# Designing architectural patterns for distributed flexibility in health information systems

By

Lars Kristian Roland

A thesis submitted in partial fulfillment of the requirements for the degree of

Philosophiae Doctor (Ph.D.)

Faculty of Mathematics and Natural Sciences,

University of Oslo, Norway

January 2018

*Series of dissertations submitted to the*
*Faculty of Mathematics and Natural Sciences, University of Oslo*
*No. 1959*

ISSN 1501-7710



Cover: Hanne Baadsgaard Utigard.
Print production: Reprosentralen, University of Oslo.

# Table of contents

## Acknowledgements

This research project has been a long journey with many detours. I would not have been able to complete this Ph.D. without the persistent help from my supervisors Kristin Braa, Margunn Aanestad, Sundeep Sahay and Terje Sanner. They helped me find the way, and took me back on course when I was astray. Eric Monteiro also provided me with substantial help during a difficult period of the Ph.D, and the work we did together on "P for Platform" became a critical part of my thesis. Richard Ling was also my supervisor a while, and I regret not being able to finish the article about messaging on which he was of great help.

I would also like to thank a long list of other fellow students, faculty and project members who helped me by reading papers and giving input: Johan Sæbø, Knut Staring, Ola Titlestad, Lars Helge Øverland, Saptarshi Purkayasth, Anne Thorseng, Bob Jolliffe, Jason Pickering, Bendik Bygstad, Ole Hanseth, Xenia Vassilakopoulou, Arunima Mukherjee, Long Ngo Thanh, Jo Størset and my master students.

I have also received so much help and learnt from people in the field. There are so many names that could be mentioned, but I would like to thank especially Prosper Behumbiize, Immaculate Ayebazibwe, Zikulah Namukwaya, Fitti Weisglass, Jaco Homsy and Mary Glenn Fowler. You opened my eyes and taught me so much about the health service in Uganda. Your exceptional competence and spirit is an inspiration to everyone around you. I wish I had done more to help you back.

Most importantly I would like to thank my family. All those discussions about health during my childhood and later, with my parents, sister, brothers, and my wife, finally made me enter the health sector both in academics and professionally.

To Marie Cecilie, Fredrik, Kristian and Magnus; you are my constant and most important inspiration. Thank you!

## Abstract

The information systems used by the health sector have seen a tremendous evolution in recent years, and health professionals are increasingly becoming dependent on these to treat patients and complete their other daily tasks. However, these systems have failed to keep up with the requirements of health professionals, management and patients. The increased proliferation of such systems has led to the convergence of many different user tasks into the same solution, the reuse of software across multiple sites and a parallel need for increased rates of innovation and change. These three drivers are difficult to combine, and this thesis provides architectural insights into how platforms can be designed, governed and used in order to address these issues of heterogeneity in the health sector. While heterogeneity in healthcare is nothing new, the degree of convergence and consolidation of systems, in contrast with the increasing requirements for innovation and flexibility, has not been sufficiently covered in the literature.

This thesis asks: "*How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"* and uses material from both developing countries and a developed nation to answer this question. It follows the health management information system DHIS2 through a platformization process, from a single application addressing very specific needs to a platform that is open to external innovation, outside of control of the core software developers. Additionally, this thesis considers three cases of application integration with electronic health registers (EHR) in Norway, where the EHR takes on the role of a health information platform.

This work contributes to the literature concerning health information systems, information infrastructure and platforms, by means of architectural insights and a discussion of how flexibility is delegated and constrained between actors in a platform value chain. One of the included papers also makes a contribution to the participatory design literature by introducing a participation typology and discussing how platform architectures relate to participation in large-scale systems.

This research makes practical contributions in terms of several typologies, architectural insights and a discussion of design principles and tradeoffs that will be helpful for practitioners who intend to implement health information systems.

# Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.) at the Faculty of Mathematics and Natural Sciences, University of Oslo, Norway. The University of Oslo funded this work. This dissertation consists of four papers as well as an introductory section.

The papers, listed below, are included in appendices. The papers were published in a different order than listed below.

- Paper 1: Lars Kristian Roland, Terje Sanner, Prosper Behumbiize, Zikula Namukwaya, and Kristin Braa. “Accommodating Multiple Rationalities in Patient Oriented Health Information System Design.”, *Selected Papers of the Information Systems Research Seminar in Scandinavia:* Nr. 4 (2013)

- Paper 2: Roland, Lars Kristian; Sanner, Terje Aksel; Sæbø, Johan Ivar; and Monteiro, Eric (2017) "P for Platform. Architectures of large-scale participatory design," *Scandinavian Journal of Information Systems*: Vol. 29 : Iss. 2 , Article 1. Available at: http://aisel.aisnet.org/sjis/vol29/iss2/1

- Paper 3: Terje Aksel Sanner, Lars Kristian Roland, and Kristin Braa. “From Pilot to Scale: Towards an mHealth Typology for Low-Resource Contexts.” *Health Policy and Technology* 1, no. 3 (September 2012): 155–64. doi:10.1016/j.hlpt.2012.07.009.

- Paper 4: Roland, L.K., Sanner, T.A. &Aanestad, M..: Flexibility in EHR ecosystems: five integration strategies and their trade-offs (2017). Paper presented at *NOKOBIT 2017, Oslo, 27-29 Nov. NOKOBIT*, vol. 25, no. 1, Bibsys Open Journal Systems, ISSN 1894-7719.

# 1 Introduction

This thesis studies the design and development of regional and national health information systems, with a particular focus on how flexibility can be distributed among actors to enable health information platforms that support different contexts, countries and user groups. The study discusses the emergence of platform architectures that have become influential innovation enablers in other industries (Tiwana 2013), but, despite some early efforts, have yet to enter the health stage with full force (Furstenau and Auschra 2016; Mandl and Kohane 2012; Sellberg and Eltes 2017).

Platform architectures have become commonplace for computers (Gawer and Cusumano 2002; Baldwin and Woodard 2009; Cusumano 2010), mobile phones (Ghazawneh and Henfridsson 2013), enterprise systems (Ceccagnoli et al. 2011), gaming (Cennamo, Ozalp, and Kretschmer 2016), social networking, and communication (Avital and Te'eni 2009). In these industries, the platforms create value chains and enable third-party innovation through external applications that run on the platform. The platforms become enablers for ecosystems of companies that cooperate to offer new functionality that is beyond the scope and capabilities of the core platform and its owner (Tiwana 2013). Through these platform architectures, multiple actors can cooperate to handle a greater number of users and use cases.

Since the platform trend is a common method of addressing the need for heterogeneity and distributed innovation in other industries, it is therefore natural to study how platform architectures can address such issues in health systems.

This thesis therefore asks: *"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"*

### 1.1.1 Concern 1: Software must fit the needs of the users

For a technology to be successful, it must be a good fit for the tasks it is supposed to support (Goodhue and Thompson 1995), and significant differences between the actual needs of users and the design of the software can lead to catastrophic failures (Heeks 2006). Such failures may occur when solutions are taken from the designer's own background and applied to different settings without sufficient adaptation. Designers are perhaps not even in a suitable position to understand these actual needs, because users have not participated closely enough

in the design process. Heeks (2006) calls this phenomenon, where the design does not match the needs, the *design-reality gap.* In healthcare, there are often complex and specialized practices, the characteristics of which may be undocumented and hidden from outsiders (Hanseth and Lundberg 2001:365), and designing for such work processes is therefore difficult.

This thesis examines national and regional health information systems, where *scaling* across users and use cases is an important goal (Braa, Monteiro, and Sahay 2004). To understand how to design architectures for large-scale deployments, we need to understand the different dimensions of heterogeneity that are introduced when systems are scaled across multiple user groups and domains.

The growth of national and regional systems raises various issues related to heterogeneity. The focus and tasks of the different health workers and managers using health information systems vary, and the effects of this diversity become increasingly evident as systems grow. This growth is challenging across several dimensions, beyond the mere increase in the number of users and deployments: i) use cases and application domains vary (Braa, Monteiro, and Sahay 2004; Shaw 2005); ii) the systems may need to support different types of workflow (Berg 1997); iii) the increasing number of users introduces different user roles, locations and rationalities (Ellingsen and Monteiro 2006; Heeks 2006); and iv) there are requirements for supporting new technologies that expand the realms of use of the solutions, including mobile phones, tablets and the web (DeRenzi et al. 2012). All these heterogeneity dimensions challenge how the software fits the users' needs. To avoid the design-reality gap, a typical solution would be to design the system with the input of users familiar with the real-world needs (Bjerknes and Bratteteig 1995). However, it is difficult to offer participation when products are shared between many different user groups in national and regional systems, because the requirements of different users can cause conflicts in the design (Pollock and Williams 2009; Neumann and Star 1996; Roland et al. 2013). This research claims that providing a good fit to users' needs is enabled by offering certain types of flexibility, distributed to the different actors involved in designing, configuring and using the system. The thesis also asks "*Which types of flexibility need to be supported in health information platforms to meet the challenges of heterogeneity?"*

According to Friedberg et al. (2013), there is considerable professional dissatisfaction with current health information systems in the US, and it is likely that the same issues exist in

other regions. Some doctors have even taken to YouTube to complain about how these systems prevent them from accomplishing their professional tasks as doctors (#LetDoctorsBeDoctors 2015). Creating large-scale systems that fit all health use cases equally well is challenging, since when a single technology is applied to multiple contexts, it is hard to provide an excellent fit for all users (Roland et al. 2013; Sæbø 2013). In other industries, innovation and heterogeneity have been addressed using platform concepts, but healthcare systems are typically more complex than systems in many other industries (Sturmberg and Martin 2013:4–8). Further research on how platform architectures can be applied to health is therefore essential.

### 1.1.2 Concern 2: Trend towards generic systems that need adaptation

Health organizations in developed countries often now use electronic health registers (EHRs) from large international vendors, although with mixed success (Koppel and Lehmann 2015; Makam et al. 2014; Mandl and Kohane 2012; Payne et al. 2015; Friedberg et al. 2013; Fitzpatrick and Ellingsen 2013). Similarly, many developing countries are using off-the-shelf health information systems that are developed globally and configured locally (Braa and Sahay 2012; Althauser et al. 2016; Broyles et al. 2016; Idris 2013; Mehl and Labrique 2014). To fit the local context and scale successfully, such generic software must be flexible and allow for local configuration (Fleck 1994; Sahay, Monteiro, and Aanestad 2009).

These generic information systems have given rise to a community of local system implementers and developers. They are a group of professionals who can help bridge the gap between the local health requirements and the generic software (Althauser et al. 2016; Titlestad, Staring, and Braa 2009; Roland et al. 2017). Such implementers often travel between different countries and sites and share information on how to configure the local information infrastructures (Titlestad, Staring, and Braa 2009). Early decisions regarding how the system is configured and used can significantly affect how easily the solution can adapt and scale later (Sanner, Roland, and Braa 2012). Designers of health information platforms therefore need to understand which strategies enable flexibility, and which types of flexibility are necessary for core developers, implementers and end-users. These different actors will relate to the various forms of flexibility in different ways, and we therefore also ask: "*How should flexibility in the platform be distributed to benefit the various actors in the wider platform ecosystem?"*

### 1.1.3 Concern 3: Increased scale leads to increased pressure on the core vendor to deliver

The increased reuse of software has helped to professionalize the development of health information systems, moving software development from ad hoc project-based financing to better-planned product development with long-term funding (Sæbø 2013; Althauser et al. 2016). Nevertheless, the scope and number of some large-scale projects can reach a level that exceeds the capabilities of the core development group of the system (Roland et al. 2017). In many cases, a global vendor may not even know the local requirements well enough to make suitable adaptations, causing design-reality gaps (Heeks 2006). To fit the local requirements and context, the core actors could instead enable external parties to create innovative solutions on top of their core platform (Tiwana 2013). It is challenging for a single vendor to provide sufficient user fit in these large-scale systems that reuse generic software across multiple sites (Mandl and Kohane 2012). The vendor can, however, provide flexibility for others to change and add functions on top of their system, hence distributing flexibility to these actors. Since multi-actor environments also make coordination and alignment more difficult, it is natural to ask whether there *are tradeoffs involved in distributing flexibility throughout health information platforms*.

This need for external innovation promotes the idea of using platform architectures to distribute flexibility in health information systems. These generic systems are designed, configured and used by somewhat independent actors through a platform value chain, where the actors involved in each step leverage the flexibility of the system to address particular concerns (Tiwana 2013). The recognition of this loosely coupled chain, where flexibility is distributed among different actors, motivates our central research question: *"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"*

This thesis builds on the assumption that platform-thinking is a possible answer to providing better-adapted and more efficient health information systems, and explores which factors must be kept in mind by platform owners, implementers, and users when engaging in this new approach to deploying health information systems.

The remainder of this chapter discusses the theoretical motivation of this thesis, introduces the research case and the practical implications of the research, and summarizes the research questions, findings and contributions.

### 1.1.4 Drivers for applying platform-thinking to health information systems

As explored above, there are several related drivers for the introduction of platform architectures in health information systems:

- The healthcare domain represents multi-actor environments, and it is challenging to develop large-scale systems that cater for the heterogeneous needs of many different healthcare users (Braa, Monteiro, and Sahay 2004; Friedberg et al. 2013; Fitzpatrick and Ellingsen 2013; Ellingsen and Monteiro 2006).
- There is a general trend in many industries, including healthcare, towards using configurable, off-the-shelf systems (Fitzpatrick and Ellingsen 2013). This trend is driven both by cost perspectives and by the fact that such systems come pre-developed, with capabilities that would be difficult to develop without cross-system learning (Gawer 2009). These systems have the experience and learning of others inscribed into them (Sæbø 2013), but must also be flexible and configurable to cater for local requirements (Sahay, Monteiro, and Aanestad 2009).
- Central vendors of large, off-the-shelf products cannot cater for all the requirements seen in healthcare at a global scale (Mandl and Kohane 2012). Additional products and integration with other vendors are inevitable at some level, but such integration can again give rise to challenges and the need for tradeoffs.

These points represent current practical problems, and are potential drivers for the introduction of platform architectures in health information systems. This thesis will show that there are certain tradeoffs that must be considered when introducing such platforms, and that the assumption should not be made that the application of platform architectures is sufficient in itself to solve the inherent problems of heterogeneity in healthcare.

## 1.2 Theory

The research question of this thesis concerns the application of platform-thinking in health information systems. The health information system domain has already been discussed in depth in extant literature, using concepts from information infrastructure theory (Monteiro and Hanseth 1996; Hanseth and Monteiro 1998; Sahay, Monteiro, and Aanestad 2009; Ellingsen and Monteiro 2006; Ellingsen and Monteiro 2008), although the platform literature

has not discussed health information systems to the same extent. Information infrastructure theory also lacks a thorough discussion of platform concepts.

When considering some of the primary drivers of why and how platform-based architectures can be applied to health, many factors are related to the different forms of flexibility. This work therefore combines theory from both information infrastructure and the platform literature, especially when discussing the relevant flexibility concepts. In particular, we consider how flexibility can be distributed across multiple actors taking different roles in the development, configuration and use of the system. The extant platform literature has gaps regarding aspects of health information systems, including large-scale health systems that can be considered infrastructures. The application of concepts from information infrastructure and information systems to the platform literature helps to fill these gaps.

The theoretical concepts related to flexibility are used as a lens to analyze and describe the characteristics of a specific class of systems that could be called *health information platforms*; these form a specific type of health information system that allows external innovation to act as an enabler in a broader ecosystem. Health information platforms share some traits with information infrastructures, and are characterized by the following: several heterogeneous groups of users; a large number of integrated systems; complex ownership and funding; long-term vendor relationships with high degree of lock-in; strict regulations curbing openness and sharing; an extreme focus on integrity, stability and availability; and a large variety of use cases.

Information infrastructures (II) are defined as "a shared, open, heterogeneous and evolving socio-technical system of Information Technology capabilities" (Hanseth and Lyytinen 2010). The process of change in IIs can be described as the evolution of an installed base and a pre-existing set of capabilities. It is difficult to change information infrastructures, and II theory involves a description of various concepts related to flexibility and change, including path dependency (flexibility over time), irreversibility, lock-in, change-flexibility and use-flexibility (Hanseth and Lyytinen 2004). This thesis also discusses other scholars' work on flexibility and concepts such as configuration and layering, which are related to information infrastructure and information systems (Orlikowski 1992; Sahay and Robey 1996; Sahay, Monteiro, and Aanestad 2009; Braa, Monteiro, and Sahay 2004; Fleck 1994; Braa et al. 2007; Baldwin and Clark 2006; Gebauer and Schober 2006).

The platform literature covers various forms of flexibility, but mainly from an architectural point of view, such as the effects that layering and modularization have on flexibility (Tiwana 2013:97, 101). Central to platform architecture is the concept of the relation between a core, its interfaces and the peripheral applications (Gawer 2014). The core platform provides an *architecture of participation* that entices external software developers to create applications through the modularity and option value in the platform (Baldwin and Clark 2006). Tiwana (2013) distinguishes between the architecture of the platform itself and the microarchitectures of the applications. Flexibility for platform users is imparted to the platform by the platform owner, through the definition of the layered platform architecture and a tightly related governance model. Limited only by specific control mechanisms, a platform owner aims to provide the external developers with a degree of autonomy in terms of how they use the platform and for what purposes. The evolution of the platform is therefore subject to a tension between autonomy and flexibility on the one hand, and control mechanisms and reuse of platform functions on the other. The platform owner must manage how much flexibility is distributed and the level at which the flexibility is provided. The theory chapter of this work summarizes platform theory in further detail.

There are relatively few stories of successful platform implementations in health, although the vision of open health information systems that facilitate innovation is a dream that many share (Mandl and Kohane 2012; Kasthurirathne et al. 2015; MoH Norway 2017). The platform literature mainly discusses platform implementations in other industries, and health information systems may have certain characteristics that are not prominent in other sectors. Since there are few truly open health information platforms, there is also a research gap in terms of the characteristics of and design principles related to such platforms. Designers and implementers of health information platforms need these design principles to guide them in making successful ecosystem enablers.

This thesis attempts to discuss these characteristics, architectural insights and design principles for health information platforms, and contributes primarily to the description of the distribution of flexibility in the overall architecture of these systems.

## 1.3 Research aims

Keeping in mind the issues and identified gaps discussed above, the objective of this research is to better understand how generic health information platforms, a particular class of health

information systems, can be designed to be flexible and to adapt to certain different user scenarios, user roles, workflows and technologies. The unit of analysis is the high-level socio-technical architecture of solutions and the characteristics, tradeoffs, and consequences of this architecture.

The central research question of this thesis is: *"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"*

The word 'distributing' indicates that an actor within the platform ecosystem distributes and another actor accepts such flexibility. The discussion of the different actors' roles in distributing and receiving flexibility is therefore important for our understanding of the research question.

In several industries, the requirements for diversity in large-scale solutions have been addressed by introducing platform architectures that allow external parties to innovate independently on top of a set of shared capabilities. This platform trend in other industries and development within the projects that form an empirical basis for this thesis give rise to research questions regarding how heterogeneity can be addressed by applying *platform architectural insights* to health information systems. These architectural insights are linked to governance and design.

While addressing the main research question, other, more specific questions are considered regarding the following aspects of platform architecture:

- *Which types of flexibility need to be supported in health information platforms to meet the challenges of heterogeneity?*
- *How should flexibility in the platform be distributed to benefit the various actors in the wider platform ecosystem?*
- *Are there tradeoffs involved in distributing flexibility throughout health information platforms?*

To answer these questions, two separate cases of health information platforms are studied: the birth of DHIS2 as a platform in some developing countries, and the integration patterns used to integrate applications into Electronic Health Records in Norway.

## 1.4 Empirical basis

The empirical basis for three of the papers in this thesis is the author's participation in several action research projects in India, Uganda, Zambia, Nigeria and Rwanda, with a focus on rolling out mobile applications for the health software DHIS2. The last paper is a case study considering three cases of applications integrated with Electronic Health Records (EHRs) in Norway.

DHIS2 is an open-source health information system developed by the University of Oslo and their partners over many years. The system has a rich set of configurable capabilities within health management, data warehousing, health data collection, visualization and reporting. Usage areas include traditional public health management, tracking of health programs, both at the aggregate and individual level, disease outbreak and response, basic EHR use cases and several other areas. The software is used in over 50 countries (Althauser et al. 2016). The author's work with DHIS2 relates to the tracking of patients who are followed up using mobile phones, for example following up on pregnant women and their babies after birth. The long-term evolution of DHIS2 offers the opportunity to examine the longitudinal aspects of a product evolving from a single application to an open health platform. The large number of countries and actors involved in developing and implementing DHIS2 provides an abundant empirical background for consideration of the infrastructural aspects of a health information system becoming a platform, beyond what is seen in single-vendor controlled platform environments.

The emergent platform capabilities seen in DHIS2 may also apply to other health information systems. In addition to the DHIS2-case, the author was fortunate enough to get the opportunity to study the concept of health information platforms in a developed world context. The last paper looks at three systems used in Norway, and how these are integrated into the Electronic Health Record (EHR) workspace. Two of these systems are national infrastructures that are shared among most healthcare institutions in Norway, while the third is a smaller application that is used locally in some hospitals. All three share the requirement that they must be integrated with the health workers' daily workflow. The paper considers the EHR as the platform and the three systems as external applications. The justification for adding a case that was methodically and contextually different from the main research cases involving DHIS2 is discussed further in Section 3.1.4. The reason for this inclusion was mainly opportunistic, and the author believes that any drawbacks to the inclusion of a

different context and method are amply balanced by the benefits of considering health information platforms in a different context.

## 1.5 Linking the included papers and their role in answering the RQ

This thesis includes four papers, each of which form part of the answers to the research questions. Figure 1 and the following description illustrate how these papers are linked.

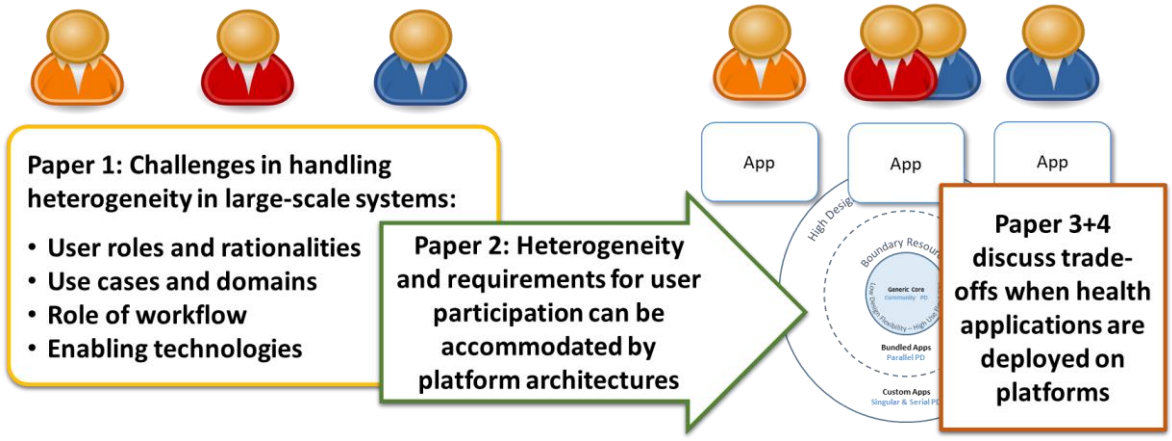


**Figure 1 illustrates how the four papers of the thesis link into the research topic.**

The first paper addresses the overall problem area, and considers the consequences of combining many different roles, sites and use cases in a single system. The paper discusses how different rationalities from multiple (or the same) users can clash when they are combined. From this observation of heterogeneity, we derive the need for user participation in the design process, and the need for flexibility in order to cover multiple use cases consistently in a single system. The paper helps us to describe the various dimensions of heterogeneity that exist in such large-scale systems. The topic of heterogeneity is not new in the health information system literature, but the paper places this problem into the context of DHIS2 and gives a background for discussing the types of heterogeneity that arise from health information system scaling.

Paper 2 goes on to discuss this heterogeneity problem in more detail, in a longitudinal study of DHIS2. The analysis covers multiple sites and discusses how the modes of participation and architecture have changed throughout the history of DHIS2. It concludes that a platform architecture can enable various levels of flexibility and user participation. The platform architecture combines some common, reusable features into a single core, and allows for flexible innovation in peripheral applications. The paper also introduces several different

types of flexibility, and indicates that these play different roles at different levels of the architecture.

The introduction of platform architectures to healthcare is not without challenges, and the last two papers help to illuminate areas in which platform designers and app developers must make tradeoffs. Although platform architecture allows some consistency to be maintained in reusable platform capabilities, the heterogeneity still causes issues, both for the applications and how they are integrated. Papers 3 and 4 consider the consequences of these platform architectures for the applications and required integrations, especially in terms of flexibility, scale, robustness and usability. These papers also highlight the lock-in effect of early architecture design decisions, and emphasize the need for proactive design decisions that keep in mind the later effects of design.

Together, as illustrated in Figure 1, these four papers provide the background necessary to address the main research question: *"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"*

## 1.6 Contributions

In this thesis, theory from Information Infrastructure is combined with Platform Theory to highlight architectural aspects that eHealth architectures should exhibit.

Paper 1 describes the heterogeneity of patient tracking information systems, and contributes a description of the needs and types of flexibility required. It argues that a rationality-aware design process with user participation is essential for addressing heterogeneity in the design of health information systems. The paper establishes that large-scale health information systems that span different user groups, work areas and regions must be flexible in order to fit the various needs of the multiple actors. The paper also discusses the problems in reusing the same generic software across sites when these sites are all involved in developing the system and have different underlying intentions of what the system should do.

Paper 2 considers how the DHIS2 architecture has evolved to accommodate heterogeneity, with a particular focus on a typology for the different modes of user participation in the design process as the solution changed from a singular application to a platform. The study highlights how the emergent platform architectures and the governance of the surrounding ecosystem co-constitute a platform for participation in design. The paper discusses how a

platform architecture that distributes flexibility among actors enables layered support for multi-actor applications and participation in large-scale health information systems.

Paper 3 contributes to the literature by discussing design tradeoffs between flexibility, robustness, cost and usability. The paper highlights how early design and stakeholder decisions for small pilots create path dependencies for the scaled solutions.

Paper 4 contributes to the information infrastructure and platform literature by proposing a typology of integration strategies that can be used to address change-flexibility for the third-party integration of applications into electronic patient health records (EHR). The paper contributes to the theory by discussing the tradeoffs between seamlessness and design flexibility, and how the related design strategy affects flexibility during and after scaling.

Papers 3 and 4 discuss cases where applications relate to multiple platforms in parallel, such as the mobile phone and DHIS2, and the EHR and a national system. The papers contribute with a problematization of the conflicting governance of multiple platforms, and the tradeoffs that stem from such multi-platform integration. From these papers, we can see that some health projects are required to relate to multiple platform providers in parallel.

The overall thesis contributes to the information infrastructure and platform literature with a discussion of flexibility within platform architecture and governance, with a particular focus on describing a concept I call *delegating flexibility*.

### 1.6.1 Practical contribution

DHIS2 was already a successful application when I started my Ph.D. in August 2011, but it had considerable potential to expand into the outer levels of the health hierarchies; this potential was being tested in some projects (Mukherjee and Purkayastha 2010; Sanner, Roland, and Braa 2012). This vertical expansion could only be facilitated by using mobile phone technologies in new and innovative ways, while staying within the restrictions of the low-resource context of developing countries. The research has as its main empirical basis the introduction of mobile technology into the DHIS2 product and its implementations, including research into the mobile projects that were already ongoing at the time.

Since I started researching DHIS2, there has been an emerging requirement to change DHIS2 into a platform for applications; this has been driven by trends in other industries and the large number of mobile and web contexts that the core DHIS2 had to support. Similar projects planning to expand their health information application into a platform will benefit

from this thesis, as it outlines and discusses some important design principles and elements to keep in mind when doing such migrations.

### 1.6.2 How can I know this is useful for practitioners? A personal perspective

In 2015, I left my Ph.D. research unfinished and started working full-time in the Norwegian Directorate of eHealth as an architect. My mission there was to help create a national platform that would support the deployment of personal connected health technology, enabling the older generation to stay at home longer and avoid hospitalization.

I was motivated by how useful the infrastructure theory, coupled with a knowledge of the DHIS2 platform, was in my new job. Inspired by the value of DHIS2 research, I decided to return to the University of Oslo to finish my Ph.D.

I felt that finishing the documentation of our work could be of practical importance, not only for platform creation in developing countries but also for developed countries that wish to create innovative ecosystems to address rising health costs and the lack of sustainable health innovation. To help show that some of the concepts and requirements are indeed similar in both developing and developed countries, my last article focused on the integration of health applications in a Norwegian context and the electronic health records (EHRs) that are already a fundamental part of the Norwegian health infrastructure. The article was a contribution to the present ongoing debate over health information platforms and ecosystems in Norway.

No two projects are the same, even when applied in similar countries and seemingly similar contexts. The contribution is therefore not one of permanent design principles that should be followed without thinking. The real knowledge lies in understanding how these design principles were formed. The rich story around the creation of principles is important in explaining how one can create or participate in one's own health ecosystem platform. In doing so, many commonly encountered problems can be dealt with, some old and some new.

## 1.7 Organization of this thesis

The following chapters will expand further on the contents of the thesis. Chapter 2 describes the extant literature, putting this work into the context of ICT within the health, Infrastructure Theory and platform literature. Chapter 3 discusses the research context and empirical cases. Chapter 4 reviews the research approach and method. Chapter 5 contains a summary of the papers, including their research findings and contributions. Chapter 6 discusses the contents

of the papers in the context of each other, and how they together help to shed light on the overall research aim. This chapter addresses both the overarching research question and the sub-questions, using flexibility as a lens. Chapter 7 concludes this work and offers some thoughts on future questions to be answered regarding platforms applied to health information systems.

# 2 Related research

In order to discuss the research questions, it is important to first highlight areas of related research. Three literature domains can be identified that relate specifically to the research topic of health information platforms:

- Health information systems
- Information infrastructures
- Platforms

This chapter therefore contains a review of the relevant literature on health information systems, information infrastructure and platforms, with particular attention to concepts that relate to flexibility. According to Merriam Webster, flexibility is "characterized by a ready capability to adapt to new, different, or changing requirements" (Definition of FLEXIBLE n.d.). As will be discussed later, the flexibility of use and changes in information systems are central themes in this thesis.

The review includes literature from the articles forming part of this thesis, although newer works such as Tiwana and Gawer's contributions to the platform literature have been given more space here, since these were not covered in depth during the research. The chapter concludes with a summary of the concepts that are most relevant to a discussion of the research questions.

The literature and debate about participatory design (PD) and the classification of different types of PD are not covered in this chapter, although participation was a central theme in Paper 2, "P for Platform". It may have been appropriate to include more literature in this Kappa that covered the participation of users in design processes, especially since Paper 2 shows that this need for participation is an important driver for platform architectures. Studies of participatory design (Bjerknes and Bratteteig 1995; Titlestad, Staring, and Braa 2009; Obendorf, Janneck, and Finck 2009; Kyng 2010) and generification (Pollock and Williams 2009; Pollock and Williams 2008) are particularly relevant to the discussion in "P for Platform", and form the basis for a wider debate on how flexible architectures are enablers for participation. Nevertheless, a decision was made to exclude literature on participation from this Kappa, and instead to emphasize the information infrastructure and platform literature, with a special focus on flexibility; the literature included here is more important for

the discussion leading to answers to the main research questions. A full inclusion of literature on participation and how it relates to the design of information systems would probably have distracted the reader away from the core research objectives, despite participation being central to one of the papers through its role as a driver for platform architectures.

## 2.1 Health information systems

Health information systems (HIS) in developing countries exist at different levels, including community health, smaller facilities, hospitals and national health management information systems (HMIS). Mobile HIS has been used as a tool to strengthen community-level data collection (Ngabo et al. 2012; DeRenzi et al. 2012), due to the cost and scale benefits of mobile technology, but these systems rarely make a national impact beyond isolated projects (Heeks 2006). The use of mobile technology and the grafting of these innovations into the national HIS backbone are fundamental requirements for creating sustainable data collection and use-mechanisms at the most local level (Sanner, Manda, and Nielsen 2014), but the implications and design of this fusion have not been carefully researched. The organization of the projects is either top-down, that is, looking at systems from the national level but lacking input from the local community; or bottom-up, that is, considering the local and often personalized clinical data at the community level, but not considering how to translate such data into nationally useful systems that scale in a sustainable manner (Braa, Monteiro, and Sahay 2004). The national and local levels are seldom single systems, and instead exist as compartmentalized architectures configured together for the local context, sometimes with standardized interfaces between the systems (Sahay, Monteiro, and Aanestad 2009).

### 2.1.1 Researching IS in a low-resource context

Many of the low-resource countries studied in this thesis have health systems that are traditionally managed and funded through independent health programs. These programs represent focused interventions aiming at improved health outcomes within their designated areas, such as antenatal care, delivery and reduction of maternal death, improved child health and vaccinations, HIV and prevention of mother-to-child transmission, TB and others (Braa et al. 2007; Shaw, Mengiste, and Braa 2007). In addition to clinics and hospitals, volunteer community health workers (CHW) play a major role in the actual work of implementing health programs at the community level. Community health workers are typically directly

involved with more than one health program and represent a local, community-level hub for health interventions (Lehmann and Sanders 2007).

Although the responsibility lies with the Ministry of Health (MoH), the initiative for improving the local health situation is spread across many actors. The timely and accurate collection, use and appropriate sharing of health data between all these actors have an enormous potential in coordinating and managing improvements to health services. Health management information systems (HMIS) are used as a data collection tool in many such countries (Althauser et al. 2016), although each country also has other information systems that play a role in the country's health information architecture. Each configuration of the systems in these countries is subject to a careful balance between the powers of the different actors at all levels of the health sector. These actors include the Ministry of Health, districts, local health workers, volunteer workers, donors, NGOs, implementing partners, commercial consultancy companies and software vendors (Braa, Monteiro, and Sahay 2004).

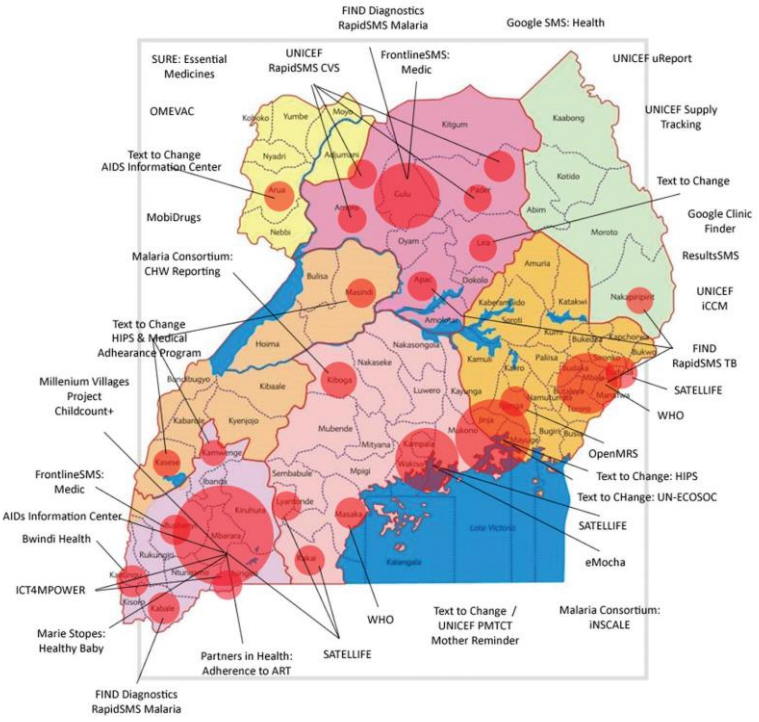


**Figure 2 shows Uganda as an example of a country with a large number of mHealth pilots, marked here with circles. Reproduced with permission from Sean Blaschke (Technology for Development Specialist at UNICEF Uganda). From https://aidleap.org/2014/11/20/is-there-too-much-innovation-in-development/ (aidleap 2014).**

Some countries have been flooded with innovation projects aiming to introduce innovative health information systems using mobile technologies, i.e. so-called mHealth (Sanner 2015). In some countries, the number of these projects is so high, and the ability to convert the

projects into sustainable infrastructure is so low, that these pilots have been compared to a disease, referred to as “pilotitis” (Kuipers et al. 2008). In the case of Uganda, the number of disjoint eHealth projects led the government to increase its focus on establishing common standards and reusable infrastructure (McCann 2012), though early efforts were hampered by a lack of open architectures and standards.

Health service outlets such as hospitals, clinics, health posts and outreach health workers employ entirely different user settings, placing vastly different requirements on workflow even for similar tasks. For example, the same use case of vaccinating children would be organized and optimized differently within a hospital, clinic or a mobile health post, because the context and number of children to vaccinate would be different (Requejo, Bryce, and Victora 2012).

Information systems supporting health service follow-up activities about individual patients across time and space are inherently complex and difficult to design, especially when they are required to accommodate a multiplicity of end-user roles and requirements (Hanseth and Lundberg 2001). Typically, these systems may need to accommodate the migration and referral of individuals across healthcare units, bridge private and public-sector interests and negotiate ownership, security and privacy concerns related to patient health data (Haggerty 2003; Braa et al. 2007). Haggerty (2003) defines the concept of continuity (of care) as “the degree to which a series of discrete healthcare events is experienced as coherent and connected and consistent with the patient’s medical needs and personal context.” The desired continuity of care in patient follow-up includes informational continuity, management continuity, and relational continuity. Patient-oriented systems supporting this level of continuity of care can have a broad functional and organizational scope (ibid).

In the case of pregnancy follow-up, direct users of historical patient information may include the pregnant woman herself, nurses, midwives, general practitioners and highly specialized physicians dealing with referral cases (Namukwaya et al. 2011). In addition, the patient-oriented information system may be drawn on to share information with billing, logistics and human resource systems, aggregate health management information systems and health insurance systems, to name a few. Different contexts and situations impose new and sometimes contradictory requirements for information systems (WHO 1994; Tamrat and Kachnowski 2012).

The advent of mobile technology has created an IT change at the local levels of the healthcare hierarchy, with mobile technologies being deployed to replace existing paper-based systems. This research project falls into the category of such technology projects, but is also solidly integrated with the national health information systems forming an information infrastructure at the higher levels of the health service (Sanner 2015). The introduction of technology at the local level brings new contrasts in workflow and more user-roles to be considered in the configuration of the systems.

### 2.1.2 The trend towards large-scale integration and standardization in health

Fitzpatrick and Ellingsen (2013) conducted an extensive review of 25 years of CSCW research on healthcare, uncovering research carried out on the use of EHRs by GPs; the use of EHRs in hospitals, both in departmental silos and across organizations; the use of IT systems to provide mobility in terms of data, patients, employees and resources; IT systems used for temporal planning, shift changes and handovers; information flow between organizations; the expansion of the context of healthcare work to new areas; multidisciplinary cooperative work; and the use of IT to move care into the home. One major trend that these authors claim to see within these domains is the move towards large-scale implementations of healthcare IT solutions, and an increased need for the integration and standardization of systems, resources, and workflows (ibid). This need for integration arises as a consequence of more integrated care models involving a group of different professionals, the patient and their social network (ibid).

### 2.1.3 Embedding work practices in information systems

Berg (1997) describes how protocols in health services are increasingly embedded into information systems, and he problematizes this trend as a "tendency to perceive and describe the management of patient's trajectories as constituted by a sequence of individual, formally rational decisions" (ibid). These protocols are formalized work practices that are meant to be general and applicable across different settings. Berg argues that multiple rationalities are present when treating patients, not all of which can be coded into a protocol. Tools must therefore support these different rationalities to be useful, and there is seldom only one right answer to a clinical problem when considering its context (ibid). Decision support systems may take a more or less active role in making the actual decision, or just facilitating a decision in being made (Silver 1990).

Information systems implementing protocols may lack the flexibility to allow health workers to make such context-aware decisions, based on observations of the social part of the treatment process. It can be concluded from Berg's article that the ideal medical workflow would be a mix of the workflow embedded into the information system used and rational decisions made by the health workers based on their experience, the context of the disease, results from medical research, the resources available and other social aspects. The clinician should make decisions based on all of these factors. Systems implementing formalized workflows through protocols must allow the user the flexibility to take into account other aspects of the context.

Gasser (1986) introduces the concept of *workarounds* to describe how users may be "intentionally using computing in ways for which it was not designed or avoiding its use and relying on an alternative means of accomplishing work." He introduces different types of workarounds, such as data adjustment, procedural adjustment, and backup systems. Damsleth (2013) applies the concept of workarounds to an integration setting, and shows how users fill gaps in the integration between systems with workarounds. When applications are poorly integrated or do not support the required workflow, users will work around this integration to fulfill their required tasks.

Berg (1999) also points out that some workflows that use structured form data entry at the point of care may be there mainly because of the intended secondary use of information, rather than to support the day-to-day tasks of the clinician directly. Friedberg et al. (2013) convincingly show that many clinicians find their EHRs overly templated; these templates are implemented partly due to top-down management decisions rather than their direct usefulness to the clinician.

For simplicity, Electronic Patient Records (EPR) and Electronic Health Records (EHR) are considered here to be the same type of system, and the term EHR is used throughout this thesis (Smolij and Dun 2006). Fitzpatrick and Ellingsen (2013) find that a long list of CSCW papers "illustrate how the tensions between process versus patient/data centric views of EPRs play out in practice, and how clinicians develop practices to work with or around EPRs." The EHR is an important tool in providing a balance between flexibility and rigidity, to support both the secondary and primary use of health data for different professions within health organizations (Berg 1999).

Together, these authors show that embedding workflows into health information systems can be troublesome, both in terms of those secondary tasks that are forced upon the end-user through the information system and those that are meant to support the end-user in their primary tasks.

