards-based, full-featured web server implemented entirely in Java. Jetty has benefited from a large user community, thus been able to establish a stable core of lead developers (Jetty, 2008a). Jetty can be used as:

a stand-alone traditional web server for static and dynamic content
a dynamic content server behind a dedicated HTTP server such as Apache using mod_proxy
an embedded component within a java application

In the DHIS2 project we use the maven jetty plug-in to run the web-server during development testing. However when we were in the field in Gujarat in India, the original Jetty server was prime choice. At the jetty website the following features are advocated as the defining features of Jetty; simplicity, scalability, efficiency, embed ability, plug ability.

[34] The Datamart aggregates data values as a batch job in order to quickly access pre-aggregated data for presentation purposes. The latter is very useful for databases with our design. The database is very flexible, but demands many joins and a little analysis in order to present useful information. If you add the fact that a Sample database for 5 out of 23 districts in a state with sample data for a few months actually contained approximately 1.6 million rows of data values, then the importance of pre-aggregated data becomes obvious.

5.12 Layered Architecture

In order to visualise different roles the frameworks play in the in java application. Hibernate works as a middleware which means that it is part of the persistence layer and have the main responsibility to connect and communicate with the database.

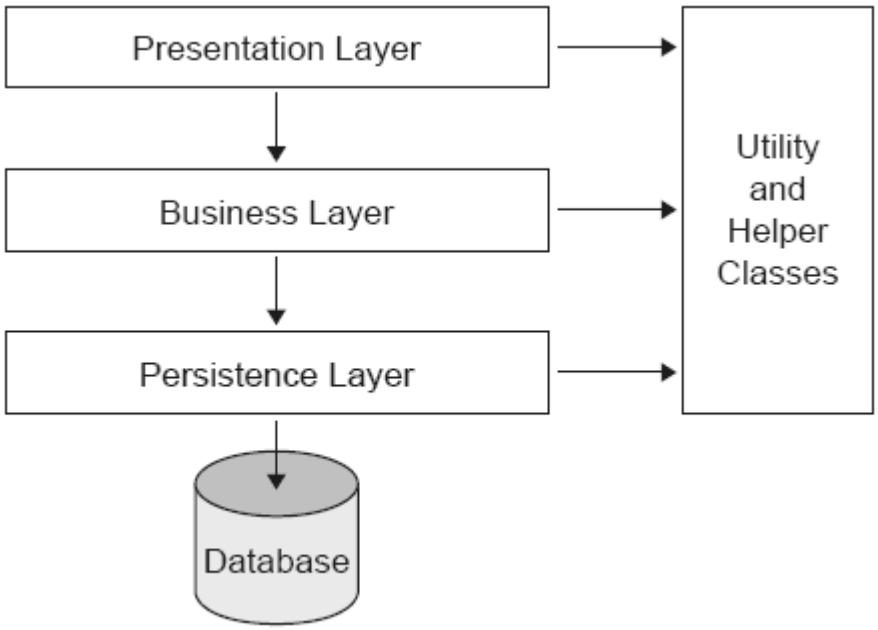


Figure 7 Three tier architecture (hibernate in action)

Below follow a description of the architecture which is

- Persistence - *The persistence layer is a group of classes and components responsible for data storage to, and retrieval from, one or more data stores. This layer necessarily includes a model of the business domain entities (even if it's only a metadata model* (*Hibernate in Action p.18*). Hibernate is as part of the persistence layer in short hibernate is suppose to be used for all the communication between the DHI2 persistence layer, and the database.

- Data base - *The database exists outside the Java application. It's the actual, persistent representation of the system state. If an SQL database is used, the database includes the relational schema and possibly stored procedures* (ibid).

- Business Layer – The business layer is supposed to handle all business logic in the application. The business logic mean; data manipulation, aggregation, business rules, calculations, etc., in short anything which is unrelated from persistence and presentation.

- Presentation – The presentation layer handles presentation related issues, such as how to present data (table, graph, list, plain text, Images, hierarchies), colours, internationalisation (languages), text formatting. WebWork framework plays a major

part of the presentation view of DHIS2. In addition to handling presentation issues WebWork also handles the application control flow through the “chaining of actions” and action results.

- A common strategy is to present the use with the data in the local language (if supported by the application), but everywhere else in the application the

- Helper/utility classes— *every application has a set of infrastructural helper or utility classes that are used in every layer of the application (for example, Exception classes for error handling). These infrastructural elements don’t form a layer, since they don’t obey the rules for interlayer dependency in a layered architecture. (Hibernate in Action p.18).* The Spring framework is playing this part in the DHIS2, by providing wiring of the different application objects. Spring is used in all parts of the application.

6 Discussion

The capacity building effort done by me and my fellow master student, were based on decisions and expectations made during the initial design and development of DHIS2. The best summaries available on the initial design and development effort are in two master theses written by former master students. The students themselves participated in the DHIS2 project, and I assume that the statements from the master thesis are based on their field notes since no other reference is given. In addition I have conducted interviews with key stakeholders in Norway and studied written communication (email) in order to grasp more of the picture than what's summarised in the former theses.

Later in this chapter I have added extracts from one other thesis (Nordal 2006), which was directly connected to the outcome of our development and capacity building effort in India spring 2007.

6.1 Sustainable Capacity building

When I started working on this thesis I had a main goal of conduction sustainable capacity building in India after the model presented in "Network of Action" described by Braa et al. (2004). During the time in the field it became apparent that to achieve my 'ultimate' goal may be a little difficult.

How to conduct sustainable capacity building for system developers in OSS based HIS projects in a Non Governmental Organisation in India?

When I now look at the capacity building effort in retrospect, it became obvious that I have not reached my ultimate goal. Since most of the developers ended up leaving the project during the next six months. After inquires with *Cordinater1 from Oslo* I learned that the high turnover rate has been a problem all along. This problem is actually the one I tried to solve by my capacity building effort. According to "Network of Action" article (ibid) building one community of practice have a great risk of falling apart if key personal is leaving the project. The effect can be countered by have multiple communities of practice and share knowledge and work experience through workshops and other similar joint efforts.

In my opinion the HISP India have the advantage of having multiple local communities (Appendix B), with developer team in each state they have implementations running. If one team dissolves or the political wind in the public administration changes, then they still have other communities of practice in which the tacit knowledge is preserved. Through our common workshop (4.1.4) and the local effort in Bhopal (4.1.2) we tired to create a knowledge foundation the developers could extend later. However this did not work out as planed (4.6).

The logical question then becomes; why did this happen? Well, there might be a multiple number of reasons, and my data give me no definite answers to the question. So instead of philosophising around a question I can't answer, I instead tried to identify the source of the problem. This initially leaded me to what appeared to be an ever so little paradox.

How can use of multiple open frameworks, design patterns, architectures, and tools create a lock-in situation for a flexible and open project?

In a different context the problems we experience might not be problem at all. It could probably been solved if we had unlimited resources, but unfortunately we don't. So it boils down to; what is the root of this problem and what can we do about it?

The new problem has become the focus of my attention, and the focus of the reminder of the thesis discussion. I try to identify the cause of the capacity issue (1.3) through reviewing the installed base using "Knowledge as Infrastructure" (2.3) theory, and exploring the nature of our 'best practices' (2.4).

6.2 Design choices and Expectations

In this subchapter I try to summarise design choices and expectations from the initial DHIS2 design in order to explore why and how the different choices were made. Below I list excerpts with information on the initial reasons for choosing the particular frameworks and of the assumptions these choices were based on. I then compare this with my experiences from the fieldwork in India.

Requirements that was highly important for the new release:

> *– Platform independence*
>
> *– It must run on most relational database servers*
>
> *– It must be possible to develop both web based and desktop modules for the system*
>
> *(Nordal, 2006)*

As mentioned earlier and in Appendix A, the development of a new system was based on multiple requirements, but the three reasons mentioned above can be linked to making the new system environment independent in order to avoid a technological lock-in situation (such as Microsoft Windows dependency), and even provide the possibility of the user to move from one environment to the other.

The use of Open Source Software technology in the public administration in developing countries is deemed a likely strategy .The choice of technologies in DHIS2 follows the latter strategy. In particular an OSS solution for Public Health care systems was requested from the Indian state of Kerela back in (Nordal, 2006, p. 71), thus DHIS2 were a suitable option.

Another way to view the three requirements is as an attempt to draw upon the installed base in HISP by exploiting experience gained from the DHIS1.X implementation. In addition to use the experience from the former project, the requirements focus on drawing upon experience and resources from other Open Source Software projects, and attempts to use a development process which is attractive to HISP's installed base in Norway (which include but are not limited to professors, PHD & master students as well as popular technologies; java, spring, subversion, hibernate). The requirements also reflect "openness" in scale, domain and function (the initial core of the system is not suppose to be limited to Routine Healthcare information).

> *but desktop modules for the system will also be needed. Advanced analysis and reporting are examples on parts of the system that will be very difficult to develop as pure web modules (Nordal, 2006).*

I strongly agree with the need of a desktop GUI, mostly because it used to be significantly faster, the web-based application tools have so far much to gain on speed, and usability in comparison with their old fashion desktop versions. The new system is designed with heterogeneity in mind, with respect to layering and multiple Graphical User Interfaces to

utilise strengths of the conventional desktop environment - speed and computational power - like availability and compatibility of websites. A desktop application can in no way compete with a web based interface on availability, thus making the web interface the superior choice with regard to availability in countries with an adequate infrastructure.

> *It is also apparent that DHIS2 will need to be able to integrate with 3rd party desktop applications, like GIS tools, spreadsheets and reporting tools.*

In order to achieve function richness, the idea was to utilise existing modules from the installed base of OSS. There exist OSS projects for supporting Business intelligence analysis and GIS.

> *Using technologies that are considered mature and stable, but which are no longer under development, might seem a safe bet from a user perspective, but it will make the project uninteresting for both researchers and potential outside involvement. (Nordal, 2006)*

The former quotation outlines the perspectives on the initial design of DHIS2. The quote shows how the local context in the other countries at the time was not considered, or to be more precise the choice of technology for DHIS2 combined with the Nordal's perspective shows that only the local context in Norway was considered. The question is, should we design after what is, or what we believe is the near future. II design theory advice that we use what we have at hand (installed base). Hence we should look at what resources that is available in the global HISP network. The cooperation with learning institutions in Vietnam (Øverland, 2006, p. 59), as well as a higher focus on OSS in the latter nation, aligned with the strength of OSS and the favoured tools at the university and industry in Norway, makes the choices of technology platform sound and rational. However when we look to the India node, it become apparent that India should perhaps not be chosen for field implementation and development of DHIS2 nor the chosen sight for core development. I have previously emphasized the poor java background of our DHIS2 developers. The background of our developers is related to HISP's ability to attract skilful java programmers in India rather than a total absence of these developers in India. Some of the latter lack of attractiveness may be related to the low wages HISP can offer the programmers compared to what they can make in other companies.

6.3 Best Practices

The fact that we rely on many different frameworks implies that any DHIS2 developer has to spend a lot of time just to learn the basic usage for each framework.

James Turner (2008) suggests that much of the resources spent on developing a piece of code is not reflected in the code it self, but rather as knowledge gained by the developer who made the solution (algorithm). In our situation this can be translated to; we spend much resources on training developers, before they can produce any code. The framework related knowledge they gain is of a more universal character than the knowledge gained by making a solution based purely on a company's designs and in house applications, thus I argue that even more of the value is added to the developer relatively to the code.

Now if we assume the framework represents project independent (universal) knowledge which gives great benefits in form of effectiveness once it is learned, then it is obvious that the investment may pay off.

The aspect of the investment in knowledge is important, since each framework gives benefits over a longer time aspect in terms of better modulation, more effective implementation (the framework provide many behind the scenes operations), more reusable code, more reusable algorithms and a smaller code base which should be easier to maintain.

6.3.1 Investment and risk in a KI perspective

The benefits from the use of framework as KI-standards (2.4) depend on a single condition; that a developer already knows the framework. On the other hand, if the developer has to learn the framework before he can perform a single implementation on the production code, then we have to expect to do an investment in terms of capacity building before further development may be initiated. After the necessary training (learning process) is complete it is just a matter of time (number of implementations) before the investment starts to be beneficial for the organisation[35], thus we have to be able to maintain the developer in our service until

[35] The view is a gross simplification, according to the 5-stage skill acquisition model of Dreyfus & Dreyfus (1980) the productivity of the developers will increase continuously over time as the developer become more proficient with the framework. The model were initially developed and studied in other settings, but have in the recent years also been applied to informatics.

the original investment is “paid off”, before we actually harvest any benefits (Return of Investment (RoI)) from using the framework

The former paragraph is exploring the a cost of using one single framework unfamiliar to a developer, but what if we use many tools and frameworks unfamiliar to the developer? The argument goes as follows. For each framework the developer has to learn in order to do implementation, we first have to invest some resources (time, money, other developers) in training. Thus the more frameworks and tools we have, the longer we have to employ the new developer in order to reap the benefits from our initial investment. We should also consider the fact that the more frameworks you use, the harder it becomes to find developers who have a competent level of knowledge to begin with (On the bright side it might be more developers who knows a few of the frameworks to start with).

If we disregard other factors, then this simply turns into a question of investment and interest. And how long time it takes before the interest on the investment exceeds the initial and running cost. The time available per developer is in some degree random, so no absolute measure is available. However the RoI split on the turnover rate gives us a relative cost/effect relationship. Again in lack of absolute measures there is difficult to say where the limit goes. Although it does tell us that there exists a relationship between knowledge complexity handling ability and the organisations development team’s stability.

In all the DHIS2 cases I have examined the cost have been high compared to the gains; Appendix H, 4.6, Nordal (2006), Øverland (2006). All the cases show that the return of investment is low compared to the resources spent, because the production code is less than expected and/or the HISP Node have a high turnover rate. My case is no different. In short the risk involved by using multiple frameworks, is higher than the gains we sp far have been able to harvest.

6.3.2 Benefits of using a standard

When we are using a popular framework it can act as de facto standard for the particular implementing language, then we have several possibilities to harvest from the “installed base” which in my case include; students and Professors at UIO, HISP, the OSS community, and developers in India and Vietnam. The use of frameworks should enable us to communicate and share knowledge between developers, projects and organisations with great ease.

At the current point, we only train developers in the frameworks and are cultivating the installed base without exploiting the possible benefits from the "installed base". Neither are we harvesting much return of investment (RoI) from the capacity building efforts (Appendix H, 4.6, Nordal (2006), Øverland (2006)). The capacity building is very resource demanding and we suffer from high turnover rate for our developers, both in India and among the student involved through the INF5750 (2007) course.

The reasons for the situation is very dependent on the local context, at UiO the frameworks are only thought and used in the INF5750 class, hence there are unlikely that the students know these frameworks when they take the class (which should be expected). The overall implementation strategy and high cost of the professional Norwegian developers prohibit us to use the professional developers from the Norwegian IT industry. Thus we are unable to gain possible benefits from using a "standard" in the Norwegian context.

In India we have been unable to recruit developers with experience in our frameworks, and very few with experience with java at all. The experience from India is general lack of java developers (at least in our price range), the developers we had in the organisation had a background in procedural languages. They did not seem familiar with OO programming ideas either.

6.3.3 Technology and design review Anno 2007

The students who made the initial design of the DIHS2 were exploring the frameworks for only a few months before starting the implementation. The latter suggest that lack of experience might have been significant. In a communication case study by Schmidt (1995) the use of practical experience when designing and using patterns are of the essence.

> *Useful patterns arise from practical experience. Therefore, we found it was important to work closely with domain experts in order to identify and document key patterns in the communication domain (Schmidt, 1995).*

In the DHIS2 case, we had domain knowledge within the organisation which outlined the original non-functional requirements of flexibility and modularisation. However the students of the initial design of the DHIS2 application had little personal experience with some of the

tools and frameworks (Appendix H). The tools were used in a short trial period in order to determine what they were capable of and which of them which was easy to learn and use. Further more the usefulness of a pattern is measured by the successful usage of the pattern by other than the author.

> *We measured the utility of design patterns by how widely they were adopted and used successfully (particularly by developers other than the original authors) (Schmidt, 1995).*

The case is not directly comparable to the DHIS2 case. However the study emphasizes that usefulness and use of patterns must be reviewed after a while, Schmidt lists three key positive outputs from such reviews.

1. To validate that the pattern is implemented correct
2. To enrich pattern vocabulary within and across development teams.
3. Sharing experience, such as usage, pros and cons.

The points Schmidt made with regard to communication pattern design in some degree relevant to our case. The frameworks in DHIS2 have some patterns implemented already and there is just a matter of setting an attribute to use pattern such as the “Singleton” with Spring. On the other hand, the use of the Inversion of Control is supported by spring, but also requires the different beans to be implemented in a certain way in order to work properly. Point 2 & 3 is just as important and useful for use of frameworks (with design pattern) as implementing the design patterns themselves. Particularly in the distributed development environment of DHIS2 where most communications is done by email, a common terminology is important. Further more Schmidt advocates the sharing of practice which we all agree upon, but it is difficult to manage when the developers in the different nodes have so different knowledge of tools, frameworks and domains.

In India we tried to bridge some of the gap by making the third training assignment (Appendix D), and later by starting the integrating the Graphical Analyser (Appendix C) into the DHIS2. The latter task was postponed for a long while because the GA (appendix C) was a pretty big and complex application written in only JSP code. The integration of the application was postponed until after the workshop since at any earlier point in time, the capacity gap between us was too big. We had the tool knowledge, but did not understand the

GA, and the Indian National developer had all the GA knowledge but little experience in using the DHIS2 tools and frameworks.