### 2.1.4 Health platforms

There have been attempts to approach health information systems with a platform mindset in order to increase flexibility in the implementation of different applications and workflows. Christensen and Ellingsen (2016) describe a case from a Norwegian hospital region where model-driven development was meant to "separate organizational issues from the underlying technology platforms." The intention was to give users a greater ability to shape the clinical behavior of the system by separating the technical design from the clinical problems. Christensen and Ellingsen (2016) conclude that although the concept has an appealing objective, the modeling process itself and the competence required to do this well across multiple contexts is challenging. International standardization and local initiatives must be able to work together to achieve this. Bygstad and Hanseth (forthcoming) use the term platformization to describe the process of creating health information platforms within a large eHealth study from Norway, and other countries also have national integration solutions that resemble platform architecture patterns (Sellberg and Eltes 2017).

Another notable initiative where platform thinking has been applied to EHR integration is the SMART-on-FHIR framework funded by the US government (Mandl, Mandel, and Kohane 2015; Mandl and Kohane 2015; Mandl and Kohane 2012; Mandl and Kohane 2009; Mandel et al. 2016). SMART-on-FHIR is an architecture where web-based applications can be deployed inside EHRs, enabling applications with a flexible user interface to be created on top of the otherwise quite rigid EHR infrastructure (Mandl, Mandel, and Kohane 2015).

Unfortunately, despite these limited efforts, the extant literature indicates that today's health information systems may not yet be ready to become full-scale platforms (D'Amore et al. 2014; Friedberg et al. 2013).

## 2.2 Infrastructure theory

### 2.2.1 What is Infrastructure theory?

There is no single definition of an information infrastructure. The concept arises from the view of the internet as an infrastructure that transports information in ways that are shared by many applications and users. Al Gore is known for calling the internet the Information Superhighway and for stressing the infrastructural traits of the internet (Hanseth, Monteiro, and Hatling 1996).

Hanseth and Monteiro (1998, ch. 3) describe information infrastructures as complex solutions that have the following characteristics: enabling, shared, open, sociotechnical, heterogeneous, and an installed base. These characteristics separate information infrastructures from traditional systems and applications. The analysis of information infrastructures in the extant literature elaborates on several concepts that are important when implementing and scaling information infrastructures, including path dependency, complexity, bootstrapping, standardization, installed base cultivation (Hanseth, Monteiro, and Hatling 1996; Hanseth and Monteiro 1998; Hanseth and Lyytinen 2010; Hanseth and Lyytinen 2004), irreversibility (Callon 1991; Hanseth, Monteiro, and Hatling 1996); and modularization (Hanseth, Monteiro, and Hatling 1996).

The installed base is the collection of pre-existing systems, information, standards, actors, processes etc. that make up the information infrastructure. Scholars describe how the installed base is continuously evolving and is out of the control of the original actors. Even what appears to be a new infrastructure will always be built on pre-existing elements, and its introduction must relate to the installed base (Hanseth and Monteiro 1998).

Dahlbom and Janlert (1996, from (Ciborra 1997; Ciborra and Hanseth 1998)) distinguish between construction and cultivation as two forms of shaping technology. Explaining cultivation, they describe how a natural process can be interfered with, supported and manipulated, rather than taking full control of building a complete system from scratch. Construction and cultivation are hence two different approaches to systems thinking (ibid). Since there is no single point of total control and the role of the installed base is so dominant in II, its design and change processes should involve cultivating the installed base rather than from-scratch-design (Hanseth and Monteiro 1998).

Information infrastructures also have a characteristic called irreversibility that affects the change process (Hanseth and Monteiro 1998). Irreversibility is seen when a network of actors converges and becomes aligned in such a way that changes are difficult to implement (Callon 1991). A change for each actor is dependent on the others, and even small changes may affect the whole network. Irreversibility is the extent to which it is hard to go back to an earlier, looser network. The irreversibility of a network also affects the future development of the network (ibid), restricting the path of possible options.

Certain decisions are made for historical reasons. When judgments in the past lead stakeholders down a certain path, forcing other compatible future decisions to be taken, this is called a path dependency (Hanseth and Monteiro 1998:151; Carlile 2004). Irreversibility and path dependency are therefore concepts that relate to flexibility along the time axis.

### 2.2.2 Design, change and use-flexibility

Hanseth et al. (1996) discuss the terms flexibility and standardization as seen in information infrastructures. Their discussion and analysis of the extant literature, backed by empirical material from information infrastructure cases, shows that there are several types of flexibility that affect systems development in different ways. They also point to there being relative degrees of flexibility and that different design options may expose varying degrees of flexibility (ibid).

Orlikowski (1992) explains that technology both is created and used by humans: "Technology is created and changed by human action, yet it is also used by humans to accomplish some action." In each of these two domains, creation and use, there are concepts of flexibility. She explains how interaction with technology can be described as having two modes, design-mode and use-mode, each of which relates to flexibility in different ways. These modes are tightly coupled, but their conceptual separation is useful for understanding to what extent users can affect redesign (ibid). The level of flexibility is also role-dependent. Users often consider technology to be an unchangeable black box, while designers can change the technology itself and have a wider view of the flexibility of the system (ibid).

For an information system, design and change-flexibility are important when introducing or adapting systems (Hanseth, Monteiro, and Hatling 1996). These are dimensions of flexibility that relate to the design mode of information systems. If an information infrastructure has low change-flexibility, it will be difficult to introduce new changes into it, while high change-

flexibility allows designers to make changes easily. Several prior works discuss which elements affect the change-flexibility in information infrastructure, and how low change-flexibility can be overcome by applying the right design decisions (Hanseth, Monteiro, and Hatling 1996; Hanseth and Monteiro 1998; Sanner, Manda, and Nielsen 2014). Change-flexibility can be affected, for example, by modularization and the right approach to standardization (Hanseth, Monteiro, and Hatling 1996). These design and change-flexibility characteristics restrict or enable the maneuvering space for designers and developers of systems. Gebauer and Schoeber (2006) describe flexibility-to-change as being related to major changes in information systems, and define major changes to include fresh system setups, including re-installation and testing.

After systems have been established, there is an element of flexibility that shapes how easily the system adapts to multiple uses. This flexibility concerns the use mode, as described by Orlikowski (1992). Systems may be more or less flexible for different use cases and local requirements, as seen from the perspective of the end-user (Orlikowski 1992, Hanseth et al. 1996). This use-flexibility is dependent both on characteristics of the technology and the perception of the users, as will be discussed later. Gebauer and Schoeber (2006) include non-major changes within the scope of use-flexibility, and classify possible use-time changes into (i) system functionality, (ii) the scope of the underlying database, (iii) user interface, and (iv) processing capacity. These changes can also be done at design time, and this categorization helps to operationalize the difference between the design and use-time changes.

### 2.2.3 Use-flexibility, inscriptions and programs of action

Software designers can entice users to perform certain actions by inscribing patterns of use into software objects (Monteiro and Hanseth 1996; Akrich 1992; Callon 1991:143). During the information systems design process, the developer works out a scenario for how the system should be used (Ciborra 2002), and this scenario is then inscribed into the software system (Akrich 1992). Inscriptions vary in strength (weak or strong), affecting the likelihood that the end-user will follow the inscription. The strength of the inscriptions embedded during the design-mode relates to the use-flexibility of a solution in use-mode (Monteiro and Hanseth 1996; Gebauer and Schober 2006).

Use-flexibility and inscriptions can also be linked to the concept of *workarounds*, which is a deviation in the planned action by the system designer (Gasser 1986; Damsleth 2013).

Workarounds are ad-hoc strategies that solve immediate and pressing problems, often conflicting with the formal ideology of system use (Gasser 1986).

Silver (1990) introduces the concepts of "system restrictiveness" and "decisional guidance" in decision support systems. These are concepts which bring extra dimensions to the strength of inscriptions and the existence or lack of flexibility. He claims that a system's restrictiveness must promote rather than inhibit the use of the system. Strategies for inducing user change also differ between "directed change" and "non-direct change", where the former often focuses on greater restrictiveness and strong guidance (strong inscriptions), and the latter includes guiding users in making their own decisions without influencing the actual direction of these decisions (ibid). Systems with too much restrictiveness will be avoided by users, while a system with too little restrictiveness may feel overwhelming and difficult to use effectively (ibid). Silver's (1990) discussion exemplifies how the concept of restrictiveness also can be useful in understanding flexibility.

### 2.2.4 Design-flexibility and modularity

Many scholars stress the importance of modularity for reducing complexity and increasing flexibility (Simon 1996:199; Schilling 2000; Yoo, Henfridsson, and Lyytinen 2010; Berente and Yoo 2012). Information infrastructure scholars also point to modularization and encapsulation (Parnas 1972b; Parnas 1972a) as important characteristics for enabling flexibility in IIs (Hanseth, Monteiro, and Hatling 1996; Hanseth and Monteiro 1998; Braa et al. 2007).

Modularity can be implemented in different ways, and correct modularization decisions greatly affect the flexibility of the solution (Hanseth and Monteiro 1998). The interfaces of information infrastructure modules are typically standardized, although the process and detail of standardization vary (Hanseth and Lyytinen 2004).

The concept of *black-boxing* (Latour 1987) helps to describe why modularization is necessary. When a component is black-boxed, the inner details of the component are no longer of interest to the public, as people only relate to its interfaces and their perception of its intended use. The designers can change the implementation inside the black box as long as they adhere to the agreed visible external interfaces, thus allowing a level of flexibility combined with stability. The black-boxing of components allows external developers to trust components that other developers have made, enabling their use in complex systems. The

independence of the inner implementation is described by Parnas (1972a) with the principle: "The specification must provide to the intended user all the information that he will need to use the program correctly, and nothing more."

Baldwin and Clark (2006) describe two critical architecture principles of open-source development: i) modularity and ii) option value. They write that modularity exists if "parts can be designed independently, but work together to support the whole." They further define the concepts of a platform and modules, where the platform supports the modules and is essential to the system. "Option value" is a property that enables the possibility of doing something in the future, although this is not an obligation. As such, option value allows the flexibility to carry out future actions. Open source software solutions that are more modular also have a higher degree of option value (Baldwin and Clark 2006). The way in which open source software projects are modularized is dependent on the project organization and the way developers cooperate, and both modularization and organization affect the flexibility in terms of changing the system architecture (MacCormack, Rusnak, and Baldwin 2006).

### 2.2.5 Flexibility in standardization

Hanseth and Lyytinen (2004) state that "open and standardized interfaces" are a defining feature of information infrastructures. Standards are "a set of technical specifications adhered to by a producer, either tacitly or as a result of a formal agreement," and a variety of different types of standards exist (David and Greenstein 1990). Standards-based components within an information infrastructure enable flexibility through modularization, but the standards themselves are also subject to varying degrees of flexibility (Braa et al. 2007). In some scenarios, flexible standards are required to cater for changing environments (ibid).

When information infrastructure standards are well-defined and simple, they can behave as attractors that draw a large number of heterogeneous actors with different use-cases to the infrastructure (ibid). Braa et al. (2007) define two important principles related to the development of standards for information infrastructures: i) the principle of flexible standards ("it is important to craft standards and their relations so that they emerge as a complex adaptive system that can adapt to a changing environment and thereby contribute to the sustainability of the HIS"); and ii) the principle of integrated independence ("[i]mproved integration of information systems is also at the center of the efforts presented in this article to enable smoother coordination and control of organizational processes and healthcare

delivery. But integration may cause less independence and less flexibility."). They do not address in great depth the questions of when integration acts as an enabler for smooth coordination, when it causes less flexibility, and whether there is a tradeoff between the two. This thesis, however, outlines some principles which illuminate this topic further.

Braa et al. (2007) also state that the total flexibility of a standard is the sum of both its use- and change-flexibility (ibid), and recognize that both the design and use modes are important.

### 2.2.6 Flexibility in deployment

As seen in earlier sections, flexibility relates both to the design- and use-modes of a solution. It is also worth considering the flexibility aspects of deploying components into existing information infrastructures. The concept of *configuration* (Fleck 1994) can be used to further elaborate on deployment flexibility.

Configurational technology allows implementers to select technological components and to combine these to fit local requirements (Fleck 1994), often combining them with other existing infrastructure components (Sahay, Monteiro, and Aanestad 2009). The process of configuration typically combines generic technology knowledge with local practical knowledge to reach contextually appropriate implementations (ibid).

The application of configuration to different cases is limited by the local context as well as the features and underlying logics of the technology; these constrain a given technology from being universally configurable to all contexts and logics (Fleck 1994; Kallinikos 2004; Doherty, Coombs, and Loan-Clarke 2006). Repeated attempts to configure the same technology in different contexts can lead the software vendor to implement common requirements and flexibility of configurability into the artifact (Fleck 1994; Titlestad, Staring, and Braa 2009), which is part of the process of making the solution generic.

Implementation projects are not always about deploying all the available features of a technology. Because new technologies are seldom implemented in a void, the configuration process sometimes concerns the fitting of selected parts of the technology into welcoming gaps in the local infrastructure (Sahay, Monteiro, and Aanestad 2009; Sanner 2015). The process of socio-technical configuration can thus also be seen as a sociotechnical and political process for how a piece of technology is positioned in relation to other components in the infrastructure (ibid), and recognizing that the socio-technical concept of configuration can extend our understanding beyond simple technical configuration of artifacts.

### 2.2.7 Flexibility in the scaling process

The scale of an information infrastructure is not merely a matter of how many users the infrastructure has (Shaw 2009; Sæbø 2013). Sahay and Walsham (2006) wrote: "scaling concerns the process through which that product or process is taken from one setting and expanded in size and scope within that same setting and/or also incorporated within other settings." Braa et al. (2004) describe scaling as a process of spreading a solution that works in one site to other locations. Shaw (2009) shows that scale for health information infrastructures can be considered in terms of both geography and scope, and for each of these, scale can imply both a deepening and widening. In Section 6.1, some dimensions of the heterogeneity caused by scaling solutions are considered, and from this discussion we will see that scale can be understood to mean more than simply size or scope.

Some scholars consider scaling a prerequisite for the success of certain types of information infrastructures, a challenge that has been called the all-or-nothing predicament (Sanner 2015; Braa, Monteiro, and Sahay 2004). In certain cases, it seems that the information infrastructure is of little use without scale, and at the same time, the first users "cannot be sure if the infrastructure will be implemented on a full scale or whether it will be an infrastructure at all" (Hanseth and Lyytinen 2004). This is a chicken-and-egg problem of what to prioritize—scale or usefulness—and this is addressed in Paper 4 of this thesis (Roland, Sanner, and Aanestad 2017).

Hanseth and Lyytinen (2004) suggest working around this conundrum of growing IIs by applying a *bootstrapping* design principle. Bootstrapping involves focusing on the immediate usefulness and simplicity for the first users, while acknowledging that the system has not yet scaled. The existing installed base is leveraged for growth by reusing the existing infrastructure where possible when diffusing the new elements, using gateways.

Heeks (2006) suggests that the successful scaling of health information systems may require flexibility in the scaling process itself. Those parts of the innovation that are scaled first, and the speed at which it is scaled, may be important for success. He claims that health information systems should be delivered in steps, and that a reduction in the project scope may even be a useful strategy for providing some parts successfully (ibid).

### 2.2.8 Interpretive flexibility

Flexibility is relative, and is dependent on the point of view. The interpretive nature of flexibility has been discussed by scholars as *interpretive flexibility*, and has been used as a lens to examine how people interpret the possible uses of technologies and artifacts (Bijker 1987; Bijker, Hughes, and Pinch 1987; Sahay and Robey 1996; Orlikowski 1992). It is open for discussion whether interpretive flexibility is a characteristic of the technology or the viewer; however, if the interpretive flexibility is high, either due to the artifact or due solely to the attitude of the users, there is more ability both to change the artifact and to use it in unintended ways. If most people have a similar, rigid interpretation of what an object is and what it can be used for, its interpretive flexibility will be low.

Interpretive flexibility was introduced by Bijker (1987) to explain how technology is socially constructed, using examples of the development of technologies such as bicycles and plastics. Bijker links interpretive flexibility to the idea of technological frames, which conceptualizes the boundaries of a social group in interpreting technology.

Based on how the design of a bicycle converged over time, Bijker (1987) claims that as a technology is institutionalized and its design converges, its interpretive flexibility is also reduced. In terms of actor network theory, this may be described as the actor network tightening as all actors are aligned around the same understanding of the technology (Monteiro and Hanseth 1996). The technology is black-boxed (Latour 1987), and a dominant design emerges (Tiwana 2013).

Interpretive flexibility can also be illustrated by contrasting a traditional telephone with a modern smartphone. A traditional phone would be considered by most people as a device for facilitating speech between two people in different locations. A smartphone, with its malleable user interface and mobility, could be interpreted flexibly as a device for talking, blogging, texting, emailing, playing games, watching movies, a fitness logging device, a fashion artifact, a device for surfing the web, opening beer bottles or reporting health information. Due to the platform architecture of smart phones, app developers can continuously add to the understanding of what a modern mobile phone is.

### 2.2.9 Information infrastructures and platforms

Hanseth and Lyytinen (2010) describe how the concept of information infrastructures relates to platforms. They define four different classes of IT solutions with increasing complexity: i)

IT capabilities; ii) applications; iii) platforms; and iv) information infrastructures. These classes differ in "their overall complexity, how they relate to their design and use environments, and how they behave over time in relation to those environments" (ibid).

Information infrastructures and platforms have some common characteristics, but IIs are typically more open and heterogeneous and less controllable than platforms. A central actor typically controls the platform (the platform owner), while infrastructures have a more distributed and dynamically negotiated control regime (ibid). Since Hanseth and Lyytinen (2004) describe three different types of IIs (universal, business sector and corporate), the difference between platforms and IIs may in fact depend on the type of II considered.

## 2.3 Platforms and ecosystems

### 2.3.1 The ecosystem: an information system flowerpot in which innovation grows

One word that is often mentioned when discussing platforms is the concept of *ecosystems*. With obvious links to ecology, the word implies an environment where certain elements, living beings and the environment, are dependent on each other (Tansley 1935). Drawing a parallel with information systems, when multiple software companies co-exist in an environment in order to engage in complementary innovation, this environment can be called a platform ecosystem (Ceccagnoli et al. 2011; Tiwana 2013). The concept of the platform ecosystem also relates to the platform value chain, that is, the network through which value is provided to the users of the platform (Tiwana 2013). Some scholars prefer to use the concept of value network rather than chain, as it adds more dimensions to the ecosystem (Christensen and Rosenbloom 1995); however, for simplicity, we will use the terms value chain and ecosystem here.

### 2.3.2 Gawer on platforms

Gawer (2014) describes how technological platforms are widely recognized, using several notable examples such as Google, Apple, and Facebook, but finds that the management research agenda in terms of platforms has been limited. Two separate theoretical perspectives have been proposed for considering platforms: as types of markets or as modular technological architectures (ibid). From an economic point of view, platforms can be considered as special kinds of markets, in which two or more different types of consumers/providers can transact with one another more easily than they could have without

the platform. An important factor of such multi-sided platforms is the network effect, which occurs when multiple players are present on the platform (Gawer 2014).

From a technological point of view, Gawer explains that all platforms share a common modular architecture comprising three elements: a core, a periphery (complements), and the interfaces between these (Gawer 2009; Gawer 2014). The interoperability between the core and the complements is made possible via the design rules or interface specifications linking them. The interfaces have traditionally been considered to be long-living and stable, allowing the internal architecture of the core or complementary applications to change as long as the interfaces remain. Gawer (2014) indicates that change and evolution of these interfaces are possible as a part of the governance and evolution of the platform, when conceptualizing a platform as an organization.

Since the platform's core capabilities are available to all applications for reuse, the cost of variety and innovation is reduced; the system does not have to be re-invented or rebuilt from scratch to generate a new product, accommodate niche requirements or respond to changes in the environment (Gawer 2009). A major challenge for designers creating a platform is to identify which parts should be common and which should be left to external application developers, enabled by the stable yet versatile interfaces of the platform (ibid).

### 2.3.3 Tiwana's platform

Tiwana (2013) describes in great detail how software vendors can evolve into platform providers through a combined architecture and governance strategy. He describes platform markets as being different from traditional markets, partly because open innovation allows for faster evolution cycles, with many actors co-evolving the ecosystem under the guidance of a dominant platform owner, thus providing access to micro-segments to which traditional software players would not be able to provide solutions.

According to Tiwana (2013), a platform is a multi-sided software system that allows different types of players to come together and interact, such as an application vendor and an application user. Most platforms are two-sided, but some have more sides, such as iOS, which involves end-users, developers and advertisers. Tiwana (2013) opines that platforms must be multi-sided, but not all scholars share this view (Gawer 2009).

The platform ecosystem can be divided into the upstream and downstream parts of a value chain. As an illustration, one can imagine a river, where the platform vendor is upstream and

application vendors are downstream. Platform vendors can add enabling features to their platform, but must relinquish control over the application and allow independent developers to use the features of the platform to innovate. Unlike in traditional software development, the platform vendor must carefully introduce capabilities in the hope that these will have the desired effect downstream, as application vendors use the platform. A significant aspect of the value of a platform ecosystem lies in the fact that application developers can add innovative applications that the platform vendor cannot implement, and that this innovation is outside the platform owner's direct control. Tiwana states that "Software platform ecosystems must be orchestrated rather than managed", which is similar to the cultivation concept as described in infrastructure theory (Hanseth and Monteiro 1998).

Dougherty and Dunne (2011) claim that in ecology-oriented business networks, new innovation is held back by centrally planned innovation. Since knowledge and other resources are distributed across a large number of actors within the ecology, multiple organizations must actively participate in the innovation to succeed, rather than being controlled by a single company. Tiwana (2013) states that coordination in platform ecosystems is achieved through defining the platform architecture, and therefore links the definition of the architecture very strongly to the governance of the ecosystem. In the context of platforms, governance is defined as "who makes which decisions and how the 'pie' is split among the platform owner and the app developers" (Tiwana 2013).

Depending on the market situation, the application vendors are typically provided with sufficient autonomy and may even have the ability to move their applications between competing ecosystems, a concept Tiwana calls multi-homing. In this sense, multi-homing allows flexibility for application vendors to switch between and reside on multiple ecosystems.

In what he describes as the seesaw problem, Tiwana (2013) explains how platform owners and their app developers are subject to a delicate balance between integration and autonomy. A platform enforcing tight integration, both with the platform and across applications, leaves less room for developers to make their own choices, a balance which must be handled carefully. Integration offers benefits in terms of creating seamlessness between apps within the ecosystem; however, tighter integration implies reduced flexibility and freedom for the app vendor. Requirements for tight integration may also limit multi-homing (ibid).

Tiwana describes how the early architecture choices of ecosystem platforms are irreversible in practice. As the architecture is adopted by application developers and becomes intertwined with governance options, the architecture becomes increasingly difficult to modify. He points out that the effects of such architecture choices may become apparent long after the choices have been made. As such, the future flexibility of the platform is affected by these early architecture decisions, as described by path dependency in infrastructure theory.

Tiwana also discusses the concept of 'real options thinking' as a way to control flexibility over time. The concept of real options was originally introduced in a discussion of financial flexibility (Tiwana 2013; Trigeorgis 1993). Real options are possibilities introduced early, through decisions that enable later choices and flexibility. For example, the exact manner in which an information system is modularized can introduce options that can be triggered later. Modularization can enable the expansion of the platform into new markets or user scenarios that had not been identified early on. Real options could therefore be considered the opposite of path dependencies, in that certain early decisions allow for later flexibility.

In the initial stages of competing ecosystems, each will typically use diverse architectures and governance models. As the market evolves, Tiwana (2013) claims that one of the vendors will establish a dominant design through its market position (Anderson and Tushman 1990), which the other platform vendors are forced to follow. Tied together via other mechanisms such as lock-in, network effects, leapfrogging, envelopment and multihoming, the platform players in the market compete and evolve, sometimes alternating as the dominant player.

The concepts of upstream and downstream should not be misunderstood to mean that platform vendors lose all means of governing the development of an application within their ecosystem. The platform vendor may activate multiple control mechanisms, which Tiwana (2013) categorizes into gatekeeping, process, metrics and relational; decision rights can also be either centralized with the platform owner or decentralized to the application developer.

Gatekeeping is the mechanism of allowing or disallowing certain apps and developers onto the platform, possibly subject to a strict yet predictable set of guidelines. Process control involves requiring the app developer to follow certain development processes, rules and regulations while developing applications. The metrics control mechanism places specific requirements on the application, such as performance, level of support, impact on platform performance and business metrics such as unit sales, downloads or end-user ratings. The relational control mechanism is a higher-level measure; for example, it establishes shared

norms and values that create a common identity for the solution. Tiwana (2013) discusses the governance process between three actor groups: the platform owner, the app developer and the end-users. Different platform types may, however, have more 'sides' and more actors who converge on the platform to co-evolve the ecosystem, and the governance process would presumably also then be more complex. In such cases, the ecosystem may resemble a value network more than a chain (Christensen and Rosenbloom 1995).

### 2.3.4 Platform architecture

As described by several scholars, the platform consists of a core, interfaces and complementary applications (Gawer 2009; Gawer 2014; Tiwana 2013). The architecture of the platform is typically modular, allowing flexibility in how applications are built and how the platform evolves. The core platform provides an *architecture of participation* that entices external software developers to make applications, through the modularity and option value in the platform (Baldwin and Clark 2006). Tiwana (2013) distinguishes between the architecture of the platform itself and the *microarchitectures* of the applications; the platform architecture includes the core and its interfaces, while each application typically has its own microarchitecture that is subject to the modularity and options available in the platform. Common to all microarchitectures is the fact that they delegate some parts of their functionality to the core platform; however, the extent to which they do this is dependent on the capabilities of the platform architecture, the choices of the developer and the governance of the platform owner. Multiple application microarchitectures may communicate through the interfaces of the platform or outside it, and through this communication, they are said to be integrated.

### 2.3.5 Resourcing and securing a platform

Ghazawneh and Henfridsson (2013) use the theoretical concept of boundary objects to describe how a platform provides interfaces to application providers, and the role that application providers play in defining these interfaces. They elaborate on the concepts of resourcing and securing. Resourcing is the process wherein the scope and diversity of a platform are enhanced, and securing denotes the process by which the platform owner maintains control of the platform and related services. They also identify four specific variants of these concepts. Self-resourcing is the creation by app developers of new services on the platform that are outside of the platform owner's control. Diversity resourcing is a

deliberate action by the platform vendor in order to expand platform services; it is intended to lead to an increased use of the platform and potentially to envelop other platforms' domains. Regulation-based securing is a non-technical means for a platform owner to control access to their platform. Sovereignty securing refers to measures taken by the platform owner to stay in control of their domain, thus protecting against being enveloped and preventing multihoming (Ghazawneh and Henfridsson 2013)

## 2.4 Summary of related research: Conceptual framework

The above review of the extant literature goes beyond what is strictly necessary to analyze the core concepts of this thesis. This is not wasted, however, as it should give the reader a solid background from which to read the four attached articles. To help summarize the basic concepts, a core conceptual framework is established which includes the most important concepts used in the rest of the thesis. This conceptual framework is based on the literature review, the author's own interpretations and the results of the theoretical analysis from the research papers in this thesis.

**Table 1: Summary of the main concepts discussed in the above literature section; these are used as lenses in later discussion**

| Concept | Description (not a formal definition) | Related references |
|---|---|---|
| Platform | A system with a core, interfaces and peripheral apps. Can also be defined as a multi-sided environment that allows different groups of actors to interact more easily with the platform than without. | (Tiwana 2013; Gawer 2014; Gawer 2009; Hanseth and Lyytinen 2010) |
| Design- and change-flexibility | Design- and change-flexibility indicate how easily major changes in a system can be made. Major changes include fresh system setups, re-installation and testing (Gebauer and Schober 2006). In this paper, we treat design- and change-flexibility as equals, although it can be argued that they are different. | (Orlikowski 1992; Hanseth, Monteiro, and Hatling 1996; Gebauer and Schober 2006) |
| Use-flexibility | The ability to make changes to a system at the time of use, often by the users themselves. Gebauer and Schoeber (2006) include non-major changes within the scope of use-flexibility. | (Orlikowski 1992; Gebauer and Schober 2006) |
| Workaround | "[I]ntentionally using computing in ways for which it was not designed or avoiding its use | (Gasser 1986; Damsleth 2013) |

| | | |
|---|---|---|
| | and relying on an alternative means of accomplishing work" (Gasser 1986). | |
| Path dependency | This describes the situation where judgments in the past lead designers down a certain path, forcing them to take other compatible future decisions. | (Hanseth and Monteiro 1998:151; Carlile 2004) |
| Configuration | Configuration is the process of adapting a technology to the local environment, and can be both strictly technical and political. Configurability is the ability of a technology to adapt in such ways. | (Fleck 1994; Sahay, Monteiro, and Aanestad 2009) |
| Inscription | Inscriptions are programs of action that are embedded into the software by the software designer at design time. | (Monteiro and Hanseth 1996; Akrich 1992; Callon 1991:143) |
| Upstream and downstream | In a platform value chain (stream), the core platform vendor is typically upstream from the app developers and users. | (Tiwana 2013) |
| Platform governance | Control over the evolution of the platform; closely linked to platform architecture. May take the shape of 'orchestration' (or cultivation) rather than strict control. | (Tiwana 2013) |
| Micro-architecture | The internal architecture of an application running on top of a platform. Typically contrasted with the overall architecture of the platform itself. | (Tiwana 2013) |
| Resourcing and securing platform | Resourcing is the process of enhancing the scope and diversity of the platform. Securing is the process of retaining control over the platform, typically by the platform owner. | Ghazawneh and Henfridsson (2013) |

We firstly establish the concept of the *platform*; this is a system that has a reusable and *modular core* at its center, complementary functions in the periphery called *applications* and more or less stable *interfaces* between them. These interfaces are sometimes called APIs, or application programming interfaces. Platform owners are *upstream* in a *value chain*, while application vendors and users are *downstream*. The modules of the platform can be *integrated*, both between the core and complementary functions and between several applications. Integration involves using the interfaces of the platform to communicate, and closer integration allows a more *seamless* flow between different applications (Roland, Sanner, and Aanestad 2017).

Secondly, a core concept that is necessary to understand and respond to the research questions is that of *flexibility*. Flexibility has many different angles and dimensions. Central to the understanding of flexibility used here is that it is *interpretive*, i.e. that its value is subject to the interpretation of the actor. Actors within the platform value chain are not a homogeneous group, and each will have a distinct understanding of flexibility. Interpretive flexibility is assumed as an underlying premise, and was a central theme in one of the papers (Roland et al. 2013).

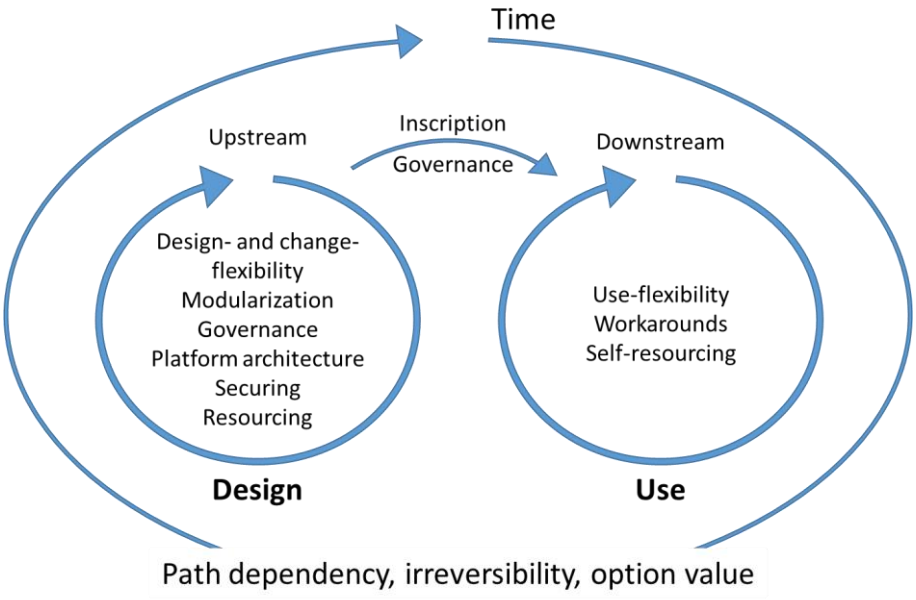


**Figure 3 shows the design and use modes, and some of the concepts of flexibility found in the literature. The figure is a visual summary of the most important concepts used in this thesis to answer the research questions. It is expanded on later by linking this figure into the discussion of the architectural insights in the attached research papers, such as the architecture figure introduced in "P for Platform" by Roland et al. (2017).**

Flexibility is applicable during the design, change, implementation and use of an artifact, and this concept is critical to the understanding of this thesis. The terms *design-flexibility* and *use-flexibility* are used to denote the flexibility required during design/change and use. The terms design and change are used interchangeably here, although it could be argued that these are different characteristics.

Design-flexibility is a property that is subject to both the *modularity* of the solution and its ability to be configured during deployment. *Configuration* here denotes the act of adapting the technology to the setting, during the deployment phase, including adapting to and integrating with existing components.

Use-flexibility is constrained by designers through *inscriptions* that may force users into certain actions; however, designers can also increase use-flexibility by inscribing *options*. Options are for example inscribed by modularization choices combined with use-time configuration, allowing users to make their own decisions of how components integrate.

When users are not given use-flexibility, they may have to resort to *workarounds*, which are unintended and sometimes undesirable actions that users have to take to fulfill their tasks. When a workflow supporting a given task flows across multiple components that communicate, these components are said to be integrated. The integration may be supportive of this task, or the user may have to resort to workarounds to achieve the task. If the level of integration between components is high, and the workflow can proceed without significant workarounds, the integrated workflow is said to be *seamless*. Use-flexibility and workarounds are related, although different, concepts. Use-flexibility is considered here as enabling users to create their own workarounds within the information system. Workarounds can, however, also be manual actions outside of the system that are forced upon the user as a result of a lack of use-flexibility (Gebauer and Schober 2006).

Flexibility also has a longitudinal dimension, described as *path dependency*. Path dependency is a lock-in based on earlier choices. Such path dependencies may be intended and apparent at design time, or may become evident as the project proceeds. *Real options* are the opposite of path dependencies, since they provide future choice rather than lock-in.

Now that these concepts have been established, based on an interpretation of the extant literature, we proceed to discuss the research context and methodology used, before considering the content of the included papers and how these papers are linked to and support the overall research questions.

# 3 Research context

## 3.1 Research context

Let us revisit the main research question:

*"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?"*

To answer this question, we need to consider cases where information systems have been implemented both in a traditional silo manner and as a platform. It would also be useful to consider different types of contexts, for example both developing and developed countries, since these are at various stages of evolution in terms of their national eHealth architectures.

This thesis is based on multiple cases from two different research contexts, the first of which encompasses low-resource developing countries and the second a highly developed and wealthy country. In each of these contexts, the drivers for platform architectures are present but take different forms.

The low-resource cases concern the development and deployment of mobile applications for DHIS2, an open source health information management (HMIS) and patient follow-up system. The high-resource context studies the integration of external applications into electronic health registers (EHRs) in Norway.

### 3.1.1 HISP and DHIS2: Information systems in a low-resource context

The main part of this research was performed as part of the HISP network. HISP is an open source research network with hubs around the world, including Norway, Vietnam, India and South Africa. Professional developers and students involved in HISP develop the open source software DHIS2, which is becoming a standard for health information management in many countries of the global south (Althauser et al. 2016; McCann 2012). There is increasing adoption of DHIS2 as a tool for collecting, managing, analyzing and acting on health data. DHIS2 has traditionally covered aggregate data collection and analysis, which concerns data that health clinics report as aggregates (for example the number of births inside the clinic) and which is analyzed across the whole nation (Shaw 2005). This thesis focuses more on data at the level of the individual patient rather than aggregate data, which adds further complexity

to the system deployments. The name-based features of DHIS2 were first implemented in India for the tracking of mothers and children (Gizaw et al. 2012), and were later followed up in several other projects (Shidende, Grisot, and Aanestad 2014).

**Figure 4 shows example paper-based systems at a local facility (left); a community health worker entering data using the DHIS Mobile application (center); and a block manager entering data into DHIS2 based on paper forms (right). These pictures were taken by the author during a field trip to Punjab and Himachal Pradesh in India, August 2011.**

Together with other software projects such as OpenMRS and OpenLMIS, DHIS2 helps to create a health IT infrastructure in developing countries with the benefits of open source software development (Braa and Sahay 2012; Althauser et al. 2016; Broyles et al. 2016; Idris 2013; Mehl and Labrique 2014). Figure 5 shows the DHIS timeline from the first launch of DHIS1 in South Africa in 1998 to 2014. The figure is adapted from the included article "P for Platform".

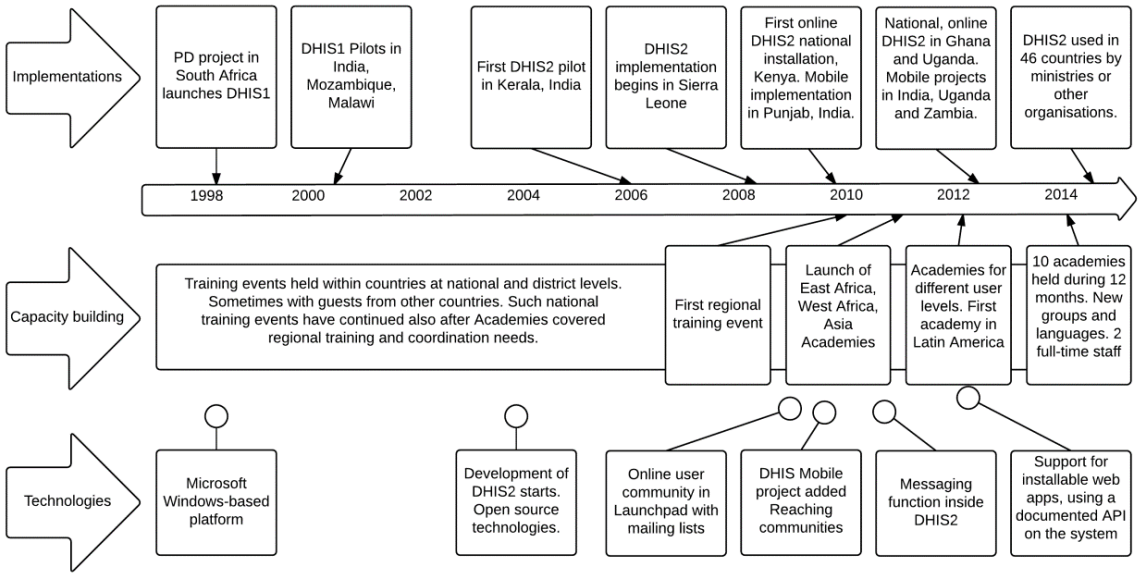


**Figure 5 shows a DHIS and DHIS2 timeline, with a focus on the evolution of implementations, capacity building and technologies. Taken from the included paper, "P for Platform" (Roland et al. 2017).**

DHIS2 software is a fundamental part of what is becoming a shared infrastructure in many countries, with a heterogeneous base of users at all levels of the healthcare hierarchy and use

cases from different independent organizations and government institutions. The loosely organized HISP movement consists of core developers and designers, who develop generic software, and implementers, who configure the DHIS2 software to the local context. Some implementers are based in a single country, while others work across regions and globally. HISP and partners run training courses around the world for implementers and advanced users called DHIS Academies. The DHIS Academies and community mailing lists are valuable tools for releasing information, discussions of requirements and support within the DHIS2 community.

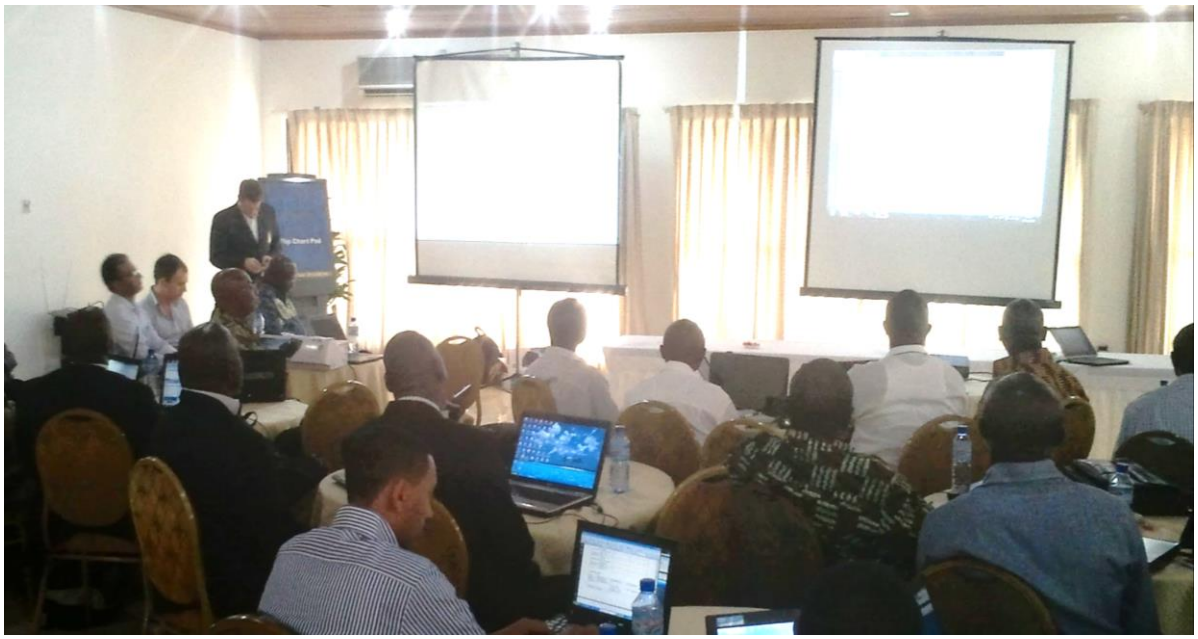

**Figure 6 shows a DHIS2 Academy in Ghana, in which the author participated in November 2011.**

The flexibility of off-the-shelf software makes the system very adaptable to the local setting, allowing local needs to be implemented without programming skills.