6.3.4 Local Capacity building

The building of local development team is much a question of evolving (2.3.3) the installed base (2.3.1) with regard to developers and DHIS2 specific architecture knowledge, in addition to organisation and infrastructure components.

> *With an object-oriented programming language like Java and some of the more complex technical frameworks planned for DHIS2, they considered the gap between advanced users and developers to be a lot wider than with MS Access and Visual Basic that were used for DHIS1.x. More effort would need to go into the training of technical personnel, for them to be able to actually write source code for the software (Nordal, 2006).*

The latter assumption can be confirmed as true based on the field experience from India spring 2007 (Field notes, 02.2007- 05.2007). The problem can not be narrowed to a single issue or topic, but are more a complex collection of problems.

> *With project coordination from Norway, it was planned to create the first international nodes in Vietnam and India. By sending Norwegian developers to these two nodes, and using them to train and integrate the local teams in the global effort, HISP was going to establish a global development team for DHIS2(Nordal, 2006).*

> *These teams would then be a part of the general development effort, in addition to develop functionality for their local needs. By the time DHIS2 is ready, the local teams would also be able to support implementation efforts, and further need for local adaptations of the software (Nordal, 2006).*

When we look back on recent events we have achieved the latter mentioned goal to a certain degree. The India developers have developed state specific reports, and the Graphical Analyser. The Vietnam node has contributed with a new webpage design. The latter mentioned design resulted in a lot of controversy since it is based on a hardcode webpage, instead of using the module for webpage customisation. Besides the webpage colour combinations is regarded as unfashionable by Norwegian Standards, but might be successful in Vietnam.

6.4 The Knowledge as Infrastructure

In this subchapter I wish to focus on the knowledge infrastructure of HISP with regard to the DHIS2 project, on a National, and global scale. I will compare the two nodes where I have experience, and supply with data and experience from Vietnam. The data from Vietnam is included to show a general problem which occurs in all three national nodes.

What is the current knowledge base of the developers?

Through the Empirical chapter (chapter 4), and the description of the DHIS2 project (Chapter 5) I find it evident that the knowledge infrastructure and the DHIS2 project and HISP itself can be defined as separate infrastructures, both on the scale of HISP Nodes and National.

On the National level the learning institutions uses different technologies, in particular Java, which is favoured by Norwegian Universities and Colleges (Chapter 5.1, Main programming language), has apparently not favoured by schools in India in the past.

6.4.1 KI related challenges in India

From my field study in India it became apparent that our Indian co-workers did not have a java background from their education (Chapter 4, 4.1-4.3), which implied that they had no training in use of the java dependent frameworks either. The student from IIT did not learn java at school either, and had to learn Java on his own, as part of a bachelor project which he did under direction of HISP.

At the Evaluation of the workshop in Noida, I inquired about the former training the HISP India developers within the boundaries of HISP India. The HISP India developer2 from Jarkhand informed us that he had received some training in the some of the tools (Chapter 4, 4.3), but not in the exact same tools we use in the project in the Norwegian Node.

In the recruitment process HISP tries to find developers with required background in java technology, but so far this has not paid off, as *Cordinater1 from Oslo* said:

"It is difficult to find unemployed developers experienced in popular technologies willing to work for minimum wage (Appendix H, Interview with Cordinater1 from Oslo)".

As mentioned in the empiric chapter, the HISP India node suffers from a high turnover rate among the developers. Much of the capacity building effort does not actually lead to any implementation being done by many of the developers.

6.4.2 KI related challenges in Norway

In Norway most project participators are recruited through the INF5750 course. The course itself is very attractive for the students. In the autumn 2007 the course attracted twice the number of students the course administration is was comfortable with[36] . But so far most of the students have only participated through mandatory project work as part of the course syllabus.

In the Norwegian context we have the advantage that all the students are familiar with java from their education, and might have used a few similar tools and techniques in prior courses or in (part time) job. This gives us an advantage with regard to necessary capacity building time span.

The course spanned 3-3.5 months and ran in parallel with 2 other arbitrary courses the student is suppose to complete during a semester. The first month is used to teach the tools and framework, while the last two months is spent solving a project assignment which includes exploring problems and implement solutions for the DHIS2 system.

Of the students from a semester on the INF5750 course, some join the HISP project and continue developing on the DHIS2 system for approximately 6 months or so. The master students might participate as teacher assistants in the INF5750 class the following year, in addition to capacity building in other HISP nodes (like Vietnam and India).

[36] The course last months is spent working on the project assignments related to the DHIS2. However the projects are very time consuming and require close supervision of the teacher assistants. Thus a large number of project groups make it difficult to give each group the proper supervision.

The projects economy is split on the national nodes. In Norway the research group is responsible for the tutoring and capacity building in local and global context (Nordal, 2006, p.35) and uses students for developing purposes. To actually hire developers in Norway is very expensive compared to the local context in India and Vietnam, and is thus out of the question.

> *but while the competence building efforts for DHIS1.3 was primarily focused on the use and administration of the software, building local development teams was going to be an additional goal for the DHIS2 project. (Nordal 2006, p.34)*

Furthermore the long time development strategy is based on funding from local government for local maintenance, support and module development as well as core development. The latter requires proficient local capacity which we are unlikely to gain by hire Norwegian developers.

The research group has until recently (January 2007) avoided hiring Norwegian developers. However the group have hired one former master student to continue implementing on the system, and tutoring the technology specific part of the INF5750 course. The latter employment was done out of necessity to keep very valuable experience and capacity within the project since many of the other master students finished their thesis in 2007 (Appendix F, Email with *Cordinater1 from Oslo* 09.05.2007), and leave the project

6.4.3 KI Challenges in Vietnam

The local capacity building in Vietnam is outside the scope of this Case study, even so I would like to underline that the turn overrate among our Vietnamese developers also is very high for various reasons. In the first capacity building effort done by Nordal in 2006, HISP used a partner company. Some of the employees assigned to the HISP project were prioritising the other projects they were working on, other quit the firm and found other jobs. And finally the firm had a disagreement regarding the partnership agreement with HISP and the collaboration broke down.

A second capacity building effort was done by Øverland (2006). He used HISP's connections at a local university who had an interest in OSS technology, and held a course in the DHIS2 frameworks with project assignments related DHS2. Several of the students were recruited to

HISP Vietnam after their graduation. However all but one of those students are no longer active in the HISP Vietnam node. Although a few of them are still in the global HISP project.

6.4.4 OSS community contributions

The OSS community we hoped to create has not contributed to this endeavour in terms attracting competent developers[37], but we do benefit from attracting many students in Norway. Being an Open Source Software project we also receive opportunities for collaboration from other OSS projects, such as OpenMRS and Pentaho (HISP, 2008e), along with opportunities to establish the project in states with an OSS friendly Public Health Administration (like Kerela in India).

But the high knowledge demanded in order to understand the code, and the size of the DHIS2 code base which makes it difficult to understand the inner workings in a short period of time. The latter has been mentioned as a reason for the lack of interest to join the project after the students have finished their INF5750 course. In addition I would like to emphasize that the first voluntaries of the OSS communities originated from a hacker[38] culture. From the definition of "hacker[32]" I presume that the high knowledge bar makes the hacker participation less likely.

6.4.5 High turnover rate

As emphasized in all the local nodes of HISP DHIS2 project we suffer from the fact that most of our "developers" only follow the project for a "short" period of time. even though the time frame can actually be several months, the limited resources in HISP prohibits a continually capacity building effort to take place in all our global nodes at the same time, thus the

---

[37] The best case scenario of an OSS community is when users involve themselves and participates in; bug tracking, problem solving and the implementation of the application code. The latter has been known to happen in successful projects.

[38] Of many definitions of "hacker" I find the following the most corresponding to my own understanding of the term "The dictionary defines 'hacker' as a slang term describing a person who carries out or manages something successful. A hacker is someone who spends many hours with the computer often successfully operating it by trial and error without first referring to the manual. A hacker is often a technical person in the computer field, such as assembly language programmer or systems programmer. Today the term hacker has taken on a negative meaning. The news media has often used the term hacker in a derogatory manner to refer to people that use their technical knowledge to gain unauthorized access and perform mischievous or destructive activity in computer systems and data banks. (U.S. Senate, 1996)"

extended time frame may still be to short. This brings us to the main research question of the thesis.

How can use of multiple open frameworks, design patterns, architectures, and tools create a lock-in situation for a flexible and open project?

As I have tried to show in the former (4,5, 6.2 -6.4) chapters, the use of the architecture, frameworks and design patterns all bring nice benefits, and frankly I have difficulties understanding how we could go one with out any of the them. The frameworks promise modularisation, flexibility and database independence, really fast and simple web view implementation through use of templates, etc.

However I also see that despite all the great benefits provided by the frameworks (Chapter 5), the high knowledge bar itself acts as heavy limitations on who of all the potential contributors that might join the project as a developer. In this case the Knowledge standards are tied through path dependency (2.2.3) back to the design choices in 2004. At the current point there is so many coding hours invested in the code base, that changing the use of framework, seems unlikely thus the installed base have reached a locking situation (2.2.2). Even though we have facilitated three layer architecture with consistent use of Inversion of Control and low coupling which makes such a transition feasible in theory.

Dependent on the different contexts the stability of the developer group has been unstable at best. In my own case many of the developers participating at the workshop in Noida left HISP during the next six months. The same happened during the first effort by Nordal back in 2005, and also in the Vietnam cases. The developer teams are quite unstable compared to the complexity of the frameworks. Due to the high risk discussed earlier (6.3.1), there should definitely been taken measures reduce complexity and to avoid further complexity.

6.5 Stakeholder and Politics

How different stake holder's expectations within the organisation can affect capacity building for system developers in OSS based HIS projects in a NGO organisation in India?

During our stay in India it became apparent that there were different interests at stake. I had a loose description of my tasks from my supervisor, but not all the other interests in the HISP

network aligned with my own. The different interests shaped the capacity building effort, and thus are interesting to study further.

There are two different problems which are worth noticing. The first event is how we are conducting the capacity building in a running production environment. Secondly we had different goals on what should be achieved by the capacity building effort. The HISP India Manager, is focused on getting "critical" tasks done as part of the workshop (such as integration of the GA), and how to do particular changes to the web view of the application (hacks[39]). While I am focusing on my role as a capacity builder and wanted to make a foundation for the developers to continue build upon after I had travelled home.

There is not necessarily a mismatch between the goals, but there is a mismatch between time limit and all the different subjects we wish to cover in that period. The hacking could have proved useful, but requires a lot of preparation time (which also was limited), since each hack demands a time consuming trial and error period. The big knowledge gap between the me and my fellow student and the GA, and the Indian developers big knowledge gap towards the DHIS2 made such resource demanding effort very unlikely to succeed, instead we made the compromise with the third training assignment, which has value for all states and align with interest on a global scale by using the DHIS2 API and uphold the database independency design (Chapter 6.2).

The problem of trying to initiate the capacity building in a running production environment was problematic because of many interruptions (Chapter 4.1.3 and 4.1.4) from high priority tasks. As explained in the empirical chapter (Chapter 4.1.3 and 4.1.4) the tasks were important and related to important presentations regarding new pilot projects, or funding.
If we look a little passed the fact that we were interrupted by highly critical requirements, and towards the cause of why the interruptions happens only few days in front of an important presentation, then we realise that the problem can be categorised as a project management problem.

---

[39] By hack I mean learning how to do a particular change with out understanding why it works. Hacking is a fine way to start implementing something, but I don't give a coherent overview of the application and it's architecture, thus can not act as a foundation for further capacity building.

On a CMMI (Royce, 2002) scale the project management would have a ranking of '1', meaning that the project delivery is saved by heroic contribution by a few of the project participants. Having a '1'rank on the CMMI scale does not necessary mean the projects fails to deliver. On the other hand it does suggest that the way the project is run should be scrutinised in order to increase product quality, and secure a more smooth development cycle. The inquiry is supposed to generate better routines and rules for the project in order to avoid the dependence of last minute heroic efforts. Since we already suffer from a high knowledge bar, and have many rules and guidelines on how to conduct coding, commenting, communication and so forth, more rules is not a favoured option. In stead measures such as having a more rigid release control in form of defining clear milestones and system testing. In addition to push last minute request to a later release.

Another perspective which might be partially responsible for the chaotic circumstance around important presentation is the fact that we are trying to use an OSS inspired development model (chapter 2.6). The model doesn't rely on heavy system testing by the developer team. In a successful OSS project the users are our best test resource. Do you remember the Linus saying; "given enough eyeballs all problems are shallow". However it is clear from my case data that the strategy does not work well when we are going to present new releases before it has been tested in the field for a while. We have a mismatch between the code quality we deliver from Oslo and what is expected or needed by HISP India. The problems manifested itself in an angry mail from the field in India (Appendix F – Mail from HISP India Cordinator1).

6.6 Infrastructure Limitations

In Norway we take availability the common infrastructure such as; accommodation, continuous electricity supply, continuous internet access and wireless internet in the office for granted. In India however, power black out, is quite common, especially during the raining season.

How infrastructure limitations can affect capacity building for system developers in OSS based HIS projects in a NGO organisation in India?

During the workshop in Noida the blackout occurred daily (scheduled blackout) and when it was raining there were more black outs. Black outs was a problem in Bhopal, but at the main

office in Noida HISP India installed an inverter[40] before the workshop started (but we experienced some start up problems). Both I and my fellow student had quite new laptops, which meant we had power for 2-3 hours after a black out, however some of the Indian developers used Desktops and some older Laptops with poor batteries, thus the blackout was a bigger problem with regard to computer access. The biggest problem however is that we rely heavily on internet for development resources (Google is an excellent reference tool for programmers). Further more many other projects artefacts such as database dumps are found on remote servers. The maven build tool also uses external resources to check for updates. As you might guess the router we used had no internal power supply. This meant that we lost access to some of our most valuable resources during blackout. The latter was a continuous problem in Bhopal, but was fixed in Noida, when we got the router plugged into a power socket powered by the inverter (the latter was harder than it seems due to lack of long enough power and net cables).

In general access to Internet was problematic. In Gujarat we had to use an Internet café a 10 min drive from the office. In Bhopal we suffered from black out, which also were a problem in the beginning in Noida. In addition the best internet access was seldom faster than 256kb, and some of the applications we use are quite heavy, and the DHIS2 project source code is also quite a huge task to update. In other words the internet was a little slow and the access might be a little unstable. During the first capacity building efforts in Bhopal, I often experienced the problem when I was supposed to show something (Murphy's Law). So after a while I stopped relying on remote sources and just used a memory stick for sharing data at the office.

None of the mentioned problems are fatal, but they do slow down the process, since we have to do many unscheduled breaks waiting for the power to return or getting internet access.

For the second part of the infrastructure review, the working environment of the HISP India is up for a closer look.

---

[40] The inverter has batteries and charges when there is power available. The inverter later serves as power source when blackouts occur.

How working environment affects capacity building?

In the different states the HISP India offices are organised after two different strategies. In Gujarat, HISP India had office space in the state public administration building. For the connection between HISP India and the state of Gujarat, such arrangement may yield many benefits on political and strategic level. As for the developers' possibilities to interact with each other, the office arrangement gave certain disadvantages. The DHIS2 project uses quite a lot tools and frameworks as you might already know. In addition to have a large code base, the job of configuring the tools on a new machine and download the necessary resources is very time consuming. For me and my fellow student the use of our laptops seemed the only real option. Since we were not allowed to connect them to the office network, we had to go to a near by café in order to work. The latter was unfortunate since we were not working with the State developers.

In Bhopal and Noida, we lived in the same house as the office. Further more the Indian developers lived in the same house which gave us a better common ground (office space) for sharing knowledge and working together.

7 Conclusion

The findings from my research, if anything outline that global development is perhaps more dependent on local capacity than the founders of the DHIS2 project assumed (Nordal, 2006). In the HISP India case we have experienced that we have a so fundamentally different installed base, that the matter of bridging the gap is very resource consuming with respect to time and human resources.

The initial technology review from (Nordal, 2006) has always been oriented against the future, by being OSS, flexible, highly modularized, in an attempt to reduce complexity. But the total number of frameworks and architecture used in this project actually add to the complexity by adding "invisible" links and "interceptors" (The complexity is moved from the application to the developers knowledge but it is still there). The number of different technologies used also puts a high demand for competence in for the individual developer in order to contribute to the system.

The use of framework in our project seems to be very reasonable since it greatly simplify the implementation job and handles common design problems in a simple manner. However the use of these frameworks is highly dependent of the projects access to necessary developer resources. In our case the main developer resource so far has been master students from Oslo, but most of them only participated a few months during the INF5750 course. A few of the students do their master thesis within the HISP project as well, which means they continue implementing for another six to twelve months. In India we had difficulties hiring qualified personal and also have a high turnover rate. Thus the use of frameworks and multiple frameworks in particular should be carefully leveraged by the organisations stability and the possibilities for attracting qualified help.

The project have always been aiming for staying "open", but because of our special domain we have so far been unable to draw upon the resources (with respect to recruiting new developers) of the open source software (OSS) community on a global basis. On the bright side the OSS course at the university in Oslo is very popular (due too the popular technologies), and combined with the OSS brand attracts new master students to participate in the project.

The project has to a great extent used opportunities when presented, but perhaps neglected a few resources which are possible to draw upon. After an interview with one of the Norwegian Coordinators, we discussed the approach on recruiting. There is perhaps possible to address the recruiting process to a more specific audience, like the PHP OSS community in India to attract more motivated developers.

> *"But to find highly experienced developers which know popular technologies and are looking for a low salary job is perhaps wishful thinking– Cordinater1 from Oslo"(Appendix H).*

My experiences from the HISP project lead me to believe that the choice of tools and technologies has paid little regard to availability of qualified developers. The initial capacity building efforts should perhaps alarmed us a little, but then again we could not have foreseen how thing would turn out in India and Vietnam in the long run. This brings me to conclude that the availability of developer resources should be given a greater concern in the project's technology review.

Even though the availability is there, then your project might not be able to attract the existing resources, nor hold on the ones you train on your own. Further more my findings suggest that if you have unstable developer team (organisation), or your future prospects with regard to resource as uncertain. Then there is a limit to the complexity your project is able to handle. The latter gives at least a few obvious paths on worth considering.

1. Reduce the initial scope of your project in order to avoid the complexity.

2. Keep the number of advanced tools and frameworks to what's necessary to keep the developers need for capacity building at a minimum to shorten the initial capacity building needs.

3. The possibility to make fast changes to your application is limited by the number of developers understanding the application and/or how fast you can get hold of one.

If we take a step back and consider, why is it important to have a flexible system? In Information Infrastructure I theory one reason is that chronological order causes in which

things are done creates path dependencies. The dependencies may lead to lock in situations (irreversible). Thus it may be irrelevant if you have a superior product when everyone else is using your competitors already. A company's ability to compete is dependent on its "agility", when the company is producing software and services agility (response speed) the latter translates to provide the customer with the required services and features. Flexibility in the software implies that it is easy (possible) to change, which should translate to that it is fast to change, which again lead us back to the company's ability to compete. My point was once expressed in better terms by a John Chambers, the Cisco's President and CEO.

> *"It's not the big that beat the small, but the fast that beat the slow." (Cook, 2004)*

8 Future work

In view of my concluding remarks a number of possible strategies outline as possibilities for further development of HISP as a knowledge infrastructure.

We can search for more alliances in form of universities as attempted in Vietnam. The initial attempt was based on a chance meeting between one of our developers and OSS interested professor from a foreign university. If we actively seek this kind of alliances in a particular audience like an OSS community in both Vietnam and India then the chance of discovering such possibilities may increase significantly.

Another way to achieve the former is to study the education options in both countries and see if any of them has special classes teaching java and our other java OSS frameworks.

We should perhaps also look into other ways of conducting the training. The training done by master students have suffered from mediocre knowledge of the technologies themselves, the capacity building effort as a whole is discontinuous, in both scale and scope, with regard to developers and technologies. In order to get some more continuous efforts, we can try to use remote learning, using video, audio, code examples.

Remote learning is quite difficult and not very much used compared to traditional class room teaching, but the model should be interesting for our particular case. If we compare the "classroom teaching" which is directed at a large audience and in such matter is similar to broadcasting with "on-demand", which is directed at individuals which suites a distributed network where it is difficult to get people together at the same time and space. The "on-demand model" can also be reviewed later, if we forgot parts of the "lesson". For some of the problems, the latter could prove to be a solution. One of the developers from India said "I have some of the spring framework book, but I did not understand it until now", right after we had a short lecture on how we used the spring framework, and how it was "wired together". During the spring 2008, the lectures were recorded, and it will be interesting to see how this works out.

Another option is to look into possible ways to link to the DHIS2 API and let the developer work on external applications based on other programming languages and frameworks, the

latter is a very short term strategy, but could perhaps be useful for prototyping. So far this has actually provided the only ‘successful’ local development effort with regard to DHIS2. From summer 2007 other projects developed in PHP has been initiated.

Something I mentioned several times is the need for a developers guide. We have extensive information on the different modules, but there is difficult to navigate in the content when you don’t know what you are looking for. What we really need is an “edited” book like version which can be red as a “how to” tutorial which describes the architecture, the modularisation, naming conventions, how to make new module, and go trough each necessary step. The biggest difference between the wiki we have now and a developer’s guide is the fact that a guide (book) should edited. The content should be ordered in an easily understandable fashion with an overall design. In comparison the wiki is mostly a collection of loosely coupled module descriptions and “how to” pages. Which works in some degree as a reference, but it is difficult to find things you don’t know exist.

The prototyping activity (creative play and possible solution exploring) could be significantly optimised by using simpler approaches such as simple all-in-on webpage using direct SQL insertion instead of using the Hibernate (5.7) and the other frameworks. Both because of the tool support and deploy speed of the simple tools, as well as the low participation bar. This makes the initial development steps much smaller. In fact this has until now proven to be the only option which have given significant output despite continues efforts to facilitate a more ‘correct‘ approach. Thus we perhaps should explore possibilities of using another prototyping approach in order to have an option for participation with a lower knowledge demand.

9 Dictionary of acronyms and abbreviations

AR – Action research an umbrella term for qualitative research methods with action as a central step in an iterative research cycle.

Ibid. – Latin ibidem, from/of same source

DHIS 2.0 – District Health Information System 2.0

DHIS 1.4 – District Health Information System 1.4

EDS – Essential Data Sets

GA – Graphical Analyser

HISP – Health Information System Program

HISP-India – Indian branch of HISP

HIS – Health Information System

ICT – Information Communication Technology

IDSP – Another health information system in India for collecting and analysing routine data (see appendix E)

II – Information Infrastructure

IT – Information Technology

IIT – Indian Institute of technology

MYSQL – OSS database, used in the development of DHIS 2.0

OSS – Open Source Software

PDF - Portable Document Format

PHC – Public Healthcare Centre

PHP – PHP: Hypertext Pre-processor

RIMS – Routine Immunization Monitoring Systems (see appendix E), another health information system in India for collecting and reporting Immunization routine data.

UIO – University of Oslo

UIP - Universal Immunization Programme (see appendix E)

10 Definitions

Definition of Capacity building: The WCO defines capacity building as "activities which strengthen the knowledge, abilities, skills and behaviour of individuals and improve institutional structures and processes such that the organization can efficiently meet its mission and goals in a sustainable way." http://en.wikipedia.org/wiki/Capacity_building

Definition of Best practice: A superior method or innovative practice that contributes to the improved performance of an organization, usually recognized as "best" by other peer organizations. www.asq.org/glossary/b.html

Definition of framework: A skeletal software component that performs functions required by a system and which is incorporated into the design of such systems. javaworkshop.sourceforge.net/glossary.html

Definition of Component: A piece of software with a clear function that can be isolated and replaced by another component with equivalent functionality. javaworkshop.sourceforge.net/glossary.html

Definition of Design Pattern: A design pattern is simply a defined and repeatable design solution to a common programming problem. Design patterns aren't code per se, but they define a proven recipe for creating the code to solve a given problem. I'll go over these in more detail in a later article. (en.wikipedia.org/wiki/Design pattern (computer science)

11 References

APACHE, 2007, "*Apache – Velocity*", [online], Apache foundation,
URL:http://velocity.apache.org/engine/devel/
[08.01.2007]

Argyris and Schön, 1991 C. Argyris and D. Schön, *Participatory action research and action science compared: a commentary. In: W.F. Whyte, Editor, Participatory Action Research,* Sage Publications, London, United Kingdom, pp. 85–98.

Baskerville, R. L., Wood-Harper, A. T., 1996, "A critical Perspective on Action Research as a Method for Information Systems Research", Myers & Avison, 2002, "*Qualitative Research on Information Systems"*, Sage, London

Benbasat, I., Goldstein, D.K., Mead, M., 1987, "Research Strategy in Studies of Information Systems", MIS Quarterly, 11, 3, 369-386, Myers, M. D., Avison, D., 2002, "*Qualitative Research in Information Systems*, Sage Publication", London.

Braa, J., Monteiro, E., Sahay, S. *Networks of Action: Sustainable Health Information Systems Across Developing Countries,* MIS Quarterly, Vol 28 No. 3 September 2004

Beck, K., 2002, *Test Driven Development: By Example,* Addison-Wesley Professional,
Creswell, J.W., 2003, *Research Design,* 2.ed., Sage Publication, , London.

Cook, B., 2004, "*Behind the Brand: Triumphs and Tragedies*",
URL:http://www.brandchannel.com/features_profile.asp?pr_id=161
[16.05.2008]

Dreyfus, S. E, Dreyfus, H. L., 1980, "*A Five-Stage Model of the Mental Activities Involved in Directed Skill Acquisition*", CALIFORNIA UNIV BERKELEY OPERATIONS RESEARCH CENTER, US
URL:http://stinet.dtic.mil/cgi-in/GetTRDoc?AD=ADA084551&Location=U2&doc=GetTRDoc.pdf
[12.05.2008]

Ecstein, R., 2006, "*James Gosling on Open Sourcing Sun's Java Platform Implementations, Part 2*", SUN,
URL:http://java.sun.com/developer/technicalArticles/Interviews/gosling_os2_qa.html

[06.01.2007]

EUROPEAN PARLIAMENT, 2001, "*Echolon Report*", FINAL, A5-0264/2001, PAR1,
URL:http://fas.org/irp/program/process/rapport_echelon_en.pdf
[27.01.2008]

FreeCol, 2007, "*FreeCol – the Colonisation of America"*, [online]
URL: http://www.freecol.org/
[30.12.2007]

Fossum, K., 2007, *Social Construction of Legacy Systems*, department of Informatics, University of Oslo, Oslo, Norway.

Gable, D., 1995 "*An introduction to action research*", National Association for Research in Science Teaching (NARST), San Francisco, April 24, 1995
URL:http://physicsed.buffalostate.edu/danowner/actionrsch.html
[08.11.2007]

Gladwell, M., 2002, "*The Tipping Point: How Little Things Can Make a Big Difference*", Back Bay Books.

Hanseth, O., Lyyttinen, K., 2005, *Theorizing about the design of Information Infrastructures: design kernel theories and principles"*
URL:http://www.ifi.uio.no/~oleha/Publications/ISRinfrastructurefinal05-12-05.pdf
[20.12.2007]

Hanseth, O., 2004, "Knowledge as an Infrastructure", In C. Avgerou, C. Ciborra and F. Land (eds) The Social Study of Information and Communication Technology: Innovation, Actors and Contexts, Oxford University Press, Oxford. 2004.
URL:http://heim.ifi.uio.no/~oleha/Publications/Knowledge%20as%20Infra%20MCC%20LSE%20book.pdf
[11.01.2007]

Herr, K, Anderson, G.L., 2005, *The Action Research Dissertation*, Sage Publication, Inc, London, pp. 1-7, 23-27,112 - 127

HISPINDIA, 2008a, "HISP India Carrers", [online], HISPIndia, India, URL:http://www.hispindia.org/index.php?section=122 [09.01.2007]

HISPWIKI, 2007a, "*DHIS overview*", [online], HISP URL:http://www.hisp.info:8080/display/DHIS2/DHIS+2+overview [12.01.2007]

HISPWIKI, 2007b, "*Mailing lists*", [online], HISP, Univeristy of Oslo, Oslo URL:http://www.hisp.info:8080/display/DOC/Mailing+lists [12.01.2008]

HISPWIKI, 2007c, "*Hisp Teams*", HISP, Univeristy of Oslo, Oslo URL:http://www.hisp.info:8080/display/HISP/HISP+Teams [15.05.2008]

HISP, 2008d, "*Installing DHIS2*", University of Oslo, Oslo URL:http://www.hisp.info:8080/display/DOC/Installing+DHIS+2

HISP, 2008e, "*Health Information System Program - HISP*", University of Oslo, Oslo URL:http://www.hisp.info:8080/display/HISP/Health+Information+Systems+Programme+-+HISP

Hustad, O. K., 2007a "*Meeting with Dr. Wikas Ben Desai Additional director of Health Services"*, 28.02.2007

Hustad, O.K., 2007c, "*Rims Export Module*", [online], HISP wiki. URL:http://hips.ifi.uio.no:8080/display/DHIS2/RIMS+Export+Module [14.11.2007]

Hustad, O.K., Strandli, S., 2006, "IDSP exporter plugin", [online], HISP Wiki, Univerisity of Oslo. URL: http://hips.ifi.uio.no:8080/display/DHIS2/IDSP+Export+Module [15.11.2007]

INF5750, 2006, *Open Source Software development and Java frameworks in global networks*, [online], University of Oslo, URL:http://www.uio.no/studier/emner/matnat/ifi/INF5750/h06/ [01.03.2007]

INF5750, 2007, *Open Source Software development and Java frameworks in global networks*, [online], University of Oslo, URL: http://www.uio.no/studier/emner/matnat/ifi/INF5750/h07/ [12.11.2007]

Johnson, R. E., 1997 "*Communications of the ACM*", Volume 40 , Issue 10 (October 1997) Pages: 39 - 42 , ISSN:0001-0782 , URL: http://portal.acm.org/citation.cfm?doid=262793.262799 [02.04.2008]

Marshall, G., Ruohonen, M., 1998, *Capacity Building for IT in Education in Developing Countries,* 1. ed., Chapman & Hall, London.

Myers, M.D., Avison, D., 2002, *Qualitative Research in Information Systems*, Sage Publication, London.

JETTY, 2008a, "*Jetty 6 – Jetty web server"*, [online], Mort Bay Consulting, URL:http://www.mortbay.org/ [09.01.2008]

JETTY, 2008b, "Introduction to Jetty6", [online], Codehaus, URL:http://docs.codehaus.org/display/JETTY/Introduction+to+Jetty6 [09.01.2007]

Lin, Y., 2003, "*The Institutionalization of Hacking Practices*", ACM, Ubiquity, Vol. 4, Issue 4, March 18 - 24, 2003
URL:http://www.acm.org/ubiquity/views/y_lin_1.html?searchterm=knowledges [15.05.2008]

Nordal, K., 2006, "The Challenge of Being Open - Building an Open Source Development Network", Master Thesis, Department of Informatics, University of Oslo, URL:http://www.hisp.info:8080/download/attachments/2374/KristianNordal.pdf?version=1

Meese, A. J., 1999. "Monopoly building in cyberspace: how many products does Microsoft sell?", The anti trust Bullitine, Spring 1999.
URL:http://papers.ssrn.com/sol3/Delivery.cfm/SSRN_ID922136_code58849.pdf?abstractid=922136&mirid=1
[27.01.2008]

OPENSYMPHONY, 2008, "*Xwork documentation*", [online], OPENSYMPHONY, URL:http://www.opensymphony.com/xwork/
[08.01.2008]

Raymond, E. S., 2000, "The Chatedral and the Bazaar", [online], Thyrsus Enterprises, URL:http://www.tuxedo.org/~esr/, [15.01.2008]

Royce, W., 2002, "*CMM vs. CMMI: From Conventional to Modern Software Management*", Rational Software.
URL:http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/feb02/ConventionalToModernFeb02.pdf
[16.05.2008]

Schmidt, D. C., 1995, "*Using Desgin Patterns to devlop reusable Object Oriented Communication software*". Communication of the ACM, October 1995, /Vol 8, No 10, pp 65 -74,
URL:http://delivery.acm.org/10.1145/230000/226255/p65-schmidt.pdf?key1=226255&key2=0795057021&coll=GUIDE&dl=GUIDE&CFID=23106844&CFTOKEN=49568300
[06.04.2008]

Sfrat, A., 1998 "*On Two Metaphors for Learning and the Dangers of Choosing Just One*", Educational Researcher, Vol. 27, No. 2 (Mar., 1998), pp. 4-13,

Stallman, R., 1986, " *Gnu's bulletin*" February 1986 G N U ' S B U L L E T I N, Volume 1 No.1, URL: http://www.gnu.org/bulletins/bull1.txt

[10.01.2008]

Silverman D., 2004, *Doing Qualitative Research,* 2 ed., Sage, London

SPRINGFRAMEWORK, 2008, "*About Spring*", [online], Springframework,
URL:http://www.springframework.org/about
[08.01.2007]

Srinath, U., *Usha Srinath-Met Vikas Ben Desai.txt*, 2007,
14 March 2007
WHO, 2007, Routine Immunization, RIMS SOFTWARE,
URL:http://www.whoindia.org/en/Section6/Section284/Section286_506.htm [14.11.2007]

Patra, N., 2006, "*Universal Immunization Programme in India: The Determinants of Childhood Immunization*". Available at SSRN: http://ssrn.com/abstract=881224
[14.11.2007]

Sfard, A., 1998, *"On Two Metaphors for Learning and the Dangers of Choosing Just One"*, Education researcher, Vol 27, No 2, pp. 4-13, 1998

Soundarajan, N., 1999, " Understanding Frameworks", Computer and Information Science The Ohio State University Columbus, OH 43210, USA , URL:http://www.cse.ohio-state.edu/~neelam/papers/uf.pdf
[01.04.2008]

Trac, 2008, "*The trac RoadMap",* online
URL:http://trac.edgewall.org/wiki/TracRoadmap
[15.05.2008]

Turner, J., 2008, "Worthlessness of Code", [online], O'Reilly
URL: http://www.oreillynet.com/onlamp/blog/2008/03/the_worthlessness_of_code.html
[04.06.2008]

U.S. Senate, 1996, "*SECURITY IN CYBERSPACE*", [online], U.S. SENATE,
PERMANENT SUBCOMMITTEE ON INVESTIGATIONS
(Minority Staff Statement) JUNE 5, 1996, APPENDIX A - Computer Terms and Definitions

URL:http://ftp.fas.org/irp/congress/1996_hr/s960605a.htm

Øverland, L. H., 2006 “Global Software Development and Local Capacity Building: A means for improving Sustainability in Information Systems Implementations”, Master thesis, University of Oslo.
URL: http://www.hisp.info:8080/download/attachments/2374/Master+Thesis+-+Lars+Helge+%C3%98verland.pdf
[03.03.2007]

12 Appendix A

At the beginning of the thesis we looked on a little of the history behind HISP, the development of the DHIS began back in 1994 (Braa et al).