This thesis draws on empirical data from implementation projects mainly in Uganda, Ghana, Rwanda, Zambia, Nigeria, Malawi, and India. These countries have health services of variable quality but provide some level of both primary care services and hospital-based care. In many of these countries, depending on the local funding and governance structure, the Ministry of Health (MoH) plays a marginalized role in the health service; however, it should be considered the main body coordinating health interventions. In many countries, the health service is fragmented between MoH-owned facilities, which are often managed in a decentralized manner through district offices, faith-based treatment facilities, military health services and traditional medicine.

### 3.1.2 DHIS Mobile/MobiHealth

This thesis forms part of a larger project called MobiHealth that aims to develop mobile interfaces on top of DHIS2, thereby facilitating an expansion of the DHIS2 use cases and user groups. The project spans a large number of countries, Master's and Ph.D. projects. DHIS Mobile functionality has been added to the DHIS2 product portfolio as part of this project, although contributions have also been made to other aspects of DHIS2.

### 3.1.3 DHIS2: The birth of a platform

The District Health Information System (DHIS) began as an activist intervention in South Africa to help collect and analyze health data from black townships after democracy was introduced in 1994 (Braa and Hedberg 2002). Version 2 of this open source software (DHIS2) was developed to simplify its spread globally, and has been the basis for deployments in more than 50 countries worldwide (Øverland 2006; Manya et al. 2012; Althauser et al. 2016; Roland et al. 2017). The scaling process has created many challenges as the scope has increased along multiple dimensions: more countries and regions within countries have been added; the numbers and types of users in these countries have increased; the software is used at more levels of the healthcare hierarchy, including access outside of healthcare; and more use cases and functions have been added (Sæbø 2013; Sahay, Sæbø, and Braa 2013). As discussed in this thesis, the tremendous increase in the number of stakeholders and user domains was made possible through an architectural shift away from a silo application to an open platform, where multiple groups of developers can innovate. This transformation was not without hurdles, and is not complete.

Successful scaling was enabled by adding layers of *flexibility* to the product. Modularity and interfaces were introduced, while a simultaneous change in governance and training also allowed for more external developers who could innovate freely.

As will be explained in this thesis, the requirement for different types of flexibility can be managed by a platform that has a generic, reusable core at the center, stable interfaces to access the core and an environment that allows for innovation at the edges. This architecture allows for user participation and flexibility at the edge while promoting a reused core across domains and sites. As noted by Øverland (2006), DHIS2 was, from its early stages, always intended to be a modularized system. In fact, he writes "the DHIS 2 design is highly modularised, and the presentation layer is composed of fairly independent modules."

(Øverland 2006). For some reason, the ability of external parties to develop their own modules was held back, perhaps because of unforeseen restrictions in terms of architecture and governance. The results from the field did not all indicate that it was easy to innovate on top of DHIS2 when the required adaptations exceeded the built-in flexibility (Shidende and Mörtberg 2014; Shidende, Grisot, and Aanestad 2014; Shidende, Chawani, and Kaasbøll 2014; Manda et al. 2014).

The birth of DHIS2 as a real platform was to a large extent enabled by the core development team some years later, who together with implementers envisioned that moving DHIS2 from an application to a platform would allow a level of innovation as yet unseen in the global HIS industry. The introduction of open Application Programming Interfaces (APIs) in 2011 was promoted further by implementers, management, NGOs, Master's teaching programs, funding organizations and ministries of health.

This move towards more openness was a result of the increasing requirements from multiple user groups and domains, and the knowledge that this innovation could not be entirely centrally controlled (Paper 2 / Roland et al. 2017). The DHIS2 platform concept was described as follows in the release announcement for DHIS2 version 2.6 in December 2011:

> "The primary purpose of the Web API is to make DHIS suitable as a platform for third-party software clients and systems such as Web portals, content management systems, dashboards and mobile device clients. It provides a machine-readable API from where such clients can retrieve the desired data over the well-known HTTP protocol, without the need for any knowledge of the DHIS internal technologies and implementation." (Lars Helge Øverland, email, 2011)

The API was developed further in subsequent versions, enhanced by student projects, tested in real-life projects, helped by the development of applications by NGOs and funded by organizations that saw the potential of DHIS2 as an open platform. Standardization efforts by the architects behind DHIS2 also helped to cement the stability of the APIs. The story of DHIS2 between 2010 and 2017 thus became a story of scaling and allowing for flexibility at multiple levels by creating a platform from what had been the DHIS2 application. This strategy is supported by Tiwana (2013), who claims that the most successful platforms start off as single-sided applications, and by Hanseth and Monteiro (1998), who describe how infrastructures are typically built from an installed base of existing components.

The figure below shows a slide from a presentation given in the autumn of 2014, illustrating how the DHIS2 platform story was maturing.

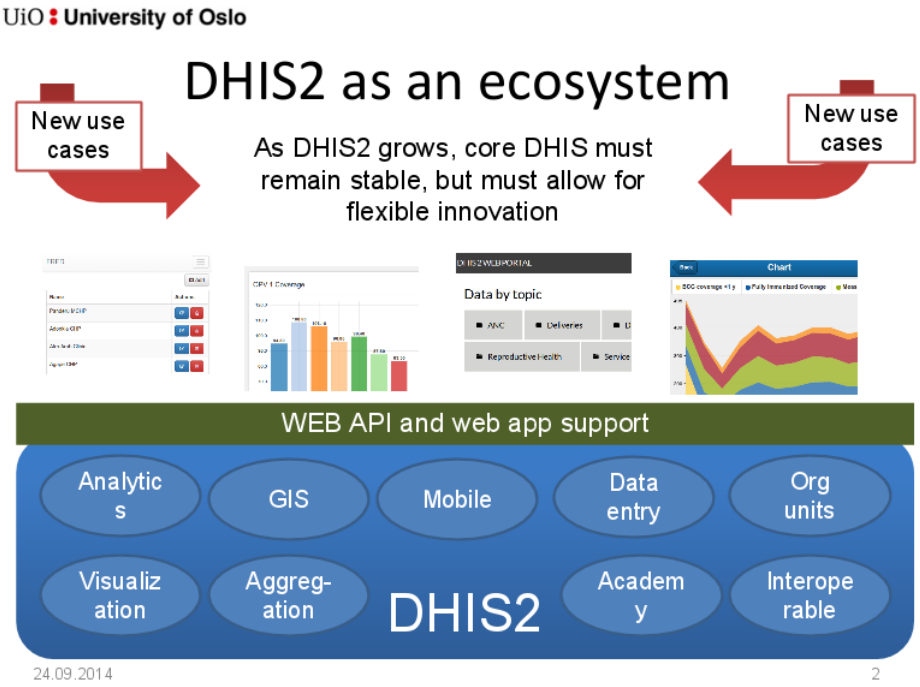


**Figure 7 shows a DHIS2 slide shown as part of a presentation about the new platform capability in DHIS2.**

### 3.1.4 Norwegian context

The last article in this thesis relates to three Norwegian cases where external applications have been integrated into Electronic Health Registers (EHRs) in hospitals and by general practitioners. Norway had an early rollout of EHRs at all levels of the health service; rollout started in 1985, and penetration reached 90% in 2000 for general practitioners and 2005 for hospitals (EPJ Monitor Årsrapport 2010 2010; EPJ Monitor Årsrapport 2008 2008; Heimly et al. 2011). In 2008, the last of the public hospitals implemented an EHR system. However, this early introduction of EHRs led to a fragmented EHR landscape, with many vendors and versions. One implication of this is that making common changes across the whole sector is difficult (EIEJ 2015). There are multiple EHR vendors for each hospital (>2), municipality (>3) and general practitioner (>3). Although there is a national program in place to consolidate EHR systems into fewer systems (Én Innbygger – Én Journal 2015), the current situation is that the rollout of any national service requires a tremendous amount of integration work with each of the EHR vendors.

The Norwegian health sector has been subject to external pressure from the industry to open up value chains and allow innovation beyond the core EHR vendors. The work done by Bygstad and Hanseth on describing the differences between lightweight and heavyweight IT in the Norwegian health sector has been significant in shaping rhetoric and plans, both in government institutions and the Norwegian health sector as a whole (Bygstad 2016; Bygstad and Hanseth 2016). Questions of how to deploy open health ecosystem platforms have been discussed in several domains, such as health registers for research and personal connected health.

As outlined in the assignment letter for 2017 from the Norwegian Ministry of Health to its Directorate of eHealth, the establishment of health information platforms is an important political objective:

> "National eHealth services contribute to reducing the complexity and meeting the needs of increased information sharing. It is a goal that these services are developed according to common national architecture principles and established standards that make information reusable and accessible. National platforms should be established for innovation and development of new services, based on open interfaces. It is our aim that national platforms should contribute to innovation and industry growth" (MoH Norway 2017; translated from Norwegian by the author).

Platform concepts and ambitions are evident in the Norwegian health sector, and it was therefore a natural step to complement this thesis with an article from this context. EHR systems in the developed world are increasingly based on large-scale integrated systems or even mega-suites from a single vendor (Fitzpatrick and Ellingsen 2013). This trend raises questions about how these systems can become platforms for external innovation, and which types of flexibility are offered and required in these systems.

Two of the national solutions aimed at health workers in Norway are ePrescription and Summary Care Records (SCR). Both these applications are integrated into EHRs using the country-specific mechanisms described in Paper 4. Hospitals in Norway currently incorporate a large number of locally adapted solutions into their EHR systems. One of these applications is a solution that offers a structured user interface documenting data on pregnancy and childbirth.

The DHIS2 and Norwegian contexts studied here represent two different contexts in which health applications have grown from silos to more open application ecosystems. These two contexts provide a strong empirical base from which to answer the research questions of this thesis.

# 4 Research method

The objective of this thesis is to better understand how generic health information platforms, a special class of health information systems, can be flexible and can adapt to a number of different user scenarios, user roles, workflows and technologies.

This chapter summarizes the research methods used in this thesis, offers some reflections on the overarching research process and clarifies the epistemological and methodological positions taken here. The chapter describes how the assumptions, interpretations, and conclusions of this work were formed by combining existing literature, theory and empirical material.

The core research for this thesis is based on action research using qualitative research methods, and draws on principles from "Networks of Action" (Braa, Monteiro, and Sahay 2004) and "Action Design Research" (ADR) (Sein et al. 2011). The ADR method provides a good fit with the research objective's focus on the artifact in a socio-technical setting, and Networks of Action adds important aspects of multi-context research in developing countries.

The final paper is, however, a separate qualitative case study without action research elements, and was included to provide a developed world context to complement the research findings (Miles and Huberman 1994; Miles, Huberman, and Saldaña 2013).

## 4.1 Research Foundation

Chua (1986) classified research into several philosophies: positivist, interpretive and critical. Ontologically, positivists assume the existence of an objective and physical social world. Positivist research focuses on testing hypotheses and predicting phenomena, and assumes that scientific inquiry is value-free. Experiments should be repeatable, and it should be straightforward for another researcher to replicate a study and confirm or falsify the hypothesis (Orlikowski and Baroudi 1991). In a 1992 study, Orlikowski and Baroudi discovered that 96.8% of all IS papers were based on a positivistic epistemology. Klein and Myers (1999) explain that research is classified as positivist if there is evidence of formal propositions, quantifiable measures of variables, hypothesis testing and drawing of inferences. They mention Yin (1994) as an example of a positivist case study description.

A positivist stance can sometimes be useful for certain types of research, but the most interesting phenomena in information systems cannot be subject to the same rigidity, objectivity and value-neutrality as natural sciences. The learning about these information systems cannot be taken entirely out of the context in which the knowledge was created. Events and outcomes will be dependent on social, cultural and other conditions that are outside the control of the researcher. Nevertheless, such non-positivist research findings can be useful, even though the cases may not be reproducible or the results repeatable.

In contrast to positivism, interpretivism adheres to a belief that truth and meaning are social products that are linked to the social actors and researchers themselves. Reality is socially constructed and is dependent on the viewer. In interpretive studies, people create and associate subjective meaning with objects as they interact with the world around them. Interpretive researchers try to understand which meaning was associated by the subjects with certain actions and artifacts (Orlikowski and Baroudi 1991).

Although they are often used together, interpretive research is not a synonym for qualitative methods (Klein and Myers 1999), and other forms of research may also use these methods. While interpretivism represents an epistemological stance or criteria for constructing and evaluating knowledge, qualitative methods represent one of the several possible methodologies for generating valid knowledge.

From an epistemological standpoint, the stance taken in this thesis is interpretivist, in the sense that I acknowledge that observations that are presented to me by informants, and that the observations that I make myself are subject to a social process of interpretation. These interpretations of reality can also be negotiated, and will change over time as circumstances change (Orlikowski and Baroudi 1991). Interview subjects will give me their interpretation of the topics being discussed, rather than a set of universal and agreed-upon facts. Through a larger number of interviews, studies of other artifacts and related research, I can as the researcher try to establish some common ground in dialog with the informants and other researchers. It is my task as the researcher to extract concepts and models that are of a more general nature through the proper use of methodology and theory. However, such results should not be seen as irrefutable and falsifiable facts that can be removed entirely from the context in which they were created, to be universally applied. I acknowledge that the resulting interpretations are some of many possible interpretations of the material. I do,

however, believe that one can make useful generalizations within some classes of information systems (Sein et al. 2011).

I believe that an interpretive stance is beneficial in highlighting the research objectives of this thesis. From a social constructionist point of view, this research represents what Orlikowski and Baroudi (1991) call *weak constructivism*, in that this research complements positivist research rather than attempting to replace it entirely.

In three of the articles, there were elements of action research, where deliberate interventions were made to learn about the artifacts and organization. I was to a large extent an *involved researcher* (Walsham 2006), with interests beyond simply understanding the meaning attached by the research subjects to their actions, and my actions were to some extent also an object of the investigation. My research therefore diverges somewhat from the most extreme interpretive research, and my methods cannot be considered a purist interpretive research approach.

Klein and Myers (1999) explain that researchers may taint a study with the preconceptions that guided the original research design, such as the initial lenses. In this Kappa I have therefore tried to be as open as possible about my theoretical and practical preconceptions, research foundation and methods, so that the reader can take this into account when interpreting my research. In what Klein and Myers (ibid) name the principle of dialogical reasoning, they recognize that prejudice is a necessary and inevitable starting point for our understanding, but that we as researchers must be aware of how our history affects us. Over the course of my Ph.D., which began in 2011, I have attended several courses and discussions that have affected how I think about information systems, and have read a substantial amount of literature concerning theory, practice and method. My reference library includes more than 500 literature references, many of which I have studied in detail. It is challenging to separate the different sources of learning and knowledge construction, but I have attempted to clarify these to the best of my ability in this thesis.

### 4.1.1 Generalizability and validity

The validation of knowledge obtained using an interpretivist stance is difficult or meaningless in terms of a natural science viewpoint, and action researchers would not typically look for characteristics such as reductionism, repeatability and refutation (Baskerville 1999). Gregor (2006) describes some different types of IS theories with different characteristics, including i)

analysis; ii) explanation; iii) prediction; iv) explanation and prediction; and v) design and action. This indicates that there certainly are ways in which one can create general theories from IS research results. Sein et al. (2011) state that as a method, action design research can be used to create general knowledge, as described in the principle of "generalized outcome". They propose areas of generalization, such as the problem instance, the solution instance and the derivation of design principles, which apply to a certain class of systems. The existence of these general outcomes is subject to adherence to other principles, including the theoretical basis of the design and the narrowing of the problem area into a certain class of problems. These principles are supported by Gregor (2006), who stresses that the definition of the level of generality or scope of a theory includes a specification of the boundaries within which it is expected to operate.

Some scholars in the IS field have shown that interpretive case studies can make generalized contributions to IS, through concepts, theories and the specific implications of rich insights (Walsham 1995; Walsham 2006:322). Lee and Baskerville (2003) suggest a framework of four types of generalizability: i) generalizing from data to description; ii) generalizing from description to theory; iii) generalizing from theory to description; and iv) generalizing from concepts to theory. They claim that the researcher may attempt to develop a theory that is generalizable within the case setting, but that "a theory may never be generalizable to a setting where it has not yet been empirically tested and confirmed."

I believe that the conclusions from this thesis can be applied more widely than to the cases in this study, but that in all cases such conclusions must be considered as guidance and learning rather than absolute truths. This thesis therefore provides some *insights* (Walsham 1995) that are based on the particular setting of these studies, but which are useful to understand and learn from when considering similar classes of systems. The methodological approach of including multiple contexts rather than single cases should help to increase the generalizability of the research (Miles, Huberman, and Saldaña 2013:101).

## 4.2 Methodology

This thesis is mainly based on action research methodology, using qualitative research methods. Specifically, it draws on principles from "Networks of Action" (Braa, Monteiro, and Sahay 2004) and "Action Design Research" (ADR) (Sein et al. 2011). The last paper is included to provide a developed world context to complement the research findings, and uses

a different method; it is a qualitative case study without action research elements (Miles and Huberman 1994; Miles, Huberman, and Saldaña 2013).

Through the DHIS community, I involved myself actively in these projects and intervened in the research setting, seeking to understand the effects of the intervention on the attitudes of the parties involved, and how these could be translated into design principles for a health information system. I was, in many meanings of the words, an involved researcher (Walsham 1995).

### 4.2.1 IS research in developing contexts: Networks of Action

There are many different varieties of action research, and it is necessary to examine what is meant by action research in this thesis. Action research is an approach that has been extensively used since the mid-twentieth century for discovering and studying change processes in organizations, and is characterized by the researchers taking an active and engaged role in implementing and studying change rather than taking a bystander and observational role (Baskerville 1999).

Baskerville (1999) describes classic action research projects as iterative, reflective and linear. Davison, Martinson, and Kock (2004) go to great lengths in describing their version of action research as a rigorous and structured process, with a planned approach containing clear phases.

Carrying out action research simultaneously in many low-resource countries requires and allows for a different research approach. These projects are far from linear, and often lack clear phases and stages. The action research projects surrounding the development and deployment of DHIS2 have therefore historically been more pragmatic and flexible than many other forms of more structured action research methodologies. As noted in the article entitled “Networks of Actions” (Braa, Monteiro, and Sahay 2004), “Rather than clear stages or phases, HISP is characterized by a multiplicity and simultaneity of ongoing processes which take on different forms at various stages, and there is rarely ever a clear start or end.” The ‘network of action’ research approach outlines how one can base research on experience from multiple countries and locations (Braa, Monteiro, and Sahay 2004). This multi-site and longitudinal view allows us to follow the “biography of an artifact” (Pollock, Williams, and Procter 2003) across parallel projects that are deployed in widely different contexts.

The DHIS2 setting is unique and rich in research potential, but does not fit snugly into the rigid forms of action research. Figure 8 is an adaptation from the included article "P for Platform" and can be used to illustrate how DHIS2 research projects differ from traditional case studies and benefit from multiple parallel projects.

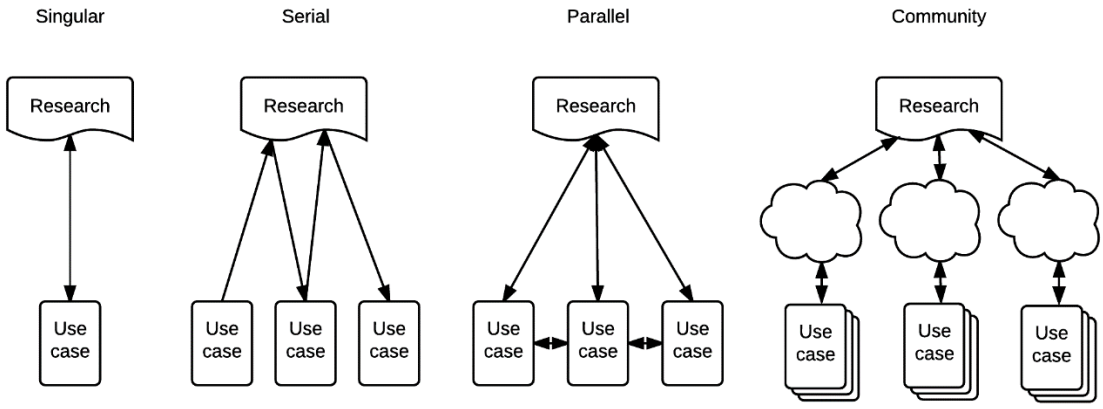


**Figure 8 shows several forms of projects with different learning involvement. The diagram is adapted from the included Paper 2, "P for Platform" (Roland et al. 2017).**

A classic action research case is constrained to a single project, and the researcher is highly involved in this single use case (the singular model). If the same researcher or research community is involved in multiple projects in series (the serial model), these projects can provide additional learning and qualification. It could also be claimed that the research results might be less context-dependent in serial research, as the learning has come from multiple contexts (Miles, Huberman, and Saldaña 2013:101). The singular and serial research approaches were common in early DHIS deployments, but the availability of many parallel projects has now opened up new research opportunities (the parallel and community models). I have been able to carry out many parallel research projects, as well as groups of projects that are represented by larger user communities. Such multi-site research would not fit all types of research questions; however, this parallel and community research model seems to fit the research objectives of this thesis particularly well. One drawback of parallel and community research projects may be that the researcher is unable to delve as deeply into the context as in single and serial projects. This limitation should be kept in mind when considering the research results. In this thesis, this concern is balanced by a combination of projects; in some of these, the researchers are more embedded, and in others there is a less deep involvement.

The broad array of DHIS projects in which I have been involved have helped to provide useful responses to the research questions. They gave a more comprehensive and cross-contextual biography of the technology used, more so than single case studies that are naturally limited in terms of context and user groups (Pollock and Williams 2010; Hyysalo 2010; Pollock, Williams, and Procter 2003; Miles, Huberman, and Saldaña 2013:101).

### 4.2.2 Action design research and ensemble artifact focus

The objective of this thesis is to better understand how generic health information platforms, a special class of health information systems, can be flexible and can adapt to a number of different user scenarios, user roles, workflows, and technologies.

In our case, the main focus is on the health information platform as an *ensemble artifact*, the combination of the physical artifact and the socio-technical environment in which it exists (Orlikowski and Iacono 2001). The chosen research method must support the research objective and its focus on the ensemble artifact.

A considerable proportion of all IS research elaborates on organizational aspects of information systems rather than the IT artifact itself (Orlikowski and Iacono 2001). Similarly, most action research methods focus on the effects of change in the organization and give the technical artifact a bystander role (Baskerville and Myers 2004; Papas, O'Keefe, and Seltsikas 2011; Baskerville 1999; Davison, Martinsons, and Kock 2004).

The *action design research* (ADR) method is a response to the lack of artifact focus in action research and the missing focus on organizational factors in design research. ADR therefore integrates design research (DR) and action research (AR) to create a method that focuses on the development of a theory-ingrained, innovative ensemble artifact (Sein et al. 2011). Sein et al. (2011) conclude that the contributions from an ADR project should be in the form of design principles, that these principles should address a class of problems and systems, and that the outcomes should be innovative.

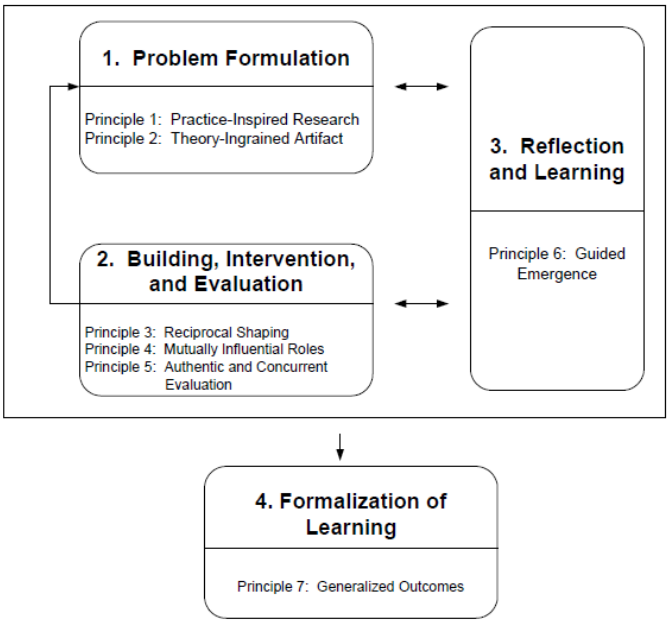


Figure 9 shows the ADR cycles (Sein et al. 2011).

In a similar way to other action research methods, ADR uses a cyclical model, where learning is achieved iteratively. The actual development cycle is the "Building, Intervention and Evaluation" (BIE) phase, which is repeated several times as the closely linked project team develops the artifact, tests it and learns. As the BIE cycles progress, the project team increasingly involves the practitioners and end users in the design and use of the artifact. When the BIE cycles are complete, the researchers formalize the learning and generalize the outcomes. The ADR method specifies seven guiding principles: practice-inspired research; theory-ingrained artifact; reciprocal shaping; mutually influential roles; authentic and concurrent evaluation; guided emergence; and generalized outcomes (Sein et al. 2011). These principles were influential in shaping the research design and have guided the research process of this study.

### 4.2.3 ADR projects in parallel or as community projects

When considering ADR in the context of the different configurations of research projects in Figure 8, it is worth noting that ADR, as described in the original article (Sein et al. 2011), relates mainly to singular research projects. In this thesis, the ADR method is extended to apply to the parallel and community research approaches.

Projects such as DHIS2 are characterized by a network of numerous different and interdependent interventions, rather than unique and independent projects (Braa, Monteiro,

and Sahay 2004). Methods such as ADR utilize a series of carefully planned and executed cycles of activities, which are much more structured in approach than is possible for DHIS2 projects in developing countries (Davison, Martinsons, and Kock 2004; Sein et al. 2011; Braa, Monteiro, and Sahay 2004).

As discussed above, classic ADR cases as described by Sein et al. (2011) relate to a single information system where the project controls the software features, rather than a network of distributed cases such as DHIS2. Classic ADR does not therefore specifically consider the learning effect across a network of projects, as is the case in DHIS. In the DHIS projects, there is rich interaction between different BIE processes spanning various projects. It would therefore be possible to consider a set of interdependent BIE cycles that span various projects in a distributed HIS program. This interaction between the different projects takes place both i) after each BIE cycle, for example by participating in formalized cross-project workshops; and ii) during the BIE cycles, for instance by sharing researchers, developers and software across several projects.

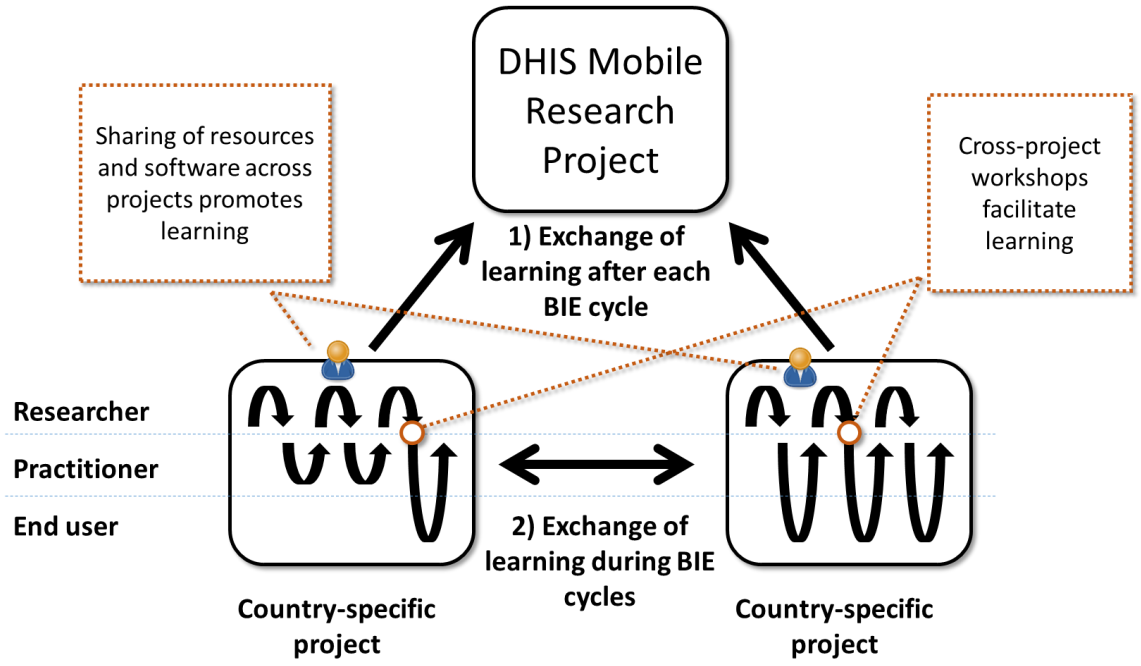


**Figure 10 shows two parallel ADR projects, with an exchange of learning both i) after each BIE cycle and ii) during the BIE cycles. The learning is also embedded into the shared, generic software resources, and is exchanged continuously due to the open source nature of the project. This thesis forms part of an overall research project called MobiHealth, which developed the DHIS Mobile product portfolio.**

Each project instance can benefit from knowledge sharing due to the learning and embedding of design principles across sites. As seen in Paper 1 (Roland et al. 2013), this sharing may also be a drawback, as it may limit the flexibility for each unique project. The choice of ADR

as a research method fits the research objective reasonably well; inspired by Networks of Action, I apply the method in this thesis in a cross-project manner that gives some additional benefits.

The fourth paper does not use action research, but is a non-interventionist case study using qualitative methods. All observations and interviews for this article were conducted late in the lifetime of the actual software projects, and it was not feasible to carry out the study as action research. Despite the use of a different method, I believe that this additional project helps to provide further substance in answering the research objective, as it applies to a non-DHIS2 context.

## 4.3 Research design

The first part of the study, resulting in three papers, is based on several implementation projects involving mobile applications for DHIS2. These were action research projects in India, Uganda, Zambia, Rwanda, and Nigeria. In all of these countries, we were involved in multiple projects. Some of the projects were multi-cycle action research projects, while others only ran for a single cycle or were not even completed. The long-term history of DHIS2 deployments and the number of countries in which the software is deployed allow for a comprehensive 'biography of an artifact' approach (Pollock, Williams, and Procter 2003). We were able to evaluate the evolution of the software artifact across many different projects and contexts, providing this thesis with a level of rigor that is hard to accomplish through single case studies. The DHIS2 research cases therefore provide a suitable empirical background for discussing how *generic health information platforms can be flexible and adapt to a number of different user scenarios, user roles, workflows, and technologies*.

Certain specific cases that highlight the research questions particularly well have been chosen as focus points for the articles; however, the additional material available to us as researchers gave us the opportunity to cross-check, substantiate and evaluate how differences in context affect the evolution of the artifact. In retrospect, some important learning about Health Information Platforms turned out to be about how seemingly non-related projects reused the software in unexpected ways, sometimes disrupting and sometimes helping other projects. These conflicts and opportunities allowed us to consider where and how flexibility should be applied in the platform architecture.

The progression throughout the thesis starts by illustrating the needs for heterogeneous user requirements, and then considering how these requirements and related user participation affect the architecture of the health information system. The thesis then goes on to discuss more specific parts of the platform architecture and the ways in which platform architectures can be used for health information system deployments.

### 4.3.1 Studying platformization by examining many parallel projects

This study has two levels, and it is important to explain why both are essential in achieving the research objectives. Consider Figure 10 as an illustration of both the multiple underlying projects and the overarching research goal of considering the platformization of health information systems. We have already argued that platformization cannot be studied effectively by examining a single project since the strongest driver for platformization is the way in which this architecture model can support multiple contexts on a single architecture. As described above, the DHIS2 environment gives us the opportunity to consider many such cases together. However, a keen reader might ask why it is necessary to delve so deeply into each single project and to use action research and the extensive qualitative methods described in this chapter to explain platformization. Why is it important to interview health workers in the field to understand how platforms are formed from health information systems? The reason why we need this level of detail in our research is that it is the individual contexts that drive the need for the platform. Without a deeper understanding of what these different needs are, how needs are catered for by the architecture, and how the end-users react to being 'forced' onto a generic platform, we cannot understand which types of flexibility are required in health information platforms. This study therefore exists at two levels. One is the level of each project, where we apply action research to solve the individual needs of the health organizations using the software; the other is a consideration of the overall development process and architecture, including an analysis of how the common architecture is shaped to fit requirements from multiple projects. We must keep these two levels in mind when we consider the data collection methods, for example. Interviews of health workers in the field are important; however, equally important to the research objective are observations and interviews pertaining to how the architecture evolves in response to the requirements of these health workers. Both of these levels use very similar data collection and analysis methods, and these are discussed in the data collection and analysis section of this thesis.

### 4.3.2 How is the fourth paper different?

The first three papers of this thesis are related to DHIS2 projects and use relatively similar research methodologies, while the fourth paper is different. This last paper differs in terms of geography, context, software product and method, and its inclusion therefore warrants some explanation. To some extent, it was included for opportunistic and personal reasons. After studying health information systems and a particular platform solution in developing countries, I took the opportunity to temporarily suspend my Ph.D. and to work with Norwegian healthcare and national health architectures for some years. When returning to finish my Ph.D., I had the choice of whether to study the platform phenomena in DHIS2 in more detail, or to leverage my recent learning from the Norwegian health sector and find a project that could contribute to the completeness of my Ph.D. from a different perspective. Expanding the study of the platform evolution in DHIS2 was certainly a feasible route, but I found great personal motivation in considering whether the insights regarding the architectural platform found in the earlier DHIS2 projects were also applicable to the Norwegian health sector. The paper contributes to an ongoing debate within the Norwegian health sector and internationally about how incumbent EHR systems should be broken up and external vendors allowed to contribute with innovative solutions. From the work on "P for Platform", I also felt that it was necessary to look further into the drawbacks of flexibility and platform architectures, in order to balance the story of how platform architectures can be applicable to health.

Given the late stage of my Ph.D., it was difficult to find an action research case that could be completed in time, and I therefore chose to conduct a retrospective, qualitative case study of three projects where external applications had been integrated into a number of different EHRs. Although the inclusion of this paper had opportunistic reasons, I believe it increases the relevance of the thesis and shows that insights regarding the platform can be analyzed across multiple contexts. As can be seen from the discussion (Section 6.5), the findings of Paper 2 ("P for Platform") can be used to further expand on the findings of the last paper. The discussion of theoretical concepts such as flexibility across the two contexts also provides a critical component of the overall contributions of this thesis.

On the surface, the inclusion of this fourth paper complicates the methodology of the thesis as a whole, as it introduces a new research method and a case from a very different setting. There are, however, some key similarities between the research methodologies. The research

for the last paper used many of the same data collection and data analysis methods as the first three papers, although not in an action research setting. Data collection was based on semi-structured interviews, observations, studying the integrated artifacts and studying project documentation, as was the case for the first three papers. Data analysis was guided by the "Ladder of analytical abstraction" (Carney 1990) and the use of multiple levels of summary notes and data displays, as explained in Section 4.6, similarly to the analysis conducted for the DHIS2-related projects. If anything, the last paper was conducted in a more rigorous, planned and structured manner than the three earlier papers, in view of my learning from the earlier research and my maturity as a researcher. To emphasize these similarities, I have chosen to present a joint description in this Kappa of the data collection and analysis for all four papers, although it could be argued that this could have been split to give the specific methodology for each paper. In reality, covering each paper's methodology separately would have been repetitive, since the methods were similar.

There is one fundamental difference between the research approaches used in these papers. The DHIS2 research was based on action research principles, where we as researchers were involved team members and were instrumental in guiding the project outcomes throughout the project period. Action research offers the benefits of access and insight, but also gives rise to complications in terms of creating sufficient distance in the data analysis phase. In the last paper, the research was conducted after the main components had been completed, and we took an external view of the projects. We were not active initiators of or participants in the project, but based our findings on retrospective interviews and observations of the finished product and documentation. Hence, although the data collection and analysis were similar, the last project lacked the observations during the project and our close involvement in the project outcome. This difference made it simpler to create the necessary distance during the data analysis phase of the fourth paper, but means that we may also have missed some details that could have been discovered during an action research project. In particular, data that evolved during the project period may have disappeared, since we interviewed informants after the projects had finished.

## 4.4 Access to field work and other empirical sources

The empirical material of this thesis comes from five main activities:

1. Work related to specific DHIS2 mobile implementations in various countries. This work was done through on-site visits, conference calls, and workshops. The DHIS Mobile team had weekly phone meetings where we discussed projects in different countries, support was given to in-country representatives, bugs and features were discussed, and requirements were gathered. We encouraged project representatives to participate in these meetings so that we could support them as a team, and so that countries could learn from each other. The papers are primarily based on empirical data from these activities.
2. General work on enhancing the open source DHIS2 mobile and DHIS2 tracker generic product portfolio. To fulfill this coordination role, I interacted with architects and developers on a daily basis. This work also included participation in DHIS2 Academies and specific workshops for DHIS2 mobile development, with a focus on evolving the DHIS2 product into a platform that enabled new users and usage domains through a rich set of mobile technologies.
3. My teaching role in one of the largest Master's-level courses in the department of informatics, where more than 100 students every year were taught to create DHIS2-based applications using the open APIs of the platform.
4. I co-supervised five Master students who were working on the same (or similar) cases as I was, participating in joint field trips and discussions about the empirical material (Korvald 2013; Berntsen 2015; Gammersvik 2015; Halvorsen 2015; Røsok 2015). Their research was also relevant as input to this research.
5. My study of the Norwegian eHealth sector and, in particular, the detailed cases of some of the applications that have been integrated into Norwegian EHR systems. My full-time employment as an enterprise architect at the Norwegian Directorate of eHealth between 2015 and 2017 also gave me a good background for understanding the challenges that the Norwegian health sector is facing.

Access to a large number of country implementation projects was implicit based on my role as the lead coordinator for DHIS2 mobile applications. Davison et al. (2004) describe the importance of a researcher-client agreement for conducting action research; my in-the-field research was conducted mainly as part of larger implementation projects, where our permission to do research was included in the implementation agreements.

Klein and Myers (1999) urge researchers to reflect on the relationship between the researcher and the subjects, and to acknowledge that the data are socially constructed in a manner that also involves the researcher. My role as a mobile coordinator may have had both positive and negative consequences. I was given access to interview informants and was able to study a large number of projects. The massive interest in using mobile technologies in DHIS2-related projects allowed us to interact with many actors who represented vastly different use cases and contexts. Our involvement varied from short discussions in order to clarify certain issues, to deep engagement in requirement discussions with multiple field trips that provided a rich set of empirical material. Being a DHIS2 Mobile expert helped in opening many such doors, and access to enough projects was never a problem.

However, the informants' reactions were affected by their perception of my role as an important stakeholder in the software development. During the various field-trips, my role was introduced in different ways, and there were times when this introduction obviously shaped the rest of the conversation. It was, therefore, challenging to grasp the full extent of the opinions of the interview subjects. In several cases, this initial hurdle to communication was overcome by getting to know the informants better. As time passed, they let their guard down and opened up about the challenges and possibilities they saw with DHIS2 deployments. I learnt that deep engagement in projects, curiosity and always digging deeper was important in obtaining more complete stories.

Much of the interaction was done using email and internet-based phone conferences, but I also got the opportunity to visit the countries on multiple occasions. Table 2 summarizes these field trips abroad during my Ph.D. In addition to these, I attended a number of workshops and DHIS Academies that were held in Norway. The field trips were typically one to two weeks long.

**Table 2 shows the field trips made outside Norway. Much of the data collection was done from Norway.**

| Countries | Time |
|---|---|
| India (Punjab and Himachal Pradesh) | Fall 2011 |
| Ghana (DHIS Academy) | Winter 2011 |
| Uganda (First mobile tracker project setup) | Spring 2012 |
| Kenya (DHIS Academy and mobile workshop) | Summer 2012 |
| Vietnam (DHIS Mobile Workshop) | Fall 2012 |
| Uganda and Rwanda | Fall 2012 |
| Uganda | Winter 2013 |
| Uganda and Zambia | Spring 2013 |
| Uganda | Fall 2013 |

The largest quantity of case material in this thesis is from Uganda. There I studied various mobile projects and worked extensively with a mobile-enabled system for tracking mothers and children through an integrated health program. During the field trips, I visited many villages, health clinics, and hospitals, interviewing health staff and community health workers. Meetings and interviews also included higher-level program officials and representatives from the Ministry of Health, CDC, USAID, US embassy, funding organizations and implementing partners. During my attendance at DHIS Academies and other training events, I could also interview implementers from a large number of other countries.

**Figure 11 shows a volunteer being tested for malaria, as part of a malaria testing program in which DHIS2 mobile applications are used to coordinate malaria eradication efforts in Zambia. The photo is from a field trip in 2013, where we attended a training event for community health workers.**

### 4.4.1 Mobile coordination role

In addition to field trips, I coordinated the mobile development work for DHIS2 for two years through weekly coordination calls, workshops, and other project management activities. This responsibility gave me a rich insight into the development process and challenges at the global level, and provided an understanding of HIS development in other countries, in particular India, Rwanda, Uganda, Tanzania, Malawi, Zambia, and Nigeria. The learning from this coordination work formed a fundamental part of the empirical material.

**Figure 12 shows mobile phones belonging to health workers at a health clinic in Uganda. The photograph was taken during a field trip in September 2012.**

### 4.4.2 Teaching experience and tutoring of students creating DHIS2 apps

As part of the teaching requirement of the Ph.D. role, I organized and was the main lecturer for the "Open Source Development" course at the University of Oslo in 2013 and 2014. This was one of the largest Master's-level courses in the Department of Informatics, with approximately 100 Master's students completing it each year. Being a lecturer was rewarding in itself, and the course also gave me a solid background in the technical aspects of DHIS2. I held these courses during the innovative years, during which DHIS2 evolved from a relatively closed application into an open platform, and the students' projects helped to grow and mature the platform APIs and documentation. In the first year, I co-lectured with the Ph.D. student who had implemented the DHIS2 app store, and whose contributions to the course and the platform also were extremely valuable.