> *The Open Source Software developed by HISP is called the District Health Information Software (DHIS) because the key issue in South Africa after apartheid was to integrate the various types of data and reporting from different health programs within the District Health system. (HISPWIKI, 2007a)*

At the time most of the HIS was fragmented on a vertical level as many different programs were present in South Africa where each had their own management and their own data collection systems. (Braa et al). The solution was to establish a *district* database (thus the District Health Information System), where data from all (health centres, hospitals, clinics etc.) and health programs (e.g. mother and child health, immunisation) were managed (HISPWIKI, 2007a).

12.1 DHIS 1.x

The term DHIS 1.X refers to any of the DHIS 1 versions, the rationale for using the 1.X term is to address all the different versions implemented in Visual Basic and Access ad one group, since we often need to extinguish between the 1.X and the Open Source version (DHIS 2.0).

The DHIS 1.x is based main target was handling routine data like Demographic data (census) and data about infrastructure (e.g. number of beds, equipment) and personnel (number of doctors, nurses etc.) were also included. The raw data in itself does not much use with setting it in a relative perspective, and the use of an indicator engine (module) makes it possible to combine and correlate different types of data into indicators; immunisation coverage (number of e.g. vaccines/target population), average length of stay (in-patient days/discharges), bed-occupancy (in-patient days/ beds), resources per population (e.g. doctors per population, beds per population) etc. (HISPWIKI,2007), hence turning raw data into useful information.

The possibility to combine all kinds of data from the health services, population,

infrastructure etc. has shown very useful in South Africa and has enhanced management, planning and monitoring activities at the various levels of administration; from hospital management up to provincial management. The system use developed from the scarce beginning - when the goal was to collect a "general" set of essential data - to the more specialized tasks (HISPWIKI, 2007a).

> *The collection and use of healthcare data has changed some of the administration practices.*
>
> *For example; budget allocation to all hospitals in South Africa is now done based on a particular set of data reported through the DHIS where targets are compared with performance (e.g. in-patient days, bed-occupancy, average length of stay, resource utilisation, budget follow-up) (ibid).*

> *Nutrition exemplifies is another area where a specific set of data is reported organisation learns to manage database technology and appropriate approaches; they invent new ways of using the technology (ibid).*

The two latter examples is part of the HISP strategy of action driven data collection. HISP is trying to introduce a practice where the local administration uses their data for planning immediate action (The action cycle follows the same steps as described in the Methodology chapter) which in the latter case proved to change the need for data gathering (ibid).

The DHIS 1.x versions are by far the most actively used today comparing to DHIS 2.0. DHIS version 1.3.0.x (the Version & BUILD as per 24 July 2005 is 1.3.0.84) was used in South Africa, Malawi, Mozambique, Tanzania, Ethiopia, Nigeria, India, China, and Uganda to a larger or lesser degree. The population covered by existing routine data is tentatively between 70 and 100 million people (ibid).

12.2 About DHIS 2

The rationale for developing DHIS 2.0 is to develop a platform independent and web-enabled version of the DHIS. The DHIS 2.0 requirements originated from the earlier work with DHIS1.X. The main reasons for re-implementing the system, was the wish to create a web based application, which could run online and offline, using completely open source software

technology (HISPWIKI, 2007). On the HISP wiki I found some of their original requirement for the DHIS 2.0, the DHIS2 can be regarded as a hybrid application suite that should cater for at least three main needs:

> *The need to rapidly design data collection tools for a variety of purposes.*
>
> *The need to efficiently capture or import/collate this variety of data in an "integrated" manner, then to monitor the processing and flow of this data, and finally to communicate with various stakeholders in the process .*
>
> *The need to analyse relatively large and complicated data sets quickly and efficiently.*
>
> *Another way of saying the same is that the term "hybrid" denotes that the DHIS actually needs to combine the typical properties of a flexible TRANSACTION database with the typical properties of a DATA WAREHOUSE and some properties of a COMMUNICATION TOOL. (HISPWIKI, 2007a)*

The DHIS 2.0 is developed using the Open Source Java frameworks and tools: The Spring Framework, Hibernate, WebWork and JUnit are Java frameworks used in the implementation. In addition the source build tool Maven and the source content management system Subversion is used for building the source code, and distribute updates, respectively. The tools and frameworks are described below.

The software development process is a global collaboration between students, researchers and experienced developers in Norway, India, South Africa, Ethiopia and Vietnam, (HISPWIKI, 2007c). There are however a large gap between which technologies developers in the different countries is familiar with, thus the challenge of capacity building. Below I have described the different technologies used in DHIS, and our capacity building effort.

12.3 Programming Language – Java SE 5.0

The DHIS2 application is developed in regular Java 5.0 Standard Edition mostly on the server side, with additional java based OSS framework such as JUnit, Hibernate, Spring, WebWork to simplify the development and secure different properties. Java is a object oriented programming language, but some of the framework we use complement java with new properties.

I will not say much about Java except that is is widely used in Norway, it has been the most common Object Oriented programming langue used in courses I have taken through my days at the University of Oslo (2002-2007). Java commonly used to learn as an introduction to programming or introduction to Object orientation in Many Norwegian Universities and Colleges; University of Oslo, NTNU, University of Environment and Biological Science (UMB), NITH, to mention a few. Java is generally known as one of the 2 major programming platforms in Norwegian IT industry, along side with the .NET platform.

In November 2006 Sun Microsystem[41] announced that they are going to "open source" java SE and ME. In a interview view with James Gosling[42] - one of the "fathers" of java – the effort was revealed (Ecstein, 2006).

12.4 IDE – Eclipse

Eclipse is a fantastic Integrated Development Environment (IDE), originally Eclipse was developed by IBM, but was released under Open Source Software license under the Eclipse foundation as a Java IDE. Eclipse has many nice features as code completion (showing a list of legal choices when programming), under line and in other ways tell you about syntactical mistakes, has JUnit as an integrated Unit testing system, and an amazing file/package/file/class explorer to navigate in your development projects.

In addition you can install plug-ins, and there are many really useful plug-ins. SubEclipse is a Subversion plug-in for eclipse, there exists a Maven plug-in, but Ant is the default build tool in Eclipse, there also exists plug-ins which enable Eclipse to give similar programming support to other programming languages, such as Python and PHP to mention a few. The list just goes on and on.

12.5 Source Code Management system - Subversion

Subversion(SVN) is an open source "source code management system" (also known as a file versioning system), used to distribute source code and updates on source code between developers. The system can be used to distribute any kind of files, but in DHIS it is primary

[41] Sun is the company that develop the Java programming language.
[42] James Gosling is the vice president of Sun Microsystems

used for source code. One of the more brilliant properties of SVN is it backup functionality through file versioning, when ever you change a source file, and "commit" the change, SVN might bring back the current and any former committed version of a file.

Subversion uses a repository to keep track of changes, the latter is placed on a server (server side). In addition every user has a local copy (working copy) on their local machine (client side). The user (client) first download a "working copy" to their computer, make the necessary changes, and send the change files back (commit) to the repository. The update will now be available whenever another user tries to check out (download for the first time) or update their working copy (check if anyone has committed something to the repository).

The issues of file versioning systems, distributed source code and parallel development is out of the scope of this thesis.

12.6 Build Tool – Maven

Maven is an open source build tool, made for managing and building source code and managing dependencies to external libraries. The term building include a seven step lifecycle process which include downloading external libraries, compiling the source code, running unit tests, packing the complied classes in to JAR[43] or WAR[44] files, and install or deploy the jar files.

Maven build uses a file project object model (POM) file to describe a project, the POM file may include links to sub projects as well as dependencies to external projects and libraries. When external references are given, maven downloads the external libraries and copies them to a local maven repository. The local repository is known as "maven home", and is placed in a folder which is on the "class path". The external libraries in "maven home" are then

[43] JAR is the acronym for a java archive. The java archive is a specific folder structure, containing in particular a folder named META-INF which contains the file MANIFEST.MF. The latter file can contain many sort of information, but most common is the path to the "main class" if such a class exists. If the latter path exists the jar file may be executed. The JAR files are compressed using a compressing algorithm similar to ZIP.
[44] WAR also known as Sun WAR is short for **W**eb **Ar**chive may refer to a JAR containing a collection of JavaServerPages, servlets, Java classes, XML files, tag libraries and static Web pages (HTML and related files) that together constitute a Web application.

available to all local maven projects. Maven also contains a reporting tool for making reports of the build process.

12.7 Spring Framework

The Spring Framework is an open source application framework for the Java platform. The strategy is not to compete with good existing solutions, but should foster integration. (For example, JDO, Toplink, and Hibernate are great O/R mapping solutions. We don't need to develop another one. The spring framework is to support centralised, automated configuration and wiring of your application objects (SPRINGFRAMEWORK, 2008). Some of the features advocated by SpringSource (Leading developer of the Spring framework) are:

The most complete lightweight container
Integration with Hibernate (ORM)
AOP functionality
A flexible MVC web application framework
In addition spring supports several other Object Relation Mapping (ORM) frameworks. Furthermore Spring provides a common layer for transaction management and JDBC abstraction layer for error handling and reduce the amount of code to write.

12.8 Test framework - JUnit

JUnit is an open source unit testing framework for java. The latter is used to control the behaviour of a particular class or method and that they provide the correct answer to a given input. To completely test a unit of code, you should check response four different domains of values; the legal values, the lower limits, the upper limits, and the illegal values.

The unit testing can also be used as regression testing, once implemented you can guarantee that the unit of code gives a certain output based on a certain input, if changes elsewhere causes your tests to fail, then it will be detected at once, the tests (if written in a correct manner) also provide information of which tests that fail, and thus provide information of what part of your system which is broken.

12.9 Web development framework - WebWork

WebWork is an open source java web application development framework. WebWork is designed with a web customised MVC pattern in mind, in fact WebWork support two different versions of the MVC pattern (Front controller and Page controller). WebWork also support internationalisation (i18n).

In addition WebWork include a lightweight servlet container, at the current date there exits many different servlet containers and each have their special features. WebWork is enabled to configure which container to use, thus the possibility to enable containers with different pros and cons.

A powerful property gained by using a lightweight container is the “Inversion of control”, we can define the coupling between different objects thorough use of a xml configuration file(s) and the lightweight container. The latter makes the links between objects clearer and easy to manage.

For easy webpage programming WebWork supports the user of Velocity, which is a Java-based template engine. One of the major strength is the easy access to objects defined in the java code (APACHE, 2008).

> *When Velocity is used for web development, Web designers can work in parallel with Java programmers to develop web sites according to the Model-View-Controller (MVC) model, meaning that web page designers can focus solely on creating a site that looks good, and programmers can focus solely on writing top-notch code. Velocity separates Java code from the web pages, making the web site more maintainable over its lifespan and providing a viable alternative to Java Server Pages (JSPs) or PHP (ibid).*

At last it is worth mentioning that WebWork build on a command-pattern framework Xwork. The command-pattern framework is used to power Struts 2 as well. (OPENSYMPHONY, 2008)

> *XWork provides an Inversion of Control container, a powerful expression language, data type conversion, validation, and pluggable configuration. (ibid)*

Xwork was created as part of the development of WebWork 2.0 and later versions as an attempt to separate the non-web related part of WebWork into another framework. The separation makes it possible to use many of WebWorks non-web related features in other types of Software.

12.10Communications

Since HISP is a global network, good, cheap and effective communication is of the essence. I separate between different kind of information communication, I separate between different types of communication.

12.10.1 Documentation – Confluence wiki

HISP has its own wiki page (www.hisp.info), containing information about HISP, and it's development projects. HISP uses a wiki application called Confluence. The application is not an Open Source Software application, but it is free to use for Educational purposes. The WIKI allow it's users to read, add, edit, correct and remove the presented content and even make new sections. The wiki is the main source of documentation of the DHIS 2.0 project which includes conceptual information about the different modules of DHIS2.0 and how to use them.

12.10.2 Bug tracker and project management - Trac

Trac is an open source wiki specialised for development projects, it therefore contains project management system, which also can be used for bug tracking.

The system is simple, first part of the project management is to outline a roadmap[45], the roadmap tells us when and where requirements are going to be completed. The road map is organised by milestones, and each milestone contains tickets. The management process is driven by splitting the requirements into tickets (a small atomic task with a specific measurable goal), and distribute the tickets on different milestones. The milestones are in chronological order and when a milestone deadline occur the project participators is supposed to have completed every task linked to that milestone. The latter is not always the case however. Tickets can prove to be irrelevant, useless or too time consuming and might be discarded or moved to a later milestone.

I the DHIS 2.0 development bugs are handled as tickets, earlier we used a system know as Jira for this task, but the Trac offer multiple purposes. Some of the developers even argued for moving the content on the confluence wiki to Trac, because the confluence version used by HISP is old, and that Trac may serve multiple purposes. The main rationale was to lessen the burden of maintaining (updating) by using only one piece of software.

[45]Roadmap in this context refers to a plan. The roadmap provides a view on the ticket system that helps planning and managing the future development of a project (Trac, 2008).

12.10.3 Email

The email service is HISP's most common information stream in addition to the word of mouth. Since the DHIS development is distributed into very different time zones the need of an asynchronous messaging service is dire. HISP mainly use four global email lists

Tabell 2 - Email lists (HISPWIKI, 2007b)

hisp AT hisp.info	HISP general list	http://www.hisp.info/mailman/listinfo/dhis-dev	
dhis-users AT hisp.info	User feedback and help list	http://www.hisp.info/mailman/listinfo/dhis-users	
dhis-dev AT hisp.info	Developer list	http://www.hisp.info/mailman/listinfo/dhis-dev	Required for developers
dhis-scm AT hisp.info	Subversion commit mails	http://www.hisp.info/mailman/listinfo/dhis-scm	Required for developers

There also exits National mailing lists to support the national need of the developers in the different countries, but most technical questions are on the international list (dhis-dev AT hisp.info), at least more than on the Indian developer list which I have been member of for the last 10 months. On the Indian mailing list I mostly receive spam mail. The latter was so I annoying at times that I actually had to start using a more aggressive spam filter on my laptop.

12.10.4 Instant messaging

Instant messenger services (like MSN[46]) is one of my personal favourites for communication and problem solving when programming. Due to a significant time zone difference (5.5 hours) between India and Norway this proved less efficient in India. Since most of my personal network did not logon until 16:00 (Local time). Our Indian friends preferred the Google-talk[47] instead of MSN, the latter is a significant blocking factor for communication.

Internet is also slow, unstable and a little expensive in many parts of India, Besides and good internet cafes which provide the necessities for tedious development effort is hard to come by.

[46] MSN is a Instant Messenger (IM) service for the Windows platform, the MSN client application is shipped with the Microsoft Windows operative system, which as you probably already know, pre-installed on approximately 92% of all sold PC's. By common users in Norway, this is the preferred IM, in India however the Google talk is more common.

Many of the HISP India developers don't have continues access to a Desktop or laptop computer at home and are confined to limited access at the various HISP state offices or at a local internet café.

12.10.5 VoIP – Skype

During our stay in India we participated in a few online meetings using Skype, or at least we were suppose to. The idea was to have coordinators, Norwegian developers in Oslo, and the Indian coordinators and developers join in. However limitations in the size of chat groups on the free Skype client, as well's as limited access to headsets and computers, made this difficult. In the end only a few (the most important decision takers) participated in the discussion and a summary was distributed to the rest us.

12.11Sql Server – MySQL

In the DHIS2 project we are open to using other tools than the most common ones. MySQL is most used by far, but PostGresSQL is generally known as a better database implementation than MySQL. But the Hibernate Framework makes the choice of database irrelevant for the most part since Hibernate is suppose to do all database communication with the database.

HISP strive for keeping the DHIS2 database independent but at a few occurrence there has been implemented directly against the database to optimise performance. The latter was the case for on implementation effort for the Datamart[48] module. The pre aggregated data is used both for DHIS2 internal data presentations, but also represents an easy way to integrate against the DHIS2 database (though integration of such features into the DHIS2 application is our preferred choice). The Indian module "Graphical Analyser" was built using the latter approach before it was integrated into the DHIS2 Spring 2007.

12.12Web server – Jetty (maven plug-in)

Jetty is an open-source, standards-based, full-featured web server implemented entirely in Java. Jetty has benefited from a large user community, thus been able to establish a stable core of lead developers (Jetty, 2008a). Jetty can be used as:

[48] The Datamart aggregates data values as a batch job in order to quickly access pre-aggregated data for presentation purposes. The latter is very useful for databases with our design. The database is very flexible, but demands many joins and a little analysis in order to present useful information. If you add the fact that a Sample database for 5 out of 23 districts in a state with sample data for a few months actually contained approximately 1.6 million rows of data values, then the importance of pre-aggregated data becomes obvious.

a stand-alone traditional web server for static and dynamic content
a dynamic content server behind a dedicated HTTP server such as Apache using mod_proxy
an embedded component within a java application
In the DHIS2 project we use the maven jetty plug-in to run the web-server during development testing. However when we were in the field in Gujarat in India, the original Jetty server was prime choice. At the jetty website the following features are advocated as the defining features of Jetty; simplicity, scalability, efficiency, embed ability, plug ability.

12.13Online DHIS2 Demo

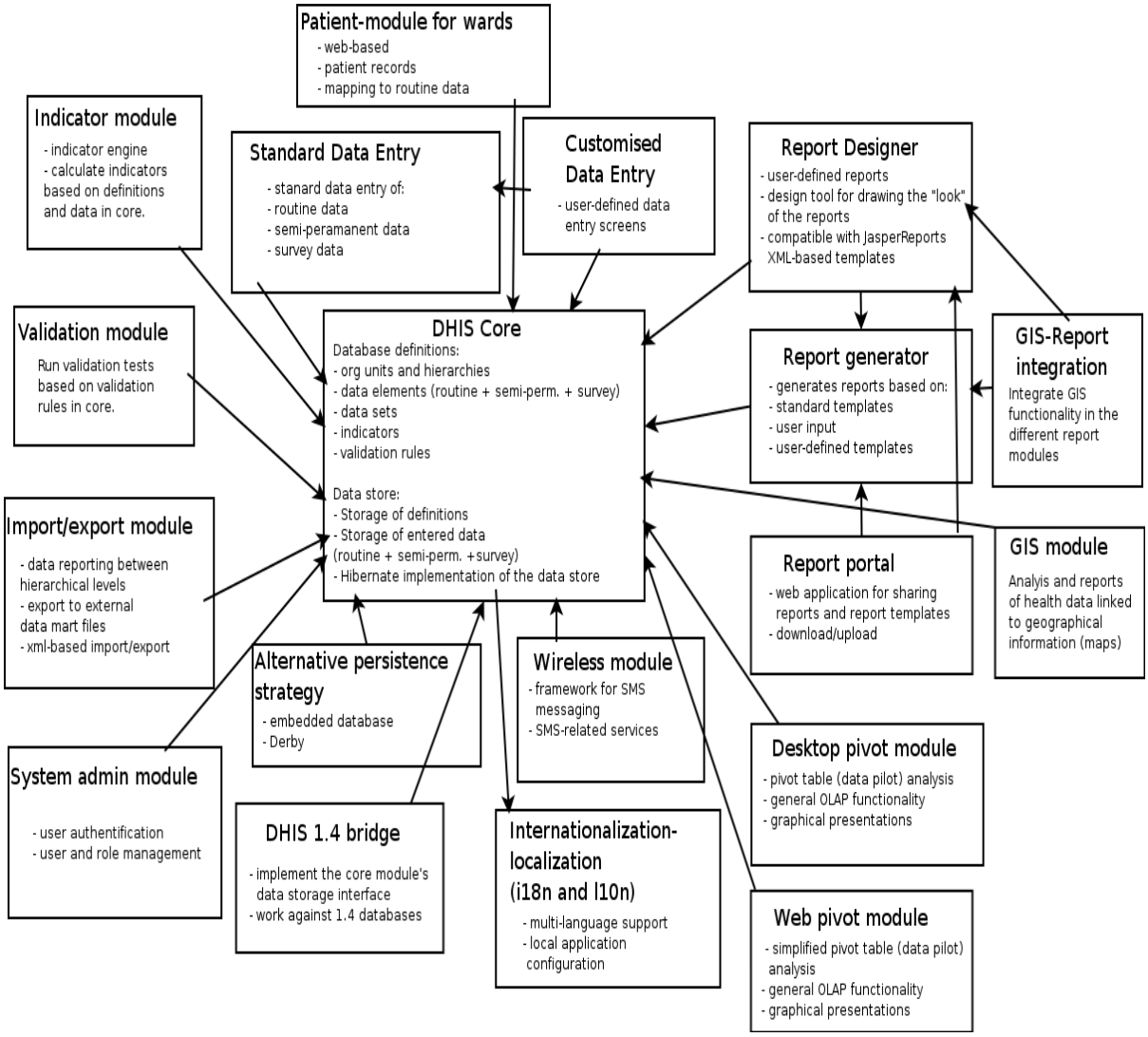
The current version of the DHIS2 is running for demonstration purposes at:

http://www.hisp.info:8090/

Username: guest

Password: guest

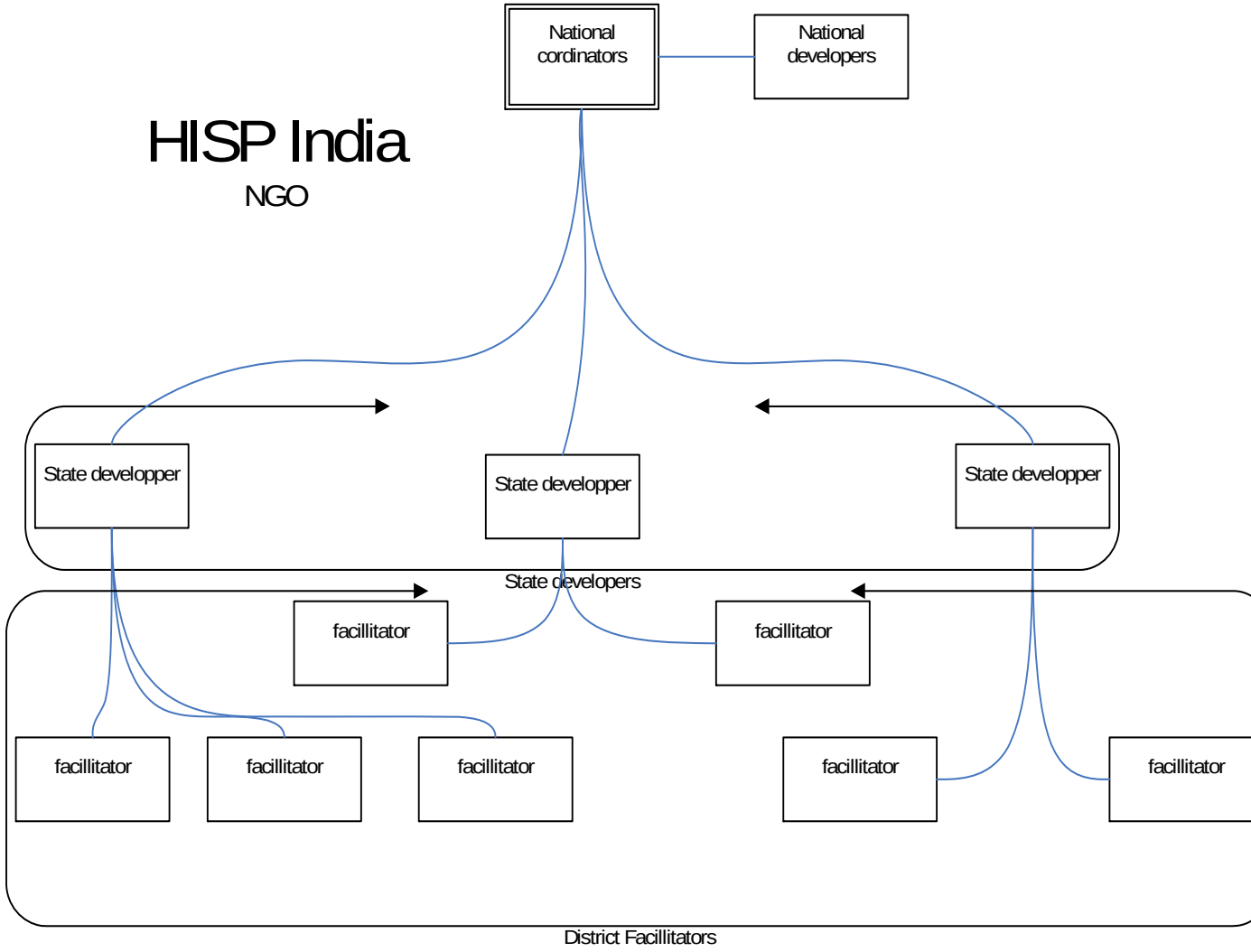
12.14A conceptual overview of DHIS 2.0



Figur 8 - Overview of the DHIS 2.0

Above I have added a conceptual overview of DHIS2 application which I found on the DHIS2 web site (HISPWIKI, 2007a). The module names is on an abstract level is self explaining, and do not intend to explain each part in detail. In stead I will comment on the divergence between the model and the DHIS2 situation at the beginning of 2008.The conceptual model is accurate enough, but hibernate is used in the current implementation. The alternative persistence strategy mentioned in the model is therefore currently unused. The Patient based module is currently not implemented. Work on an external “name based system” have been started .The wireless module is currently on ice due to low human resources.

13 Appendix B - HISP India



Figur 9 -HISP India organisation structure 2007/2008

HISP India, is a National NGO branch of HISP in India. India and Vietnam, were the main targets for implementing the DHIS2. A strategy for further development of the project is to "out source" the core development of DHIS2 to India or Vietnam. The capacity building effort in India was a step to support this move of core development activities.

The model of the HISP India network is supposed to show the working hierarchy in HISP India at the time. I have focused on management, developers and technical support (facilitators), though many other roles exits in the network. HISP is led by coordinators for the University in Oslo, but also by the national level coordinators in India.

The overall strategy is to have one or more state developer in each of the states HISP India is currently having a pilot project. The responsibility of the state developer is to gather and implement state dependent requirements (Mostly reports). While more general development is suppose to be done by the national (graphical analyser (Appendix C)) and global developers. In each of the respective states HISP India has a main state office. The main office is often a medium sized house (or a floor) which functions as accommodations for developers and coordinators.

The national coordinators organize the different implementation[49] efforts, engage in government meetings, presenting the recent development with field data. Thereby gaining continued contracts for pilot projects, support and further development and perhaps even increasing the number of healthcare units enrolled into the HISP India network.

HISP India is active in many states, but can not afford to have a coordinator at all state offices. Thus the facilitators travel from state to state usually in order to have meetings with government, check up on the working progress, organise user training, gather requirements, pay salary and fix miscellaneous administrative tasks.

At the different Healthcare administration offices, there are engaged facilitators to help with the immediate needs of the healthcare workers regarding use of the DHIS2 software. There can be many facilitators at a single office.

The coordinators also have the overall economical responsibilities, in particular this includes presenting HISP's work for the different customers, show that the requirements are met, and collect payment. As well as distributing salary. The use of bank accounts is less common in India, in fact some of the HISP India employees does not have one (Field notes, 24.04.2007).

[49] In this context implementation also refers to the socio technical development like meeting, user training and so forth.

13.1 Role descriptions

The following job descriptions as they are described on the HIP India website, this is the role as it is intended to be. This is however not always written in stone. It has been practices for higher ranking members of HISP India, to start as facilitator, then work a little as developer and become a manager/coordinator role at some later point in time.

13.1.1 Facilitator

The system facilitator will be responsible for the HISP India project implementation in a cluster of geographically proximate primary health facilities.

Implementation responsibilities will include activities of (HISPINDIA, 2008a).

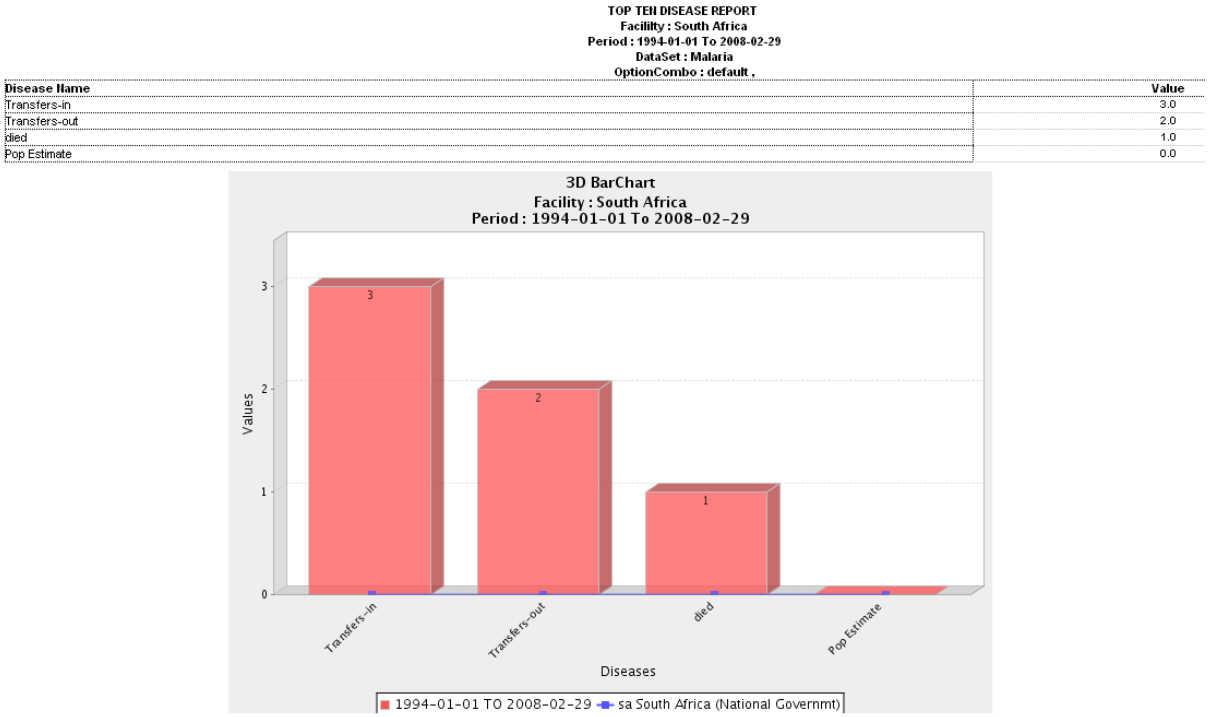
Database installation and management (update of data and software),
Facilitating data entry, providing technical support when needed, and report generation based on understanding of public health issues,
Training health staff on data entry, report generation, and online sharing of data and reports,
Preparation of training content and its implementation,
Interaction with all field staff (JPHN/HS/HI/LHI/LHS) and doctors to understand their analysis needs, and convey the same to HISP coordinators,
Ensuring the provision of all reports as defined by the HISP India reporting manual for the district, and corrective action to fill the required gaps,
Discussion and improvement (and cleaning) of data dictionary based on understanding of local public health issues and interactions with health personnel,
Active participation in planning, coordinating, and implementing deadlines of HISP field activities,
Being the point of interface between health personnel and HIS, and responsibility for all interactions between them, including reporting on progress, informing HISP plans to health facilities and responding to specific requests the health facilities might have.

13.1.2 State Implementation Coordinators

HISP India is now looking for "STATE IMPLEMENTATION COORDINATORS" in three States, Madhya Pradesh, Chattisgarh and Kerala. Persons with Medical Background and MPH (Master in Public Health) with a minimum of not less than two years experience are eligible for applying for this position. Aspirants for this position should have sound knowledge of Health Domain preferably Public Health and good management skills with fluency in English and local languages of the mentioned States. (HISPINDIA, 2008a).

14 Appendix C - Graphical Analyser

The Graphical Analyser (GA) is a dashboard designed to use the “aggregated data values” from Datamart (see appendix A) and present them in a easy understandable manner. The GA goes under the term “dashboard” due to the layout and its functions.



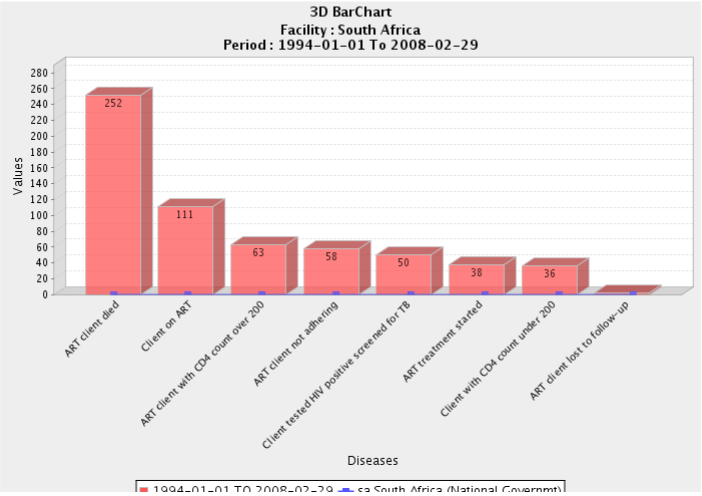
TOP TEN DISEASE REPORT
Facility : South Africa
Period : 1994-01-01 To 2008-02-29
DataSet : Malaria
OptionCombo : default ,

Disease Name	Value
Transfers-in	3.0
Transfers-out	2.0
died	1.0
Pop Estimate	0.0

Figur 10 - The dashboard picture is from the integrated version in the DHIS2 demo

The Graphical Analyser was originally written in JSP and used JDBC connection to the DHIS2 database. The original GA was implemented as an external application which presented statistics based on pre calculated data from the DHIS2 Datamart.

Disease Name	Value
ART client died	252.0
Client on ART	111.0
ART client with CD4 count over 200	63.0
ART client not adhering	58.0
Client tested HIV positive screened for TB	50.0
ART treatment started	38.0
Client with CD4 count under 200	36.0
ART client lost to follow-up	2.0



Figur 11 - The dash board picture is from the integrated version in the DHIS2 demo

The Original implementation effort was done by the Indian national developer, the GA represents data in an easy understandable manner through use of charts.

The functions of the GA adds much value to the DHIS2 and such an integration had been requested from Oslo, furthermore the Indian coordinators had promised get the GA integrated. The integration of the GA was one of the primary goals for the global HISP organisation (Field notes, 11.04.2007), and started after the workshop in India. The integration would give the rest of the DHIS2 network access to the GA features. In addition integration against the DHIS2 API will make the GA more independent of database changes and database independent (due to hibernate).