The students took part in group work, where with our guidance they created web applications utilizing the DHIS2 APIs. More than 30 web apps running on top of the DHIS2 platform were created each year. The breadth and innovativeness of the applications exemplified how powerful the DHIS2 system was becoming through opening up the APIs. Several of these apps were subsequently deployed in DHIS2 countries or otherwise became foundations for

further work, and some of the top students were recruited as developers for DHIS2. This teaching experience thus gave me a practical look at both the difficulties and possibilities of having a large group of developers creating applications on top of a platform that was in the making. This teaching work is not directly included in any of the articles, but has been an essential empirical foundation for me in studying the overall research objectives and writing my thesis. I use some observations from this teaching experience when discussing the concepts of delegating flexibility in Section 6.3.6, to illustrate how this experience was important in understanding how different actors in the platform value chain relate to flexibility. Listing experience from teaching as an empirical data source is probably uncommon; however, it was relevant in this thesis as the teaching and guidance of students was very closely related to the platformization of DHIS2.

## 4.5 Data Collection

The informants in my research were mainly the project members and implementers of various projects using DHIS2 applications: developers and architects working on DHIS2; health workers at hospitals, clinics, and villages; health managers; stakeholders within state and health programs; and mobile operators. I did not interview patients directly, and did not analyze any patient-sensitive data, but rather analyzed the collection and analysis processes concerning those data. In some cases, patients were observed in the course of observing the practice, but only in public areas such as waiting rooms. All interviews were anonymized.

Although his positivistic stance should be taken into account, one can consider how Yin (2013) explains that empirical material for case studies may come from i) documents (letters, agendas, progress reports); ii) archival records (service records, organizational charts, budgets etc.); ii) interviews (typically open-ended but also focused in structure, and surveys are possible); iv) direct observations (formal or casual; multiple observers are useful); v) participant observation (assuming a role in the situation and getting an inside view of events); and vi) physical artifacts. Walsham (1995) claims that interviews should be the primary data source for interpretive studies, as this gives the researcher more direct access to the interpretations of the participants. I believe that interviews cannot stand alone, and as a researcher I value the use of a variety of data sources.

Data collection for this thesis was based on a mixture of the following: semi-structured interviews; ad-hoc discussions; extensive observations during training sessions, workshops,

clinic visits and meetings; study of official documents; and reviews of health information tools.

### 4.5.1 Interviews

The interviews were important sources of empirical material. In some of these interviews, the interaction between the researcher and subject was a deliberate and planned event, and was structured as an interview. Often, however, the data collection was more ad-hoc, for example sitting at a café in a group discussing the implementation, with an ever-present notebook for jotting down important concepts and comments. During field trips, I used all my spare time to ask questions and dig into the case, for example while sitting on the bus with the informants or at lunch. Interviews were conducted both as closed sessions, with only one informant and the researcher present, and as group sessions with multiple participants.

Klein and Myers (1999) emphasize that researchers must be sensitive "to possible differences in interpretations among the participants." It was important in our research design and data collection to deliberately include interviewees that we thought may have conflicting interpretations. In some cases, such informants were interviewed separately so that they did not affect each other during the interviews. In other cases, a discussion between informants with different opinions gave better results, as long as they knew each other well and dared to speak up. This interviewee selection principle was critical, for example, in discovering the extent of diverging rationalities amongst different actors in the first paper, a result that was not expected during the initial research design. These conflicts in rationality and their manifestation in the artifact became apparent through the principle of guided emergence (Sein et al. 2011).

On my more recent field trips, I used a pen with an integrated microphone, which links audio recordings to notes taken on paper. This technology allowed me to make a note on paper during the interview when an interesting comment was made, or a concept emerged or was discussed. Later, I could then access that particular event in the recording directly, rather than listening to the entire session. I always listened to the full recordings after interviews, but the pen allowed me to be more focused on the analysis process and to quickly go back to check if this was what the informant had said. This tool therefore allowed me to zoom in and out of the notes quickly during the analysis phase, making it easier to condense the interviews into common concepts and themes. After the field trips, some of the interviews were transcribed,

although the pen with the integrated microphone seemed to be a more efficient way to extract data, as it helped in tagging important parts of the interview during the interview, and made it easier to return to selected parts of the recording and hear the actual words from the informant directly.

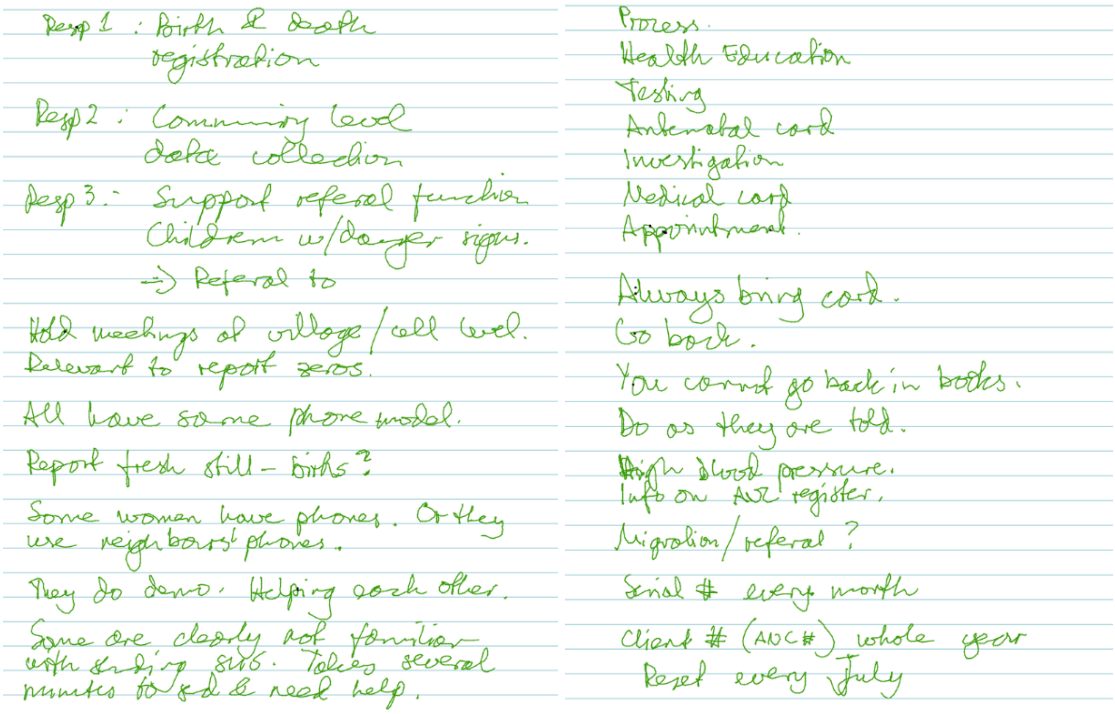


**Figure 13 shows two pages from the notes taken during a demo session of DHIS2 Mobile in Uganda on 26th February 2013. A pen with an integrated microphone was used during interviews. The picture is a screenshot from the PC screen. Clicking a word will play the audio recorded at the time the word was written.**

### 4.5.2 Observations

Data collection also included observations of real-life patient flow at hospitals and clinics. In some cases, we arrived at busy clinics and were able to view the patient flow and use of information artifacts firsthand. However, arriving early typically made it difficult to interview staff, as they were busy taking care of their patients. On some days, we arrived later in the afternoon when the sites were less busy, and the local staff explained to us how the patients and their information were handled.

This mixture of our own observations and an explanation of the process in interviews was necessary for us to understand the artifacts and the reasoning behind them. When we only observed without an explanation, we sometimes misunderstood the background. For example, the staff explained how they used the different registries; however, browsing through them to see which data the health workers actually entered for patients allowed us to

understand what the registries were used for, which challenges these artifacts posed and which tasks they solved well.

### 4.5.3 Training sessions and focus groups

At many of the sites we attended, we had the opportunity to talk to a larger number of people, but setting up individual interviews was not practical. Instead, we organized joint training sessions or focus groups with the health workers who would be using the system. In these sessions, staff members were presented with the intentions of the project and given a demonstration of the current state of the product. The subsequent discussion was very useful in shaping the next version of the software, and also helped us uncover new requirements that were not obvious to the central project team. These larger group sessions allowed us to gather a broader set of opinions more effectively (Lazar, Feng, and Hochheiser 2017). At some of these sites, we invited individuals to be interviewed directly after the focus group.

**Figure 14 shows an interview situation from a multi-day training session.**

Participation in multi-day training sessions turned out to be an extremely valuable source of empirical material. The training setting allowed us to ‘zoom out’ and to consider the whole group, identify concepts, opinions and discussion topics, and then to select certain people who we thought might be able to represent different views. We used the breaks in the program to move around the room, conducting interviews with these specific informants. One example of this was in Zambia, where we attended a training event for voluntary health workers participating in a program aiming to eradicate malaria from certain regions in the country. As a group of four researchers from the DHIS mobile project, we followed the

health workers through both the theoretical training program and the practical sessions in which they carried out malaria testing in rural villages. The training lasted for several days, and we were able to use the different social settings and spaces in the program to interview the same people about their perspectives on using mobile technologies in general and DHIS2 in particular. In parallel, we discussed the findings within the research team and were able to do some on-site analysis before returning to gather more empirical material. I participated in similar sessions in Zambia, India, Uganda, and Rwanda. The DHIS Academies I attended in Oslo, Ghana, and Kenya also represented similar opportunities for gathering data from both groups and individuals.

### 4.5.4 Design sessions, workshops, and project meetings

As part of the development process, we set up design sessions during which the internal project members acted out usage scenarios. These scenarios included, for example, receiving a pregnant mother at a health clinic and going through the proposed procedure for treating her and registering data about her at the clinic. The project members were assigned roles such as "pregnant woman," "health worker at registry desk" and "midwife," and we tested the system with these user roles in mind.


**Figure 15 shows a project meeting in September 2012; the project team met to summarize requirements for antenatal follow-up in Uganda.**

We also organized some workshops where implementers from different projects gathered to discuss features and their experiences across these projects. These sessions made it possible to observe the developers and architects, and the ways in which they responded to the different user scenarios and contexts. These cross-project meetings varied from weekly phone conferences, where team members from the projects raised concerns and discussed experiences, to joint physical workshops lasting several days or a week.

### 4.5.5 Taking pictures and recording video

We brought cameras on the field trips and used these as often as possible to document the environment, whenever this was suitable and we were given permission to do so.


**Figure 16 shows the bookshelf outside a delivery ward where the paper-based journals were stored. The second picture shows the storage room (previously a laboratory) where the papers were kept after the mother had delivered her baby.**

Many of the training and demonstration sessions were video-recorded, and I was able to review these, take notes and summarize them more effectively after the event. One example of this was the DHIS Academy in Kenya, where we video-recorded a two-hour discussion about DHIS Mobile. The session involved participants from numerous countries who were interested in implementing mobile HIS tools. The video was not only useful for research purposes as empirical material but was also shared with developers in Vietnam who were not present, giving them the possibility to hear firsthand which features were requested by the

implementers. Another example was the video-recording of a DHIS2 training session for hospital doctors, where some of the more vocal participants voiced concerns about the contrast between using health information systems for collecting statistics and for treating patients, corresponding to the rationality conflict between primary and secondary usage discussed in the first paper. These videos provided a richer documentation of these concerns than the limited paper notes of a researcher.

### 4.5.6 Document analysis

The data collection also included a review of the existing project literature. The amount of existing literature varied between projects. For the DHIS2 cases, this included both national documents and WHO documents, for example guidelines for health programs, guides involving mother-and-child cards, and other literature. For the Norwegian case, I reviewed an extensive amount of internal project documentation as well as reports from the usability studies carried out on the implementations.

### 4.5.7 Studying the existing tools: artifact analysis

Studying the existing information artifacts, the documents describing their use and the explanations given by health workers about these tools was a beneficial way of understanding the legacy information processes in the health service. On my first field trip to India in August 2011, one of my strongest impressions was the number of paper-based registers that the health workers had to fill in, often with duplication, and the time it took them to manage these books. We studied the existing paper-based tools, how these books were passed around in the clinic, how the data in the books were reported to authorities, the workaround tools that health workers maintained unofficially, the tools that had been introduced but had failed, and ideas for new tools; this gave us an impression of what both health workers and program officials felt was important.


**Figure 17 shows pictures of the different types of health registers and forms that the health workers had to maintain at a health facility in Himachal Pradesh, India.**

During my field trips, I was fortunate enough to travel with people who were very knowledgeable about the registers and the intended processes. For example, I participated in an annual reporting quality review in Rwanda; this was an activity where specialists from the Ministry of Health traveled to rural clinics to review their use of the registries, checking for anomalies between the actual data in the registries and the reported data. When issues were found, staff members were given training and constructive explanations of how to use the tools. Listening to the discussions during these reviews gave rich insight into the different requirements that the various informants had for a health information system.

This data collection also included data about how DHIS2 has developed as a tool, by reviewing release notes, source code, and documentation. This data collection was necessary to identify how DHIS2 was responding to the growing number of use cases and contexts.

### 4.5.8 Looking for inconsistencies in the data

Klein and Myers (1999) describe "the principle of suspicion," whereby the researcher should be sensitive to biases or deliberate distortions in the interpretations. Examples of this could be found in our repeated visits to health clinics; when we returned the following year, it became evident that we had not completely understood what the full process was. In one case, a piece of paper had been presented as something different from what it actually was (a receipt had become the mother's health card), and in another case, it was not mentioned that the tracking process for was different for mothers who were HIV positive and negative. Either the stories

that we had been told were different from what we later discovered or, equally probably, we had simply misunderstood the initial account of the details, and we understood more as we learned more about the context.

Let us consider the process of tracking mothers with or without HIV as an example of this principle of suspicion. On our first visit to a certain health clinic, we discussed the need to track women throughout their pregnancies, over multiple antenatal care visits. The health workers seemed to have little interest in such tracking, and did not see why this was important. Their paper-based artifacts also seemed to lack the possibility of implementing a feasible tracking scheme. A mother could equally well visit another clinic for the next visit, and this was not seen as a problem; it was the mother's concern if she became *lost to follow-up*. A year later we visited exactly the same clinic, and were presented with a completely different process. They told us about how diligently they tracked mothers throughout their pregnancies, how they used calendars to track appointments, and how they sent voluntary health workers out to those mothers who did not show up for appointments. We were confused. This situation could have been understood as them lying to us about the status quo, or as the process having changed dramatically over the last year, but the reason for this contradiction was something else entirely. In the first year, they had described mothers without HIV, and in the next, mothers with HIV; the latter group of patients were always followed up, using a completely different regime and process. Our repeated visits to the same sites, the patience of the health staff who explained the process repeatedly and our persistence in trying to understand helped to unveil these important details. The scope for misinterpretation of such interview data is tremendous, and can have dramatic consequences for the story that is told. It is therefore essential for the researcher to be suspicious not only of their data, but also of their own understanding of the data. The story is not necessarily wrong because the informant gave the wrong information, but because the researcher did not understand the background of the informant's statements. The researcher (myself, in this case) did not understand until later that whether or not the mother was HIV positive had a huge effect on the process that was followed. This was obvious to the informant, but not to me, the researcher.

I have interacted with some of the same informants multiple times over several years. In the case of the developers and architects working on DHIS2, we talked almost daily throughout the research period. During field trips, I took notes, recorded audio from interviews, took

pictures, and recorded video from training sessions and interviews while participating actively in meetings, workshops and training sessions. Combining these empirical sources helped to uncover inconsistencies and offered multiple channels for checking interpretations. A considerable amount of communication with each project took place via email and Skype calls, and this data also provided valuable material and the possibility of re-checking open questions.

Each day of the field trips concluded with a session where I reviewed and organized my notes, sometimes transcribing important parts of audio interviews, condensing data, writing summary emails to other project members and considering which topics were crucial to cover the next day. This daily summary was also a useful method for finding inconsistencies that could be addressed with the relevant people the following day.

## 4.6 Data analysis

In action design research, the project contributes learning in the form of theory-ingrained design principles. The ADR team learns from testing and evolves the software during the stage of the project called "Building, Intervention, and Evaluation"; this stage is repeated multiple times and comprises the increasingly deep involvement of the organization as part of the development and testing of the artifact. The final stage, "Formalization of Learning," focuses on casting the problem-instance into a class of problems and formalizing learning through a description of design principles. With further reflection, these design principles can contribute to the original theories used during the ADR project.

Data analysis in ADR is not a strongly delineated and serial process, but rather an iterative process where multiple cycles of software development and testing followed by reflection lead to formalized learning (Sein et al. 2011). The ADR method does not describe in detail which analytic process should form the foundation for this learning, other than the ADR principles and stages.

In this thesis, I used a qualitative data analysis process based on the 'ladder of analytical abstraction,' which defines a series of steps through which the analysis moves (Carney 1990; Miles and Huberman 1994; Bygstad and Munkvold 2011). These steps include: i) summarizing and packaging the data; ii) repackaging and aggregating the data; and iii) developing and testing propositions to construct an explanatory framework. Figure 18 shows

a simplified version of the analytical framework, modified here to resemble the actual analytical steps taken during the current research.

The analytical ladder and action design research principles were central to my data analysis methodology. In this chapter, I will summarize and exemplify how the three steps of abstraction were manifested in this research. Although represented as a ladder, it is important to understand that the actual data analysis was an iterative process, in which we often returned to a more basic analysis and even collected more data in order to confirm or expand on certain aspects. The iterative nature of this analysis conforms well to the ADR iterative model. The ladder of analytical abstraction (Carney 1990) helps to describe the analytical process that took place within our ADR "Building, Intervention, and Evaluation" cycles, as well as in the final "Reflection and Learning" phase.

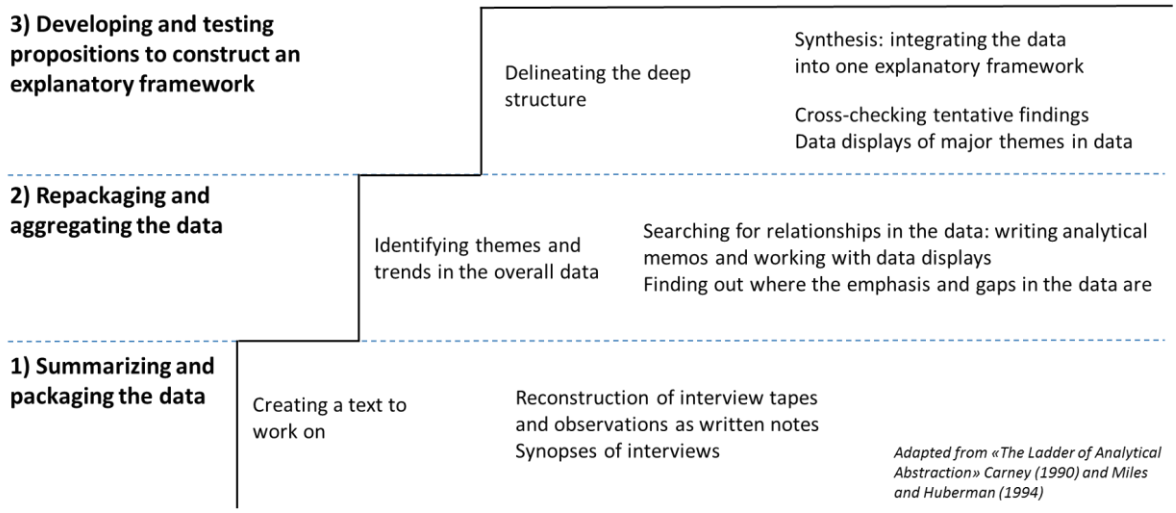


**Figure 18 shows the analytical model used in this thesis. The model is a simplification of "The Ladder of Analytical Abstraction", adapted from Carney (1990), with additional inspiration from Miles and Huberman (1994).**

### 4.6.1 Summarizing and packaging the data

The first analytical step should take place soon after the data is collected, and consists of going through handwritten notes, listening to interviews, transcribing recordings and condensing empirical evidence. In my research, this step typically resulted in a set of condensed notes from field trips, meetings or workshops, or from artifact studies. The summary notes included quotes that I found particularly interesting, observations that seemed important, actions that had been taken and comments about the development process and how the artifact had been affected by the learning in the field. I made a deliberate attempt to retain some of the softer, contextual information and multiple interpretations in these summary

notes, rather than just including 'factual' information that related strictly to my intended research question. While the research questions guided this part of the process, I allowed for deviations where these seemed interesting. These notes were distributed to my fellow researchers within the team, the developers and sometimes also the informants. At this stage of the analysis, the material was highly contextual and tainted by the situations in which the data had been collected. While distributing such notes could result in misunderstandings and false generalizations, it also provided the team with a condensed overview of what was happening in the field.

The following notes from a visit to a semi-rural health clinic offer an example of how I summarized these observations from one day, and exemplify the first step of data analysis. This report was shared immediately with the teams in Uganda, Norway, and Vietnam. I tried to capture part of the surrounding environment, rather than just facts and quotes.

***Wednesday afternoon - ABC Health Center III***

*It was 14.30 in the afternoon. We drove out of Kampala for about # minutes along a main road, towards the ABC health center. The turn-off from the road was an unpaved road down a steep hill from the main road, with no signs indicating that a health center was down the road. There might have been a sign if approaching the other way on the main road. We drove along the dirt road for a minute or so and passed a few women walking the other direction. X said they were the women working at the health center, and she wondered if they had walked off instead of meeting us as planned. [Note: See note below] However, she also said that there would be people in the delivery ward, even though the bulk of the midwives handling ANC might have left. ANC visits seem to happen in the morning typically, and the load reduces after lunchtime.*

*We arrived at the health center, which was a collection of one-story brick houses. We parked alongside the top building, which according to X had the office and out-patient functions, and walked down 20 meters to another building. There we met with several midwives who spent the rest of the afternoon describing their work processes and also helped to test our mobile and PC solutions.*

*We were first shown around. There was a front desk in the hall at the entrance. On one side there was a large-ish room with 4-5 beds in it. None seem occupied by pregnant*

*women, but we didn't see the whole room. A newborn baby was lying in one of the beds, and we did see pregnant women walking about later on. On the other side were an examination room and a room used for deliveries.*

*They said they had about 25 ANC visits daily, with five deliveries. We did not see anybody in labor, and it seemed remarkably quiet for a labor ward.*

*We went through their use of registries and discussed how they used them and which information they filled in. They reset the ANC number every New Year (the instruction document says 1st July, but X said many health centers use New Year instead). The books seemed to have been filled in with diligence. They wrote the date for the next visit in the mother-passport (woman takes this home), but they did not keep a note of this appointment themselves. They didn't seem too interested in tracking the mother. It didn't really seem to be of their concern. They said this was the responsibility of the mother. [Note: When we returned to the same facility the next year, we discovered that HIV-positive mothers were treated differently and were tracked much more carefully. Note the above comment about how the people we were supposed to meet may have left. This may have been a reason why we got different stories, since we met people with different responsibilities during the two visits. ]*

*Mothers often gave birth at other facilities. The road to this health clinic was difficult to use at night. They didn't seem to see it as a problem that mothers moved around. In principle, I guess it isn't an issue either, as long as they meet up to the appointments they should meet to.*

*They didn't have any maternal deaths in this site the last years. If they observed that the delivery was not going well, they would call the husband to get transport to Mulago. They even seemed to pay out of their own pockets to have the mother transported to the hospital. X said this was a general issue; health centers send the mothers to Mulago Hospital, sometimes too late, with the mother already in shock. The death is therefore logged at Mulago and not attached to the health center. The maternal death audits should typically follow up on such issues.*

*They told us that most pregnant mothers came with their mother-passport, but that some didn't because they didn't want to show to their husband that they had been there. Perhaps because they were with another man.*

*They first tested the PC solution, guided by O. They entered a fake patient assigned to their health center and filled in the ANC information for her. The person testing struggled to use the computer and had difficulty with scrolling and selecting drop-down selection boxes. The option selection boxes are not very user-friendly on DHIS2 currently due to the automatic filter but are probably very effective for trained personnel.*

*They registered their own mobile phone number on the 'fake' patient they had entered, and Ola showed them the possibility to send SMS messages from the web interfaces. They seemed impressed when the messages came on the phone (but I was perhaps the happiest that it worked as expected).*

*They seemed more comfortable with the mobile phone than the PC. The person testing it typed quickly (quicker than I) and understood drop-down boxes etc in the Opera Mini interface. When we remarked this, they laughed and said 'The mobile is the PC in Uganda.' None of them had PCs at home, but they proudly presented their mobile phones (I took a picture).*

*One of the younger employees said she reported into DHIS2 using a PC, and that she had attended DHIS2 training. P pulled out his PC and we looked at the data they had supplied, confirming the earlier numbers they had told us about. She was very proud to have done this job.*

*Cervical cancer is a problem in Uganda, and most women don't come in to test this specifically. The PNC visit is a good time to do this and is one of the reasons (in addition to family planning etc) that PNC is important. But X said that many now just called PNC for family planning and that there wasn't much focus on the cervical cancer test. X was concerned about the state of PNC.*

*I tried to make a point of telling them that they should be open about telling us about things that didn't work. The software is intended to make their day simpler, not more*

*difficult. So if it makes it more difficult, please ... (I was interrupted by a midwife who jokingly said "resign our jobs hehe...") let us know.*

*That evening we sent improvement suggestions to T, who implemented most of the fixes by the next morning so we could show them at the next site.*

The above is an extract from a summary note written after one day of a field trip to Uganda, 2012. Comments in square brackets [] were added later for explanation. The actors in the text are anonymized.

### 4.6.2 Repackaging and aggregating the data

The next step in the analysis process was to identify themes and trends in the material. In this part of the process, we used data displays extensively (Miles and Huberman 1994; Miles, Huberman, and Saldaña 2013). Data displays were created as a team effort from the underlying summary material, in order to represent different views of the observed environment, including the use cases and scenarios, roles and organizational hierarchy, clinical and documentation processes, and aspects relating closely to the artifacts themselves, such as diagrams with information models and the screen flow of applications. These data displays were actively used by the core team for internal discussions and external communication. The data displays helped us to visualize the proposed solution for the informants and practitioners. In turn, they guided us in creating improved versions of the software artifacts, as well as enhancing our understanding of the underlying organization and processes. These data displays were contextual and were linked to the project environment, rather than generalized examples that could apply to a class of systems.

Data displays such as those in Figure 19 and Figure 20 helped us to understand the underlying process of a particular antenatal care project, and guided the design of the software. The diagrams were refined as different stakeholders gave their feedback. From an analytical viewpoint, we repeatedly returned to data collection, gathering new data that was summarized and which subsequently resulted in new changes in the data displays.

Due to the highly configurable nature of the software used during the project, some of the data displays were also actual configurations of the software. The software artifact could therefore be interpreted as a powerful data display that was used to demonstrate and visualize concepts seen in the field.

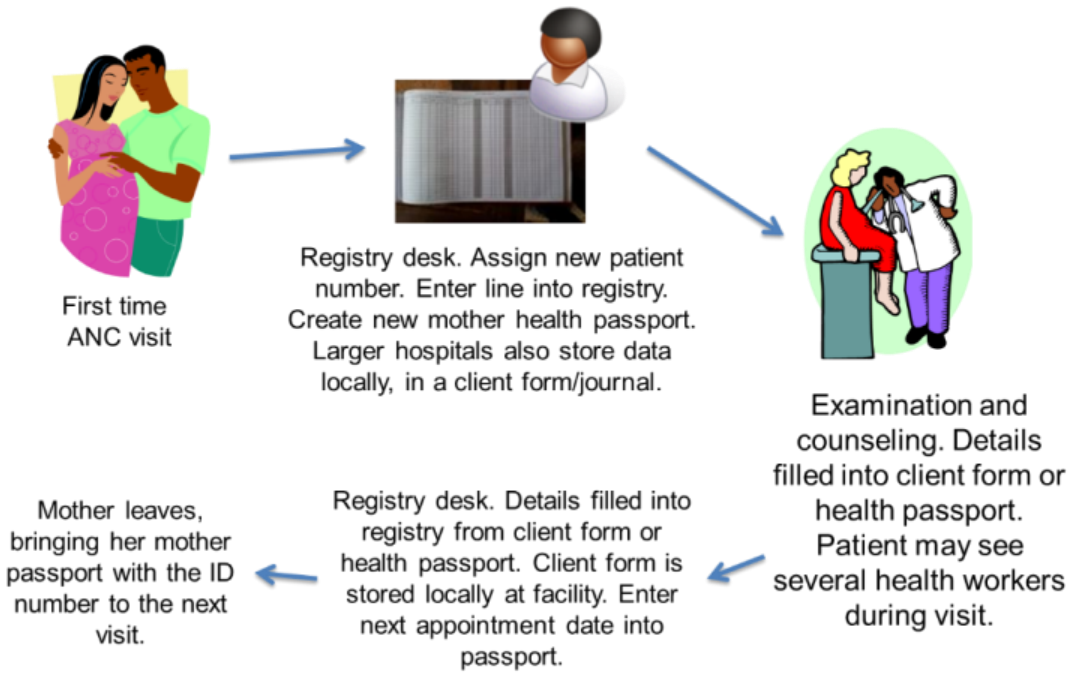


**Figure 19 shows a data display representing the patient flow during an antenatal care visit. This illustration was created in the evening after the first visit to the largest hospital, and was discussed and refined the following day with the project members.**

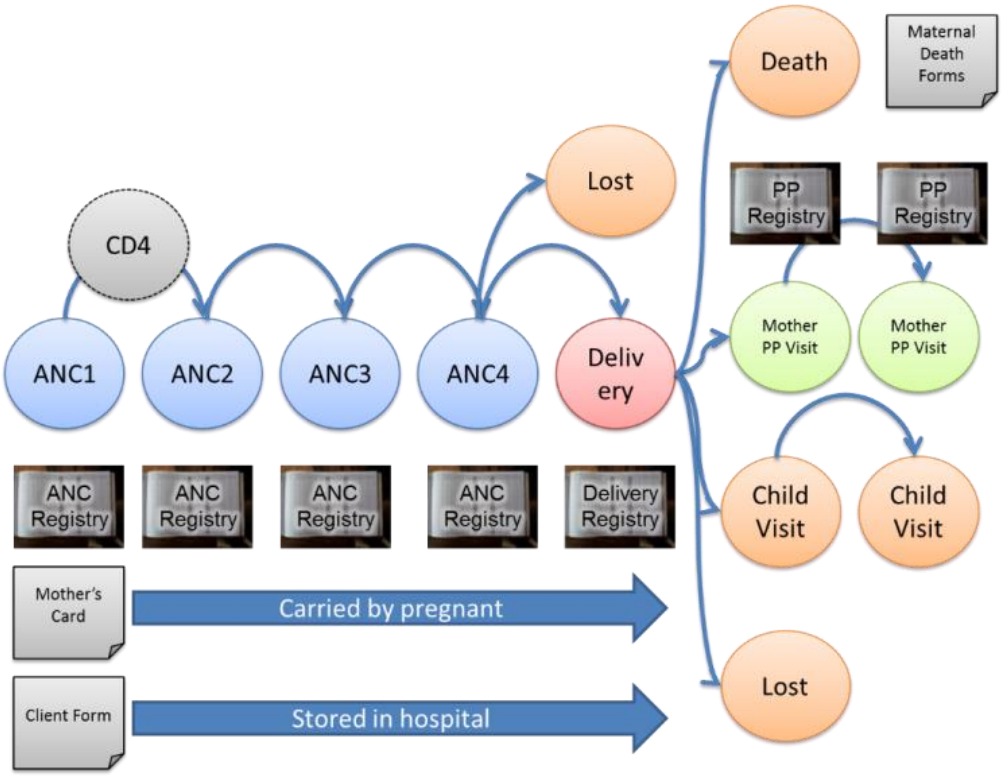


**Figure 20 shows a refined data display of the visits a pregnant woman should ideally make on the way to delivery. The first version of this data display was created during the first field trip to Uganda, and was reviewed and revised multiple times after this.**

Another example of the use of data displays in this phase is closely related to the user interface of the artifact. The required design of the user interface was documented using data displays such as that in Figure 21, which shows the flow between multiple mobile screens. These mobile screens are naturally limited in size and function, and inspired the use of a mobile-first development methodology (Wroblewski 2011). The small space on the mobile phone screen and the importance of making the data entry and use processes efficient forced practitioners to make tradeoffs in terms of the functional scope. These diagrams helped them to visualize the flow and to prioritize functions. The background guiding these functional decisions differed depending on the practitioners we asked. The data displays showing user interfaces thus helped us to visualize and analyze the underlying priorities of different practitioners.

Reusing the data displays for other contexts helped to identify heterogeneity between projects and between various sites within projects. As described by Sein et al. (2011) through the principle of "Guided Emergence," the resulting design emerged during the project period, based on both technical and social perspectives that were unearthed due to a combination of both anticipated and unanticipated outcomes.

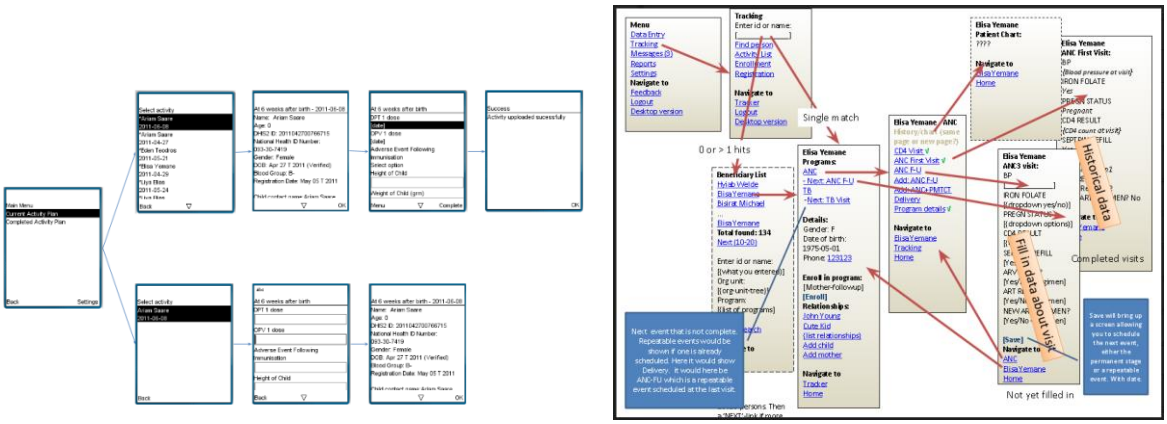


**Figure 21 shows screenshot mock-ups that we used in the design process. These mock-ups were simple enough so that more functionally competent people within the project could give valuable feedback and design the actual application. These slides were sent to the development team in Vietnam, who within a few days had implemented the interface. In this way, multiple design cycles could be achieved during the same field visit.**

### 4.6.3 Developing and testing propositions to construct an explanatory framework

The next analytical step was to move from data-driven data displays to more general representations that also were described in terms of theoretical constructs. As stated in the principle of the "Theory-Ingrained Artifact," the ADR method should look for research outcomes that are informed by theory. The use of both original and newly visible theoretical

concepts was useful during this phase of analysis. Using theoretical lenses helped us to generalize project-specific issues by introducing a vocabulary that was more general and less contextual.

In this phase, the earlier data displays were generalized and combined with theoretical constructs. The working products at this stage were more data displays, and also theoretical memos that explored different theoretical perspectives. As an example of this stage of data analysis, Figure 22 shows part of a draft data display that was used during the analysis phase of the "P for Platform" article. The figure shows a circle with the user in the middle, and illustrates how the user interacts with the developers through implementers. The data display demonstrates the role of the user in relation to the stakeholders of the project, and we used theoretical constructs such as flexibility, inscriptions, configuration, path dependency, generification, participatory design and boundary objects to discuss this figure. The circular figure later evolved into the 'onion shaped' platform representation published in the article, which is based on platform architecture theory (Gawer 2009; Tiwana 2013; Ghazawneh and Henfridsson 2013) and flexibility concepts from information infrastructure theory (Hanseth and Lyytinen 2004).

It should be noted that Figure 22 shows the user at the center, while the final figure used a more traditional platform view with the core platform in the center and the user at the periphery. Several different perspectives were tested during this analysis phase, and the results were shaped both by our material and by the theoretical concepts.

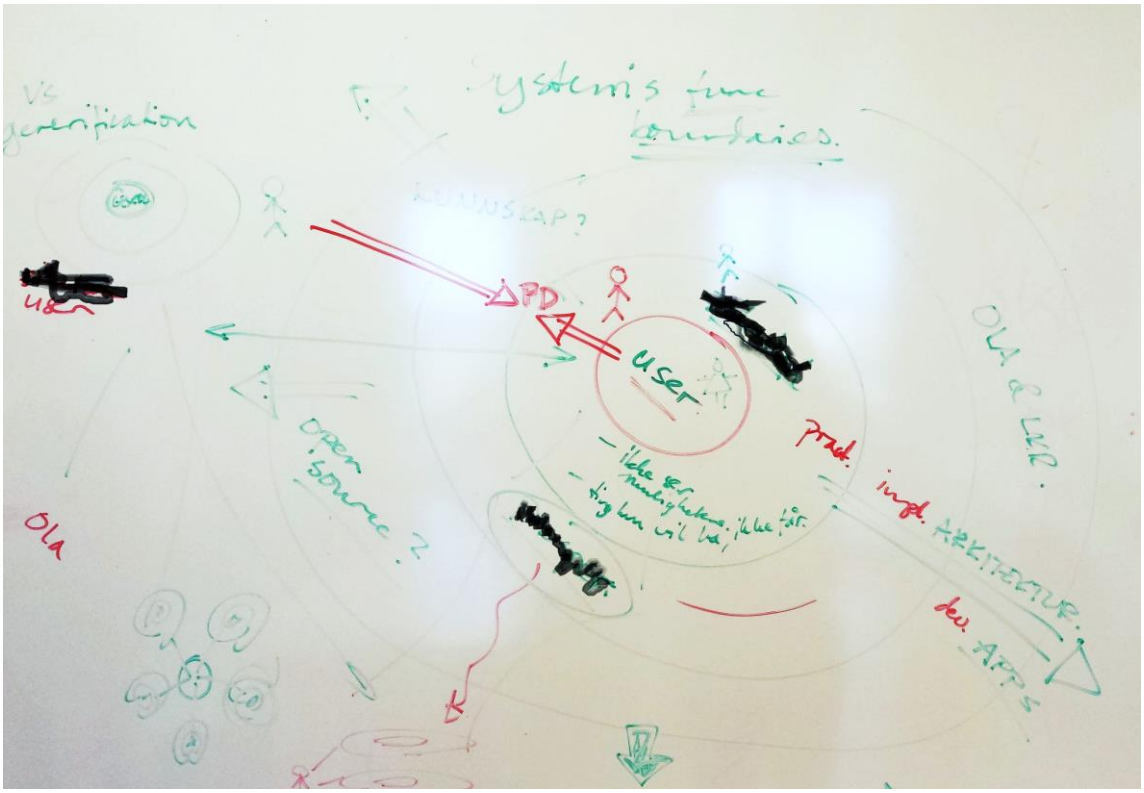


**Figure 22 shows a data display on a whiteboard, where different actors were placed according to the role they played in the platform hierarchy and the level of participation was discussed. The figure was part of the analysis that led to the "P for Platform" article and is a forerunner of the 'onion' platform figure that has the platform at the core. Some names have been erased.**

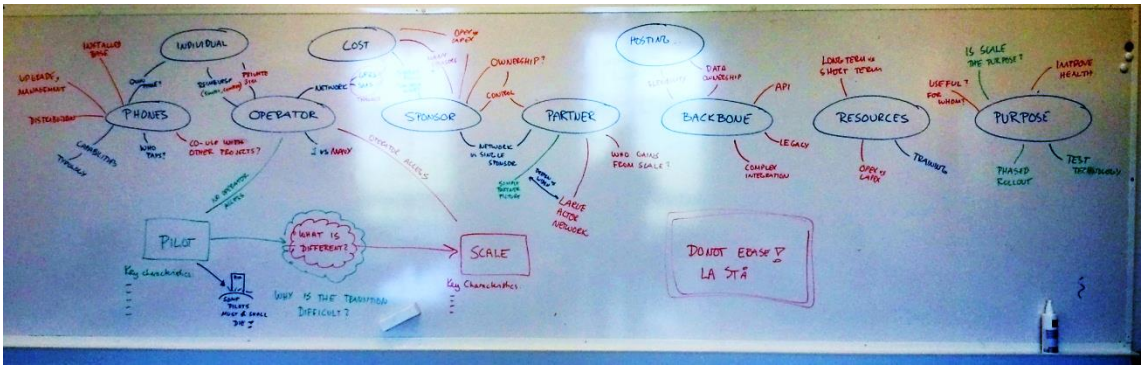


**Figure 23 shows a data display used during the analysis phase of Paper 3, "From Pilot to Scale…" (Sanner, Roland, and Braa 2012). The display focused on the different stakeholders and factors related to scaling and their relationships.**

The original research always began with certain theoretical constructs that informed our initial research question, such as concepts from infrastructure theory. However, at this stage of the analysis we allowed ourselves to introduce new theories to see if this could help in highlighting interesting aspects of our material. We also discussed the material with researchers who had not been involved in the earlier phases of the project, but who had a

strong theoretical background. In some cases, we then had to return to previous phases to review material and re-do the analysis.

For example, in one paper we introduced Weber's concepts of different sets of rationality during the analysis phase (Kalberg 1980), which helped us to illustrate some unforeseen conflicts. The introduction of new theory at this late stage in the research led us to carry out another round of analysis to confirm that our application of this theory seemed valid for the observed data. Additional data were also collected during the next field trip to investigate these theoretical concepts in more detail.

### 4.6.4 Iterative data collection and analytical process

As explained above, the data were presented and analyzed using data displays of varying maturity (Miles and Huberman 1994). These data displays were discussed within the project team and refined. After refinement and internal discussion, we used the data displays both internally and in external events such as DHIS Academies, workshops, and even academic conferences, in order to get feedback on the results.

We also re-examined the data repeatedly, through listening to the audio or video recordings of many of the sessions, to check whether the concepts we had identified were indeed valid and accurate. The longitudinal nature of the research project, and the repeated field trips and interaction with the same informants over several years, allowed us to continue data collection and further add to the empirical evidence. The multi-case nature of the study also allowed us to analyze data from multiple contexts, providing more rigor in our analytical work.

### 4.6.5 Analysis for the overall thesis

The overall research theme of this thesis goes beyond those of the individual papers. Additional analysis was therefore required to highlight the overall research questions. The final analytical process was similar to those of the individual papers and included both an analysis of the findings of each paper and a return to the empirical material of the research to highlight significant findings. In general, however, the results of this thesis are a combined and refined summary of the conclusions in each of the papers. In this final phase of the thesis, the main focus of the analysis was to provide what Carney (1990) describes as the top level

of the ladder of analytical abstraction: “Synthesis: integrating the data into one explanatory framework”.

As described earlier, the research underpinning this thesis has two layers: one focuses on the implementation of single instances of the systems within each project, and the other considers how the ensemble artifact, represented by its platform architecture, evolved as a result of opposing requirements and tradeoffs. An analysis of the overall ensemble of architectural elements was the main focus of the final analysis of the joint thesis.

To provide a coherent and effective response to the research objective, it is necessary to discuss the research using a common conceptual framework. This Kappa establishes such a common conceptual framework in Chapter 2, summarizes the findings in Chapter 5 and integrates the findings into one explanatory framework in Chapter 6.

## 4.7 Reflections on and limitations of the research method

### 4.7.1 Ethical considerations

Several aspects of the research for this thesis required careful ethical consideration. Firstly, the research involves observing patients as part of the patient flow, although it does not directly involve patient data or patient interviews. All research involving informants, interviews, and observations is by its very nature sensitive and subject to strict ethical guidelines. Secondly, the majority of the thesis involved doing information system research in low-resource contexts, where there are considerable challenges in handling the existing health service. The issues covered in this thesis are potentially sensitive from a political and organizational standpoint, particularly since some of the research was carried out in countries with a volatile political environment.

### 4.7.2 Qualitative research involving informants

None of the research in this thesis specifically involved patient data, and all interview notes were anonymized. The research did however include observing patients, browsing through registers where patient data is documented and, in a few cases and with explicit permission, taking pictures that included patients. This type of data collection can be problematic, and it is important for the researcher to maintain a high standard of ethics throughout the research, including anonymization and deletion of data after the study.

The importance of anonymization cannot be overstated. The effects of publishing certain types of comments can be considered problem-free for a researcher from Norway who is used to openness and has a willingness to accept feedback, but can be hugely problematic for a person working in certain developing countries, for example, where the consequences of openness and criticism may be much more dramatic than in Norway. In our case, there are no negative effects of anonymization on the research results, because we are seeking an understanding that is independent of the specific people holding these opinions. The identities of the individuals are therefore not interesting for this research, although their roles and backgrounds may be.

For some parts of this research, such as the last paper, for which the data collection was conducted in Norway, the interview situations were planned and the informants signed agreements before taking part in the research. They were explicitly asked if it was acceptable to record the session, and no recording was made if they declined. The recordings were deleted after the paper was published. In these cases, it is clear that the informants were in a position to understand the consequences of participating and could make an informed consent about their participation.

However, some of the data collection relating to the three first papers was potentially more problematic. For some interviews, we sought explicit consent; in others, the consent was considered implicit since I was presented as a researcher, showed the participants the notebook and took notes during the session. Given the hierarchical political systems of some developing countries, however, these informants may have been unlikely to refuse participation. To illustrate how vulnerable some people probably felt in this position, consider the midwife who responded that the only option she saw if the system did not accommodate her needs was to quit her job.

Although we did not record sessions without informing the people in the room, there were several situations in the DHIS2-related research where it is likely that the participants felt forced to accept this. We also taped large plenary sessions, where it may not have been obvious to the participants that the meeting was recorded.

The weak position of an informant in a larger system may be considered a drawback of this research method. Although we conducted research in all cases with agreement from the authorities, and always asked and sought permission locally in clinics, there are bound to have been times when we overstepped the boundary of the strictest ethics guidelines. In view

of this, it is imperative for us as researchers to maintain the highest standards for data storage, i.e. deleting data as soon as possible, anonymizing it and making attempts to avoid causing problems for the original informants.

### 4.7.3 Research in low-resource settings

My research concerns the development and deployment of health information systems in developing countries. These are countries with fewer resources, low technological know-how, vulnerable health workers and patients, restricted and fragmented health funding, poor health services and sometimes political instability. We have activist-like goals of making a difference by improving health systems in these countries through better information systems. The introduction of these systems can sometimes be completely disruptive to existing processes, replacing the existing paper-based solutions entirely. Many of the solutions we replace are very poor, and if successful, intervention can make a substantial positive difference. However, in some cases, our projects fail. It is therefore relevant to discuss whether it is morally permissible to do research in these developing countries which involves disrupting their processes and organizations with new technology that we cannot guarantee will work.

Research in developing countries and low-resource contexts raises some specific ethical concerns. Dearden (2012) provides a comprehensive review of ethical issues in research on ICT for developing countries. Many of the topics he raises are also valid for this thesis.

Research involving low-resource countries with disadvantaged or vulnerable communities is a special case that requires extra attention and ethical considerations. One source describing this is the World Medical Organization (WMA): "Medical research involving a disadvantaged or vulnerable population or community is only justified if the research is responsive to the health needs and priorities of this population or community and if there is a reasonable likelihood *that this population or community stands to benefit from the results of the research*" (WMA, 2008, clause 17, cited in Dearden 2012, my emphasis).

These guidelines are for medical research, but I believe that health information systems research involving the same groups should be governed by similar ethical guidelines. My opinion is that when involving disempowered groups in ICT research, one must be particularly cautious that the results of the study, including the software development, benefit them. These groups should not merely be seen as easily accessed user groups where one can

collect information, involve them in software development and use their resources for the development of solutions that mainly are intended for other users. How the projects are financed, and the way in which potential profit later is extracted from the software development should also be considered (Dearden 2012, Sanner and Sæbø 2014).

When doing action-research based ICT research, one intentionally disrupts the existing processes and tools and examines the effects of introducing new tools. These projects may require that health workers are taken away from existing health tasks to receive training, or that extra work is needed on a regular basis that takes time away from clinical work. Sometimes the new tools are introduced in parallel with the existing paper-based tools, adding more work and potentially taking the focus away from the core health-saving tasks that form the main job of these workers. New ICT tools are introduced into existing workflows without guarantees that they will improve the work situation of the individual worker, as this is the subject of the research. Health management tools are also added without direct benefits for the health worker on the ground, in order to assist in national public health management (Manya et al. 2012; Braa and Sahay 2012; Sahay, Sæbø, and Braa 2013; Sahay, Monteiro, and Aanestad 2009). When these research interventions fail, they may still provide important learning for the global research community, but the disruption at the local level would typically disadvantage rather than benefit the local community (Sanner and Sæbø 2014). These failures therefore raise ethical concerns for ICT research.

It is worth noting that health workers who attend training events in developing countries typically receive generous per diem payments that can constitute a considerable part of their annual salary (Sanner and Sæbø 2014). Training events are also meant to be capacity-building for the health staff, and would typically also provide opportunities for networking and coordination across health facilities.

As noted by the US National Bioethics Advisory Commission (NBAC 2001, p. 4, cited in Dearden 2012), the exclusion of these groups from research may also be problematic. These are groups of users who may have special needs, and who may not easily be able to use generic software developed for other purposes. Bhutta claims that restricting research can "effectively stop much-needed public health and epidemiological research that often generates precisely the information that might influence future public health policy" (2002, p. 116, cited by Dearden 2012).

Vulnerable users might be able to give their consent to being research subjects, but may not be in a position to see all the consequences of such research, and may also have false expectations about how the research will benefit them (Dearden 2012).

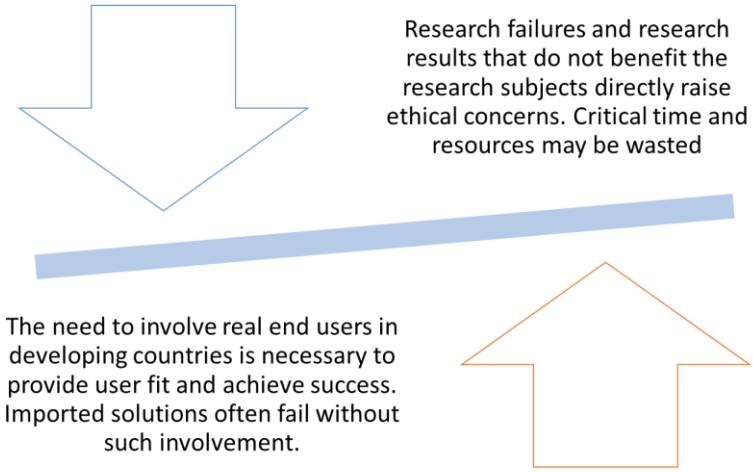


**Figure 24 illustrates the balance between concerns over avoiding disruption and the unnecessary involvement of low-resource groups, and the need to involve these groups in developing systems to fit their needs.**

There is a balance between restricting research because of the risk of failure and time wastage, and the benefits of involving the real end-users in designing the solutions. The active recruitment and funding of Master's and Ph.D. students by DHIS2 in these countries are important measures for ensuring local understanding and guaranteeing that knowledge is left behind when the project leaves (Braa, Monteiro, and Sahay 2004).

Throughout the history of DHIS versions 1 and 2, the empowerment of low-resource groups and an increasing focus on them through discovering knowledge about their health and how their health status compares to other groups have been core drivers for the research projects and software development. The DHIS2 implementation projects have therefore included interventionist studies aiming to uncover such information since the start of DHIS in South Africa in 1996.

Braa and Hedberg (2002) write: "The politics of apartheid were deeply inscribed into the "old" health information systems", "In its first pilot phase (1996–1998), HISP aimed at developing district health information systems to support the emerging decentralized administrative structures in three pilot districts in Cape Town" and "It is important to note that the original key members of the HISP team have background as social/political activists

in the anti-apartheid struggle and other social movements, and that we have always explicitly and implicitly seen ourselves as political actors in a larger development process."

They successfully started and managed a global network of action that became the basis for one of the largest health information system solutions in the world, and the global impact of their solution is now evident (Althauser et al. 2016). A key factor in the success of DHIS has been that local universities and individuals have been recruited into the network and have helped the project align with local needs and requirements (Braa et al. 2004). It is important for such research projects to focus on local capacity building amongst local academics and government institutions and within the health service. If this is achieved, local learning can be provided independently of the success of the cases.

I believe that the potential negative impacts of researching low-resource groups have been balanced in DHIS2-related work by ensuring that the results benefit the countries in question. Using a consequentialist argument, I would say this potential benefit justifies the potential drawback of being research subjects.

# 5 Research findings: summary of papers

As described in Section 1.5 and Figure 1, the four papers in this thesis all play a role in answering different aspects of the research questions: Paper 1 discusses the types and effects of heterogeneity; Paper 2 discusses how a platform architecture can be opened up to different types of user participation; and Papers 3 and 4 discuss the benefits and tradeoffs that application developers meet when adapting to a platform.

This chapter summarizes the findings of the four research papers included in this thesis.

1. ***"Accommodating Multiple Rationalities in Patient-Oriented Health Information System Design"***: This paper discusses how health workers within the same health system have different concerns over and requirements for the shared information infrastructure, predominantly covering the conflict between primary and secondary use when designing shared information systems. The paper uses rationality, interpretive flexibility and inscriptions as theoretical lenses, and discusses how these affect the design.
2. ***"P for Platform: Architectures of large-scale participatory design"***: This paper considers the role of participation when systems scale vertically and horizontally, and how participation relates to the architecture of the system. The proposed platform architecture promotes a generic core that has high use-flexibility, but less flexible participation characteristics. The platform enables user participation and flexible innovation in the fringes using applications. The paper discusses how different types of flexibility and user participation are available at various levels of the platform.
3. ***"From pilot to scale: Towards an mHealth typology for low-resource contexts"***: This paper discusses different types of mHealth architecture, and considers the scaling of pilot implementations as the intention, and robustness and flexibility as dimensions when analyzing the ability of each type of architecture to scale.
4. ***"Flexibility in EHR ecosystems: Five integration strategies and their tradeoffs":*** This paper discusses the different ways in which applications can be integrated into a health worker's workspace, and highlights the tradeoff between change-flexibility and seamlessness. The term seamlessness is a usability-related concept that is also related to use-flexibility.

The papers were not originally written in this order, but have been organized in this way in this thesis to reflect the way in which they represent sequential parts of the answer to the overall research question.

## 5.1 Paper 1: "Accommodating Multiple Rationalities in Patient-Oriented Health Information System Design"

### 5.1.1 Purpose

The global development of health information architectures is shaped by two broad trends: a move towards more integrated health information architectures, and more health data about individuals being captured and stored in databases (Braa and Sahay 2012). The implementation of patient-oriented information systems that span several contexts can impose contradictory requirements. The paper illustrates how the conflicting rationalities of different user groups within the health service can lead to contradictory requirements for the software development of shared software systems. Unearthing these rationalities is an important part of the design process, and can help in supporting multiple contexts of use within the integrated system. The article describes which rationalities are prevalent amongst users and designers of person-oriented information systems, and which design principles can be derived to accommodate these rationalities.

### 5.1.2 Research approach

The research project was informed by action design research (Sein et al. 2011), and covered the development of an information system for integrating antenatal care with delivery and postnatal care across clinics in Uganda. An existing DHIS2 patient tracking solution was redesigned and changed to fit the project context, and multiple development cycles were performed to adjust the software and extract design principles for this class of systems. The empirical data was collected as part of the design and development process and included semi-structured interviews and observations from meetings and training sessions. The informants included a large number of stakeholders, including end-users. Although the project was informed by action design research, its real-life context coupled with the simultaneous development of parallel projects made it difficult to adhere strictly to the ADR process as described by Sein et al. (2011). The development of the system had to be seen in a

shared context with other projects using the same code base, used to deploy similar solutions in parallel worldwide.

### 5.1.3 Findings

The development resulted in a software system that was tested over multiple action design research cycles (BIE), with increasing involvement from practitioners and end-users. As the solution was demonstrated and tested in clinics, it became evident that different user groups had conflicting strands of reasoning (rationalities) about patient-oriented health information systems. Some advanced users showed the ability to bridge multiple rationalities, while others argued strongly for one or the other.

Firstly, there was the choice of how rigidly the health system should be tracking users, or whether patients should be left to their own decisions and take the consequences of not attending their antenatal check-ups. The rationality of patient choice vs. institutional control of care proved to be central to how the solution was designed.

Secondly, some saw the most significant requirement for a patient tracking system as providing the best possible treatment at the point of care. Others considered the most important reason for such a system to be the possibility of shaping wider-reaching public health management decisions based on the collected data, and replacing existing tools for public health management data collection. These two latter rationalities were labeled *clinical tracking* and *public health tracking*. The study is supported by the literature (Berg 1999) showing that data collection at the point of care and data collection for public health management may impose conflicting requirements. The resulting design principles addressed all of these three rationalities within the software, carefully balancing them when their implications were contradictory.

Identifying these different user rationalities helped guide the design of the software system, allowing the designers to incorporate separate but compatible views that catered for each rationality. The identification of these rationalities should, however, have been part of the early requirement work, and would have made the adoption smoother.

The study also identified a difference between these rationalities in terms of the importance of structured data. Public health tracking requires the structure to give useful statistics across facilities, but clinical tracking is sometimes better accommodated by free-text fields, which can capture nuances that are lost in structured data.

The project involved the redesign of an existing software system already used in India in a public health tracking context, and this software was developed and applied simultaneously to other similar projects in different countries. These somewhat similar use cases caused conflicts between the multiple parallel software projects reusing the same code base. A further examination of the effects of the co-negotiation of both aligned and conflicting design principles across projects was identified as an area of further study, leading to the "P for platform" paper included in this Ph.D. thesis.

### 5.1.4 Implications

The paper uses the concepts of inscription and interpretive flexibility to discuss how multiple rationalities are identified and understood through the design and use of a patient-oriented health information system. The study highlights several parallel rationalities that are more or less in conflict when designing a patient tracking system. The contribution of this paper was the identification of these rationalities and a discussion of how they may align and conflict, along with the identification of other rationality-driven design principles.

### 5.1.5 Contribution to Ph.D. research

The most important implication of this study for the Ph.D. research was not to identify which user rationalities are evident in patient-oriented tracking systems, but rather the acknowledgment that multiple rationalities exist when creating integrated health systems, and a discussion of how these rationalities are accommodated and negotiated in multi-site projects during the design process. The remaining articles continue the discussion on how to support requirements from opposing rationalities in a single integrated system using a platform architecture approach.

## 5.2 Paper 2: "P for Platform: Architectures of large-scale participatory design"

### 5.2.1 Purpose

Participatory design (PD) has been an important and widely used method for empowering the end-user in settings where powerful management and a powerless end-user have different agendas and requirements. With today's increased focus on large-scale, integrated health information systems, the role of the end-user can again become marginalized and the

principles of participatory design forgotten. To some extent, there is no longer a clear definition of the end-user in such an integrated system, but rather a collection of multiple user communities who are affected by the information system introduction in different ways. As shown in Paper 1, in integrated health information systems the various user groups exhibit different rationalities and requirements that must be carefully negotiated to accommodate all users. This second paper considers the role that architecture can play in enabling end-user participation in large-scale projects that span multiple rationalities and in which generic off-the-shelf products are used. The study highlights how an emergent platform architecture and its surrounding ecosystem can allow for different forms of user participation at multiple levels of the platform.

### 5.2.2 Research approach

The paper reports on a longitudinal case study of the development of the DHIS system, which started as a local district health information system in South Africa in 1994 and is now a massive-scale health information platform, used in more than 50 countries. The case study consists of three vignettes that illustrate the different architecture and participation modes throughout the lifecycle of DHIS2. These vignettes are used to discuss the changing patterns of user participation and participatory design in response to the challenges of scale. Given the longitudinal context of the paper, the four authors participated at varying levels in the three vignettes. The empirical data consisted of structured interviews and observations conducted at the time of each vignette, and these were re-interpreted and analyzed for this paper.

### 5.2.3 Conceptual framework

The paper uses the extant literature to arrive at a classification of four different types of PD in terms of scale: i) singular, ii) serial, iii) parallel, and iv) community participatory design. Singular PD is the traditional method, where a system is built from scratch with the use of tools such as prototypes, mock-ups, and workshops to facilitate end-user participation in design. In this traditional scenario, the end-user is in control. The user does not only participate but takes actual design decisions. In serial and parallel PD, the distance between the developer and the end-user increases and is largely facilitated by experts who represent the end-users' requirements. In these scenarios, the existence of multiple sites and different user groups introduces a level of negotiation of end-user requirements and a favoring of generic requirements. In community PD, the end-users and sites are represented by

communities that help shape general requirements for the architects and developers of the solution. Community PD is similar to generification, as described by Pollock and Williams (2009), where individual user requirements are placed in the background and universal concepts that apply to multiple sites are preferred. The idea of generification was in fact included in early revisions of the paper, but was removed following reviewer comments.

Use- and change-flexibility are chosen in the paper as theoretical lenses to help analyze the relation between architecture and modes of participation.

### 5.2.4 Findings

The typology of these four modes of participation is applied to the three vignettes to discuss the role of architecture in the different forms of PD. Vignette 1 (Sierra Leone) illustrates the case of serial PD, with experts and developers who travel to different sites and focus mainly on one project at a time; they develop the solution according to that context and then move on to the next site. Vignette 2 (Uganda) shows a case of parallel PD, where end-users in multiple parallel projects arrive at conflicting requirements and the central developers begin to focus on negotiating generic functions rather than listening equally to all end-users. Vignette 3 (DHIS2 Academies) describes how regional workshops are set up to bring together experts from different countries to learn about new functions and discuss common requirements. In this case, the communities are used to help arrive at a generic functionality, and developers have little or no contact with end-users.

As the type of user participation changes, the system architecture also evolves. The first architecture of the system was modular in theory, although in practice, only core developers added functions to any part of the common software repository. External modules were created only in certain particular cases. In most deployments, the implementers and end-users on the ground were in reality only able to configure the solution rather than develop their own modules. The options for setting up and adapting the solution without development were extensive, however, and scaling across similar projects could be achieved successfully without involving software development.

As DHIS2 was adopted by more users and, in particular, within new usage domains such as patient tracking, integrated disease surveillance and response, logistics, education and food safety, the levels of configurability were not sufficient to fit all contexts, and the participation model broke. The DHIS2 system then evolved into a real platform that allowed external

software developers to create their own applications with their own data models and user interfaces. The inner core of the platform is still protected and difficult to change, but greater end-user participation is again possible on the fringes of the solution, by creating applications on top of the platform.

### 5.2.5 Implications

The research arrives at a typology of four different modes of participatory design, and uses these to discuss how architecture affects the various possibilities for user participation. The idea of a platform architecture is analyzed using the design and use-flexibility as lenses and discussing how these two types of flexibility can be different at each level of the platform. From this, we deduce that different modes of participation can be applied at each level, with singular PD being feasible only at the fringes of the platform through the development of apps, while community PD is important for core platform features. The paper lacks a consideration of governance as well as a discussion of how different actors relate to flexibility. These two missing topics are picked up in Section 6.3 of this Kappa, in order to complete the discussion of distributing flexibility throughout platform ecosystems.

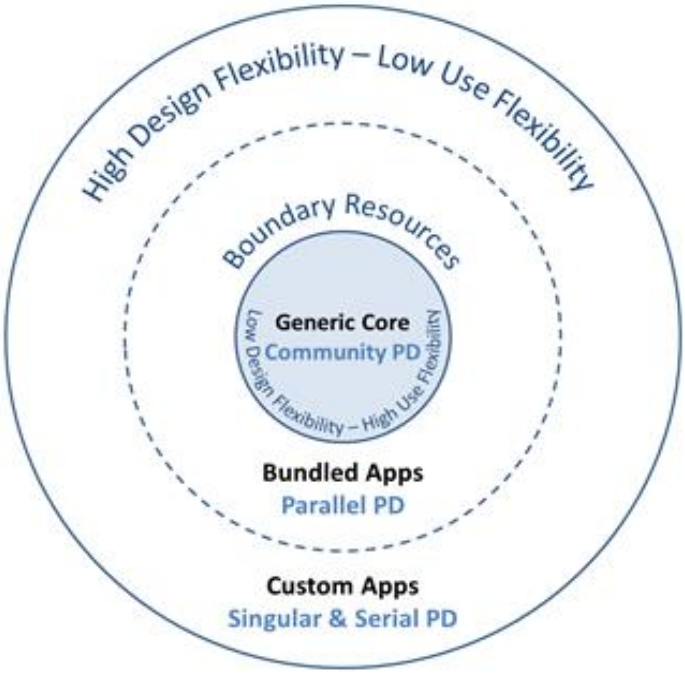


**Figure 25 shows the platform model and how this relates to flexibility.**

### 5.2.6 Contribution to Ph.D. research

The paper "P for Platform" is a central building block in this thesis, and explains how a platform architecture and the different participation modes can help to accommodate the multiple rationalities seen in Paper 1. The article establishes flexibility as a critical feature of a platform. The discussion also provides relevant background for the last two articles, which go into more detail about the concept of flexibility and how this relates to other characteristics of information systems.

## 5.3 Paper 3: "From pilot to scale: Towards an mHealth typology for low-resource contexts"

### 5.3.1 Purpose

This paper discusses the well-documented problem of the scaling and sustainability of mHealth implementations and shows how scaling of pilots may be unintentionally affected by early design decisions. It uses the concept of path dependency as a theoretical lens to explain the long-term effects of early choices. The paper develops a reference typology for mHealth deployments, classifying mobile phone-based solutions in low-resource contexts into four types: interactive voice response (IVR); plain-text SMS; locally installed mobile apps and SIM applications; and browser-based solutions. The typology discussion illustrates how the four solution types differ in terms of (i) robustness towards low-resource contexts; (ii) flexibility towards organizational and functional change; (iii) usability; and (iv) financial costs. The paper also highlights the importance of the mobile operator in rolling out mHealth solutions. In this case, the mobile phones and the backend DHIS2 data warehouse system can be considered as two joint platforms on which the mobile applications are built. The paper outlines the characteristics of such joint platforms and problematizes the conflicting roles of of multiple actors on the overall information infrastructure.

### 5.3.2 Research approach

The case study is guided by a 'network of action' approach to research (Braa, Monteiro, and Sahay 2004) and studies mHealth projects in Malawi, Zambia and several states in India; it mainly concerns the deployment of various DHIS Mobile solutions that the authors were involved in rolling out. The empirical evidence includes findings from small mHealth pilots

as well as large-scale deployments such as the statewide roll-out of 5000 mobile phones with Java applications for field nurses in Punjab, India.

### 5.3.3 Findings

The paper classifies mobile phone-based solutions for HIS interventions in low-resource contexts into four empirically derived types: interactive voice response (IVR); plain-text SMS; locally installed mobile phone and SIM applications; and browser-based solutions. It shows how the strengths and disadvantages of these solution types become evident when the scaling of implementations is attempted in low-resource settings.

Early decisions about how to leverage health workers' equipment, initial agreements with mobile operators and requirements for supporting offline work can shape the choices of early solution types and have lasting effects on the architecture. Once established during pilots, the architecture, financial agreements and organizational decisions are difficult to change.

The different solution types described in the paper vary regarding robustness (R), flexibility (F), usability (U), and cost (C). Figure 26 highlights the particular tradeoffs between flexibility and robustness, showing for example how a browser-based application has high flexibility yet low robustness to offline use, whereas installed mobile applications have opposite characteristics.

### 5.3.4 Implications

The paper contributes to the theory by applying the concepts of installed base cultivation and path dependency to a particular class of systems, HIS interventions in low-resource contexts; this is used to highlight the benefits and drawbacks of different solution types and related architectures. The findings show that early design decisions taken in the pilot phases of projects may lead to path dependencies that restrict future use- and change-flexibility.

As explained by Gregor (2006), such an analysis can be a theoretical contribution when applied in terms of design principles guiding the creation of artifacts.

The paper makes a substantial practical contribution through its creation and discussion of a mHealth typology for HIS interventions within low-resource contexts. This typology can help practitioners to make informed decisions early on and to better understand how the choice of solution types may affect the future scaling of their projects, enabling them to plan for the limitations of their selected solution types.

### 5.3.5 Contribution to Ph.D. research

The paper contributes to the discussion of applications running on a platform, and the ways in which different types of applications differ in terms of how they relate to the characteristics of the platform such as its flexibility, robustness, cost, and usability. This analysis contributes to a discussion of tradeoffs between different ways to integrate with a mobile platform, as well as a discussion of how developers are often required to relate to different platform architectures and governance structures when creating health applications.

The paper also contributes a framework for typology analysis that was evolved further in Paper 4.

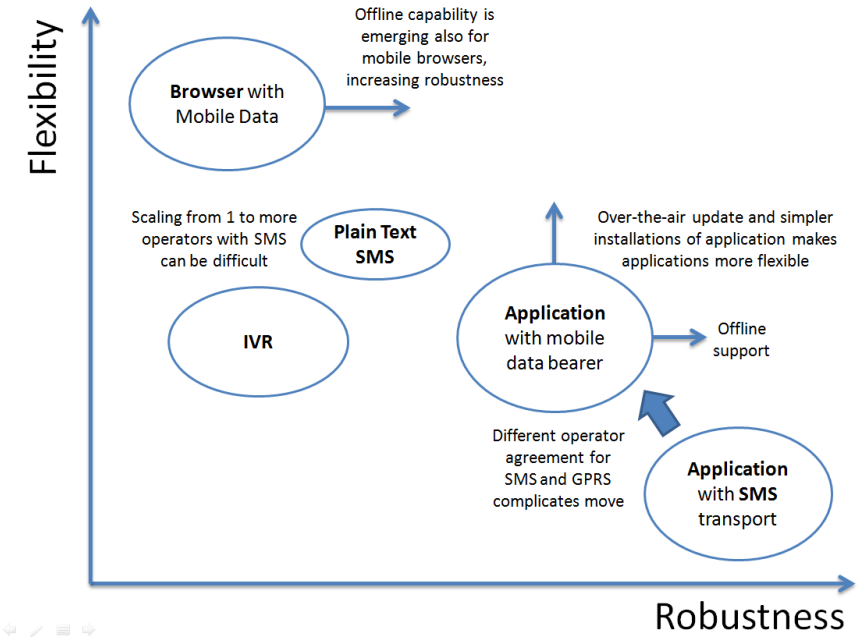


Figure 26 shows the typology, analyzed against the dimensions of flexibility and robustness.

## 5.4 Paper 4: "Flexibility in EHR ecosystems: Five integration strategies and their tradeoffs"

### 5.4.1 Purpose

Many health workers are required to relate to a large number of processes and actors as part of their daily work, and the IT tools that support them are often not designed to provide a good user experience across these multiple tasks (Friedberg et al. 2013). While national systems can be introduced to provide access to cross-organizational health information, the

integration of such external systems and information into the EHR workspace is hard to get right. IT staff who integrate health information systems need help in structuring the two seemingly opposing goals of seamlessness and flexibility. The paper uses learning from earlier papers on platforms and health information systems and focuses on tradeoffs in developing platforms for healthcare solutions.

### 5.4.2 Research approach

This research is based on a qualitative case study (Miles and Huberman 1994) covering three Norwegian cases that highlight different aspects of EHR application integration. Data collection included participation by the primary author in meetings and user forums, informal discussions, a review of documents, and semi-structured interviews. All cases were from Norway, and data collection took place while the author was also employed by the Norwegian Directorate of eHealth, the organization responsible for several of the systems studied. Data analysis used principles from the "ladder of analytical abstraction" (Carney 1990), as described in the method in Section 4.6.

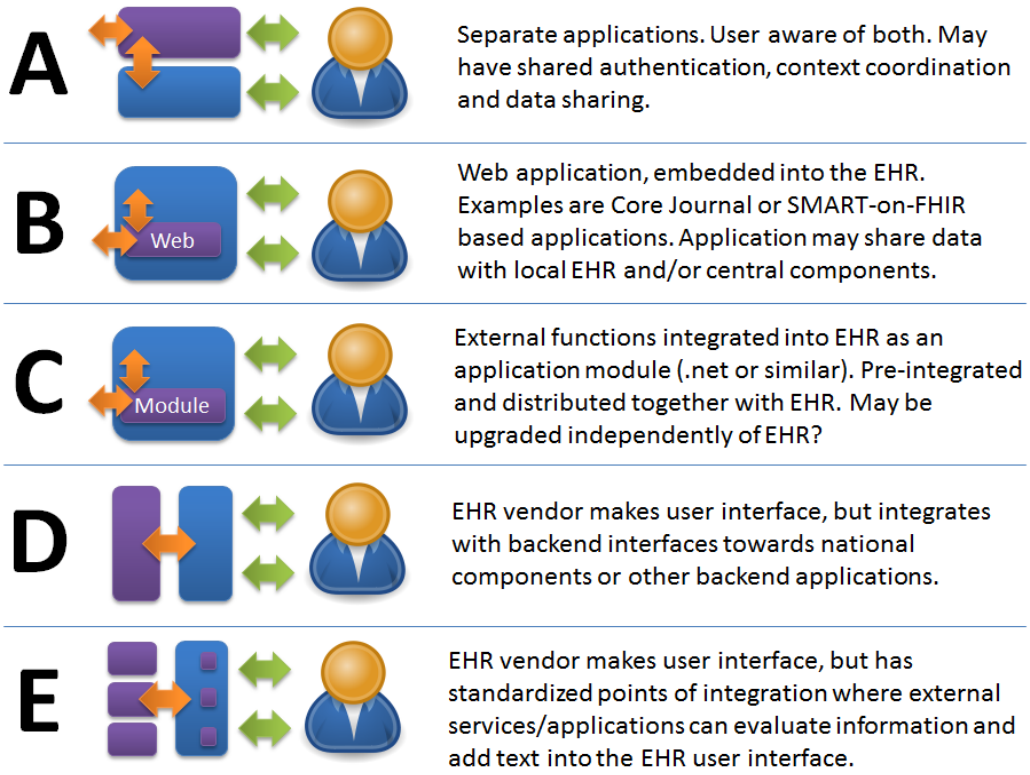


Figure 27 shows the integration typology presented in the paper.

### 5.4.3 Findings

The paper outlines five different ways to integrate an external eHealth application into the dominant workspace for the health worker, typically the EHR workspace. The typology is analyzed using the empirically supported dimensions of seamlessness and flexibility, and outlining the pros and cons of each integration method. Four different characteristics of seamlessness were found, and these were used in the discussion to outline differences that are important for IT staff to consider. The paper also explains how different strategies may be chosen and combined over time, depending on the priority of scaling the solution across multiple sites versus the need for optimal usability.

### 5.4.4 Implications

The choice of some of these integration methods is dependent on the establishment of what could be called a platform for application development; which distributes flexibility to external developers, who can then add new applications without the involvement of the core vendor. For certain integration types, the EHR plays a role as an ecosystem enabler and becomes a platform for future application development. The infrastructure must undergo a platform enablement phase in which the EHR is adapted to support a platform model. Once the platform is in place, new applications can be established more rapidly, without changes in the EHR, as shown in Figure 28. In the case of integration type D, which is advantageous in terms of usability and seamlessness, the integration requires more work from the EHR vendor and no platform capability is established.

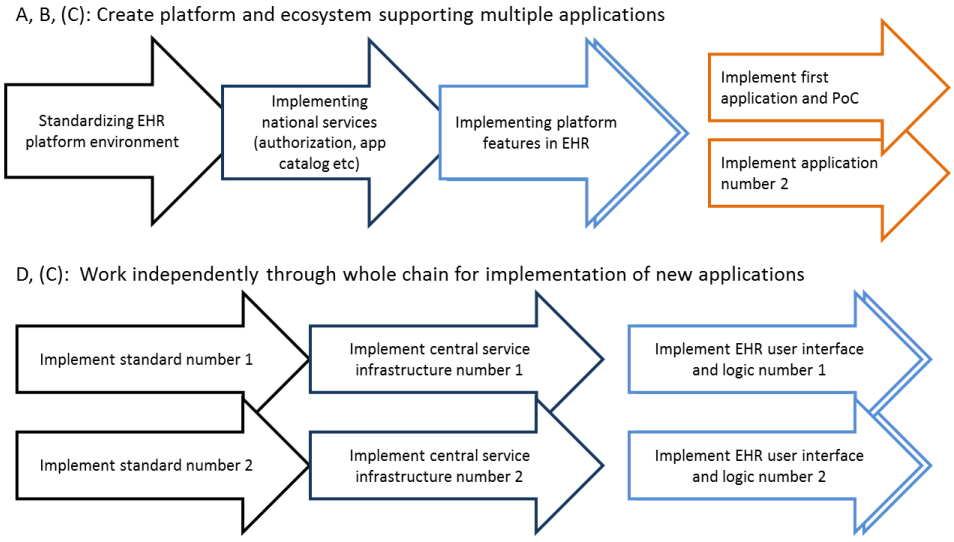


**Figure 28 shows the platform enablement phase followed by an innovation phase, in contrast to a model in which all innovation is created from scratch and involves all actors (Roland, Sanner, and Aanestad 2017).**

The paper illustrates how the five integration types enable different levels of change-flexibility and seamlessness. Given this framework, IT professionals can make more informed decisions regarding the choice of integration methods that are applicable to the priorities in their projects. The paper also discusses a promising new architecture pattern called CDS-Hooks, which potentially allows developers to combine both flexibility and seamlessness. The new technology introduces platform APIs that allow for closer integration between each application module and the core, and between independent application modules.

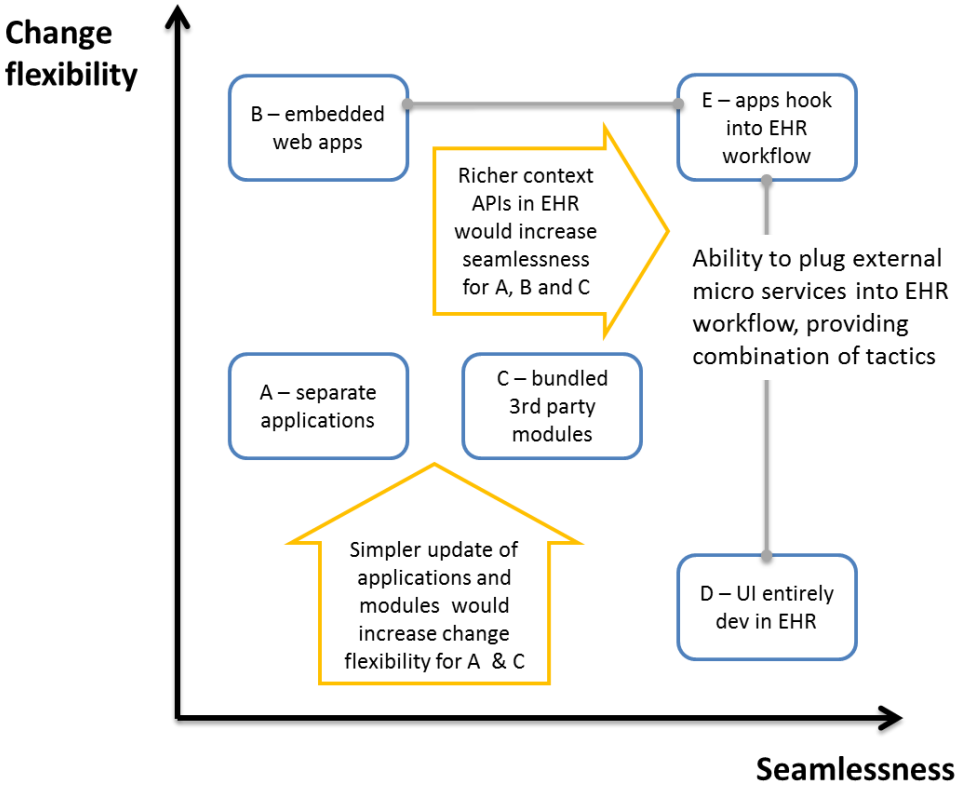


**Figure 29 shows the tradeoff in integration patterns between change-flexibility and seamlessness of integrated functions (Roland, Sanner, and Aanestad 2017).**

### 5.4.5 Contribution to Ph.D. research

This paper contributes to the current Ph.D. thesis by further elaborating and qualifying some of the platform aspects seen in "P for platform" through a discussion of how applications can be integrated into a platform architecture. The integration typology outlined in the paper can be used in the platform framework established in "P for platform", thus demonstrating how different integration types relate to platforms in a variety of ways and therefore exhibit specific types of flexibility. The research also corresponds to a more advanced, developed

world context and helps provide additional background that establishes the relevance of this thesis beyond a low-resource context. The paper also contributes significantly to an understanding of some of the key principles to take into consideration when shaping health information platforms, and identifies the tradeoffs involved.

# 6 Discussion

## 6.1 Challenges in deploying integrated health information systems

*"Which types of flexibility need to be supported in health information platforms to meet the challenges of heterogeneity?*

To understand more about how information system architectures can be designed, we need to consider which dimensions of heterogeneity place requirements on these systems. The first paper of this thesis (Roland et al. 2013) discusses this problem of heterogeneity in design, a problem also found elsewhere in the literature.

The requirement for a system to fit the exact needs of its users (Goodhue and Thompson 1995) often leads to different user groups being served by isolated applications or so-called silos (Bygstad and Hanseth 2016). As seen in Paper 2 and 4, the increasing need to integrate systems across user groups that have traditionally been catered for by separate systems has posed some interesting challenges for architecture and governance. These observations are also supported by other scholars (Ellingsen and Monteiro 2006).

As discussed in the introduction, a key driver of eHealth innovation projects is the move from bespoke development to generic, off-the-shelf systems that are configured to fit the local context (Fitzpatrick and Ellingsen 2013). These generic systems allow for the reuse of processes, design principles and learning across sites. In industries other than health, there is also a trend towards platform architectures where applications can reuse logic while innovating independently. This thesis focuses on furthering an understanding of platform concepts applied to health, or health information platforms.

The research literature lacks a discussion of cases of health information platforms in general, and, in particular, how these can be used to integrate functions across multiple user groups. Expanding on this topic is a central contribution of this thesis. To identify the specific challenges of deploying integrated health information platforms across multiple user groups and domains, it is necessary to examine which type of heterogeneity is manifested in large health information systems.