15 Appendix D - Training Exercises

In this appendix I have added the assignments we used in the training. It is a fast introduction to some of the basic frameworks. For each assignment we held a small lecture about the terminology and use of the necessary applications and frameworks. We distributed the lecture documents (PDF[50]), and applications using a USB memory stick. This proved to be the most safe way, since we often lost the internet connection due to power black outs. In addition we had a shortage of working Ethernet (RJ45 Network connection) cables.

The first Training assignment is an introduction to Subversion(See Appendix A) and Maven (See Appendix A). The participators are supposed to make a repository on a central server, check out the repository to their local machine (Working Copy), add files to the working copy, and commit them back to the server. The files include a simple java "HelloWorld" program, and a maven "POM.xml" for compiling it and make an executable JAR file. The participator is supposed to commit all source files to the central repository for evaluation.

The second assignment is about the Spring Framework(see Appendix A). The participator is supposed to learn and understand the concept of "Inversion of Control" and how to wire together loosely coupled objects, using a simple configuration file (bean.xml). As in the last assignment the participator is supposed to commit the files to an central repository for evaluation.

The final exercise was one we made in Bhopal in April 2007 (before the workshop). In every State in India, they have a great number of specially formatted reports which they want to produce each month. The former solutions are made with JSP and direct database connection. The coordinators and developers in Oslo strongly advocated that these forms should use the DHIS2 API for data access thus making all the forms more resilient to database changes. So back in Bhopal I and my fellow student made a prototype report using the DHIS2 API instead of the direct database connection. We found a way to integrate the current solutions which enabled both the new and the old to coexist at the same time. Afterwards I did another reimplementation of reports in collaboration with the national developer. We never wrote an

[50] PDF is the acronym for Adobe's Portable Document Format(PDF)

assignment specification for the last assignment. But planned on splitting the workshop participants in three groups, and then let me, my fellow master student, and the HISP India national developers guide one group each.

Below you can find the exact assignment specification for training assignment I & II.

15.1 Training Assignment I HISP India Workshop 2007

15.2 Software Requirements

15.2.1 Linux

- Subversion
- Maven2
- Eclipse

15.2.2 Windows

- Cygwin with Subversion or TortoiseSVN + Putty
- Maven2
- Eclipse

15.2.3 Frequently asked questions (FAQ)

- If you don't have a USER_HOME/.subversion directory when trying to configure Subversion, try to follow the first few steps of the assignment: Create the repository, and check it out on your computer. At this point, the .subversion directory should be created. Make sure to modify it before you continue working with the assignment, i.e. before you add anything to your new repository.
- Maven will not run if you have a empty src/test folder, but it can run without the folder.

15.2.4 Assignment description

Step	What	Why	Resource
0	Get an account on the project server (hips.ifi.uio.no). Make sure the course administrators register your names and user account.	Get access to the tools we use	Notes from lecture, your peers and course administrators.
1	Create a repository called “assignments” in your account at hips.ifi.uio.	Learn how to create a Subversion repository. The reason for doing this at hips.ifi.uio.no is that we can access your assignment and correct it.	http://svnbook.red-bean.com/
2	Check out the repository to a local folder on your computer	In order to use the repository.	http://svnbook.red-bean.com/
3	Create a directory called assignment1-<username> and add and commit it to SVN.		
4	Create a Maven 2.0 project descriptor file (pom.xml) by hand! (Not m2 archetype:create…)		http://maven.apache.org/maven2/getting-started.html
5	Make these directories in your project root: src/main/java, src/main/resources	This is a standard way to organize code and resources. Maven uses these directories as it’s default build paths.	
6	Add and commit pom.xml and the directories to the repository	Not necessary, but it’s for the sake of the exercise.	http://svnbook.red-bean.com/
7	Build project files for Eclipse using Maven	To be able to import the project into Eclipse.	http://maven.apache.org/maven2/getting-started.html
8	Import the project into Eclipse		
9	Create a package with the name no.uio.hispindia.assignment1.<username> in src/main/java	This is the standard naming convention for packages, e.g. org.apache.<project name>	
10	Create a java file in this package, with a main method that prints “Hello World!” to stdout (HelloWorld.java)	Get used to Eclipse, and to see that it is possible to make an application out of all this.	
11	Add manifest information to		http://java.sun.com/docs/books/tutorial/j

	pom.xml (see below) so that Java can find the main class.		ar/manifest/ See hint.
12	Build the project with Maven, run the jar file and see that it works		http://maven.apache.org/maven2/getting-started.html
13	Add and commit changes to the repository		http://svnbook.red-bean.com/
14	Notify course administrators when you reach this step to complete the assignment	To pass the first assignment.	

15.3 Hints

15.3.1 Read documentation!

It would be a good idea to read through the lecture slides and the first chapters of the SVN book before getting started. You need an understanding of what SCM and SVN is in order to get this assignment done.

15.3.2 Finished directory structure

This is what the directory structure should look like when you're done coding.

- assignments/
  - assignment1-<username>/
    - pom.xml
    - src/
      - main/
        - resources/
        - java/
          - no
            - uio
              - hispindia
                - assignment1
- <username>
  - HelloWorld.java

15.3.3 Getting Maven to make a manifest file

Information needed in pom.xml so Maven is able to create the Manifest file for the jar archive:

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-jar-plugin</artifactId>
      <configuration>
        <archive>
          <manifest>
```

```
<mainClass>no.uio.hispindia.assignment1.<username>.<className></mainClass>
          </manifest>
        </archive>
      </configuration>
    </plugin>
  </plugins>
</build>
```

15.4 Training Assignment II HISP India Workshop 2007

15.5 Goal

Understand the concept of Inversion of Control (IoC) and be able to apply IoC to an exisiting project using the Spring API.

15.6 Description

In this assignment you will make improvements to assignment 1, so start by making a copy of it.

The output of this program should be exactly the same as in assignment 1. However, this time two beans should do the job. One bean should be responsible for the content of the message and one bean should be responsible for printing the message. The bean printing the message will depend on a specific instance of the Message bean to get the message content (the "Hello World!" String), use dependency injection to achieve this.

Also provide a short text explaining the concept of Inversion of Control, in your own words!

15.7 Requirements

- The program needs to be fully working and configured as described in this document.
- It will suffice that the program runs from Eclipse. In other words there is no need to make a run-able jar file this time. This is more complicated now because the necessary dependencies would also have to be included in the archive.

15.8 Detailed description, hints and guide

- Copy Assignment 1 to assignment2-<username> by using "svn copy" - http://svnbook.red-bean.com/

- Do the necessary changes to the project including updating the dependencies.
- Create eclipse project files.
- Create interfaces named “PrintMessage” and “Message” (package: “no.uio.hispindia.assignment2.<username>.service”). The Message interface should have a method called “getMessage”. The PrintMessage interface should have a method called “print” with no attributes. It will use the Message bean to get the message.
- Implement the interfaces, please use DefaultPrintMessage and DefaultMessage as classnames. This indicates that this is the default implementations of the services.
- Create bean configurations for your beans - http://static.springframework.org/spring/docs/1.2.x/reference/beans.html. Let src/main/resources/META-INF/assignment2/beans.xml be your beans configuration file.
- Update the main class to use the PrintMessage bean instead of just printing to the screen, remember to change the package of this class to “no.uio.hispindia.assignment2.<username>”.
- Verify that it works by running the main class from Eclipse.
- Complete your short text about IoC
- Notify course administrators when you are done, to complete the assignment

16 Appendix E - IDSP and RIMS

16.1.1 The RIMS

The RIMS is an acronym for the Routine Immunization Monitoring Systems (RIMS) (WHO 2007). Below follows a short description of the RIMS software found on the World health organisation website.

> *ROUTINE IMMUNIZATION MONITORING SYSTEM (RIMS) is a computerized implementation, to enter data, generate reports and perform queries. The system is presently developed in Microsoft ACCESS as a standalone CD version. It is user friendly and no special training is required to operate the system. Online system is under development in a different platform using other database and programming tools.*
>
> *The data are collected at district level from PHC's /Reporting Units in the standard pre-designed UIP format and entered on five broad categories namely (A) Immunization & Vitamin A, (B) Vaccine Supply, (C) VPD Surveillance, (D) Status of Cold Chain Equipment and (E) AEFI (Adverse Event following immunization).*
>
> *The system is capable of performing data analyses and generating useful reports for the use of UPI managers at all levels i.e. district, state and national. RIMS will be very useful tool to monitor UIP program as reports from all the 600 districts will be collected in a short period and then analyzed automatically by the software. (WHO, 2007)*

UIP is short for Universal Immunization Programme (Patra, 2006). During the time of writing I discovered a few new sources of information about UIP which would have answered a lot of questions about the background of the formats and names and acronyms of the reports in the RIMS software. The main reason for not finding this earlier was the very limited internet access I had at the time of previous research of the matter. The paper by Patra (2006) is very useful reading for those who wish to gain a thoroughly understanding of the Immunization Programme of India.

Now back to the RIMS software, I tried to run it on my laptop running an Access 2003 installation with a Norwegian Microsoft Windows Xp. This proved to be very difficult due to the very strict use of Date formats in Rims. I have added my user Experience of RIMS at the

bottom of this subchapter. The reminder of this chapter is about the strategy to enable DHIS 2.0 to integrate with the RIMS software.

> *One period in RIMS is one calendar month. You only export from one month at the time. And the exporter only mimic data exported from district level. That is for all RIMS related data for all PHC's in a district for the given month. When an export is done, then the export result is saved in a excel workbook at the location you give in the export view. If there is no values entered the sheet's will contain no data, but all the 6 sheets in the workbook is always made (Hustad, 2007c).*

The Rims software clients can communicate through the use of excel workbooks by using the UIP format. The main idea was to mimic this approach and enable the DHIS 2.0 to make the exact same Excel exports. Below we see the name of the different sheets in the work book.

Table 3 (Hustad, 2007c)

Sheet name	What is contains
AEFI_Detail	
Immunization_Detail	
Vaccine_Supply_Detail	
Surveillance	
ColdChainEquipment_Status	
DIO_in_Position_Detail	

Below we can see the standard information each sheet is loaded with, in addition to the listed elements each sheet contain a grate number of data columns:

Table 4 (Hustad, 2007c)

Name	Value
state_code	Id number of the state
state_name	Name of the state
district_code	Id number of the district
district_name	Name of the district
phc_code	Id number of the PHC

phc_name Name of the PHC

month What month the export belongs to

Year What year the export belongs to

(Hustad, 2007c)

16.1.2

I used the HSSF-POI framework from Apache to generate the workbook and write to an excel file format. The mapping between data elements in the DHIS 2.0 and the RIMS-export format is contained in a single txt file with a very simple mapping format (Hustad, 2007c).

phc_name = ¤tobeadded from dhis¤
mnth = ¤tobeadded from dhis¤
yr = ¤tobeadded from dhis¤
Session Planned = Form6_DE31
Session Held = Form6_DE32
No of sessions where vaccines received = Form6_DE33
No of volunteers engaged = Form6_DE34
Private Vaccinators Hired - ANIM Hired = Form6_DE35
(Hustad, 2007c)

A prototype of the RIMS export plugin can be found in the branch folder of the DHIS 2.0 repository, but the mappig file currently implemented as default has flaws. There was also created a GUI tool for editing the file. The tool is found in the same branch.

There was however a great difficulty concerning the mapping of the elements between the systems, since the DHIS 2.0 at that time did not contain the neccessary data elements. Since many data elements must be created anyway, the most useful approach would probably be to create one or several datasets (one for all the six reports in the UIP format, or perhaps one per report) with the exact same data elements as the RIMS, as an initial step towards integration, and then at some later stage create essential data sets containing only the useful elements from several different reporting schemes. Below we can find my personal experiences with the Rims application.

16.1.2.1 RIMS pilot application User Experience

Test subject: Me

Date: 08.03.2007

Test description: “test all features available for a District Data Entry user”

The application runs on:

Machine Paccar Bell Easy note GN45

Windows XP home edition

16.1.2.1.1 Initialising tests

I ran in to some problems when I was setting up the test environment. The problems, the possible solutions and the outcome are described in the next paragraphs.

16.1.2.1.2 Create new user (National User)

The date format is fixed, but the format set is different from the format used in the validation. Furthermore, it is impossible to use another format and therefore impossible to add users by using the create user function in RIMS.

16.1.2.1.3 Direct manipulation (Trying to work around the latter error)

The database (RIMS.mdb) is protected by a password. I do not have the rims database password, so I am unable to directly manipulate the database.

16.1.2.1.4 Temporary solution

I edited the user rights of IND11 to from “District User”, to “District data entry user”, and saved the changes. I also edited the help manual, where the default users are described.

16.1.2.1.5 Evaluation of usability

In this part of the evaluation, I describe the functions which did not work according to expectations. Most of the “fat-italic” headings (except “in general”) correspond to a main menu option in RIMS. The text represented with “fat” only corresponds to a (sub) menu option.

16.1.2.1.6 In general

The entry validation is too strict. A problem occurs when you by accident skip a field (or enter wrong value), and are unable to go back until you have filed out all the values in the form. Another problem occurs when there is dependency between two values, like session planned and session held in data entry. Here session planned must be higher than session held. If we skip session planned; the session held must be less or equal than session planned, but not blank. So we have to enter another value that’s wrong too. The main problem in all forms

is that you can not move back to an earlier field until you have filled in all the mandatory values. The latter is just not user friendly.

16.1.2.1.7 Create User

When I try to create a new user I have a problem with the entry of date, because the fixed format is invalid and give "invalid format error". Data entry is too strict, there are many obligatory fields, and you can't go back if you fill in wrong information before you have filled inn all the obligatory fields.

16.1.2.2 Data Entry

16.1.2.2.1 Cold Chain equipment

Here the PHC have their names written with capital letters. I don't know if the system is case insensitive or not. One PHC had small letters, but no machines was connected to this PHC name. The entry has the same date format problem, which I had with “create user” form.

16.1.2.2.2 Vaccine Supply and logistics

Always store data in year 2005, instead of the given year (2007)

16.1.2.3 *“Data transport”*

16.1.2.3.1 Import function

I did not understand how you are supposed to use the import function, most of the data elements I can import have no correspondent value in the database afterwards. Most common missing values are; year, month, phc, etc ... (organisation units and time).

16.1.2.4 Export

Does not export the right values. I filled in an immunisation form, with 1 in all fields, for Jan 2007 bannadevi, ALIGARH, UTTAR Pradesh (PHC), however when I tried to export from RIMS to an excel sheet all but the identification values (state code, state name, district code, district name, PHC code, PHC name, mnth, yr) were blank. NB! On export all fields are optional.

I selected all PHC in Uttar Pradesh, ALIGARH, Jan, 2007, Dataformat="excel" The export is very slow. It seemed like the application crashed, but it did not. It started 23:22, and ended 23:25.

Export to txt files can only export one "sheet" or type at the time. Excel export worked okay. It was a bit slow, but was the only export which managed to export all values from all forms today.

Since DHIS already have a similar excel export to IDSP, then this would be my preferred choice for DHIS-RIMS export. The logic in the export is as follows.
The export result is a work book with "1-6" sheets. All sheet start with the same 8 elements: state_code, state_name, district_code, district_name, phc_code, phc_name, mnth, yr.
And then the sheets appear in the order they were selected in the export interface.
As said earlier all data values are optional, if no value is exported then the corresponding excel cell is empty.

16.1.2.5 Reports

Immunization Performance does not work, get error message: "objekt referanse er ikke satt til en objekt forekomst"(this was displayed in Norwegian)
Vaccine Utilisation rate does not work, get error message: "Surveillance Detail not found"
Immunization drop out rate does not work, get error message: "objekt referanse er ikke satt til en objekt forekomst "(this was displayed in Norwegian)
Cold chain status returns no values
Vaccine stock position does not work, get error message: "objekt referanse er ikke satt til en objekt forekomst (This was displayed in Norwegian)"
Vaccine stock position (Yearly): return '0' in all value fields. I think that's strange since I have entered 1 in all value fields.
PHC/Reporting Unit completeness: Unable to select a district value
PHC's not reported: Selected values "UTTAR PRADESH", "ALIGARH", "January", "2007" Give index out of bonds exception
Generate Profile for PHC: "Bannadevi", give: "System.IndexOutOfRangeException" at System.Array.InternalGetValue(Int32 index1, Int32 index2, Int32 Inc

16.1.2.6 Possible solutions

16.1.2.7 Date validation

A possible explanation for the Date validation Problem in RIMS, was that I run a different version of the MS Access database than the one used with the application in other offices. After a little discussion, it turned out that the application is used in the field with Microsoft

Windows XP – same as me – so I cannot think of any other good reason for the problems I am experiencing.

16.1.3 IDSP

As part of my student project in the INF5750 course in the fall of 2006 and later as a part of my master assignment I worked on an export solution towards a system named IDSP. The IDSP worked a little better than the RIMS. And we had a great advantage of being able to access the database, and see that our solution actually worked.

The IDSP export solution was used as origin for the RIMS export solution, many of the tasks were similar, and I tried to re-factor common components in order to reuse it in the new export modul. The solutions is therefore quite similar thus a further description of the IDSP export module is avoided.

At the finalizing stage of the implementation my fellow master student discovered that the IDSP could not import the excel workbook produced by our plugin, even though it seemed to be identical to the workbook produced by IDSP itself. The problem might have been caused by the HSSF-POI library, or simply some missing cell formatting from our part. After spending a few days trying to find the actual cause, my fellow masterstudent implemented a direct export to the Access database going around the entire IDSP export-import system (Hustad, Strandli 2006).

17 Appendix F – Important Emails

17.1 Mail from HISP India Cordinator1

The first e-mail was from a rather frustrated coordinator in India after multiple attempts to get the DHIS2 up and running.

…. Very important …
Subject: DHIS 2 status and its Value in Field
From: HISP India Cordinator1
Date: Sat, 26 May 2007 17:21:10 +0530

Hello,
Sorry to say this we have to stop the implementation of DHIS 2 until is gets better. In Jharkhand we have installed in 32 places where we are getting the data monthly. It has been a battle with DHIS 2 since 36 hrs of 4 people with out sleep and rest. We still couldn't do the data mart nor could we manage to produce any report.

The HISP survived all these years because of its analytical capability and flexibility now we have lost it. HISP now don't have a strong back up from the software. We have collected data from 11 districts for one year the total size of the database is very very hung its 270mb. Since both the Organizational unit data and monthly data are in datavalue it is impossible to do the datamart as it contain text field. In data mart user screen in horrible, we cant even export one district data since there is no option for that either. And the datamart is based on the groups not the dataset and simple datamart never works may be the reason is the combination of text and int data. In data entry screen it is hard for the user to see which orgunit, month, dataset or data elements they are entering the data. As in India and other place we have lot of data elements its hard for the user to keep the track. There are lot of complain from the field that the data what they have entered are lot and also the export and import doesn't work. I personally had problems with export and import. Since the data base size is huge we tried to split the database in to 8 parts and then removed all the zero values and tried the data mart still it didn't work said error operation failed internal server error. Later we tried deleting all the data value and added two district data still the data mart couldn't work. Note: when we do the data mart is just selected 15 dataelements and 150 orgunits and 6 months. One of the state official has meeting on HIMS implementation and training program at national level on Monday, He was leaving from Jharkhand by 6 pm today (25 may 07) even after four people effort in 5 high configuration computer we couldn't manage to deliver the data. As it was too huge and we couldn't complete the data mart.

In DHIS 1.4 software was based on the knowledge and experience of field now in DHIS 2 it is merely technology focus and talk about using different high-tech framework but has lost the focus on field realities.

Sorry to say this but people are very unhappy about the DHIS 2software all the System facilitators are saying that its much better to use excel than dhis2 at least we got generate report and its easier and flexible to create different reports. Since we have started implementation on DHIS 2 right from Kerala, Gujarat, Jharkhand and MP we have facing humiliation from the system facilitators and the health staff as they are cursing us for the

software and making there life miserable. They have come to a stage saying that the paper form were much better than entry the data in DHIS2.

This is the reality and the value of millions NoK Software.

17.1.1 Response from one of the core developer in Oslo

Thank you for that clarification.

Please note, the issues raised in this mail have not been raised before as far as I can tell. If you get feedback from the field, you have to report it to us (i.e. the dev-list) immediately. Then at least we are aware of the problem and can fix it. We're not seeing the actual use and can't then anticipate all problems. That means we have to -keep- -talking-!

Some of the issues, like showing data element, org unit and period in the bottom of the screen, have been raised by instant messaging, and I'm sorry I didn't carry the message onward. But we should have a habit for using the list or the issue tracker. This way it goes into the collective memory.

Please believe me when I say that we want to be helpful, we want to build software you can use, and we want to supply you the functionality you need. There is a technology focus in DHIS, yes, in the sense that we want to build the functionality in a good way. This, however, may lead to a slow process. But it's not like the devs work with the technology for the technology's sake.

So, HISP India Cordinator1, if you could compile a list of more of the feedback you've been getting, we'd appreciate it.

17.1.2 Response from another of the core developers in Oslo

not a long time ago Barath reported that datamart was working fast and fine, what have happened since then? Could it be related to the change from int to string in the aggregation operator field in the database? Can you be more specific about the erros you get and post error messages on the list?

We can make the "export for all units in district" functionality quickly.

17.1.3 Response from coordinator in Oslo

t is very sad to hear about all your problems and frustrations, HISP India Cordinator1.
It unfortunately not realistic to expect a focus on field realities to
come from Oslo. Rather, it is you yourself who have to engage with the
DHIS 2 development process on a regular basis, both from India and
when you are in Oslo.

You refer to a range of problems. Please provide details on each of
them in isolation, with an estimate of the seriousness/urgency of
each, using the tool we have now set up specifically for such feedback
from the field:
http://www.hisp.info/trac/

Then we must try and prioritize and allocate our scarce development
resources accordingly.

17.2 E-mail correspondence with Former core developer1:

The following email is in Norwegian, above the email I have listed the key points from the email exchange.

- Disappointing that the Indian developers still did not use the DHIS2 API or the Frameworks
- OSS values was not highly regarded among the hired developers
- Difficulty to convince the Indian developers of the true value of using “unit testing” and “dependency injection”, since the latter concepts have little value in smaller projects.
- Language barrier, heavy accent a complicated topics may make it difficult to understand each other.
- One major issue for success is to get the developers started on something which they can continue on their own after the workshop.
- A second issue is to get the Indian developers more active in the global development, get them to use the mailing lists more actively.

- A five day workshop is to little time to reach the level of fully trained developers, so don't expect too much results from the effort.

On 4/15/07, Ole Kristian Hustad <okhustad@student.matnat.uio.no> wrote:
Hei *former core developer1*,
Jeg er på feltstudie i India i forbindelse med masteren min i
informatikk, og har blant annet fått ansvaret for opplæring av
utviklere(*the Hisp India national developer* blant andre, tror du kjenner han). Jeg har lest
masteroppgaven din og la merke til at du hadde vært her og gjort noe av
det samme for en liten stund siden. Jeg lurte på om du hadde noen
nyttige innspill å komme med for en som "følger i dine fotspor" ?.

Hei,

Kan ta et kjapt svar nå i hvert fall, så kan du jo bare sende e-post om det er noe du lurer på eller om det er noe du ønsker at jeg kommenterer.

Det aller aller aller viktigste er å få de integrert i utviklingen av DHIS2 på lik linje med utviklerne i Norge. Hvis de fortsetter å utvikle egne JSP løsninger og jdbc-databaseintegrasjonsløsninger, så bare skyter vi oss i foten. Vi ser det allerede nå - vi må være forsiktige med å ikke brekke (gjøre APIet ukompatibelt) "hacks" de har laget i India og Vietnam, dette kan (om ikke det allerede er slik) nærmest stagnere hele DHIS2 utviklingen. Desto mer sånne ting de lager, jo mer problemer kommer vi til å få. Vi er (slik jeg ser det) nødt til å få på plass stabile APIer som de kan forholde seg til, som det er nødt til å bruke - hvis de ønsker at vi skal ta hensyn til løsningene deres. Kjernen må løsrives fra all "ekstern" (alt som ikke er utviklet av de samme folkene som lager kjernen og med potensielt annen teknologi) utvikling.

Jeg prøver å sette av tid om dagen til å involvere med i utviklingen igjen, så ikke nøl med å sende spørsmål eller hva som helst.

--
Mvh,
former core developer1
On 4/15/07, Ole Kristian Hustad <okhustad@student.matnat.uio.no> wrote:
takker, jeg har ca 15 dager på gruble ut ett 5 dagers opplegg for en
workshop, jeg kommer til å ha ca 10 utviklere, der og av den grunn så
hadde jeg tenkt å bruke en del kurs materiell fra 5750 for innføring i
tools og rammeverk, og prøve å komme i gang med re-implemetering av ett
par av de indiske modulene slik at de følger DHIS "standarden".

Hvem er disse 10 utviklerne?

Men hvor
langt vi kommer gjenstår og se. Jeg har noen små økter med barrath nå,
går igjennom subversjon, maven, Junit, WW, spring og for forhåpentligvis
startet på re implementeringen. Etter workshopen kommer jeg til å ha
ytterligere 2 og en halv uke med *the Hisp India national developer* og annen utvikler. Målet er

vel
at de kan bli kjerneutviklere en gang i fremtiden.

Pass på å jobb sammen med de. Tett oppfølging er helt essensielt her. Jeg ville forsøkt å sørge for at de har et godt utgangspunkt når du drar, som de skal jobbe videre på - ikke som er et eksempel. Hadde det kanskje vært en ide å splittet deres oppgaver med norske utviklere? Slik at en av de + en norsk jobber på samme modul f.eks. Vi har jo sett gang på gang nå at det ikke kommer noe særlig ut av rene indiske eller vietnamesiske moduler - i hvert fall mtp generell DHIS2 arkitektur/teknologier. At ikke en eller flere av disse er kjerneutviklere nå er veldig synd, men det er egentlig ikke så rart - siden de sitter og utvikler sine helt egne ting, som ikke er i nærheten av hva som blir utviklet generelt i DHIS2; e. g. disse JSP rapportmodulene...

Litt bekymringer ....
Jeg forventer at *the Hisp India national developer* og the *HISP India developer from Bhopal* kommer til å være rimelig oppdatert på
rammeverk og liknende, under workshopen, men jeg tror ikke noen av de
andre kjenner noen av rammeverkene vi kommer til å bruke.

The HISP India developer from Bhopal kjenner jeg ikke til. Har *the Hisp India national developer* egentlig jobbet noe med rammeverkene vi bruker i DHIS2 i senere tid? Jeg kjørte en del innføring for de, men de kom aldri ordentlig i gang med noe utvikling (med de rammeverkene).

Jeg har
hovedsakelig to mål jeg skal prøve å nå; for det første skal jeg prøve å
lære dem rammeverkene, for det andre vil gjerne koordinatorene i Oslo at
vi skal jobbe direkte med de indiske modulene og fokusere på praktisk
problemløsning. Jeg regner med at du har gjort noe lignende og lurer
litt på hvor mye tror du vi kan greie å oppnå på fem dager, og om dette
vil gi en tilstrekkelig god innføring i verktøy og rammeverk eller om det
kan bli litt for mye på en gang?

5 dager er veldig lite. Ikke ha noen forventninger om at dette skal føre til noe videre utvikling fra deres side. I hvert fall hvis nivået på utviklerne er det samme som de vi har jobbet med tidligere. Det er dyktige folk, men de ligger noen år bak når det gjelder verktøy og rammeverk. Har du god tid til å forberede dette? Praktisk jobbing er det beste, du vil nok bare overkjøre de fullstendig hvis du kjører på med slides/forelesninger. Dette er noe jeg tenker på nå, så er ikke akkurat godt gjennomtenkt - men jeg ville prøvd å få de i gang med noe helt konkret. Del de opp i grupper og få de til å starte å lage noe DHIS2 spesifikt så fort som mulig. To dager innføring, så tre dager med utvikling av noe opplegg for DHIS2. Dette kan være en modul de skal jobbe videre med og hvor de blir støttet av en norsk mentor(utvikler).

Det vanskeligste er konseptene - skrive enhetstester, dependency injection, ... - ting de har 0 forhold til og som faktisk er veldig vanskelig å forklare for noen som ikke har erfaring fra arbeidslivet / reel erfaring fra større prosjekter. En del av disse konseptene/teknikkene/rammeverkene gir tilsynelatende liten verdi i mindre prosjekter eller for utviklere som aldri har brukt tilsvarende - derfor blir det ekstra vanskelig å få de til å forstå hvorfor vi faktisk bruker det.

Erfaringsmessig gir det 0 verdi å snakke om Open Source for de innleide utviklerne - dette

hadde de jeg jobbet med 0 forhold til og liten eller ingen interesse av. Med dette mener jeg for dem som motivasjonsfaktor - men her gjelder det nok bare å møte / få tak i de "riktige" personene.

Hvis de ikke er i gang med noe helt konkret innen du drar, som de kan jobbe videre med - så vil jeg tippe det ikke kommer til å skje så fryktelig mye...

Spørsmål av praktisk av art ...
Mange av spørsmålene mine er mer av en praktisk type, spesielt kulturelle spørsmål. Jeg forventer egentlig at ca halvparten ikke kommer til å forstå hva alt jeg sier p.g.a. aksenten min, og det jeg lurer på i den forbindelse er om de bare kommer til å nikke og smile, eller faktisk si i fra at de har problemer med å forstå hva jeg sier.

De kommer bare til å nikke og smile.

Og hvis jeg eksplisitt nevner at jeg ønsker at de skal si i fra hvis de ikke forstår hva jeg snakker om, kommer de da til å si ifra??

Nei :-)

Et godt triks er å be de som forstod deg rekke opp hånden (i motsetning til å spørre om noen ikke forstod). Kanskje la de selv prøve å forklare konseptet - for å få i gang en dialog.

Jeg er spesielt interessert i ting som er India og trenings spesifikke, ting som har høy sannsynlighet for å komme opp, men kan være litt vanskelig å forutse. Er det noen brannfakler jeg bør være klar over?

Språk er problematisk, men i India er det ikke så ille - folk kan som regel engelsk godt. Jeg har møtt mange indere som er langt bedre enn meg i Engelsk. Det er ikke alltid like lett å forstå de og de har det nok ofte like vanskelig med å forstå deg. Har hatt en del merkelige samtaler - høydaren var kanskje under en presentasjon i Saigon, hvor jeg trodde en av de som hørte på spurte om jeg ville være vennen hans. Det han ville si var at han hadde hatt fransk på skolen... Ble litt klein stemning... Trikset er nok å snakke rolig..

Jeg syntes det vanskeligste var å få de til å prøve å delta i DHIS2 utviklingsmiljøet. Utrolig vanskelig å f.eks. få de til å sende e-post til listene eller commite kode til SVN repositoriet. Her bør du kanskje jobbe mye. Slik jeg ser det så er noe av de viktigste å sørge for at det ikke blir isolerte utviklingsnoder. Enten bør vi ha folk plassert der til vi ser at det lokale teamet er 100% integrert, eller så bør noen integrere seg med de i utviklingsprosess - for å tvinge de til å jobbe mot det internasjonale miljøet.

Kom ikke på så fryktelig mye fornuftig å skrive her nå, skal tenke mer på det. Fyr løs hvis det er noe.

<-- cut -->

--
Mvh,
former core developer1

On 4/15/07, Ole Kristian Hustad <okhustad@student.matnat.uio.no> wrote:
takker, jeg har ca 15 dager på gruble ut ett 5 dagers opplegg for en
workshop, jeg kommer til å ha ca 10 utviklere, der og av den grunn så
hadde jeg tenkt å bruke en del kurs materiell fra 5750 for innføring i
tools og rammeverk, og prøve å komme i gang med re-implemetering av ett
par av de indiske modulene slik at de følger DHIS "standarden".

Hvem er disse 10 utviklerne?

Men hvor
langt vi kommer gjenstår og se. Jeg har noen små økter med barrath nå,
går igjennom subversjon, maven, Junit, WW, spring og for forhåpentligvis
startet på re implementeringen. Etter workshopen kommer jeg til å ha
ytterligere 2 og en halv uke med *the Hisp India national developer* og annen utvikler. Målet er vel
at de kan bli kjerneutviklere en gang i fremtiden.

Pass på å jobb sammen med de. Tett oppfølging er helt essensielt her. Jeg ville forsøkt å sørge for at de har et godt utgangspunkt når du drar, som de skal jobbe videre på - ikke som er et eksempel. Hadde det kanskje vært en ide å splittet deres oppgaver med norske utviklere? Slik at en av de + en norsk jobber på samme modul f.eks. Vi har jo sett gang på gang nå at det ikke kommer noe særlig ut av rene indiske eller vietnamesiske moduler - i hvert fall mtp generell DHIS2 arkitektur/teknologier. At ikke en eller flere av disse er kjerneutviklere nå er veldig synd, men det er egentlig ikke så rart - siden de sitter og utvikler sine helt egne ting, som ikke er i nærheten av hva som blir utviklet generelt i DHIS2; e. g. disse JSP rapportmodulene...

Litt bekymringer ....
Jeg forventer at *the Hisp India national developer* og *the HISP India developer from Bhopal* kommer til å være rimelig oppdatert på
rammeverk og liknende, under workshopen, men jeg tror ikke noen av de
andre kjenner noen av rammeverkene vi kommer til å bruke.

The HISP India developer from Bhopal kjenner jeg ikke til. Har *the Hisp India national developer* egentlig jobbet noe med rammeverkene vi bruker i DHIS2 i senere tid? Jeg kjørte en del innføring for de, men de kom aldri ordentlig i gang med noe utvikling (med de rammeverkene).

Jeg har
hovedsakelig to mål jeg skal prøve å nå; for det første skal jeg prøve å
lære dem rammeverkene, for det andre vil gjerne koordinatorene i Oslo at
vi skal jobbe direkte med de indiske modulene og fokusere på praktisk
problemløsning. Jeg regner med at du har gjort noe lignende og lurer
litt på hvor mye tror du vi kan greie å oppnå på fem dager, og om dette
vil gi en tilstrekkelig god innføring i verktøy og rammeverk eller om det
kan bli litt for mye på en gang?

5 dager er veldig lite. Ikke ha noen forventninger om at dette skal føre til noe videre utvikling fra deres side. I hvert fall hvis nivået på utviklerne er det samme som de vi har jobbet med tidligere. Det er dyktige folk, men de ligger noen år bak når det gjelder verktøy og

rammeverk. Har du god tid til å forberede dette? Praktisk jobbing er det beste, du vil nok bare overkjøre de fullstendig hvis du kjører på med slides/forelesninger. Dette er noe jeg tenker på nå, så er ikke akkurat godt gjennomtenkt - men jeg ville prøvd å få de i gang med noe helt konkret. Del de opp i grupper og få de til å starte å lage noe DHIS2 spesifikt så fort som mulig. To dager innføring, så tre dager med utvikling av noe opplegg for DHIS2. Dette kan være en modul de skal jobbe videre med og hvor de blir støttet av en norsk mentor(utvikler).

Det vanskeligste er konseptene - skrive enhetstester, dependency injection, ... - ting de har 0 forhold til og som faktisk er veldig vanskelig å forklare for noen som ikke har erfaring fra arbeidslivet / reel erfaring fra større prosjekter. en del av disse konseptene/teknikkene/rammeverkene gir tilsynelatende liten verdi i mindre prosjekter eller for utviklere som aldri har brukt tilsvarende - derfor blir det ekstra vanskelig å få de til å forstå hvorfor vi faktisk bruker det.

Erfaringsmessig gir det 0 verdi å snakke om Open Source for de innleide utviklerne - dette hadde de jeg jobbet med 0 forhold til og liten eller ingen interesse av. Med dette mener jeg for dem som motivasjonsfaktor - men her gjelder det nok bare å møte / få tak i de "riktige" personene.

Hvis de ikke er i gang med noe helt konkret innen du drar, som de kan jobbe videre med - så vil jeg tippe det ikke kommer til å skje så fryktelig mye...

Spørsmål av praktisk av art ...
Mange av spørsmålene mine er mer av en praktisk type, spesielt kulturelle spørsmål. Jeg forventer egentlig at ca halvparten ikke kommer til å forstå hva alt jeg sier p. g. a. aksenten min, og det jeg lurer på i den forbindelse er om de bare kommer til å nikke og smile, eller faktisk si i fra at de har problemer med å forstå hva jeg sier.

De kommer bare til å nikke og smile.

Og hvis jeg eksplisitt nevner at jeg ønsker at de skal si i fra hvis de ikke forstår hva jeg snakker om, kommer de da til å si ifra??

Nei :-)

Et godt triks er å be de som forstod deg rekke opp hånden (i motsetning til å spørre om noen ikke forstod). Kanskje la de selv prøve å forklare konseptet - for å få i gang en dialog.

Jeg er spesielt interessert i ting som er India og trenings spesifikke, ting som har høy sannsynlighet for å komme opp, men kan være litt vanskelig å forutse. Er det noen brannfakler jeg bør være klar over?

Språk er problematisk, men i India er det ikke så ille - folk kan som regel engelsk godt. Jeg har møtt mange indere som er langt bedre enn meg i Engelsk. Det er ikke alltid like lett å forstå de og de har det nok ofte like vanskelig med å forstå deg. Har hatt en del merkelige samtaler - høydaren var kanskje under en presentasjon i Saigon, hvor jeg trodde en av de som hørte på spurte om jeg ville være vennen hans. Det han ville si var at han hadde hatt fransk på skolen... Ble litt klein stemning... Trikset er nok å snakke rolig..

Jeg syntes det vanskeligste var å få de til å prøve å delta i DHIS2 utviklingsmiljøet. Utrolig vanskelig å f.eks. få de til å sende e-post til listene eller commite kode til svn repositoriet. Her bør du kanskje jobbe mye. Slik jeg ser det så er noe av de viktigste å sørge for at det ikke blir isolerte utviklingsnoder. Enten bør vi ha folk plassert der til vi ser at det lokale teamet er 100% integrert, eller så bør noen integrere seg med de i utviklingsprosess - for å tvinge de til å jobbe mot det internasjonale miljøet.

Kom ikke på så fryktelig mye fornuftig å skrive her nå, skal tenke mer på det. Fyr løs hvis det er noe.

<-- cut -->

--
Mvh,
former core developer1

17.3 Capacity building effort in India 2004