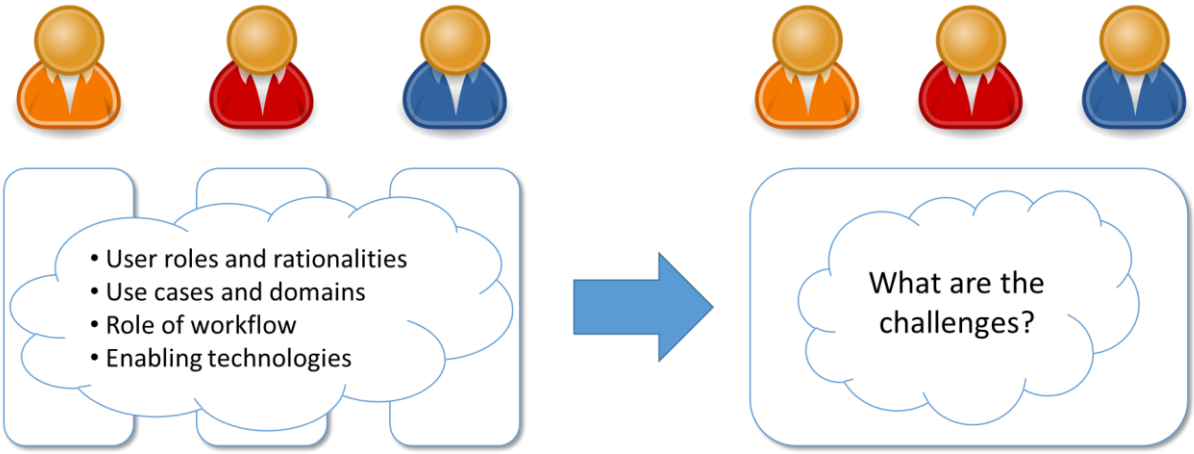


Figure 30 illustrates four different dimensions in which flexibility is required.

There are at least four overlapping areas in which generic requirements and capabilities may be different across multiple sites and thus represent elements of scale beyond mere numbers.

**Firstly, the *use cases and application domains* vary.** Some software tracks mothers throughout their pregnancies and the children throughout their first years; other software focuses on tracking patients with diseases such as HIV or TB, improving logistics and cold-chain support for the distribution of medicine, implementing results-based financing of health services, or any number of other use cases. Scaling across different use cases imposes new requirements on the solution, even if the user group itself is the same, as seen in Paper 1 and other literature (Fleck 1994; Fitzpatrick and Ellingsen 2013).

**Secondly, systems address *different user roles and rationalities*,** such as managers, doctors, midwives, nurses, counselors, or community health workers. Paper 1 describes how sometimes, the same health worker can also take on different roles and therefore also need to negotiate conflicting rationalities. This heterogeneity can impose conflicting requirements on the system, since the people in these roles understand and use the applications differently. Even for comparable use cases, there can be fundamental differences in underlying logic (Paper 1). We have seen in this research that in some cases the whole rationality behind tasks may be different, and that system design should be a process of negotiation and combination of these identified rationalities (Paper 1).

A similar concern arises when software is shared between different levels of a health organization hierarchy. The national, provincial, district, and facility levels have different requirements for the system and will use the data and software in different ways. The exact local configuration of these differentiated user needs also varies with the level of

decentralization and funding processes in each country. In the case of health programs funded by international organizations, information needs extend beyond national borders to donor agencies that wish to monitor how their money is being spent in each country (Shaw 2005; Braa et al. 2007).

**Thirdly, some systems may impose a more or less strict *workflow* on the user**, including, for example, the exact protocol to be followed when treating a patient (Both in Paper 4 and Berg 1997). This is particularly challenging in heterogeneous environments both within and between developing countries. There are huge differences between the workflows in large hospitals, clinics, mobile health posts and between community health workers, although in principle many of these may have similar tasks in terms of following up and treating patients (Lehmann and Sanders 2007). As Berg (1997) argues, protocols tend to focus on codeable data rather than the sensitivities of other social aspects; however, the clinicians' choice of action must also take into account several softer contextual considerations when executing protocols.

**Fourthly, there is the role of *new technologies*** such as mobile phones; these have further engaged the *outer levels of health services*, with health workers using mobiles to submit and retrieve health data about patients. This extended reach of national health information systems has further extended the challenges of generic software development, for example by including new users, adding new players such as mobile operators and introducing new issues in terms of technology choice, robustness, reimbursement and control of costs (Paper 3). Due to the complexities involved in their introduction, new silos often tend to form around certain new technologies and technical distribution channels, without appropriate integration with existing systems (Paper 3; Sanner, Manda, and Nielsen 2014).

The co-configuration between new and existing technologies can be a political process and should be sensitive to a number of local factors (Sahay, Monteiro, and Aanestad 2009). The co-configuration of technology such as locally available mobile phones with generic software can be subject to unintended effects that have a tremendous impact on the long-term success of projects in a negative or positive way (Paper 3). Technologies that are initially expected to contribute to flexibility and ease of generic development by designers may sometimes fail to meet the requirements on the ground, leading to design-reality gaps (Paper 3; Heeks 2006).

Technical configurability includes the devices on which the system is deployed, such as the choice of mobile phones, computers, tablets, etc. The generic software system is configured

locally by combining it with already existing or new mobile phones, adding a whole network of new and unforeseen issues and actors (ibid).

Large regional and national systems face challenges involving all four dimensions of scale described above: i) use cases, ii) user roles, iii) workflow, and iv) technology. These dimensions of heterogeneity are important in understanding the level and types of flexibility required in health information systems. Identifying heterogeneity therefore helps us to answer the question: *"Which types of flexibility need to be supported in health information platforms to meet the challenges of heterogeneity?"* From the above, we can deduce that flexibility is required in both use and design, and that flexibility in the time dimension as systems scale and change is also central to a successful scaling process. This flexibility cannot be centered at one point, but must be distributed to different actors taking on roles in various parts of the platform value chain (Paper 1 and 2).

I would claim that the solution to these problems lies in providing the right types of flexibility in the system, such that these challenges can be met at the appropriate levels of the development and deployment processes. The challenges must be addressed in turn by the developers of the off-the-shelf software, by the implementers and by the users themselves. Each of these actors must be offered the right types of flexibility and the necessary adjustments made. The next two sections present a further discussion of which kinds of flexibility are important, and how flexibility relates to the platform architecture, its actors and its governance.

## 6.2 The shaping of a platform, enforced by requirements of flexibility

*"How should flexibility in the platform be distributed to benefit the various actors in the wider platform ecosystem?"*

As discussed above, various types of flexibility are required in order to adapt the system to the use cases, domains, workflows, user roles and rationalities, and technologies seen in the large-scale health information domain. Increasing the robustness and flexibility of the solution increases the ability to scale (Paper 3).

Large health information infrastructures have users representing various rationalities (Paper 1). The design of the platform needs to keep these users in mind, and when applied across multiple generic settings, the system must be able to accommodate each rationality

throughout its design and use. As seen in Paper 2, however, no single solution can easily support a broad range of application domains and rationalities without being modularized into core and complementary functions with stable interfaces between them (Gawer 2014; Gawer 2009; Tiwana 2013). As such, according to Paper 1 and 2, accommodating these multiple rationalities involves identifying them and keeping them in mind during design, and combining them with an architecture that allows multiple rationalities in separate applications to run on a common core. Platform architectures enable this layering of flexibility and the distribution of design-flexibility to developers who have an appropriate understanding of the end-users' needs. The design process includes decisions on which types of flexibility should be handled locally in the platform chain or delegated downstream to other platform actors.

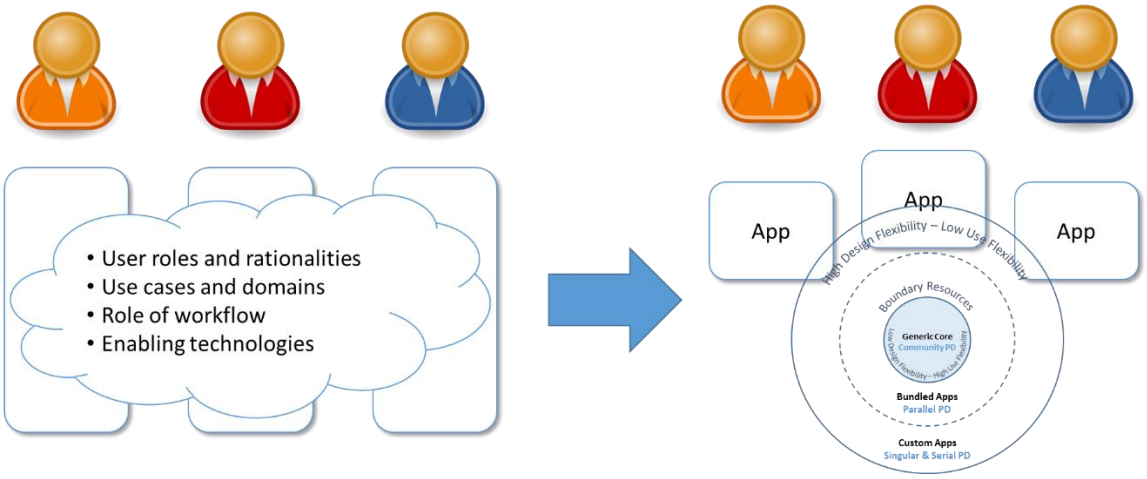


**Figure 31 illustrates how a platform allows the reuse of some core capabilities while providing the ability to specialize to each user group. The figure combines learning from all the articles in this thesis.**

Paper 2 outlines a platform model, and shows how the different tiers of the platform and the accompanying applications exhibit varying flexibility and enable different modes of participation from stakeholders. The generic core of the platform has low design-flexibility, but can be used by many different applications and users and therefore must have a high level of use-flexibility to be successful as a platform. The fringe of the platform, where loosely coupled applications reside, enables higher design-flexibility and can allow for more specialized applications that are targeted at less generic user scenarios. These applications are given design-flexibility through the option value of the platform's interfaces and modularity.

If the application designers wish, they can design strong inscriptions into the applications that restrict use-flexibility and only allow the application to be used for certain very specialized tasks. The designer of the individual applications running on the platform can choose to

decrease the use-flexibility of these applications without affecting the use-flexibility of the core platform. Within the health domain, enforcing strong inscriptions and thus restricting use-flexibility may, for example, be a tool to force adherence to certain regulatory requirements.

## 6.3 Delegating flexibility in health information platforms

*"How should flexibility in the platform be distributed to benefit the various actors in the wider platform ecosystem?"*

The circular platform figure (Figure 25) from Paper 2 illustrates a layered platform architecture and shows how this can provide different types of flexibility at different layers of the architecture, accommodating the need for participation from users. This section uses the findings from the four papers to elaborate on how flexibility is divided between actors in the different parts of the platform and the platform value chain.

There are some important points to note when discussing flexibility within platforms, which were not covered in Paper 2, "P for Platform". The actors in the platform value chain are not homogeneous, and there is both tension and cooperation between them. In addition, the level of flexibility is not merely a matter of technical architecture but is a political process linked to concepts of governance. The role of flexibility in platforms therefore requires more discussion with regard to both actors and governance.

In this section, we expand on the platform model described in "P for Platform" (Paper 2) and discuss how the roles and governance decisions of some actors in the value chain affect the level and types of flexibility for others. This discussion uses concepts described in both the platform and information infrastructure literature.

Figure 32 is built on the circular platform model shown in Figure 25 and taken from "P for Platform" (Paper 2), and the following discussion uses this diagram to describe the various types of flexibility and how different actors relate to flexibility-related concepts. The diagram shows the platform circles as part of a linear platform value chain, with the core functions upstream and the users downstream, moving from left to right in the diagram.

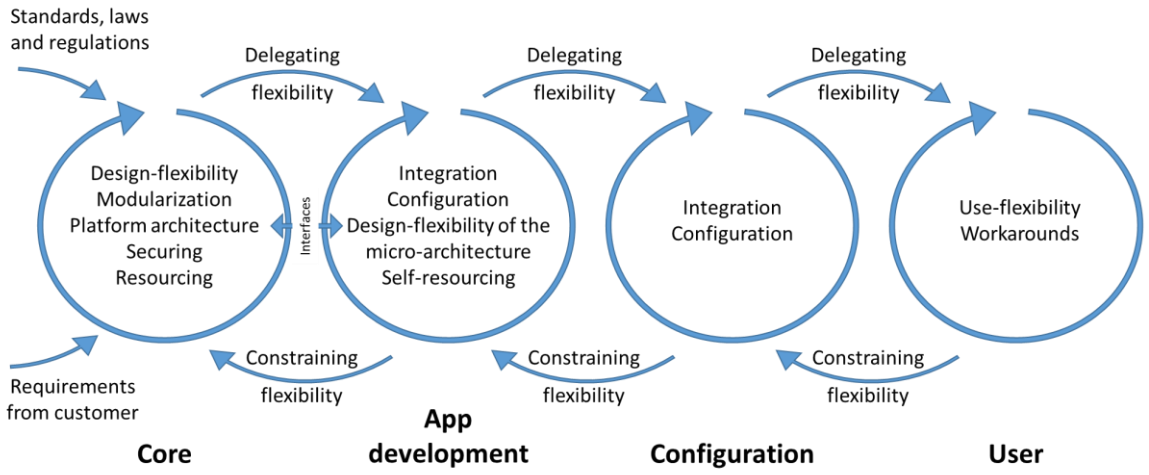


**Figure 32 shows a framework for analyzing how different roles relate to types of flexibility.**

In this thesis, I coin the term *delegating flexibility*, which encapsulates how an upstream actor in a platform value chain makes a conjoint governance and architecture decision that decides how much flexibility to distribute to downstream actors.

Although the term 'delegating flexibility' is not explicitly used in the papers in this thesis, they all describe the empirical process of platformization to which this concept applies (Paper 2, 3 and 4). Delegating flexibility is a governance decision and is related to other terms used in literature such as securing and resourcing (Ghazawneh and Henfridsson 2013), inscriptions (Monteiro and Hanseth 1996), modularity and option value (Tiwana 2013). Tiwana (2013) defines governance, in the context of platforms, as "who makes which decisions and how the 'pie' is split among the platform owner and the app developers." This thesis claims that the decision as to who has design- and use-flexibility is a central part of these governance decisions. While platform owners and other actors in the ecosystem delegate flexibility, it is also worth noting that delegation does not necessarily imply that actors accept or use what has been delegated (Oncken Jr and Wass 1974). The actor to whom flexibility is delegated may ignore it, ask for the delegator to do the tasks instead or pass this flexibility on to others.

The downstream actors will also affect the flexibility of upstream actors, although this may seem counterintuitive. As Paper 3 shows, path dependencies and irreversibility caused by scaling, increased usage and the success of functions and interfaces make it difficult for upstream actors to change existing decisions.

These feedback effects in information infrastructures and platforms are not new (Latour 1987; Callon 1991; Hanseth and Monteiro 1998); however, the ways in which they relate to

the platform architecture and the roles in the value chain have not been extensively discussed. These feedback concepts are therefore discussed further in Section 6.3.5.

### 6.3.1 Design-flexibility of the core vendor

As described in Paper 2 "P for Platform", the types of flexibility vary in different parts of the platform. However, the flexibility will also depend on the role of the actor and their position in the platform value chain. A core platform developer has design-flexibility in terms of platform functions and architecture, but has no direct design-flexibility regarding the external applications, which are developed by external application developers. The core developer can, however, affect the downstream actors through the architecture, modularization and governance choices in the core.

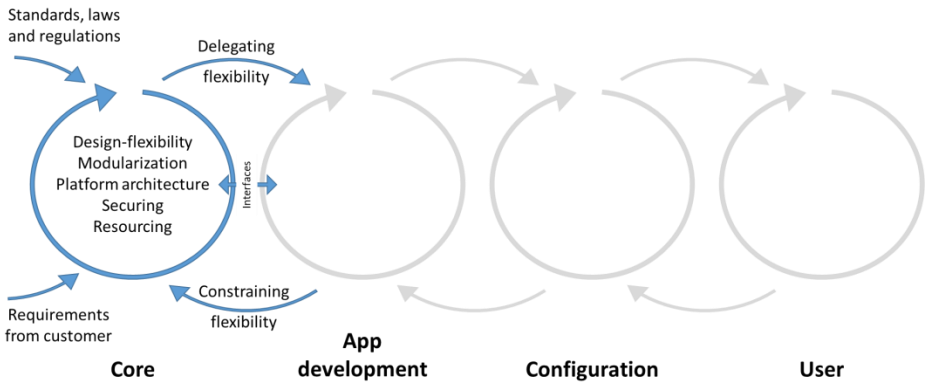


**Figure 33 shows how the core vendor can delegate flexibility, but also experiences limits to flexibility caused by downstream lock-in effects.**

The platform owner should in principle have full design-flexibility for their platform, but this flexibility is limited by the installed base, the path dependencies and other lock-in effects of scale (Paper 3). These restrictions will be discussed further in Section 6.3.5.

In the health domain, vendors are also subject to stringent regulations regarding which functions and interfaces they offer, again restricting their design-flexibility. Paper 4 discusses the effects of such regulations, indicating that there is also sometimes inertia at the core vendor in terms of implementing regulatory requirements. In public health information systems where the EHR vendors sell their systems to health organizations, customer requirements also play a significant role in defining the platform architecture. The vendor's decision on core functions, or resourcing, as described by Ghazawneh and Henfridsson (2013), is therefore typically based on a mix of proprietary strategy, external feedback and external forces such as standards, laws, and regulations. However, the ability of large, generic

software vendors to follow numerous specific requirements is limited (Pollock and Williams 2009).

The platform vendor may include specific applications, components from other vendors and integration between external applications as part of the platform. This inclusion is an enveloping mechanism where the platform vendor takes responsibility for specific application areas (Paper 4 and Tiwana 2013). This kind of generic application development is typically based on an identified need from large user communities rather than single customers. The introduction of shared functions typically affects the core of the platform and may introduce new interfaces that also can be used by external applications (Paper 2 and 4, but also supported by Pollock and Williams 2009; Tiwana 2013).

Core platform vendors may focus on enabling technologies and interfaces rather than the actual applications (Paper 4). In these cases, core vendors do not implement their own applications, but delegate design-flexibility to application developers or implementers to create applications downstream.

As described in the platform literature, the core vendor has several ways of securing and governing the platform, although the governance of downstream actors is a process of orchestration/cultivation rather than direct control (Tiwana 2013; Ghazawneh and Henfridsson 2013; Ciborra and Hanseth 1998). Using infrastructure concepts, it can be said that governance is achieved by introducing inscriptions that are intended to encourage or prevent specific behavior from actors downstream.

### 6.3.2 Restricted design-flexibility of the application developers and implementers

In Figure 32, the next two circles are “App development” and “Configuration.” The distinction between these processes may be more theoretical than practical, and in real life these functions might blend into each other.

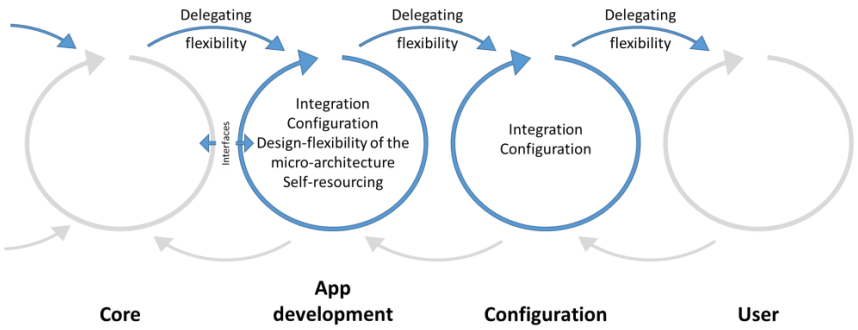


**Figure 34 shows the next two elements in the value chain, "App development" and "Configuration".**

App development involves using the interfaces offered by the core platform to implement functions within an application's microarchitecture. This development can include the use of interfaces to communicate or integrate with other applications or the core platform.

Configuration is an action typically carried out by the implementer; this is the case for DHIS2, where there are professional implementers who travel from site to site and help put the solution into production (Paper 2, but also in Titlestad, Staring, and Braa 2009; Fleck 1994; Fleck 1993). The configuration may include non-technical and strategic choices regarding which functions to include or exclude, and decisions on combining multiple products or platforms in an overall architecture (Sahay, Monteiro, and Aanestad 2009). In open source systems such as DHIS2, the configuration could also include developing custom source code or modifying source code within the system. It has become more common in DHIS2 to make these changes in new applications that run on top of the DHIS2 platform's interfaces, rather than modifying the platform core itself (Paper 2). When the core team observes that these applications implement functions that could be useful across the community, these functions are enveloped into the core.

App developers have design-flexibility within their apps and are generally relatively free to define the app's functions and microarchitecture. This design-flexibility can be restricted or extended by the core vendor or other external parties that secure and resource the platform as part of a governance process (Paper 2 and 3, but also Ghazawneh and Henfridsson 2013). We describe this action of deliberately passing flexibility to a downstream actor using the term delegated flexibility. The extent of delegated flexibility varies based on the governance principles of the platform owner, as well as other external factors. Paper 3, for example, describes the role of the mobile operator in restricting the options for the mobile applications running within the mobile ecosystem. Paper 4 describes how government regulations on

medication prescriptions are intended to shape the functions of the EHR prescription modules.

When delegating flexibility downstream, the core developer allows external implementers and application developers to make design-oriented changes. For example, when implementing an EHR or HMIS in an institution or country, the local implementer may add applications or in some cases even make changes to the source code of the product. In this case, the core vendor delegates design-flexibility to the implementer by giving access to interfaces and source code on which application development can be carried out.

### 6.3.3 Use-flexibility for organizations and users

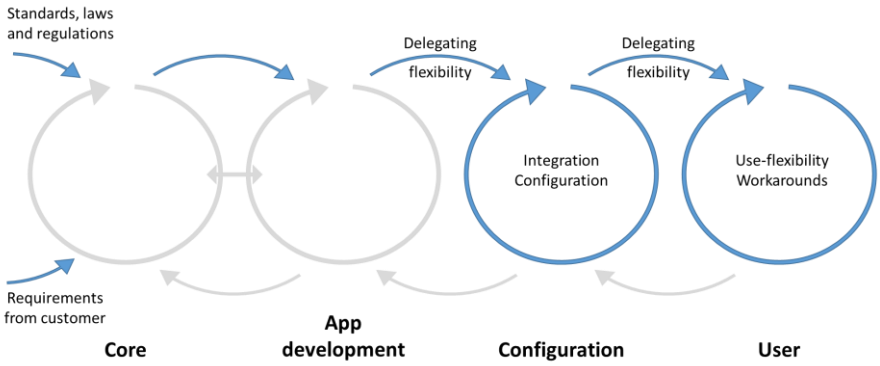


**Figure 35 shows how organizations and their users relate mainly to integration, configuration, use-flexibility and workarounds. Flexibility is delegated from upstream players. In health information systems, standards, regulations and laws also play an important role in restricting or delegating flexibility.**

Some user-level configuration may be done by the organization or end users without any actual development. This configuration would be subject to the use-flexibility offered by the particular application and platform. Use-flexibility has been delegated from upstream actors using more or less strict inscriptions in the system (Paper 2).

An application may be intended to have more or less use-flexibility through a varying strictness of the inscriptions of the intended work processes. In a medical setting, there would be regulation-based securing that mandates strong inscriptions and constrains users from configuring their systems (Paper 1 and 4). As an example of the need to limit use-flexibility, consider a case where the application displays critical allergy information as part of the workflow. In this case, the health worker should not be allowed to bypass this display by configuring the applications differently. The existence of strict external regulations thus partly inhibits the ability to delegate flexibility to end users in health information systems.

Both the ePrescription and Summary Care Record cases described in Paper 4 have these types of restrictions on use-flexibility based on regulatory requirements.

According to Silver (1990), this restrictiveness may have positive traits such as promoting normative approaches, increasing coordination and consistency, providing structure to processes and fostering structured learning. Less restrictive systems support multiple or changing decision-making environments, promote creativity and foster exploratory learning (ibid). In health information systems, a careful balance between these effects of restrictiveness and flexibility is important (Paper 4).

The level of flexibility that users see in artifacts is interpretative, and is subject to the users' understanding (Paper 1 and Sahay and Robey 1996). One example of this would be the midwife described in Paper 1, who, despite being told that she could affect the solution, jokingly said that the only way to confront non-working systems was to quit. She presumably did not believe her voice would be heard when designing the system.

### 6.3.4 Last resort workarounds

The applications and platform may have very strong inscriptions that prevent the user from handling the tasks they wish to complete (Paper 1). Such inscriptions may be intended or unintended (ibid). In some cases, the use-flexibility of the application is so low that the only solution for the end user is to use workarounds outside of the formal system (Paper 4 and Damsleth 2013; Gebauer and Schober 2006). These workarounds could, for example, include writing information on a piece of paper or sharing sensitive health information through other means such as external cloud solutions. Many workarounds might be prohibited by laws and regulations, and may often oppose the intentions of the original designer; however, in some cases, the health worker may still consider such workarounds necessary to complete their tasks.

### 6.3.5 Restrictions on flexibility due to lock-in effects

Both design- and use-flexibility may be affected by path dependencies, which are earlier decisions that restrict future possibilities (Paper 3 and Tiwana 2013). A core platform developer may have less design-flexibility as time goes by, since the interfaces are virtually impossible to change when many independent application developers use them. These are

examples of early architecture decisions with network-driven lock-in effects, and this effect is shown as feedback arrows with the text “Constraining flexibility” in Figure 32.

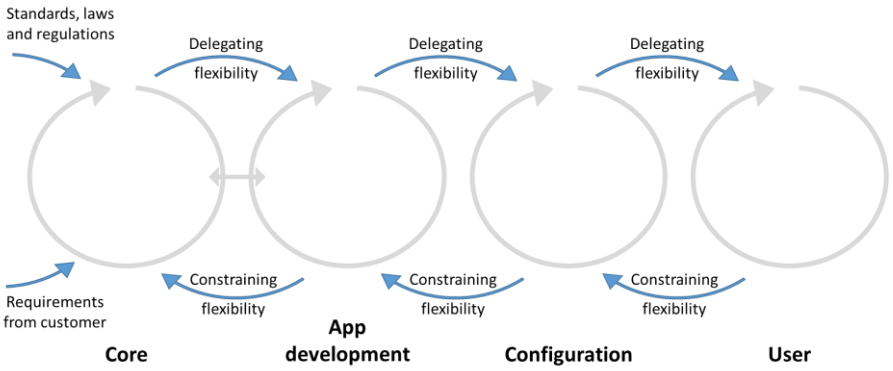


**Figure 36 shows how delegating flexibility works downstream, and the resulting constraining effect caused by downstream actors leveraging the flexibility and causing lock-in for upstream actors.**

These restricting factors are fed back from downstream actors such as app vendors and users who use and depend on the platform and its interfaces (Paper 3). The success of the platform architecture comes at the cost of flexibility, since increased scale causes path dependencies (ibid). Braa et al. (2007) write that although in principle interfaces and standardization promote flexibility, “…integration may cause less independence and less flexibility.” Tiwana (2013) claims that such lock-in effects can severely restrict the design-flexibility of core vendors, but that early modularization and interface choices can enable future flexibility. These are not the intended restrictions designed into the system, as described by Silver (1990), but are unintended restrictions on designers that arise from downstream patterns of use.

Opposite to the constricting path dependencies are real options, which provide future possibilities for change. The introduction of real options, for example through a modular architecture or flexible interfaces, offers possibilities for change and use in the future (Tiwana 2013; Trigeorgis 1993; MacCormack, Rusnak, and Baldwin 2006). Delegating flexibility generously to downstream players through such options may lead to innovation; however, this innovation may also come at the cost of future lock-in as the downstream players innovate in unintended ways. How the flexibility is constrained or delegated upstream affects the level of flexibility for the downstream roles, and also cycles back via the network effects of lock-in.

Figure 37 shows how the actors in the value chain have varying degrees of the different types of flexibility at their disposal.

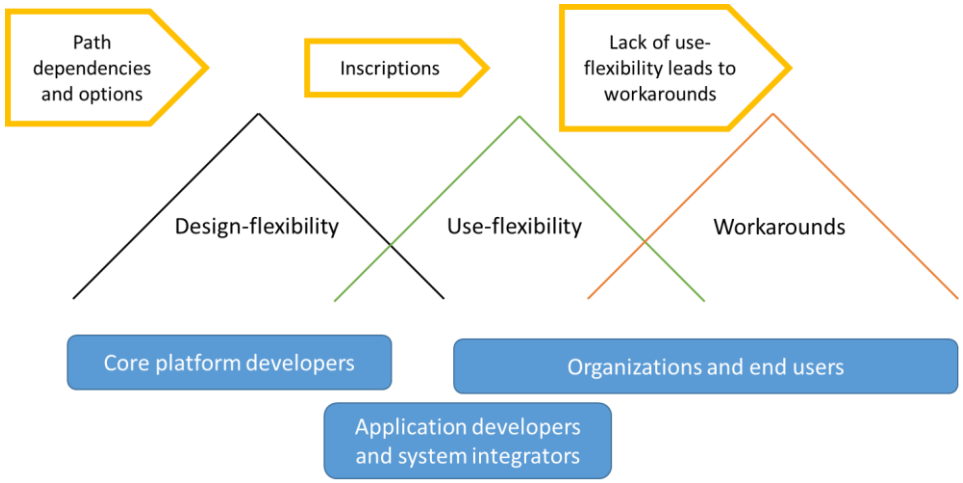


**Figure 37 shows how different types of flexibility are accessible to various actors. Core platform developers and application developers typically have design-flexibility for parts of the solution, while organizations and end users have use-flexibility and the ability to implement workarounds. The level of use-flexibility affects the need for external workarounds.**

### 6.3.6 Exemplifying flexibility delegation

The experience of teaching master students how to create applications on top of DHIS2, a platform in the making, provided some examples of how delegating flexibility worked in practice. In the two years during which I taught the course, there were 200 students, split into about 60 groups, who created applications using the interfaces of the immature DHIS2 platform.

In the first year, the capabilities of the interfaces were limited. The students started off with ideas about which applications to make, but had to adjust their plans to fit the actual interfaces. One example of a gap between the students' requirements and the actual platform functions was the need for data storage. DHIS2 had a rigid data model that was focused on certain types of health reporting, but the application developers wanted to store other types of data that did not fit into the model. The core developers, on the other hand, were unwilling to open up the data model, as it was central to how the aggregation and analysis of health information were implemented. Consequently, very little flexibility in terms of data storage was delegated to app developers, and most of the app developers failed to provide the full functionality that they had initially intended. The most creative app developers found external workarounds such as using web browser storage, and some even changed the source code of their DHIS2 instances to expand storage functions. Since DHIS2 is open source, the most

experienced software developers among the students self-resourced their storage flexibility (Ghazawneh and Henfridsson 2013).

In Year 2, the core developers had added the possibility of storing some data within DHIS2 in a very flexible manner. This storage was not linked to the data analysis functionality in DHIS2, and was intended to be used for storing non-secure application configurations and settings. The functionality was well received by the new group of students, and they used it eagerly. Some students also used this flexible storage to save sensitive, health-related data, which was not the intention of the core developers. In this case, the core developers had been generous in delegating flexibility, which in their opinion led to some unfortunate side-effects.

The quality of documentation also played an interesting role in how the students created applications. Since the documentation was weak in the first year, not all students discovered the full possibilities of the interfaces and the platform. Their interpretation of the flexibility was destroyed, not because of the lack of 'real' flexibility, but because of poor documentation. This shows that the resourcing of a platform also includes documentation and training.

The most knowledgeable students worked around the absence of documentation by reading the source code of DHIS2 to figure out how the interfaces worked. This issue of documentation illustrates how flexibility is interpretative, and that the same system can be considered as being more or less flexible based on the frame of reference of the viewer (Sahay and Robey 1996; Bijker 1987).

This chapter has shown that different actors relate to and delegate flexibility as part of the design and governance processes of platforms, but that not all health information systems start off as platform architectures. Paper 3 and 4 describe different types of systems and integration patterns that more or less adhere to the above platform principles, and discuss the process of becoming a platform, as well as certain tradeoffs involved in designing such systems. The following two sections discuss some of these tradeoffs in the context of flexibility delegation.

## 6.4 Enablement and innovation phases: upstream and downstream

*"Are there tradeoffs involved in distributing flexibility throughout health information platforms?"*

Paper 4 outlines five ways of integrating external functions into the workspace of an electronic health record (EHR). The paper considers the EHR as a multi-sided platform as described by Tiwana (2013), where the EHR vendor, national authorities, hospital IT departments, pharmacies, health workers and sometimes external developers engage in an ecosystem. The paper discusses how these five integration types differ, including how they handle change-flexibility and integration seamlessness.

Paper 4 explains that there is a platform enablement phase followed by an innovation phase where applications can be developed without strong interdependence with the platform. Tiwana (2013) explains this process in terms of the upstream vs. downstream parts of a value chain. The EHR vendor adds upstream features such as web application support, SMART-on-FHIR, CDS-Hooks and other, and the application developers use these features downstream. Paper 4 gives examples of how vendors have relinquished end-to-end control and allowed for external innovation by enabling downstream innovation. As described in an earlier section, the implementation of these features is a way of delegating flexibility to the application developers.

The paper also problematizes letting go of the responsibility for the user interface and integration, since the delegation of flexibility leads to less consistent user experiences and problems with seamlessness across tasks. For real seamlessness, the microarchitectures of each application must be aligned and integrated, creating a consistent user experience. This micro-architecture alignment is difficult when there is an underlying aim of breaking free from EHR lock-in and an implicit requirement for applications to multihome across platforms. The seamlessness issue illustrates the downside of flexibility delegation; however, it may be possible to counter these delegation drawbacks by introducing features that allow for more seamless integration between applications and the core, effectively introducing real options, as described earlier.

## 6.5 Analysis of Papers 2 and 4: designing in flexibility tradeoffs

*"Are there tradeoffs involved in distributing flexibility throughout health information platforms?"*

The empirical material in Paper 4 highlights potential conflicts of interest between national authorities pushing for certain changes and the EHR vendor holding these back, as explained by an informant: *"It is difficult to establish new functions. Our intention is to secure a good*

*cooperation chain in ePrescription, but they wish to sell their product to their customers. So the agenda is not the same."* In this case, the EHR vendor is the platform owner, who controls the core, and the downstream national authorities are trying to either develop an application on top of the EHR or to induce the EHR vendor themselves to make changes to the included set of functionalities. The modularity and degree of the platform capabilities in the EHR set the ground rules for how the national authorities can implement such changes.

The national authorities can also be considered as being upstream to the EHR vendor, and the national authorities' regulations as representing a platform-of-platforms strategy, where the national systems include multiple downstream platforms that the authorities try to affect. The national authorities could, therefore, be considered to be both upstream and downstream to the platform vendor. Regulations are used to both restrict and distribute the flexibility of the overall solution, as illustrated in Figure 32.

We can consider the findings of these papers in conjunction by applying the types of EHR integration in Paper 4 to the platform model derived from Paper 2. The comparison is considered in the innovation phase, after the platform features have stabilized.

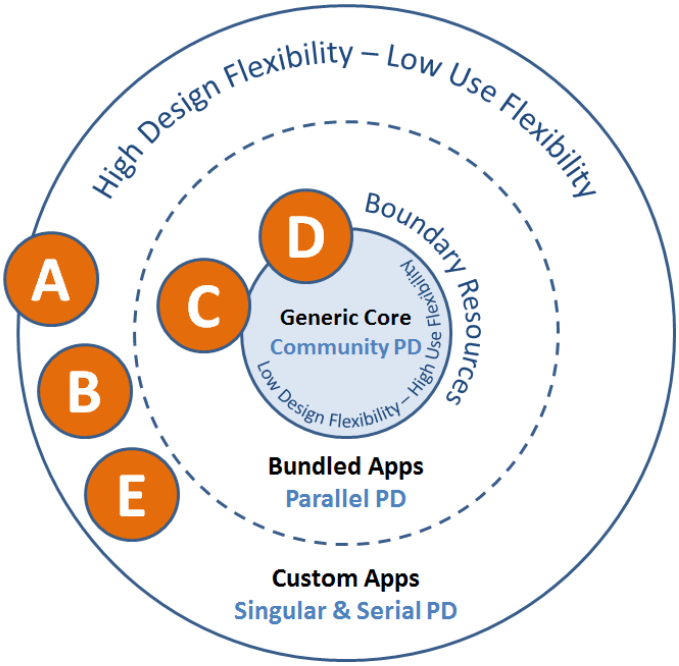


**Figure 38 shows the platform model from Paper 2 combined with the typology derived in Paper 4.**

Integration types A, B and E use the platform features of the EHR. Boundary resources such as APIs allow these applications to be integrated with the data and to some extent with the user experience of the rest of the EHR. For these application types, the platform vendor has

delegated flexibility and allowed for downstream innovation. The deepening specialization of healthcare may force EHR vendors to consider such platform concepts in order to stay competitive and innovative.

The bundled app (C) and the EHR-developed user interface (D) are still mainly within the control of the EHR vendor. For these applications, the EHR vendor takes more control of the amount of use-flexibility and ensures that applications can be utilized more consistently and supported better across their own customer base. A, B and E can, however, be easier to scale across EHR vendors; this concept is called multihoming by Tiwana (2013).

Paper 4 shows that implementations at the fringe vs. at the core of the platform have different pros and cons, and that strategic decisions on where to place functionality should be based on the needs for scaling (including multihoming) and seamlessness. Locking the application tightly into the ecosystem of the platform, as is the case for C and D, gives greater integration seamlessness across same-platform applications. Paper 4 proposes that alternative E, represented by technologies such as CDS-Hooks, may provide both seamlessness and design-flexibility. These technologies have strong linkages in the microarchitecture of the application and the platform architecture, allowing for stronger integration between the application and the core, and between applications, despite flexibility delegation. These concepts also promote standardized micro-architectures across different vendors' platforms, thus simplifying multihoming.

Whether this promise of flexibility holds true is a subject for future research. As Tiwana (2013) writes, there is a fine balance between integration and autonomy. A platform enforcing tight integration, both with the platform and across applications, leaves less room for developers to make their own choices. Therefore, the closer integration of applications to promote seamlessness may ultimately limit the individual flexibility of developers to innovate and multihome. There may be no solution to this tradeoff between flexibility and seamlessness. In the design and governance of health information platforms, flexibility must be delegated with care.

# 7 Conclusions

## 7.1 Summary of contributions

The overall findings of this research contribute to the information infrastructure and platform literature. The thesis set out to achieve a better understanding of how generic health information platforms, a particular class of health information systems, can be designed to be flexible and to adapt to different user scenarios, user roles, workflows, and technologies. Central to the contribution to platform architecture insights is a consideration of how heterogeneity can be handled by distributing flexibility among multiple actors. As defined by Tiwana (2013), the concept of platform governance is "who makes which decisions and how the 'pie' is split among the platform owner and the app developers." Understanding more about how flexibility is split and distributed therefore contributes to the understanding of health information platforms.

Paper 1 discusses how dealing with heterogeneity should play a major role in designing large-scale health information systems, and together with the discussion in Chapter 6 it addresses sub-RQ 1: *"Which types of flexibility need to be supported in health information platforms to meet the challenges of heterogeneity?"* Paper 2 described how platform architectures can facilitate a layered approach in which the layers and associated actors relate to flexibility in different ways. The paper addresses sub-RQ 2: *"How should flexibility in the platform be distributed to benefit the various actors in the wider platform ecosystem?"*; a question that is further expanded upon in in Section 6.3.

Papers 3 and 4 went on to address the tradeoffs and opportunities of such platform architectures, and hence addressed sub-RQ 3: *"Are there tradeoffs involved in distributing flexibility throughout health information platforms?"*

The thesis contributes to the health information systems literature with a classification of four dimensions of heterogeneity that pose challenges when health information systems grow into large-scale regional and national systems: i) use cases; ii) user roles; iii) workflow; and iv) technology. These dimensions are reflected as requirements for flexibility in large-scale health systems.

A contribution to the platform and information infrastructure literature is made through a discussion of health information platforms; this particularly focuses on the different types of flexibility required when establishing health information platforms, which roles require the different types of flexibility, and how the different layers of roles (core vendor, implementer, organization, end user) affect each other's flexibility through the use of inscriptions, options and cultivation of the installed base. The term *delegating flexibility* is coined as a concept that combines architecture design and governance in describing how flexibility is passed through a platform value chain. The thesis also links the tradeoff and lock-in effects caused by path dependency, and discusses how delegating flexibility can cause feedback effects that restrict upstream actors. Considering a platform as a solution where flexibility is delegated adds a new perspective when considering platforms, in addition to the two common business and technical views presented in literature (multi-sided and core-interface-apps respectively).

In response to the RQ: *"How can health information platforms handle heterogeneity by distributing flexibility among multiple actors?",* a claim is put forward that the most appropriate response to accommodating heterogeneous needs in an information system would be to offer layered support for user participation, flexibility, and robustness, and that these characteristics are best provided through a platform architecture where both use- and design-flexibility can be delegated with care to downstream vendors (Paper 1, 2 and 4). In recognizing that contexts and projects are different, it is not possible to state exact rules for how such flexibility should be distributed, but this discussion contributes to a better understanding of the process and need for delegating flexibility. Explaining the benefits and drawbacks of flexibility delegation can help in making better decisions when designing such health platforms.

The thesis identifies that the delegation of flexibility can have drawbacks such as lock-in and rogue use of flexibility. In health information systems, careful planning of flexibility delegation is necessary to avoid breaking regulations, usability and overall governance strategies. In some cases, flexibility may also need to be restricted. The discussion of flexibility delegation and awareness of path dependencies guides us towards answers to the overall research question.

### Contribution to method

This thesis draws on principles from the research methodologies of Action Design Research (Sein et al. 2011) and Networks of Action (Braa, Monteiro, and Sahay 2004), to combine artifact-focused action research with multi-case action research in developing countries. The combined execution and discussion of these methodologies could be considered a contribution to action research methodology.

### Future research: What is upstream and downstream in national HIS?

Tiwana's (2013) discussion of software ecosystem platforms relates mainly to settings where a single vendor creates a platform in competition with other platform providers, and one platform eventually establishes the dominant design to which other vendors then adhere. A single platform owner often controls the governance process within their ecosystem and it is this vendor's responsive balance between architecture decisions and governance that feeds the success of their platform. The platform owner enables or resources platform features upstream, and has limited control over whether functions are picked up by application vendors downstream. The owner can, at least in the early stages, cultivate upstream capabilities dependent on the uptake of the platform.

The environment for scaling public health information systems is different, especially when considering national and global health systems. This thesis discusses cases where there are several connected platforms, and where multiple actors have a role in owning and controlling the platforms, or in other ways take part in the governance process. There is a heterogeneous set of actors who together shape the overall flexibility of the system. Parts of the platform ownership lie with the Ministry of Health, national institutions, infrastructure providers, distributed health organizations within the country and the developer of the information system. There is no single actor involved in resourcing and securing the platform (Ghazawneh and Henfridsson 2013). Paper 2, "P for Platform", concerns how several such actors can participate in the creation of a platform and its complementary applications.

EHR platforms are not rivals in the same way as Android and IOS, for example. A health organization makes a very long-term decision on which EHR to buy, and switching costs are astronomical in comparison to switching mobile phones. Once installed, the EHR vendor becomes a dominant and long-term player for that particular organization's IT strategy, although increasing standardization helps open it up for cross-organization ecosystems.

Given appropriate standards and the willingness of the EHR vendor, external applications may run on different EHRs (multihome), both between hospitals that run the same EHR and among the various vendors.

In health, there is seldom a single actor who controls the platform on their own, and the platform is also subject to stronger regulations and a larger number of decision-makers. A number of different platforms are also often integrated together to fulfill the overall eHealth architecture of the health sector, such as several EHRs, national e-prescription and summary care record systems, which all could be considered individual multi-sided platforms. Further research is necessary to see how such a network of different types of platforms can be governed and how their architectures can be cultivated, where each platform fills different roles and is subject to various ownership and governance structures. It would therefore be useful for further research to identify a typology of platforms and how the architecture patterns and governance processes vary between the platform types.

It is debatable whether any large-scale platform success has been achieved in the health information space today, although certain vendors are making platform progress in this space (Christensen and Ellingsen 2016; Atalag et al. 2011; Mandl and Kohane 2015; Sellberg and Eltes 2017). Health systems are, however, still immature as platforms, and it is likely that the dominant design and related governance processes are yet to be established.

A topic that has gone unmentioned in this thesis is the concept of cloud computing, or solutions such as Infrastructure-as-a-Service, Platform-as-a-Service, Software-as-a-Service, Integration-Platform-as-a-Service etc. These recent ways of offering infrastructure and platform services may have a tremendous impact on how health information systems are provided and governed in the years to come. We can hope for extensive research within the platform and cloud domain applied to health systems, which will help us understand better how different governance models and architectures will shape the flexibility of the health information systems of tomorrow. This research may also help us categorize different types of platforms and understand better how various types of public sector platforms are linked and used in different ways.

## References


aidleap
2014 Is There Too Much “Innovation” in Development? AID LEAP. https://aidleap.org/2014/11/20/is-there-too-much-innovation-in-development/, accessed October 2, 2017.

Akrich, Madeleine
1992 The De-Scription of Technical Objects. Shaping Technology/Building Society: 205–224.

Althauser, C, R Andersen, K Gallo, et al.
2016 An Interim Review of the Health Information Systems Programme with Recommendations for Future Action. www.path.org. https://www.mn.uio.no/ifi/english/research/networks/hisp/hisp-assessment-report.pdf.

Anderson, Philip, and Michael L. Tushman
1990 Technological Discontinuities and Dominant Designs: A Cyclical Model of Technological Change. Administrative Science Quarterly 35(4): 604–633.

Atalag, Koray, Hong Yul Yang, Ewan Tempero, and James Warren
2011 Model Driven Development of Clinical Information Systems Using OpenEHR. https://researchspace.auckland.ac.nz/handle/2292/27894, accessed June 12, 2017.

Avital, Michel, and Dov Te’eni
2009 From Generative Fit to Generative Capacity: Exploring an Emerging Dimension of Information Systems Design and Task Performance. Information Systems Journal 19(4): 345–367.

Baldwin, Carliss Y., and Kim B. Clark
2006 The Architecture of Participation: Does Code Architecture Mitigate Free Riding in the Open Source Development Model? Management Science 52(7): 1116–1127.

Baldwin, Carliss Y., and C. Jason Woodard
2009 The Architecture of Platforms: A Unified View. *In* Platforms, Markets and Innovation. Edward Elgar.

Baskerville, Richard L.
1999 Investigating Information Systems with Action Research. Communications of the AIS 2(3es): 4.

Baskerville, Richard, and Michael D. Myers
2004 Special Issue on Action Research in Information Systems: Making Is Research Relevant to Practice–foreword. Mis Quarterly 28(3): 329–335.

Berente, Nicholas, and Youngjin Yoo

2012 Institutional Contradictions and Loose Coupling: Postimplementation of NASA's Enterprise Information System. Information Systems Research 23(2): 376–396.

Berg, Marc
1997 Problems and Promises of the Protocol. Social Science & Medicine 44(8): 1081–1088.
1999 Accumulating and Coordinating: Occasions for Information Technologies in Medical Work. Computer Supported Cooperative Work (CSCW) 8(4): 373–401.

Berntsen, Sindre Nicolaysen
2015 Enabling Participatory Design in Low Resource Contexts - A Study of Distributed Software Development of DHIS2. https://www.duo.uio.no/handle/10852/45112, accessed June 12, 2017.

Bijker, W. E., T. P. Hughes, and T. J. Pinch
1987 The Social Construction of Technological Systems: New Directions in the Sociology and History of Technology. the MIT Press.

Bijker, Wiebe E.
1987 The Social Construction of Bakelite: Toward a Theory of Invention. The Social Construction of Technological Systems: 159–187.

Bjerknes, Gro, and Tone Bratteteig
1995 User Participation and Democracy: A Discussion of Scandinavian Research on System Development. Scandinavian Journal of Information Systems 7(1): 1.

Braa, J, Ole Hanseth, Arthur Heywood, Woinshet Mohammed, and Vincent Shaw
2007 Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy. MIS Quarterly 31(2): 381–402.

Braa, J., E. Monteiro, and S. Sahay
2004 Networks of Action: Sustainable Health Information Systems across Developing Countries. Mis Quarterly: 337–362.

Braa, J., and S. Sahay
2012 Integrated Health Information Architecture. https://www.mn.uio.no/ifi/english/research/networks/hisp/integrated-health-information-architecture/integrated_health_information_architecture.pdf.

Braa, Jørn, and Calle Hedberg
2002 The Struggle for District-Based Health Information Systems in South Africa. The Information Society 18(2): 113–127.

Broyles, David, Brian E. Dixon, Ryan Crichton, Paul Biondich, and Shaun J. Grannis
2016 The Evolving Health Information Infrastructure. *In* Health Information Exchange: Navigating and Managing a Network of Health Information Systems. Elsevier Inc. https://indiana.pure.elsevier.com/en/publications/the-evolving-health-information-infrastructure, accessed August 29, 2017.

Bygstad, Bendik
2016 Generative Innovation: A Comparison of Lightweight and Heavyweight IT. Journal of Information Technology. https://link.springer.com/article/10.1057/jit.2016.15, accessed February 28, 2017.

Bygstad, Bendik, and Ole Hanseth
2016 Governing E-Health Infrastructures: Dealing with Tensions. ICIS 2016 Proceedings. http://aisel.aisnet.org/icis2016/ISHealthcare/Presentations/2.

Bygstad, Bendik, and Bjørn Erik Munkvold
2011 Exploring the Role of Informants in Interpretive Case Study Research in IS. Journal of Information Technology 26(1): 32–45.

Callon, Michel
1991 Techno-Economic Networks and Irreversibility. A Sociology of Monsters: Essays on Power, Technology and Domination 38: 132–161.

Carlile, Paul R.
2004 Transferring, Translating, and Transforming: An Integrative Framework for Managing Knowledge Across Boundaries. Organization Science 15(5): 555–568.

Carney, T. F.
1990 Collaborative Inquiry Methodology. Division for Instructional Development, University of Windsor.

Ceccagnoli, Marco, Chris Forman, Peng Huang, and D. J. Wu
2011 Co-Creation of Value in a Platform Ecosystem: The Case of Enterprise Software. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1824389, accessed June 20, 2017.

Cennamo, Carmelo, Hakan Ozalp, and Tobias Kretschmer
2016 Platform Architecture, Multihoming and Complement Quality: Evidence from the US Video Game Industry. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2839741, accessed August 29, 2017.

Christensen, Bente, and Gunnar Ellingsen
2016 Evaluating Model-Driven Development for Large-Scale EHRs through the OpenEHR Approach. International Journal of Medical Informatics 89: 43–54.

Christensen, Clayton M., and Richard S. Rosenbloom
1995 Explaining the Attacker's Advantage: Technological Paradigms, Organizational Dynamics, and the Value Network. Research Policy 24(2): 233–257.

Chua, Wai Fong
1986 Radical Developments in Accounting Thought. Accounting Review: 601–632.

Ciborra, Claudio
2002 The Labyrinths of Information: Challenging the Wisdom of Systems: Challenging the Wisdom of Systems. OUP Oxford.

Ciborra, Claudio U.
1997 De Profundis? Deconstructing the Concept of Strategic Alignment. Scandinavian Journal of Information Systems 9(1): 2.

Ciborra, Claudio U., and Ole Hanseth
1998 Toward a Contingency View of Infrastructure and Knowledge: An Exploratory Study. *In* Proceedings of the International Conference on Information Systems Pp. 263–272. Association for Information Systems. http://dl.acm.org/citation.cfm?id=353078, accessed January 27, 2014.

Cusumano, Michael
2010 Technology Strategy and Management The Evolution of Platform Thinking. Communications of the ACM 53(1): 32–34.

D'Amore, John D., Joshua C. Mandel, David A. Kreda, et al.
2014 Are Meaningful Use Stage 2 Certified EHRs Ready for Interoperability? Findings from the SMART C-CDA Collaborative. Journal of the American Medical Informatics Association 21(6): 1060–1068.

Damsleth, Wilhelm Arthur Sandberg
2013 Filling the Holes with Workarounds: Watching Maps Work in the Terrain. https://www.duo.uio.no/handle/10852/37420, accessed January 13, 2017.

David, Paul A., and Shane Greenstein
1990 The Economics of Compatibility Standards: An Introduction to Recent Research 1. Economics of Innovation and New Technology 1(1–2): 3–41.

Davison, Robert, Maris G. Martinsons, and Ned Kock
2004 Principles of Canonical Action Research. Information Systems Journal 14(1): 65–86.

Dearden, Andy
2012 See No Evil?: Ethics in an Interventionist ICTD. *In* Proceedings of the Fifth International Conference on Information and Communication Technologies and Development Pp. 46–55. ACM. http://dl.acm.org/citation.cfm?id=2160680, accessed June 6, 2017.

Definition of FLEXIBLE
N.d. https://www.merriam-webster.com/dictionary/flexible, accessed November 2, 2017.

DeRenzi, Brian, Leah Findlater, Jonathan Payne, et al.
2012 Improving Community Health Worker Performance through Automated SMS. *In* Proceedings of the Fifth International Conference on Information and Communication Technologies and Development Pp. 25–34. ICTD '12. New York, NY, USA: ACM. http://doi.acm.org/10.1145/2160673.2160677, accessed January 27, 2013.

Doherty, Neil F., Crispin R. Coombs, and John Loan-Clarke
2006 A Re-Conceptualization of the Interpretive Flexibility of Information Technologies: Redressing the Balance between the Social and the Technical. European Journal of Information Systems 15(6): 569–582.

Dougherty, Deborah, and Danielle D. Dunne
2011 Organizing Ecologies of Complex Innovation. Organization Science 22(5): 1214–1223.

Ellingsen, Gunnar, and Eric Monteiro
2006 Seamless Integration: Standardisation across Multiple Local Settings. Computer Supported Cooperative Work (CSCW) 15(5–6): 443–466.
2008 The Organizing Vision of Integrated Health Information Systems. Health Informatics Journal 14(3): 223–236.

Én Innbygger – Én Journal
2015 Ehelse.No. https://ehelse.no/strategi/n-innbygger-n-journal, accessed October 3, 2017.

EPJ Monitor Årsrapport 2008
2008. hiwiki.idi.ntnu.no/images/7/79/EPJ-monitor-2008-hovedrapport.pdf.

EPJ Monitor Årsrapport 2010
2010. hiwiki.idi.ntnu.no/images/c/c5/EPJ-monitor-2010-v1.2.pdf.

Fitzpatrick, Geraldine, and Gunnar Ellingsen
2013 A Review of 25 Years of CSCW Research in Healthcare: Contributions, Challenges and Future Agendas. Computer Supported Cooperative Work (CSCW) 22(4–6): 609–665.

Fleck, James
1993 Configurations: Crystallizing Contingency. International Journal of Human Factors in Manufacturing 3(1): 15–36.
1994 Learning by Trying: The Implementation of Configurational Technology. Research Policy 23(6): 637–652.

Friedberg, Mark W., Peggy G. Chen, Kristin R. Van Busum, et al.
2013 Factors Affecting Physician Professional Satisfaction and Their Implications for Patient Care, Health Systems, and Health Policy. Product Page. http://www.rand.org/pubs/research_reports/RR439.html, accessed March 7, 2017.

Furstenau, Daniel, and Carolin Auschra
2016 Open Digital Platforms in Health Care: Implementation and Scaling Strategies. ICIS 2016 Proceedings. http://aisel.aisnet.org/icis2016/ISHealthcare/Presentations/19.

Gammersvik, Øystein
2015 Balancing Institutional Needs with Generic Functionality in Information Systems: The Biography and Evolution of the DHIS 2 Tracker Following Two Implementations in Palestine. https://www.duo.uio.no/handle/10852/49051, accessed June 12, 2017.

Gasser, Les
1986 The Integration of Computing and Routine Work. ACM Trans. Inf. Syst. 4(3): 205–225.

Gawer, Annabelle
2009 Platforms, Markets and Innovation. Edward Elgar Publishing.
2014 Bridging Differing Perspectives on Technological Platforms: Toward an Integrative Framework. Research Policy 43(7): 1239–1249.

Gawer, Annabelle, and Michael A. Cusumano
2002 Platform Leadership: How Intel, Microsoft, and Cisco Drive Industry Innovation. Harvard Business School Press Boston.

Gebauer, Judith, and Franz Schober
2006 Information System Flexibility and the Cost Efficiency of Business Processes. Journal of the Association for Information Systems 7(3): 8.

Ghazawneh, Ahmad, and Ola Henfridsson
2013 Balancing Platform Control and External Contribution in Third-Party Development: The Boundary Resources Model. Information Systems Journal 23(2): 173–192.

Gizaw, A. A., A. Mukherjee, J. Lewis, and S. Sahay
2012 Understanding Design-Use Interaction: The Case of Name Based Health Information System in India. *In* Proceedings of IADIS International Conference on Information Systems.

Goodhue, D. L., and R. L. Thompson
1995 Task-Technology Fit and Individual Performance. MIS Quarterly: 213–236.

Gregor, Shirley
2006 The Nature of Theory in Information Systems. MIS Q. 30(3): 611–642.

Haggerty, J. L
2003 Continuity of Care: A Multidisciplinary Review. BMJ 327(7425): 1219–1221.

Halvorsen, Niclas Hammer
2015 Effektene av mobil- og data-mediert kommunikasjon i et helseinformasjonsystem. https://www.duo.uio.no/handle/10852/47650, accessed June 12, 2017.

Hanseth, O., and Lyytinen
2010 Design Theory for Dynamic Complexity in Information Infrastructures: The Case of Building Internet. Journal of Information Technology 25(1): 1–19.

Hanseth, O., and K. Lyytinen
2004 Theorizing about the Design of Information Infrastructures: Design Kernel Theories and Principles. http://sprouts.aisnet.org/124/, accessed September 5, 2012.

Hanseth, O., and E. Monteiro
1998 Understanding Information Infrastructure. Unpublished Manuscript, Http://Heim. Ifi. Uio. No/~ Oleha/Publications/Bok. Pdf. http://heim.ifi.uio.no/oleha/Publications/bok.pdf, accessed September 5, 2012.

Hanseth, O., E. Monteiro, and M. Hatling

1996 Developing Information Infrastructure: The Tension between Standardization and Flexibility. Science, Technology & Human Values 21(4): 407–426.

Hanseth, Ole, and Nina Lundberg
2001 Designing Work Oriented Infrastructures. Computer Supported Cooperative Work (CSCW) 10(3–4): 347–372.

Heeks, R.
2006 Health Information Systems: Failure, Success and Improvisation. International Journal of Medical Informatics 75(2): 125–137.

Heimly, V., A. Grimsmo, A. Faxvaag, and others
2011 Diffusion of Electronic Health Records and Electronic Communication in Norway. Applied Clinical Informatics 2(3): 355–364.

Hyysalo, S
2010 Health Technology Development and Use: From Practice-Bound Imagination to Evolving Impacts, vol.7. Routledge.

Idris, Nuran
2013 Collaborating with Global Level Partners to Computerize Health Logistics Management Information Systems in Tanzania and Zambia through Open Source Technology. *In* 141st APHA Annual Meeting (November 2-November 6, 2013). APHA.

Kalberg, Stephen
1980 Max Weber's Types of Rationality: Cornerstones for the Analysis of Rationalization Processes in History. American Journal of Sociology 85(5): 1145–1179.

Kallinikos, J
2004 Deconstructing Information Packages: Organizational and Behavioural Implications of ERP Systems. Information Technology & People 17(1): 8–30.

Kasthurirathne, Suranga N., Burke Mamlin, Harsha Kumara, Grahame Grieve, and Paul Biondich
2015 Enabling Better Interoperability for HealthCare: Lessons in Developing a Standards Based Application Programing Interface for Electronic Medical Record Systems. Journal of Medical Systems 39(11): 182.

Klein, Heinz K., and Michael D. Myers
1999 A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. MIS Quarterly: 67–93.

Koppel, Ross, and Christoph U. Lehmann
2015 Implications of an Emerging EHR Monoculture for Hospitals and Healthcare Systems. Journal of the American Medical Informatics Association 22(2): 465–471.

Korvald, Magnus
2013 Integrating Mobile and Web Health Infrastructures in Low Resource Contexts. https://www.duo.uio.no/handle/10852/37426, accessed June 12, 2017.

Kuipers, Pim, John S. Humphreys, John Wakerman, et al.
2008 Collaborative Review of Pilot Projects to Inform Policy: A Methodological Remedy for Pilotitis? Australia and New Zealand Health Policy 5: 17.

Kyng, Morten
2010 Bridging the Gap Between Politics and Techniques: On the next Practices of Participatory Design. Scandinavian Journal of Information Systems 22(1). http://aisel.aisnet.org/sjis/vol22/iss1/5.

Latour, Bruno
1987 Science in Action: How to Follow Scientists and Engineers through Society. Harvard university press.

Lazar, Jonathan, Jinjuan Heidi Feng, and Harry Hochheiser
2017 Research Methods in Human-Computer Interaction. Morgan Kaufmann.

Lee, Allen S., and Richard L. Baskerville
2003 Generalizing Generalizability in Information Systems Research. Information Systems Research 14(3): 221–243.

Lehmann, Uta, and David Sanders
2007 Community Health Workers: What Do We Know about Them? http://www.hciproject.org/chw-central/resources/community-health-workers-what-do-we-know-about-them-state-evidence-programmes-, accessed January 27, 2013.

#LetDoctorsBeDoctors
2015 ZDoggMD. http://zdoggmd.com/ehr-state-of-mind/, accessed November 18, 2017.

MacCormack, Alan, John Rusnak, and Carliss Y. Baldwin
2006 Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code. Management Science 52(7): 1015–1030.

Makam, Anil N., Holly J. Lanham, Kim Batchelor, et al.
2014 The Good, the Bad and the Early Adopters: Providers' Attitudes about a Common, Commercial EHR. Journal of Evaluation in Clinical Practice 20(1): 36–42.

Manda, T., M. Davis, S. Chawani, and C. Ngoma
2014 Software Architecture as Ground for Exercising Decision-Making Power in Distributed Software Development: A Case of DHIS Tracker. http://www.pdcpolitics.no/wp-content/uploads/2014/09/TDM-MSC-paper.pdf, accessed June 20, 2017.

Mandel, Joshua C., David A. Kreda, Kenneth D. Mandl, Isaac S. Kohane, and Rachel B. Ramoni
2016 SMART on FHIR: A Standards-Based, Interoperable Apps Platform for Electronic Health Records. Journal of the American Medical Informatics Association: ocv189.

Mandl, Kenneth D., and Isaac S. Kohane

2009 No Small Change for the Health Information Economy. New England Journal of Medicine 360(13): 1278–1281.
2012 Escaping the EHR Trap — The Future of Health IT. New England Journal of Medicine 366(24): 2240–2242.
2015 Federalist Principles for Healthcare Data Networks. Nature Biotechnology 33(4): 360–363.

Mandl, Kenneth D., Joshua C. Mandel, and Isaac S. Kohane
2015 Driving Innovation in Health Systems through an Apps-Based Information Economy. Cell Systems 1(1): 8–13.

Manya, Ayub, Jørn Braa, Lars Helge Øverland, et al.
2012 National Roll out of District Health Information Software (DHIS 2) in Kenya, 2011– Central Server and Cloud Based Infrastructure. *In* IST-Africa 2012 Conference Proceedings. http://www.academia.edu/download/35120789/ISTAfrica_Paper_ref_139_doc_4776.pdf, accessed August 5, 2017.

McCann, David
2012 A Ugandan MHealth Moratorium Is a Good Thing. ICT Works. http://www.ictworks.org/2012/02/22/ugandan-mhealth-moratorium-good-thing/, accessed October 2, 2017.

Mehl, Garrett, and Alain Labrique
2014 Prioritizing Integrated MHealth Strategies for Universal Health Coverage. Science 345(6202): 1284–1287.

Miles, Matthew B., and A. Michael Huberman
1994 Qualitative Data Analysis: An Expanded Sourcebook. Sage Publications, Incorporated.

Miles, Matthew B., A. Michael Huberman, and Johnny Saldaña
2013 Qualitative Data Analysis: A Methods Sourcebook. 3 edition. Thousand Oaks, Californinia: SAGE Publications, Inc.

MoH Norway
2017 Tildelingsbrev Til Direktoratet for E-Helse for 2017.

Monteiro, Eric, and Ole Hanseth
1996 Social Shaping of Information Infrastructure: On Being Specific about the Technology. Information Technology and Changes in Organizational Work: 325–343.

Mukherjee, Arunima, and Saptarshi Purkayastha
2010 Exploring the Potential and Challenges of Using Mobile Based Technology in Strengthening Health Information Systems: Experiences from a Pilot Study. https://works.bepress.com/saptarshi_purkayastha/1/, accessed June 19, 2017.

Namukwaya, Zikulah, Peter Mudiope, Adeodata Kekitiinwa, et al.
2011 The Impact of Maternal Highly Active Antiretroviral Therapy and Short-Course Combination Antiretrovirals for Prevention of Mother-to-Child Transmission on Early Infant

Infection Rates at the Mulago National Referral Hospital in Kampala, Uganda, January 2007 to May 2009. JAIDS Journal of Acquired Immune Deficiency Syndromes 56(1): 69–75.

Neumann, Laura J., and Susan Leigh Star
1996 Making Infrastructure: The Dream of a Common Language. *In* PDC Pp. 231–240. http://rossy.ruc.dk/ojs/index.php/pdc/article/view/153, accessed September 1, 2017.

Ngabo, Fidele, Judith Nguimfack, Friday Nwaigwe, et al.
2012 Designing and Implementing an Innovative SMS-Based Alert System (RapidSMS-MCH) to Monitor Pregnancy and Reduce Maternal and Child Deaths in Rwanda. The Pan African Medical Journal 13.

Obendorf, Hartmut, Monique Janneck, and Matthias Finck
2009 Inter-Contextual Distributed Participatory Design. Scandinavian Journal of Information Systems 21(1). http://aisel.aisnet.org/sjis/vol21/iss1/2.

Oncken Jr, William, and Donald L. Wass
1974 Management Time: Who's Got the Monkey? Harvard Business Review 52(6): 75–80.

Orlikowski, Wanda J.
1992 The Duality of Technology: Rethinking the Concept of Technology in Organizations. Organization Science 3(3): 398–427.

Orlikowski, Wanda J., and Jack J. Baroudi
1991 Studying Information Technology in Organizations: Research Approaches and Assumptions. Information Systems Research 2(1): 1–28.

Orlikowski, Wanda J., and C. Suzanne Iacono
2001 Research Commentary: Desperately Seeking the "IT" in IT Research—A Call to Theorizing the IT Artifact. Information Systems Research 12(2): 121–134.

Øverland, Lars Helge
2006 Global Software Development and Local Capacity Building : A means for improving Sustainability in Information Systems Implementations. https://www.duo.uio.no/handle/10852/9565, accessed June 19, 2017.

Papas, N., R. M. O'Keefe, and P. Seltsikas
2011 The Action Research vs Design Science Debate: Reflections from an Intervention in EGovernment. European Journal of Information Systems. http://www.palgrave-journals.com/ejis/journal/vaop/ncurrent/abs/ejis201150a.html, accessed September 3, 2012.

Parnas, D. L.
1972a A Technique for Software Module Specification with Examples. Commun. ACM 15(5): 330–336.
1972b On the Criteria to Be Used in Decomposing Systems into Modules. Commun. ACM 15(12): 1053–1058.

Payne, Thomas H., Sarah Corley, Theresa A. Cullen, et al.

2015 Report of the AMIA EHR-2020 Task Force on the Status and Future Direction of EHRs. Journal of the American Medical Informatics Association 22(5): 1102–1110.

Pollock, Neil, and Robin Williams
2008 Software and Organisations: The Biography of the Enterprise-Wide System or How SAP Conquered the World. Routledge.
2009 Global Software and Its Provenance: Generification Work in the Production of Organisational Software Packages. *In* Configuring User-Designer Relations Pp. 193–218. Springer. http://link.springer.com/chapter/10.1007/978-1-84628-925-5_9/fulltext.html, accessed April 1, 2014.
2010 E-Infrastructures: How Do We Know and Understand Them? Strategic Ethnography and the Biography of Artefacts. Computer Supported Cooperative Work (CSCW) 19(6): 521–556.

Pollock, Neil, Robin Williams, and Rob Procter
2003 Fitting Standard Software Packages to Non-Standard Organizations: The 'Biography' of an Enterprise-Wide System. Technology Analysis & Strategic Management 15(3): 317–332.

Requejo, J, J Bryce, and C Victora
2012 The 2012 Report - Countdown to 2015 - WHO. http://www.countdown2015mnch.org/documents/2012Report/2012-Complete.pdf, accessed January 27, 2013.

Roland, Lars Kristian, Terje Aksel Sanner, and Margunn Aanestad
2017 Flexibility in EHR Ecosystems: Five Integration Strategies and Their Trade-Offs. *In* .

Roland, Lars Kristian, Terje Aksel Sanner, Johan Ivar Sæbø, and Eric Monteiro
2017 P for Platform. Scandinavian Journal of Information Systems 30(2).

Roland, Lars, Terje Sanner, Prosper Behumbiize, Zikulah Namukwaya, and Kristin Braa
2013 Accommodating Multiple Rationalities in Patient Oriented Health Information System Design. *In* IRIS.

Røsok, Ole Andreas
2015 Interface Standards in HMIS. https://www.duo.uio.no/handle/10852/47655, accessed June 12, 2017.

Sæbø, Johan Ivar
2013 Global Scaling of Health Information Infrastructures: Circulating Translations. https://www.duo.uio.no/handle/10852/37463, accessed June 20, 2017.

Sahay, Sundeep, Eric Monteiro, and Margunn Aanestad
2009 Configurable Politics and Asymmetric Integration: Health e-Infrastructures in India. Journal of the Association for Information Systems 10(5): 399–414.

Sahay, Sundeep, and Daniel Robey
1996 Organizational Context, Social Interpretation, and the Implementation and

Consequences of Geographic Information Systems. Accounting, Management and Information Technologies 6(4): 255–282.

Sahay, Sundeep, Johan Sæbø, and Jørn Braa
2013 Scaling of HIS in a Global Context: Same, Same, but Different. Information and Organization 23(4): 294–323.

Sahay, Sundeep, and Geoff Walsham
2006 Scaling of Health Information Systems in India: Challenges and Approaches. Information Technology for Development 12(3): 185–200.

Sanner, Terje Aksel
2015 Grafting Information Infrastructure: Mobile Phone-based Health Information System Implementations in India and Malawi. https://www.duo.uio.no/handle/10852/45871, accessed June 20, 2017.

Sanner, Terje Aksel, Tiwonge Davis Manda, and Petter Nielsen
2014 Grafting: Balancing Control and Cultivation in Information Infrastructure Innovation. Journal of the Association for Information Systems 15(4): 220.

Sanner, Terje Aksel, Lars Kristian Roland, and Kristin Braa
2012 From Pilot to Scale: Towards an MHealth Typology for Low-Resource Contexts. Health Policy and Technology 1(3): 155–164.

Sanner, Terje Aksel, and Johan Ivar Sæbø
2014 Paying per Diems for ICT4D Project Participation: A Sustainability Challenge. Information Technologies & International Development 10(2): pp–33.

Schilling, Melissa A.
2000 Toward a General Modular Systems Theory and Its Application to Interfirm Product Modularity. Academy of Management Review 25(2): 312–334.

Sein, Ola Henfridsson, Sandeep Purao, Matti Rossi, and Rikard Lindgren
2011 Action Design Research. MIS Q. 35(1): 37–56.

Sellberg, Nina, and Johan Eltes
2017 The Swedish Patient Portal and Its Relation to the National Reference Architecture and the Overall EHealth Infrastructure. *In* Information Infrastructures within European Health Care Pp. 225–244. Health Informatics. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-319-51020-0_14, accessed November 20, 2017.

Shaw, Vincent
2005 Health Information System Reform in South Africa: Developing an Essential Data Set. Bulletin of the World Health Organization 83(8): 632–636.
2009 A Complexity INSPIRED APPROACH TO CO-EVOLUTIONARY HOSPITAL MANAGEMENT INFORMATION SYSTEMS DEVELOPMENT. University of Oslo Norway 21.

http://folk.uio.no/vshaw/Files/VShaw%20Kappa%20Final%20Version/1_V_Shaw_Complete%20Thesis.pdf, accessed May 16, 2017.

Shaw, Vincent, Shegaw Mengiste, and Jorn Braa
2007 Scaling of Health Information Systems in Nigeria and Ethiopia-Considering the Options. *In* IFIP WG. https://www.researchgate.net/profile/Vincent_Shaw/publication/228707091_Scaling_of_health_information_systems_in_Nigeria_and_Ethiopia-Considering_the_options/links/544695650cf2f14fb80f824f.pdf, accessed May 16, 2017.

Shidende, Nima Herman, Marlen Stacey Chawani, and Jens Kaasbøll
2014 Challenges in Participation and Implications on Human Development: Experiences from Health Information Systems Implementations in Tanzania and Malawi. http://www.itu.dk/people/ydi/final/participation4d_submission_3.pdf, accessed June 20, 2017.

Shidende, Nima Herman, Miria Grisot, and Margunn Aanestad
2014 Coordination Challenges in Collaborative Practices in the Prevention of Mother to Child Transmission of HIV in Tanzania. Journal of Health Informatics in Africa 2(1). https://www.jhia-online.org/index.php/jhia/article/view/88, accessed June 12, 2017.

Shidende, Nima Herman, and Christina Mörtberg
2014 Re-Visiting Design-after-Design: Reflecting Implementation Mediators Connectedness in Distributed Participatory Design Activities. *In* Proceedings of the 13th Participatory Design Conference: Research Papers - Volume 1 Pp. 61–70. PDC '14. New York, NY, USA: ACM. http://doi.acm.org/10.1145/2661435.2661437, accessed June 19, 2017.

Silver, Mark S.
1990 Decision Support Systems: Directed and Nondirected Change. Information Systems Research 1(1): 47–70.

Simon, Herbert A.
1996 The Sciences of the Artificial. MIT Press.

Smolij, Kamila, and Kim Dun
2006 Patient Health Information Management: Searching for the Right Model. Perspectives in Health Information Management / AHIMA, American Health Information Management Association 3. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2047307/, accessed October 3, 2017.

Sturmberg, Joachim P., and Carmel Martin
2013 Handbook of Systems and Complexity in Health. Springer Science & Business Media.

Tamrat, Tigest, and Stan Kachnowski
2012 Special Delivery: An Analysis of MHealth in Maternal and Newborn Health Programs and Their Outcomes Around the World. Maternal and Child Health Journal 16(5): 1092–1101.

Tansley, Arthur G.
1935 The Use and Abuse of Vegetational Concepts and Terms. Ecology 16(3): 284–307.

Titlestad, Ola, Knut Staring, and Jørn Braa
2009 Distributed Development to Enable User Participation: Multilevel Design in the HISP Network. Scandinavian Journal of Information Systems 21(1). http://aisel.aisnet.org/sjis/vol21/iss1/3.

Tiwana, Amrit
2013 Platform Ecosystems: Aligning Architecture, Governance, and Strategy. Newnes.

Trigeorgis, Lenos
1993 Real Options and Interactions with Financial Flexibility. Financial Management: 202–224.

Walsham, G.
1995 Interpretive Case Studies in IS Research: Nature and Method. European Journal of Information Systems 4(2): 74–81.

Walsham, Geoff
2006 Doing Interpretive Research. European Journal of Information Systems 15(3): 320–330.

WHO
1994 WHO Home Based Maternal Records - Guidelines. WHO.

Wroblewski, Luke
2011 Mobile First. First Edition edition. New York, NY: A Book Apart.

Yin, Robert K.
2013 Case Study Research: Design and Methods. Sage publications.

Yoo, Youngjin, Ola Henfridsson, and Kalle Lyytinen
2010 Research Commentary—The New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research. Information Systems Research 21(4): 724–735.

# I

# Accommodating multiple rationalities in patient oriented health information system design

Lars Kristian Roland[1], Terje A. Sanner[1], Prosper Behumbiize[2], Zikulah Namukwaya[3] and Kristin Braa[1].
[1]Department of Informatics, Faculty of Mathematics and Natural Sciences, University of Oslo, [2]CDC Uganda, [3]MU-JHU Research collaboration, Uganda.

roland@ifi.uio.no, terjeasa@ifi.uio.no

**Abstract.** Informed by an action research study concerning a Ugandan mother and child health programme, this paper identifies three different strands of reasoning (rationalities) pertaining to patient oriented health information systems; i) patient freedom to choose health services; ii) clinical professionalism at point of care; iii) and public health epidemiology and management. The initial identification of these rationalities emerged from our study of existing work practices utilizing paper-based information tools for following up pregnant women in Uganda. The study investigates how these different coexisting rationalities have found their way into the design process of a new electronic web and mobile phone based information tool that covers multiple application domains and contexts of use.

**Keywords:** health management information system, eHealth, patient tracking, continuity of care, rationality, action design research

## 1 Introduction

Information systems supporting health service follow-up activities for individual patients may be used to refer individuals across health care units; bridge private and public sector health services; and balance ownership, security and privacy concerns related to patient health data. A patient oriented information system's functional scope can range from supporting the work of a nearly isolated private practitioner in a rural community clinic, to facilitating information flow and enable comprehensive health care across hospital wards, health care units, and health programmes within and across large regional or national health systems.

The implementation of patient oriented information systems across contexts imposes different and sometimes contradictory requirements. Understanding the underlying rationalities of heterogeneous information users at various stages of patient follow-up and related activities is important in guiding the design process of such information systems (Robey and Boudreau 1999). This paper focuses on how different foundations for sense making (i.e. rationalities) play out in informing the design of an integrated patient oriented electronic information system. We study the emergent design of an *ensemble artifact* which includes social processes as well as the software (Orlikowski

and Iacono 2001). Noting the importance of acknowledging multiple rationalities in information systems design we pose the following research question: which rationalities are prevalent amongst users and designers of person oriented information systems and which design principles can we derive to accommodate these rationalities?

Empirically, we follow a collaborative project aimed at integrating antenatal care with delivery and postnatal care in Uganda, with particular focus on preventing mother to child transmission of HIV (PMTCT). The study illustrates how some practitioners (e.g., health workers) are concerned with utilizing information systems for following up the pregnant lady while others - for valid reasons – argue for leaving the responsibility with the lady herself. Furthermore, amongst those who consider systematic tracking of individual patients important, there are multiple coexisting rationalities legitimizing and justifying such claims (e.g., clinical and epidemiological). While drawing on these rationalities, individual practitioners are able to argue for the inclusion of certain assumptions about appropriate behaviors and practices to be reflected in the design of a patient oriented information system. Through an engaged action research approach the researchers have themselves been involved in trying to iteratively accommodate various concerns and translate them into design principles. These have in turn been attempted inscribed into both software (e.g., user interface, screen sequences and terminology) and relevant work practices.

## 1.1 Tracking patients throughout health programs

For the purpose of this paper, we define patient oriented health program tracking (just tracking for short) as the process of collecting and acting on individualized health data throughout a patient's enrollment in a health program. Thus, *tracking* is for instance used to facilitate *continuity of care* - a term widely used in the primary health service (Haggerty 2003), while it could equally well be used to manage health insurances or social security schemes targeting individuals eligible for refunds on health expenditures.

Mobile phones can play a crucial role in implementing patient oriented public health schemes in low resource contexts. Projects involving health workers at the periphery of the health system have successfully leveraged mobile technology to enable patient oriented health program tracking (Ngabo et al. 2012). For instance, such systems have been used to send an SMS to health workers with lists of pregnant women who have missed appointments (MacLeod et al. 2012). Sending such reminders to the health workers' managers, if necessary, has been demonstrated to improve the follow-up of patients (DeRenzi et al. 2012). However, for the mobile phone to be a powerful tool the recipients must be acquainted with the intentions, purpose and rationale of the messages, or these reminders will simply be ignored (Chib et al. 2012; Ngoma et al. 2012; Berg 1997).

Implementing a patient oriented health information system is challenging when the system spans multiple organizations and health programs and utilizes information and communication technologies and software packages that may be designed elsewhere based on different assumptions (Sanner et al. 2012; R. Heeks 2006; Lehmann and Sanders 2013). Grounding information system design in a single dominant rationality

may lead to tensions and conflicts if the information is to be shared and utilized across programmes and disciplines (Chilundo and Aanestad 2005). Globally, health information systems are shaped by two broad trends i) a move towards more integrated health information architectures (Braa and Sahay 2012) and ii) more health data about individuals is being captured and stored in databases. These trends suggest that health information systems purposefully designed with one particular practice in mind will need to find ways share data with other systems, and that more of this data stems from the recording of uniquely identifiable and traceable encounters between patients and available health services. Recognizing the underlying rationalities on which different information systems are designed and how they can be explicated is crucial if we want to understand how this rich information can be meaningfully shared to facilitate more equitable, integrated and comprehensive health services.

The remainder of the paper is structured as follows: first we review relevant literature on how multiple rationalities inform information system design; secondly, our methodology and empirical setting is described; then the case description and our research findings are presented; followed by a discussion highlighting how the different rationalities of public health management and clinical point of care relate to each other and shape the design of patient tracking information systems.

# 2 Theory and conceptual framework

The recognition of multiple rationalities offer a tool for exploring how and why people arrive at decisions to act and behave in certain different ways. Max Weber, in his critique of neoclassical works' limited exploration of rationality, distinguishes between several types of rationality (Kalberg 1980). Of these, he claims some types of rationality are based on values rather than a simple means-end calculation. Values are not necessarily demonstrable or justified through Western scientific methods, but may still provide the basis for a consistent rationality towards a substantive end goal.

Weber's classification of rationalities represents a radical perspectivism. "Something is not of itself irrational, but rather becomes so when examined from a specific rational standpoint" (Weber, 1930 translated by Kalberg 1980). These rationalities may be based on different logical assumptions (McNeill 1978; Karpik 1972; in Bryman 1984). Furthermore, what seems irrational at one level of analysis may appear perfectly rational at another level of analysis (Bryman 1984). A single individual may draw on multiple rationalities, even seemingly contradictory ones, to explain or justify an action or event.

In relation to health information systems research, Heeks et al. (1999; 2006) identify and explore what they call three *archetypes of rationality*: technical, managerial, and medical. The technical rationality understands information systems as objective and neutral models of reality, with a lack of appreciation for political or cultural values. Managerial rationality is depicted as concerned with efficiency and stakeholders' financial interests, while a medical rationality is oriented towards the interpretation of health information transmitted through information systems. These three rationalities, it is argued, inform different health information system design decisions, but they do not recognize some of the - in Heeks et al.'s words - "softer" rationalities, such as *health for all* and patient centered care.

Lending her attention to softer rationalities, Mol (2008) contrasts the "logic of choice" with the "logic of care", indicating that when you give a person the option to choose between health treatments, this flexibility may contradict with providing the best possible medical care. Similarly, tracking and systematically following up patients through health programs may interfere with individuals' freedom to choose their own health treatment. It is therefore important to recognize that the rationalities drawn on and *inscribed* into the design of an information system has implications for its subsequent appropriation and use.

## 2.1 Inscribing rationalities in Information Systems

*Inscription* refers to the way technical artifacts, like information systems, embody patterns of use. Inscription may be seen as the embedding of a certain world view into an information system. During the information system design process, the developer works out a scenario for how the system will be used (Ciborra 2002). This scenario is then inscribed into the technical artifact (Akrich 1992). The inscription includes *programmes of action,* preconceived sequence of behaviors, for the users to follow (Latour 1992, 255; Akrich 1992), and defines roles to be played by users and the information system. Programmes of action both constrain and guide the interaction between the user and the information system.

In situations where the inscribed programme of action does not fit with the actual practice of the user, workarounds may be deduced. By appropriating the information system in a different way than what was intended by the system designer, users leverage the information system's *interpretative flexibility* - the capacity of a specific technology to sustain divergent opinions (Sahay and Robey 1996, 260). The space for users to deduce new programmes of action is thus limited by the interpretive flexibility of the information system at a given point in time and in a specific context of use. Interpretive flexibility does not only mean that there is flexibility in how people think of or interpret an information system or a technical artifact, "but also that there is flexibility in how artifacts are designed" (W. E. Bijker et al. 1987, 40).