```
> Dato: Fri, 20 Aug 2004 11:01:03 +0200 (MEST)
> Fra: "Former Masterstudent3"
> Til: Cordinater1 from Oslo
> Emne: Re: Fwd: Re: Java in India
>
> Hi Cordinater1 from Oslo,
>
> Web application or not, I will run the test using hibernate here as
> we
> had agreed in Oslo. Already yesterday, we wrote a considerable amount
> of
> code, using tomcat and hibernate + Struct, then built it using ant.
> The
> code is
> suposed to connect to mysql database, show the data in the table
> orgUnit
> and then provide a module for entering more orgUnits! That's all. The
> building part failed because we didn't have j2ee,  only j2sdk --I am
> downloading it now and the test and debugging will be done today.
>
> A small swing prototype would be the best way to get something done
> quickly. But a web interface should come in handy. However, due to
> lack
> of even the most basic amount of experience in java web technologies,
> we
> will end up with a very immature code that will have to be rewriten
> if to
> be used anywhere in the real world. So, lets go for a very well
> designed
> simple and small swing deskup. The report module can be web enable
> using a
> simple applet. BUT, the web version should continue to be designed...
>
> About the books, I'll get some copies!
>
>
> Best regards
> --
```

```
> Former Masterstudent3
>
```

17.4 Email from Cordinater1 from Oslo (09.05.2007)

Translation of part of email correspondence with *Cordinater1 from Oslo*.

The author: My question is related to knowledge management. We lose many competent developers from the DHIS2 project, because they are completing their master assignment this Autumn. Many of them have been in the project from its beginning and is the core of the DHIS2 developers at the moment. Who is left on the core developer team this autumn?

Cordinater1 from Oslo: As I just mentioned Core developer1 and Core developer2 is staying in the project (they will be hired).

Below I have added the extract in the original language.
….
>Spørsmålet mitt dreier seg mye om kunnskapsstyring, vi "mister" mange
>flinke folk som har vært med å bygge opp systemet fra bunnen av, disse
>studentene utgjør vel kjernen av utviklere på DHIS 2.0 for øyeblikket?
>Hvem blir igjen i utviklerteamet fra høsten?

Som jeg nettopp skrev, ser det ut til at Core developer1 og Core developer2 er med videre (de blir ansatt på prosjektet).

….

18 Appendix G – User Training

During the field work in India, I participated in some user training sessions, below you find an accurate account of the sessions as well as the aftermath, including an review of the events and a meeting with the Joint Director of Gujarat

18.1.1 Training sessions

During our stay in Gujarat we attended two training sessions with the Gujarat team. During these sessions we experienced for the first time what the users thought of HISP-India and the system we had developed.

The first session was in a medical centre in Ahmedabad, where the centre had pretty high technical standard. They even had their own computer lab. The session went on and there was much focus on data quality and I quote:

Garbage in → Garbage out

The latter pretty much sum up the motivation for doing the training as well as the motivation for entering data properly. You can have the most advanced and sophisticated analysis system

in the world, but if you put garbage into it the output will be garbage as well. Later we went to the computer lab and the Gujarat team helped the doctors with starting up the DHIS 2.0 application. The doctors entered min and max values to limit the error range of the data[51]. They also entered data within and without the error range, in order to learn how to handle exceptions. When the basic training was completed, the doctors started working on real data from their districts and generated reports[52].

The next day *the dhis2-user-trainer* held an open session and asked the doctors for their opinions in the matter validation rules. Some of the data elements are dependent on other data elements. The validation rules role is to discover the corrupted and/or missing data. One of the exercises was the use of an Eyeballing technique in order to identify the misleading data. During the exercises many of the doctors discovered dependencies in the forms, and were able to identify some validity rules.

We did another training session in a smaller town a couple of days later. Surendra Nagar did not have its own computer lab, so HISP-India had to rent computers and bring them to the Healthcare Centre. What was remarkable different between this session and the first one was that only 4 of the attendances had any prior experience with computers. In both cases mostly the head officers participated in the discussions, but more so in Surendra Nagar than Ahmedabad (okhustad-reference- 26.02.2007 – Surendra Nagar, Gujarat).

The Administrative hierarchy in Indian Health

1. National
2. State
3. District
4. Block
5. PHC
6. Sub centre

[51] The application has data filters to catch obviously wrong valued data, by decreasing the number of outlier's you drastically improve the quality of your data. Since just a few outliers may have significant impact on you statistics.

[52] The Indian health system is very hierarchical. On each level there is dozens of different reports which is supposed to be filled in by the health workers. DHIS is trying to make this job easier by generating as many reports as possible from a minimum number of different data elements, thus save a lot of plotting time.

Meeting with the Additional Joint Director of Health services

After all the training sessions in Gujarat were completed, *the dhis2-user-trainer* had a meeting with the additional joint director of Health Services in Gujarat. During this informal meeting *the dhis2-user-trainer* gave a quick summary of the training sessions. I have added some written account from the meeting

> *We discussed the concept of data dictionary and she was a little sceptical. She said 'Who will look at it? It will only be a text book'. I suggested that it could be more like a reference manual. One other thing which emerged from this training program was that it is very difficult to arrive at a consensus for the description of data elements. One argument at Ahmedabad went on for 20 minutes (about postnatal care) and finally I had to cut it short. Dr.Vikas says she is in the process of preparing a manual on the 'to do' things for all the programs and she will share it with us as soon as It is ready and we can base our data dictionary on that. (Srinath, 2007)*

There might be many reasons for the confusion. One might be the large number of data elements in all the different forms, and that it is difficult to remember the correct definition of each data element. Another reason might be different ideas about how the data is used and for what purpose. One manager might be more interested in good statistics than the actual quality of the service. The latter might be the primary concern of a doctor or a different manager. The different interests and stakeholders in the healthcare domain derive different practices for a rather simple data element.

> *One of the recurring problems on the training days was the in consensus on the definition of the different data elements. Dr Desai mentioned that she was currently working on a dictionary, but it was not prioritized since the doctor should have learned this at medical school. This was brought up because the intentions of several of the data elements are not clear even to the health director. One of the problems is Antenatal/Postnatal care, and the dhis2-user-trainer tried to clarify the intention of these elements. Each of them involve up to three visits, and each visit should involve certain activities and should be within a certain time interval. The question is if you should register the visits if it did not include all the prescribed activities, or if it was outside the correct time span. There are two sides here; one is about health care quality and the other is about coverage. (Hustad, 2007a)*

One of the problems was to get hold of the tacit knowledge kept by the health workers in Gujarat. Below we find a quote from a written account of the meeting.

> *Finally the discussion turned to the problem concerning the doctors, who have expert knowledge about their fields, are afraid to speak up. The local doctors possess domain knowledge, but don't challenge their superior officers even if they know that their superiors are wrong. Because that is not the way things work in Gujarat. The director admitted that this is a problem also in the higher levels in the Health administration. (Hustad, 2007a)*

The latter describe much about Gujarat as a state. There are great differences between the states of India. We got the impression on the two training days that there is much difference from district to district as well even within the administration.

19 Appendix H - Interview with *Cordinater1 from Oslo*

Date: 23.01.2008.

Time: 16:30

Place: Forsknings Parken

Topic: Technology review

Subject: *Cordinater1 from Oslo*

Role in HISP: Coordinator

The interview was unstructured, even though the questions were written beforehand only a few of them were asked explicitly. The rest was answered by the narratives own momentum. The notes were partially written during the interview and then braided together in a more presentable fashion afterwards.

What was the choice of technology for DHIS 2.0?

Java SE
Spring,
Hibernate,
Maven,
Webwork,
JUnit

When was the choices for the implementation technologies of DHIS 2.0 made?

The technology review started already in March 2004, and the complete list was ready in November 2004.

Who attended the technology reviews?

First draft members:
Therese Steensen – quit
Hanne Vibekk – quite
Erling Svanberg Mytting -
Trond Andresen -
Cordinater2 from Oslo -

Additional contributers:
former core developer1

What were the significant reasons for choosing the technologies?

The main idea was to choose open source software frameworks, which were growing in popularity and showed great promise for the future.

We were suppose to build a desktop app, but web was more important since there already existed a desktop application; the DHIS 1.4.
Main goal: OSS, Good plug-in structure, Distributed development,
It must be possible to develop small plug-ins and other modules

Had little knowledge plug-ins, and were looking for plug-ins when they stumbled over Eclipse. Eclipse has an extensive plug-in architecture. But it was made for desktop apps.

Was there any reasons for choosing other technologies?

The frameworks discussion was also quite broad

Many different programming languages and frameworks were pulled forth as possible candidates. Both PHP and Python were suggested, but the Indians In HISP had not heard about PHP or Python, so the only real option was Java.

Tapestry was just released in version 4, the framework started to grow mature. But there existed little documentation of the framework. A new book about Tapestry was released as well.

Tapestry – had a good online Tutorial, 3 av 8 examples were done at the time. The written examples were good, but the rest were not published when they were supposed too.
Tapestry is components based rather than web based. The book Tapestry in Action is difficult to use, since it is focused on the Tapestry design idea, and not how to actually use it.

JSF – was on the way, we had heard that it was oriented at developing large enterprise systems for big companies with tons of cash.

Struts 1.0 – Was a sure thing, but it was old, and presumably over the peak.

Java was a good platform based on the education students have from universities and colleges in Norway.

.Net, is not Open Source (which is a requirement) and cost a lot of money, therefore not an option.

Ruby on Rails was not released at the time.

PHP – 4 was still an function oriented language

Wicket – Web framework

OpenLaszlo (become open source) nice, but spring supported more functions.
Flex (was not OSS)

Spring emerged as an option, since several key contributors had prior experience, and advocated the framework. (Spring in action, Spring live by Matt Raible, the latter author had

great impact on *Cordinater2 from Oslo* and *Cordinater1 from Oslo* through his blog and sourcebeat.com which published Spring live chapter for chapter as it was written.

But the same people had experience with WebWork, and advocated in combination for spring and WebWork.

What was the overall out roll strategy of the new implementation?

What was the long term development strategy?

The long term development was suppose to based on the India and Vietnam Nodes, through extensive training using master students from Norway.

Was local context considered for choice of long term development strategy?

Well actually no, we did not expect it to be difficult to recruit and train developers in India

What was the expected enrolment strategy for developers?
We have used the networks of the HISP members to recruit new developers, and forge alliances through opportunistic choices as with *former core developer1*'s case in Vietnam. The entire recruitment strategy has in many cases proved unsuccessful. But there is difficult to find highly competent people, with experience in popular technologies which are willing to work for a low salary. They are usually busy with something else.

What was the expected capacity building cost?
Remark by author; The question was not asked nor answered specifically, but in the next question and answer enough information is given to presume that it was expected to be less trouble some.

When did the initial capacity building in India start, and with whom?

In summer of 2004 when I was in India, then I was supposed to study the tools myself and give a brief training in Java for a group of developers. There was a group of 4-5 people.

HISP India Cordinator1, HISP India Cordniator2, *HISP India developer1 from Jarkhand*, the Former HISP India developer2, the Former HISP India developer3, the Former HISP India developer4

HISP India Cordniator2 and HISP India Cordinator1 Did not have time to learn, since they were busy with other things.
The others did not understand enough of java.

The initial capacity building was no big success. The persons involved had little knowledge of java and even less of object oriented programming. It appeared that there was a long time since any of them had done any programming, and then it is likely that they did not use any Object Orientation.

The initial training in java was followed by one on hibernate, but the effort seemed to be wasted since they had problems grasping the object orientation theory.

April interview candidates, and hired 3 people. Best guy left before *Cordinater1 from Oslo* and *the former core developer1* arrived in India. The former HISP India developer5 and *the HISP India developer from Kerela*, had some experience with java.
May 205, reviewed the developer training in India,

And 2 students from Indian Institute of Technology (IIT)(from north India). They had no prior experience with Java or web development.

We presumed that they knew java, and had a 5 days crash course in; hibernate, spring WebWork, JUnit, Subversion, Maven. We used some exercises. We focused on the data sources. Exercises was to tough and to few, there was a problem that none of the visual components of the system was ready, so it was not possible to see what they were working on.

By august and September, they had to start showing some of what they were working on, so an Indian working in Oslo made some "visual display" in PERL. This proved to be enough, and later HISP India Cordinator1 and developers they hired were supposed to remake this using DHIS2 technology.

The former HISP India developer1 and the HISP India national developer both come from HISP India Cordniator2's home Area. They wrote an application in pure JSP and copied the GUI from DHIS1.4. It was close to identical.

January 2006, the M1 of DHIS2 was released. Did not have a report module, but could use the reports Bahrat had designed.

Vietnam
There was also a more success full capacity building effort going on in Vietnam. Kristian worked with a company (Thans company – "outer soft"), and was suppose to train to 3 students + some other employee.

There was a huge communication problem. We had big problems understanding each other. Than was working on translating the DHIS1.3 to Vietnamese.

One of the students in cooperation with the "Outersoft" employee produced a patient model, but it was never incorporated into dhis2.

2 students never produced anything interesting.

We did not get much out of the arrangement but the company did, they hired all the students. This was not a great loss, for us, though one of the students showed some promise.

Through his work in Vietnam *the former core developer1* met the Dean from Nonlam university, and had the opportunity to work with some of the students at the Nonlam university. *Cordinater1 from Oslo* visited them in June, and made big plans for the fall 2005.

The professor managed to gather some good students and gave them training in Java and Hibernate before a course in Open Source technologies was held by HISP.

HISP hired 3 of the students in 2006.

HISP sent master students from Oslo, first Core developer1, Core developer1, then *Former HISP master student1* continued the capacity building effort. No one were there in the summer. But former master student2, former master student3 and former master student4 were sent there by the fall. However the Vietnamese needed much guidance, they worked for a year, but produced little compared to the resources.