Finally, closure occurs when a consensus emerges, that is, when social groups involved in the designing and using of technology decide that problems arising during the development of a technology have been solved. Closure stabilizes the technology (W. E. Bijker et al. 1987; Wiebe E. Bijker 1993). We can think of the process of stabilization as the collaborative molding of an artifact made out of slowly, but never drying clay. As the clay becomes progressively more difficult to shape we recognize that initial choices about shape and material have long lasting implications. This is true even for software, which becomes progressively harder to change as a result of distributed and uncoordinated adoption and use. The initial assumptions informing design choices of health information systems, i.e., underlying data models and technical standards, may have long lasting implications.

This paper uses the concepts of inscription and interpretive flexibility to discuss how multiple rationalities are accommodated and reinterpreted through the design and use of a patient oriented information system.

# 3 Research methods

Action Design Research is a research method that focuses on the development of a theory-ingrained, innovative *ensemble artifact* (Sein et al. 2011). An ensemble artifact is one whose scope includes the social processes surrounding the artifact rather than just the physical software (Orlikowski and Iacono 2001). Sein et al (2011) state that Action Design Research focus on explaining a phenomenon, or on describing how to implement an artifact. A design principle describe how to design and implement an artifact, but the definition of a design principle varies in literature from a universal testable truth to more of a guiding principle (Braa et al. 2004). According to Sein et al (2011) and Gregor (2006), design principles are also theoretical contributions if they relate to a class of systems (eg. Hanseth and Lyytinen 2010). The design principles of artifact ensembles are however not as testable and generic as those one may find in a repeatable lab setting, though they may be more applicable when embedded into a social setting (Sein et al. 2011).

## 3.1 Research design

This paper is the result of an action research project informed by the Action Design Research method (Sein et al. 2011). The main objective of the project is to implement a class of systems for tracking patients in low resource contexts. The design and implementation has evolved through several cycles of development involving an increasing number of practitioners and users. The project is authorized by the Ugandan Ministry of Health and is headed by the MU-JHU institute. Several of the authors are part of a global action research program called HISP (Braa et al. 2004; Braa et al. 2007), which develops and deploys instances of an open source and web based Health Management Information System platform called DHIS2 and its mobile extension DHIS-Mobile in more than 20 countries.

The action research project is based on the deployment and further development of a patient oriented health information system (DHIS Tracker) that extends DHIS2 and DHIS-Mobile. The DHIS Tracker module has been tested and deployed in India, Tanzania, Vietnam and Malawi. The module required substantial changes in the software in order to accommodate requirements from the project in Uganda. The project conducted training, demonstrations and simulated data-entry sessions involving the authors as patients and the actual health staff doing the data entry, to guide the design. After each session, new features were implemented and again tested.

Two of the authors live and work in Uganda, and are involved in medical research and interventions as well as implementation of digital health information systems at national, district and sub-district levels, in close cooperation with the Ministry of Health. The first author, a Norwegian national, performed five field visits to Uganda over two years, visiting hospitals, health clinics, health program implementers' offices, training sites, Ministry of Health facilities and the US Embassy. By being close to the design and implementation this study obtained access to various stakeholders' diverging and sometimes conflicting responses to agendas and events pertaining to one patient oriented health information system innovation.

### 3.2 Data collection and analysis

We visited and interviewed health staff at three health facilities near the capital city of Uganda and three health facilities in a rural setting, some of which were visited two times. The data collection was a mix of observations during more than 20 encounters including project meetings within the team; semi- and unstructured interviews with health staff, community health workers and program officials working on tracking mothers; and observations from training and demonstrations of the software. Some interviews were performed in groups with health staff and researchers present, similar to ad-hoc focus groups, while some interviews were done between a single researcher and individual informants. The authors were introduced before the interviews as researchers and developers of the information system. Interview data collection was supported by detailed note taking, audio recording and video-taping of training sessions.

During this study, we have focused on *following the data*, looking at how health data is noted, stored and used. We assume that the handling and transferability of content also indicates to us the underlying rationalities that are inscribed into the existing tools and work practices. Studying the situated use of the existing paper based tools and existing socio-technical arrangements in general has been an essential part of the data collection.

Notes and observations were summarized and shared after the interviews, and concepts were discussed in project meetings. The authors later refined the identified concepts through the use of data displays and in-depth theoretical analysis (Miles and Huberman 1994). A detailed analysis of the different existing information systems was also performed.

## 4 The case of tracking information systems for mother and child health in Uganda

In Uganda, the health care during pregnancy and follow-up of the mother and child has traditionally been split into separate parts focused around antenatal care, delivery, postnatal care and immunization of the child respectively. This paper looks at an information system designed to support a more integrated approach, driven by a general wish expressed by the Ministry of Health in Uganda and development partners to improve maternal and child health as well as a special focus on prevention of mother to child transmission of HIV.

### 4.1 Following the mother and child through the health program

An integrated program for mother and child health would typically include four antenatal care (ANC) visits, the delivery itself, and postpartum visits. The program can be adjusted to reflect the mother having HIV or some other condition, making more visits necessary and also involving more health professionals at different locations. Supporting multiple programs and staff with a single information system is difficult because it requires a great deal of flexibility to cater for both the individual clinical conditions of patients and maintain a rigid data source that can be monitored by program

administrators. The system also needs to accommodate health workers who have very different focus and performance indicators by which their works is evaluated throughout the cycle of ANC, delivery and post-delivery. In essence, the tool needs to align different practices, programmes and rationalities that have evolved separately over an extended period of time.

## 4.2 Transferring and tracking patients across health units

Each health facility in Uganda has its own internal patient ID numbering system, such that a pregnant lady with ID number 100 at one facility is a different person from number 100 at another facility. This number is used in the facility registry, in the hospital journal if there is one and in the mother passport. In Uganda there is not yet an operational national identity number scheme providing a unique identifier per patient.

Pregnant women often move between different health clinics either as part of a formal referral process or based on their own preferences. For instance, pregnant women may choose to deliver at another health clinic because their regular health clinic has a bad road or is difficult to reach at night; or because a different clinic has more advanced facilities. Not all services are available at all facilities, and the larger hospitals naturally handle more of the difficult cases.

At one facility where this study was being conducted the health staff explained that they would sometimes pay for transportation out of their own personal funds to transfer laboring women in very critical conditions to a larger hospital. On the other hand, the over-crowded larger hospital has ambulances ready to transfer non-critical patients in labor to smaller facilities if these could deliver elsewhere.

## 4.3 Institutionalized patient tracking artifacts

Historically patient data concerning the ANC program has been recorded and physically kept in several places using multiple tracking artifacts; mother health passports, registry books and patient journals. Larger hospitals keep paper-based journal systems where they store patient-related historical data, though for some patient groups (e.g., HIV positive patients) paper-based journals are widespread also at clinics.

### *4.3.1 Mother health passport*

Many countries let persons carry their own individualized health information on paper between visits (Turner and Fuller 2011; WHO 1994). In Uganda, the health data is literally in the hands of the mother, as she brings a paper booklet called the “Mother’s health passport” to each health visit. The predecessors of the health passports are simpler and less comprehensive health cards. In practice, these cards are still frequently used instead of the newly designed, more expensive and elaborate health passport. Sometimes women must use photocopies of the original card, as the clinic’s own supply has run out. At hospitals that have paper based journal systems, the health passports and health cards are often not filled in or employed in the work flow, because patient data is registered and kept at the hospital.

One midwife at a sub-district health facility explained the importance of the mother health passport: "Whenever they come to ANC, they have to come with them [the health card]. Because the first thing we ask them is for the card, and it's first come first served. … [If she didn't bring her card] she would have to come back another day. We cannot go back in the [registry] books for all those people."

The health passport is normally brought to every appointment, but there are times when the mother loses or forgets the booklet or does not bring it intentionally. In addition to mothers carrying their health passports, health clinics have been provided with a schedule appointment book to write down the next appointments so they can plan and potentially see which mothers do not show up for their appointments. This book is used primarily for HIV infected mothers if at all.

#### *4.3.2 Registry books*

The health registry books are physically heavy and wide with one long line of data elements to be entered per visit. The data is registered chronologically as patients enter the clinic. Subsequent visits by the same patient are thus logged later in the book.

While registry books support the administrative reporting of summarized health management information well, they are not well suited for reviewing the progress of an individual patient throughout a health program (Chi et al. 2011; Garrib et al. 2008). One health worker put it accordingly: "paper is difficult to analyze backwards".

A recent pilot study in Uganda has tested the feasibility of using longitudinal registry books, logging all visits for antenatal care, delivery and postnatal care on a single line, thus maintaining the tracking function within the registry book. This registry was piloted in a number of districts, but it has been discontinued because it could not be applied comprehensively across all envisioned settings, in clinics with complex work flow.

## 4.4 New electronic person tracking system

One aim with the new and integrated mother and child health information system project was to adapt and further develop the DHIS Tracking module so that it could be deployed to replace or complement the existing paper based tools. This open source tracking software was used in India, Tanzania, Malawi and Vietnam prior to and during the unfolding of the project in Uganda, and had accumulated design features through adaptation to previous projects. All deployments build on the same source code and receive support from the same software developers. During the project there were several occasions when requirements from the different contexts were contradictory. In one example, a new tracking screen was included for the Uganda project, but was picked up by another project and adapted to their context, loosing important requirements from Uganda. The screen then was changed back, but the content and labels on the screen was changed to indicate more clearly which use case the screen was meant for, and the underlying rationality of the use case.

The earlier version of DHIS Tracker implemented in a state in India was based on each health worker performing house visits to patients, and reporting post-hoc on data collected during these visits. The software originally had functions for displaying a visit schedule for upcoming appointments, but had fewer features specifically targeted at

patients visiting a health clinic and tracking those who missed their appointment - so called *lost-to-follow-up*. These follow-up functions were enhanced during the project in Uganda, with the options to easily add comments to each patient, send text messages and inform the system about their temporal health program status. One practitioner explained this as follows: "The current selection criteria for both of these is overlapping and confusing to the health workers. … Again from their tracking perspective, this should only list events of the past. eg they would want to list mother who did not turn up for their visit yesterday, a week or month ago."

The medical practitioners in the project wanted a *patient dashboard* that quickly gave an overview of the patient's medical history and medical information at the point of care. This screen would be used by medical staff while examining the patient. The resulting screen has information such as personal data, list of active and previous health programs the patient has been enrolled in, link to a patient's child or parent and an overview of comments entered and text messages sent to this patient. The screen also allows for data entry, schedule management and linking this patient to a specific health worker responsible for follow-up.

The DHIS Tracker design accumulated from previous implementations had been rigid with strong inscriptions, and the software was changed to allow for more flexible health program definitions and work flows. New requirements included the need for combining multiple programs (e.g., mother and child health and HIV/AIDS) into one, and customizing how many encounters there would be for each patient enrolled in a programme. Optional automatic reminders were added, allowing configurable text alerts and multiple reminders per program stage. The new mobile features included text reminders to patients about scheduled visits, educational text messages and improved interfaces on mobile phones to register patients, enroll them in programs, access and enter patient data. The software also went through several revisions where the privacy of health data was improved. The definition of user roles and access rights were improved, and audit logs for access to private data were also introduced. A feature to hide or not use patient's real names was also implemented.

During training sessions, the health staff was clearly more used to mobile phones than computers with web browsers. One of the health staff smiled when we pointed this out and said that "In Uganda, the mobile is the PC". It became obvious during the project that the solution had to be accessible to different health workers on different devices and technologies, including SMS, Mobile Java, Mobile browsers, web and even tablets. Work in the new integrated programme are performed in heterogeneous contexts, and supporting diverse needs was imperative for making an accessible solution across the health service, while allowing for interpretive flexibility in how individual users employ the tracking information system.

## 4.5 Summary of data capturing tools

Existing paper based information system tools support different tasks and work practices in a way that do not smoothly unite clinical and health information management functions. These differences are exemplified by tensions such as: Which

data should be stored and where? Who should enter the data and who should have access to it? When is the data entered and retrieved? Table 1 summarizes how some of these questions have been addressed by different paper based tools and the new electronic patient oriented information system in the case of the integrated mother and child health programme.

| | **What data?** | **How is data stored?** | **Who has access?** | **When and how is data entered?** | **How can data be retrieved?** |
|---|---|---|---|---|---|
| **Mother health passport** | Medical information about pregnancy | Booklet carried by mother | Only people the patient shows it to | At point-of-care | When the mother wants it to |
| **Registry (employed nationwide)** | Information relevant for public health management | Book at the registry desk | Health staff at one clinic | Before and after patient visit | By sifting through the registry |
| **Patient Journal (some hospitals)** | Rich clinical information | Stored in delivery ward | Health staff at one hospital | At point-of-care | All visits stored in one file |
| **Electronic Person Tracking Information (pilot)** | Subset of data, important for clinical follow-up | Shared online database | Health staff at clinics. SMS reminders to patient and health workers | At point-of- care, or after visit through a computer | Using mobile phone, tablet or PC |

**Table 1.** Summary of the characteristics of data capture tools

# 5 Analysis

The recursive stages of Action Design Research, with building software, intervening and evaluating the outcome led to an emerging understanding of multiple rationalities present across health practitioners and other information system stakeholders. Our recognition of these multiple rationalities became an important starting point for arriving at design principles. At the outset of the project, it was important to learn how the existing paper based tools were being used and understand their benefits and drawbacks. While studying these tools, it became apparent that they had different rationalities inscribed into them, exhibiting conflicts and synergies with different practices. Developing a single integrated information system would mean accommodating multiple rationalities into a single software package.

Whenever it was brought to the system developers and implementers' attention that different rationalities informed conflicting design principles, there were negotiations within the team on how they should be resolved in the design. In some cases, the multiple rationalities could be supported simultaneously by adding multiple screens in the user interface of the system. In other cases, trade-offs had to be made which partly favored one rationality in the design of the ensemble artifact over another. This was the case when it was argued whether or not the electronic system should replace the registry books entirely or just collect data relevant for the clinical follow-up of the patient. In some cases, tensions were negotiated by loosening inscriptions, such as making program

definitions and labels customizable and allowing for free-text notes, thus increasing the interpretive flexibility of the solution.

The project built on knowledge generated from other deployments, both because previous projects had inscribed design principles into the open source software artifact, but also through meetings and discussions with other project teams (mentioned in the methods section). Because all projects used a common open source code base and also relied on the same development team, the co-negotiation of shared design principles that bridged contexts and use cases was important. When the rationalities informing previously implemented functionality were not clearly communicated, it led other projects to misunderstand the intentions inscribed into the system and, in some cases facilitated redesigns of the common source code which broke down the derived design principles.

## 5.1 Who is responsible for continuity of care?

We have identified multiple coexisting rationalities affording a view on how pregnant women should be followed up. As reflected by the mother health passport, some people argue that patients should be in control of their own data and decide for themselves which health facility they wish to attend. Others argue that the health service should take on the responsibility for following up patients, because too many mothers fail to meet for appointments, which is not helped by sparingly used scheduling books.

The argument for personal-choice in seeking health services has the following implications for the electronic health information system: When the mother is free to move between health facilities, it makes planning more difficult and also creates the demand for electronically sharing health information across these facilities, which in turn triggers a whole avalanche of privacy, data ownership and confidentiality concerns. This is particularly so when it comes to sensitive data about stigmatized health conditions.

## 5.2 Two different tracking rationalities – management and clinical

We also found that the rationale for implementing a patient oriented tracking information system varied among its proponents. From an epidemiological or public health management perspective, tracking is important as a tool to monitor the health situation in a country or a region more accurately. Information fetched from tracking programs that use individual patient data can enable powerful analysis informing health program management and the distribution of resources. From this perspective, tracking is not there so much to improve the health of the individual patient, but to monitor and improve the effectiveness of the overall health services. Collecting and centrally storing historical personal data may be justified from this perspective because individualized data is perceived to be more accurate than aggregate figures and can be traced back to the actual event for verification. More detailed analysis on the correlation between types of health related challenges in a population can be performed based on personal data.

On the other hand, many health staffs such as clinicians have a different view of tracking that is less centered on epidemiological concepts and focused more on

diagnosing and treating the individual patient. For them, the follow-up process extends beyond the recording of historical data, to incorporate tasks such as calling or going to see the patient, adapting the medication to the patient's symptoms, referring risk patients to more advanced health facilities etc. This process is pursued to improve the health outcome of the individual patient being tracked, not to collect data that is useful at a national level. This rationality was more apparent with doctors and midwives who had been in direct contact with patients and who also had been trained in professional medical care.

One of the participants in a training session clearly expressed this difference between epidemiological and clinical rationalities. He was a medical doctor at a hospital, and had also worked with electronic medical record systems. "We want to link the data to the patient directly. I would have preferred to keep the [health management information] system at the district level, because the district is aggregating. The facility would like to analyze something more to the benefit for the patient. This is what I'm talking about. The patient - what benefits [can this system have to the patient]".

We label these sometimes opposing rationalities: *clinical tracking* and the *public health management tracking*, while keeping in mind that they are not always contradictory in informing information system design. In fact, we have seen individuals 'change hats' during a discussion, arguing for the virtues of both rationalities.

During the design process, arguments drawing on the clinical tracking rationality would tend to be drawn on more to inform design of point-of-care access to data and a good patient dashboard. However, individuals drawing on this strand of reasoning also accepted trade-offs such as only logging data for critical cases or restricting the dataset to what is clinically important to share between facilities. On the other hand, people drawing on the public health management rationality would argue that enough data has to be collected to be analyzed post-hoc, and argued that all the data in the registries should be collected through the electronic system. Supporters of this view wanted to use the new electronic system to completely replace the registry books. There was a clear design conflict in the project between streamlining the design around the minimum clinical dataset required to follow up and identify critical cases, contradicted by the complete set of data filled into the existing HMIS registry books.

One of the major information system design revisions was the introduction of the *lost to follow-up screen*, listing persons who were lost-to-follow-up and providing simple interfaces for sending SMS messages and reminders to patients and community health workers. These features were added to support a clinical tracking rationality, and had not been perceived as relevant from the public health management perspective.

Realizing that there were multiple rationalities also led to the design decision of enabling two separate data entry screens, in order to accommodate multiple programmes of action. The *patient dashboard* was streamlined to provide clinical information and data entry at the point-of-care, while the post-hoc data entry screen was optimized for data entry by a data entry clerk after the patient had left.

### 5.3 Structured data vs flexible use of information

In HMIS systems, it is a good design principle to use structured data models that can be aggregated and analyzed for comparison and identification of trends across time and space. Many medical journal systems are much less structured, and allow for more flexibility through free text fields. To wit, doctors are infamous for their frequent and sloppy handwriting, but certainly rely on it to record clinical data. The historical DHIS2 software development has followed a public health management rationality informing the design of structured data and forms. However, during the development of DHIS Tracker in Uganda more free-text fields were incorporated, allowing health professionals to comments on programme stages and patients. These comments are not useful to aggregate and analyze at a national level, but are useful within the clinical setting to provide flexibility in individual treatment and care.

## 6 Discussion and concluding remarks

We have identified three important health information systems rationalities. Some actors view tracking as unnecessary, unmanageable or not cost-effective, and would rather place the responsibility for follow-up with the patient. For those who see tracking as important, the tracking rationality seems to manifest itself through two distinct views. One is a managerial rationality focused on running equitable and cost-efficient health services. The other (clinical rationality) is focuses on treatment and improving the services provided to the individual at the point-of-care.

Three design principles emerged from the negotiated accommodation of multiple rationalities in the development and implementation of the patient oriented health information system which needs to support: i) personal choice and freedom for the patient to select when and where to seek health services, ii) clinical use and sharing of person specific health data across health facilities, including the two activities of planning visits and finding patients who have been lost to follow-up, iii) support health management data collection for statistical analysis and public health management. These theory ingrained design principles were inscribed into the ensemble artefact through changes in wording, new screens, implementing new technologies such as SMS messaging and change of work practices. Because each of the design principles represent different and to some extent conflicting rationalities, the design process involved a careful balancing of design principles, where the iterative research process was used actively to explore the effects of giving weight to single principles.

The information system now supports the combination of point-of-care data management and post-visit data management; merging a data warehouse containing structured patient data with support for unstructured comments to be used at the clinical level. Features for connecting the community health workers to the health staff have also been implemented, integrating the community health worker into the process of providing continuity of care.

Conflicting rationalities between designers and potential users can lend explanatory power to the lack of adoption of a particular system, or failures to integrate and coordinate the system with relevant health programs or supporting activities. It is

therefore important to give voice to diverging rationalities and recognize the multiplicity of resulting design principles at a point in time when the solution is still being moulded, in order to embrace interpretive flexibility where it is needed. Supporting different rationalities is expected to provide a more successful deployment of the software platform, enabling it to be part of a shared and open health information infrastructure that more actors have an interest in supporting, maintaining and extending.

This paper has focused on a single project where the DHIS2 Tracker software was being deployed, but because the identical software is deployed simultaneously in different projects, each affecting the code base in varying ways, it may also have been useful to look more at the co-negotiation of design principles across projects, considering the longitudinal biography of the DHIS2 tracker artefact.

# 7 References


Akrich, Madeleine. (1992). "The De-scription of Technical Objects." *Shaping Technology/building Society*, pp. 205–224.

Berg, Marc. (1997). "Problems and Promises of the Protocol." *Social Science & Medicine* 44 (8) (April), pp. 1081–1088.

Bijker, W. E., T. P. Hughes, and T. J. Pinch. (1987). *The Social Construction of Technological Systems: New Directions in the Sociology and History of Technology*. the MIT Press.

Bijker, Wiebe E. (1993). "Do Not Despair: There Is Life after Constructivism." *Science, Technology & Human Values* 18 (1) (January 1), pp. 113–138.

Braa, J et al. (2007). "Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy." *MIS Quarterly* 31 (2) (June 1), pp. 381–402.

Braa, J., E. Monteiro, and S. Sahay. (2004). "Networks of Action: Sustainable Health Information Systems Across Developing Countries." *Mis Quarterly*, pp. 337–362.

Braa, J., and S. Sahay. (2012). "Integrated Health Information Architecture." *Power to the Users*.

Bryman, Alan. (1984). "Organization Studies and the Concept of Rationality." *Journal of Management Studies* 21 (4), pp. 391–408.

Chi, Benjamin H. et al. (2011). "Implementation of the Zambia Electronic Perinatal Record System for Comprehensive Prenatal and Delivery Care." *International Journal of Gynecology & Obstetrics* 113 (2) (May), pp. 131–136.

Chib, Arul et al. (2012). "You Have an Important Message! Evaluating the Effectiveness of a Text Message HIV/AIDS Campaign in Northwest Uganda." *Journal of Health Communication* 17 (sup1), pp. 146–157.

Chilundo, Baltazar, and Marguun Aanestad. (2005). "Negotiating Multiple Rationalities in the Process of Integrating the Information Systems of Disease Specific Health Programmes." *EJISDC: The Electronic Journal on Information Systems in Developing Countries* (20), pp. 2.

Ciborra, Claudio. (2002). *The Labyrinths of Information: Challenging the Wisdom of Systems: Challenging the Wisdom of Systems*. OUP Oxford.

DeRenzi, Brian et al. (2012). "Improving Community Health Worker Performance through Automated SMS." In *Proceedings of the Fifth International Conference on Information and Communication Technologies and Development*, 25–34. ICTD '12. New York, NY, USA: ACM.

Garrib, A. et al. (2008). "An Evaluation of the District Health Information System in Rural South Africa." *South African Medical Journal* 98 (7) (July 18), pp. 549–552.

Gregor, Shirley. (2006). "The Nature of Theory in Information Systems." *MIS Q.* 30 (3) (September), pp. 611–642.
Haggerty, J. L. (2003). "Continuity of Care: a Multidisciplinary Review." *BMJ* 327 (7425) (November 22), pp. 1219–1221.
Hanseth, O., and K. Lyytinen. (2010). "Design Theory for Dynamic Complexity in Information Infrastructures: The Case of Building Internet." *Journal of Information Technology* 25 (1), pp. 1–19.
Heeks, R. (2006). "Health Information Systems: Failure, Success and Improvisation." *International Journal of Medical Informatics* 75 (2), pp. 125–137.
Heeks, Richard, David Mundy, and Angel Salazar. (1999). "Why Health Care Information Systems Succeed or Fail." *Information Systems for Public Sector Management*.
Kalberg, Stephen. (1980). "Max Weber's Types of Rationality: Cornerstones for the Analysis of Rationalization Processes in History." *American Journal of Sociology* 85 (5) (March 1), pp. 1145–1179.
Latour, Bruno. (1992). "Where Are the Missing Masses? The Sociology of a Few Mundane Artifacts." *Shaping Technology/building Society: Studies in Sociotechnical Change*, pp. 225–258.
Lehmann, Uta, and David Sanders. (2013). "Community Health Workers: What Do We Know About Them?"
MacLeod, Bruce Bradford et al. (2012). "The Architecture of a Software System for Supporting Community-based Primary Health Care with Mobile Technology." *Online Journal of Public Health Informatics* 4 (1).
Miles, Matthew B., and A. Michael Huberman. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. Sage Publications, Incorporated.
Mol, Annemarie. (2008). *The Logic of Care: Health and the Problem of Patient Choice*. Routledge.
Ngabo, Fidele et al. (2012). "Designing and Implementing an Innovative SMS-based Alert System (RapidSMS-MCH) to Monitor Pregnancy and Reduce Maternal and Child Deaths in Rwanda." *The Pan African Medical Journal* 13 (October 14).
Ngoma, Caroline Abdul et al. (2012). "Challenges in Implementing a Computerized Name-based Information Tracking System: Practical Experiences from Maternal Health Care" (October 2).
Orlikowski, Wanda J., and C. Suzanne Iacono. (2001). "Research Commentary: Desperately Seeking the 'IT' in IT Research—A Call to Theorizing the IT Artifact." *Information Systems Research* 12 (2) (June 1), pp. 121–134.
Robey, Daniel, and Marie-Claude Boudreau. (1999). "Accounting for the Contradictory Organizational Consequences of Information Technology." *Information Systems Research* 10 (2) (June 1), pp. 167–185.
Sahay, Sundeep, and Daniel Robey. (1996). "Organizational Context, Social Interpretation, and the Implementation and Consequences of Geographic Information Systems." *Accounting, Management and Information Technologies* 6 (4), pp. 255–282.
Sanner, Terje Aksel, Lars Kristian Roland, and Kristin Braa. (2012). "From Pilot to Scale: Towards an mHealth Typology for Low-resource Contexts." *Health Policy and Technology* 1 (3) (September), pp. 155–164.
Sein et al. (2011). "Action Design Research." *MIS Q.* 35 (1) (March), pp. 37–56.
Turner, Kathleen E., and Sherrilynne Fuller. (2011). "Patient-Held Maternal And/or Child Health Records." *Online Journal of Public Health Informatics* 3 (2).
WHO. (1994). "WHO Home Based Maternal Records - Guidelines". WHO.

II

# Scandinavian Journal of Information Systems

## Volume 29, No. 2



Publication date: 31 December 2017

eISSN 1901-0990

# P for Platform

## Architectures of large-scale participatory design

Lars Kristian Roland, Terje Aksel Sanner and Johan Sæbø
Department of Informatics, University of Oslo
*roland@ifi.uio.no; terjeasa@ifi.uio.no; johansa@ifi.uio.no*

Eric Monteiro
Department of Computer and Information Sciences, Norwegian University of Science and Technology (NTNU)
*Eric.Monteiro@idi.ntnu.no*

**Abstract.** Participatory Design (PD) has traditionally been committed to extensive interaction between developers and situated users to mitigate the disempowering consequences of computerization, such as the deskilling of labour workers. However, the widespread adoption of off-the-shelf software and the emergence of complex information system architectures with interdependencies across user groups and organizations challenge the applicability of traditional custom PD. Pressure is put on PD to scale with initiatives that span an increasing number and distribution of heterogeneous settings, developers, users and uses over time. In this article, we follow a PD project that started out in post-apartheid South Africa more than two decades ago. The project, which centres on the development of a software product for decentralised public health care management, has since grown into a venture with a significant footprint in the Global South. In order to problematise the scaling of key aspects of PD, such as the politics of design, the nature of participation and participatory design techniques, we first review extant literature and develop a classification of four different types of PD with respect to scale. We then apply the typology to our empirical case to discuss PD in relation to architectural traits at different stages of project scale. We contribute to PD literature by addressing the exploratory research question: What role does architecture play in large scale PD? Specifically, the study highlights how an emergent platform architecture and its surrounding ecosystem co-constitute a platform for participation in design.

*Keywords:* participatory design (PD), healthcare management, architecture, and platform.

Accepting editor: Magnus Bergquist

# 1 Introduction

Two decades ago, Neumann and Leigh Star (1996) questioned to what extent Participatory Design (PD) could be applied in large-scale information system projects. Unfortunately, the question remains largely unanswered (see; e.g.; Kyng 2010) despite its rapidly increasing pertinence. Today, PD researchers and practitioners increasingly find themselves in the midst of complex IT mega projects with many stakeholders that potentially dilute, derail or overrun small-scale participatory efforts (Karasti 2014). Yet most studies of PD still focus on the development of a single custom system that supports work within one particular client organization, department or group (Obendorf et al. 2009; Oostveen and Van den Besselaar 2004; Pilemalm and Timpka 2008). The conditions for small-scale politically motivated prototype-based approaches (Bødker and Pekkola 2010) are increasingly absent in contemporary software development processes in general (Kyng 2010; Shapiro 2005) and in health care in particular (Balka 2006; Pilemalm and Timpka 2008).

To remain relevant, PD needs to find ways to accommodate more distributed development settings (Gumm 2006; Obendorf et al. 2009; Titlestad et al. 2009) and interplay with complex and evolving ecosystems of ICT capabilities (Hess et al. 2008; Monteiro et al. 2012). We refer to this as the problem of scale in PD. In relation to PD, we define scale as the number and distribution of heterogeneous settings, developers, users and uses of a (software) product over time. As Karasti (2014) notes in a recent literature review, challenges to the sustainability of short-term PD projects have been explicitly problematised, while spatial scale, as we have defined it above, has received less attention. However, as our literature review in the following section highlights, contemporary PD projects encounter severe challenges with physical, temporal and organisational scale in relation to distribution of both designers and users (Gumm 2006).

To study PD in relation to scale, we draw on a longitudinal case study of a software development project that started out in support of decentralised health care management in post-apartheid South Africa more than two decades ago (see Braa and Hedberg 2002). Throughout the duration of the project, key individuals such as lead software developers and project managers from Norway, our colleagues, have explicitly sought to reflect on the applicability of PD (Braa and Hedberg 2002; Braa and Sahay 2012; Shaw and Braa 2014; Titlestad et al. 2009). The software has evolved from a custom-made desktop application to a modular web-based platform used globally. Consequently, the project has accumulated complexity along each dimension in our definition of PD scale: From a handful to tens of thousands of heterogeneous users, from a pilot in one district to distributed implementations in more than 50 countries, from clearly defined work routines of monthly public health data reporting and analysis to a diverse range of application domains.

To mitigate complexity and manage dependencies, information system architectures have been developed into modular and loosely coupled systems of systems that interact through standardised interfaces (Yoo et al. 2010). A prominent technical architecture employed to tackle complexity is the platform (Tiwana 2013), where reusable and generic functions are bundled as a platform core while specific services are developed as compliments, called apps. However, with the term architecture, we here refer not only to software layers and modules and the interfaces between them (Van Schewick 2012), but also to the structure of a corresponding socio-technical

ecosystem (Tiwana 2013) that potentially both enables and constrains participation in design. Contemporary PD takes place in this new architectural environment, for which traditional small-scale participatory approaches do not necessarily fit well (Shapiro 2005). This challenge motivates us to ask:

*What role does architecture play in large-scale PD?*

The contribution of this study is twofold. First, in section two, we provide a narrative review of extant literature and develop a typology of PD with respect to scale. Second, we employ this topology to our empirical case to discuss the enabling and constraining role of architecture in relation to PD. Section three outlines our research approach based on longitudinal involvement with one large-scale PD project and our theoretically informed sampling of three case vignettes. The vignettes are presented in section four. Our discussion in section five highlights how the architecture of the software product and the ecosystem that surrounds it shapes the politics of design, the nature of participation and the use of PD tools and techniques. We offer concluding comments in section six.

## 2 Scaling participatory design

Traditional PD in information systems development, with roots in Scandinavia, was driven mainly by two key concerns: workplace democracy and usefulness of design (Bjerknes and Bratteteig 1995). Firstly, end-user involvement was an explicit democratic goal (Bjerknes et al. 1987; Clement and Van den Besselaar 1993; Sandberg 1979). It was argued that the workplace, where computerised tools came into play, should follow principles of democracy that applied elsewhere in society. Secondly, PD was also seen as a means to increase the quality, efficiency and usefulness of new IT-solutions (Kyng 2010). This was based on the pragmatic view that there is a great deal that users can contribute to design (Shapiro 2010). Early PD techniques included mock-ups, prototyping, workshops and scenario building (Ehn 1988). In the seminal Utopia project 1985-1987 at the newspaper Aftonbladet, the developers went through several rounds of prototyping to articulate the needs of the graphics workers for which the system was intended (Bødker et al. 1987; Ehn 1988). These PD techniques have remained essential to date, despite their apparent shortcomings in addressing some of the challenges to user involvement in contemporary software development processes characterised by the physical, temporal and organisational distribution of both designers and users (Gumm 2006; Obendorf et al. 2009). These broad dimensions align with our definition of PD project scale as the number and distribution of heterogeneous settings, developers, users and uses of a (software) product over time.

One apparent problem with multisite and multi-user PD is logistical: how can developers engage intimately with situated practices spread out across a large number of sites? The more fundamental problem arises when the supported practices also become increasingly heterogeneous (Obendorf et al. 2009). The current pressure to modernise PD was debated in an issue of the Scandinavian Journal of Information Systems. The geographical distribution of projects and the heterogeneity of users and uses were problematised as obstacles to PD (Balka 2010; Kyng

2010; Karasti 2010; Shapiro 2010;). New forms of PD are needed to ensure that diverse user interests can be represented and recognised in large-scale projects with powerful stakeholders, novel funding mechanisms, and complex inter-organisational information system dependencies (Kyng 2010; Oostveen and Van den Besselaar 2004). In the following subsection, we review literature concerned with approaches to overcome challenges to PD project scale and synthesise this into a typology.

## 2.1 Singular, serial, parallel and community PD

We use the term 'singular PD' to denote key principles and techniques associated with custom PD such as use of prototypes, mock-ups, and workshops to facilitate end-user participation in design (Clement and Van den Besselaar 1993; Ehn 1988; Kensing 1987). In short, singular PD emphasises the one-to-one relation between developers and situated users. In recognition of the shortcomings with the classical singular approach to PD in contemporary projects, we review literature concerned with PD at scale to discern three additional types, which we dub 'serial PD', 'parallel PD' and 'community PD'.

Titlestad et al. (2009) consider challenges in a distributed PD environment and describe a software product that gradually improves through the accumulation of local innovations and learnings. They highlight the role of software implementers and customisers who, over time and across organizational and geographical settings, mediate requirements to a global development process (Figure 1). We dub this sequential model of PD across sites and over time as serial PD, where situated design iterations build on previous PD in another setting through distillation of global standards. Through accumulation of experiences from distributed yet relatively similar settings, implementation teams gain capacity to configure the software in collaboration with users. In the words of Titlestad et al. (2009, p. 33), the gap between designers and users "can best be bridged by technically conversant people who engage in implementing the system and training end users, and who are also adept at communicating with the core developer team." Traditional PD techniques are difficult to employ when co-location and homogenous work practices are no longer given. Obendorf et al. (2009) identify two techniques that facilitate PD across distributed sites. They refer to one technique as *inter-contextual workshops* that enable experience sharing, communication and requirements negotiation between dispersed users who have similar interests towards the same software product. The nature of user participation in such workshops takes the form of representation. Skilled implementers and local experts who understand the needs of different user groups professionalise participation. The second technique identified by Obendorf et al. (2009) is referred to as commented *case studies.* Case studies constitute descriptions of the appropriation and local customisation of a flexible software product to specific uses. Power users typically develop such case studies for circulation to the wider software community. On the downside, rich qualitative descriptions are time consuming to produce and may quickly become obsolete following software releases with new or revised functionality (ibid). Both of the identified PD techniques reflect a transition from situated design to representation of distributed and heterogeneous sites, users and uses.

Hansson et al. (2006) report on a distributed software development effort that spans multiple end-user groups and settings. They note that due to the relatively small size of the software

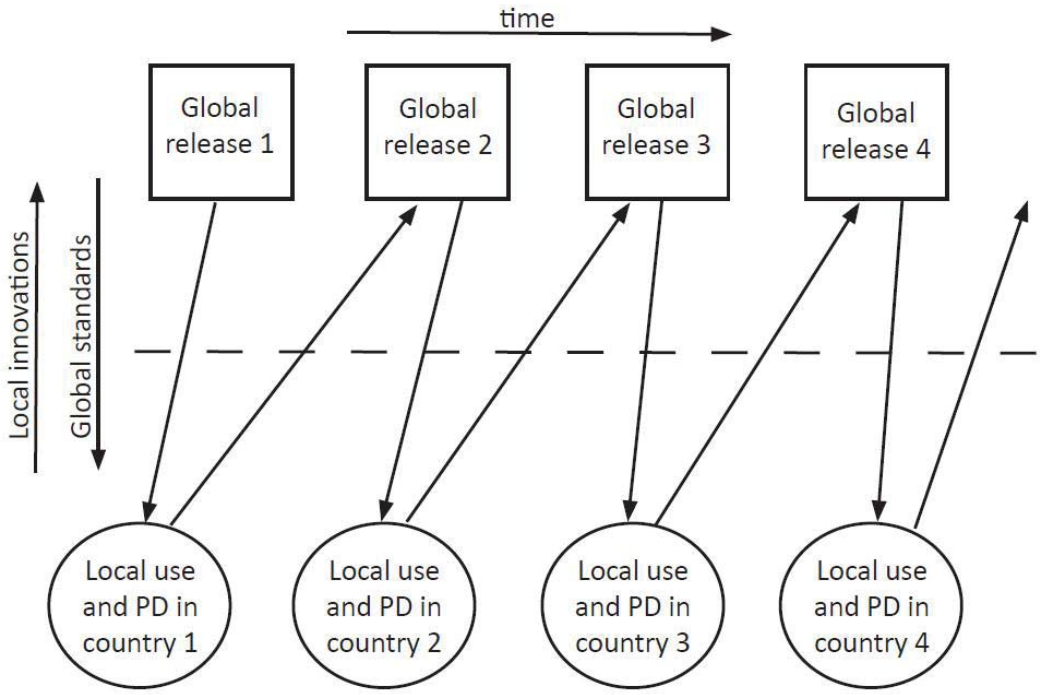


Figure 1. The Titlestad et al. (2009, p. 42) model of PD across multiple country sites informing the longitudinal development of a generic software product.

development team and the fact that end users are located at different workplaces all over Scandinavia, "[users] affect the development through user meetings, technical support and courses, but they never actually participate in design activities like workshops or mock-up evaluation" (ibid, page 5). The authors further note that as design and use becomes distributed, user participation shifts from designing a system to *using* and *evolving* it through various feedback channels. Similarly, Obendorf et al. (2009, page 21) point out that when PD was applied "[i]n increasingly distributed contexts, the focus shifted from custom software development to empowering users in assessing their own practice and technology use". Balka (2006) also notes the importance of customisation work in contemporary PD, as the process of getting off-the-shelf software products to work across multiple sites strongly resembles custom design. We note a transition from serial to parallel PD, as the overall design approach shifts from iterations of situated PD in a sequential manner (Figure 1), to collective filtering of requirements and local appropriation of generic features rather than custom design. Parallel PD emerges when software developers can no longer facilitate participation by travelling and engaging with users on site.

Parallel PD encompasses workshops and the circulation of software implementers, case studies, software documentation and the software itself. However, with a focus on even greater scale, a few PD projects have looked at how communities, with thousands of users, inform design. Hess et al. (2008), for instance, discuss the formation of online wikis and web forums to support a steering committee and a user parliament that represent more than 130,000 potential users in the design of a web-based net-TV solution. They found that outputs from the two user-centred governance bodies, the committee and the parliament, were difficult to coordinate, and that specific needs were hard to translate into general requirements. More specifically, their study

found that potential end users who participated in online forum discussions started to doubt their influence on the design process because most actualised suggestions originated with the project's product manager. Oostveen and Van den Besselaar (2004) note that conflicting interests and misunderstandings are likely to be commonplace in very large PD projects. Their study highlights that mutual understanding, rather than consensus, within the community, is a key priority, albeit difficult to achieve. Beyond PD, the transition towards community distillation of generic requirement, by increasingly heterogeneous and distributed users, has been observed in the development of global commercial software packages such as Enterprise Resource Planning (ERP) software (Pollock and Williams 2007).

Table 1 summarises key points from the narrative literature review provided above and sorts PD into four types: singular, serial, parallel and community PD. The typology characterises PD at different levels of scale. For each type we describe three key aspects of PD as synthesised by Kensing and Blomberg (1998): (1) the politics of design, (2) the nature of participation, and (3) methods, tools and techniques for carrying out design projects. The politics of design concerns "the empowerment of workers so they can co-determine the development" (Clement and Van den Besselaar 1993, p. 29). Politics of PD also relates to the type and range of skill development a technology affords, as well as how issues of power and inequality are dealt with, for instance through system flexibility (Brigham and Introna 2007). The nature of participation concerns how and to what extent user interests may be accommodated in design by, for example, mitigating the constraints of hierarchical structures, narrowing the distance between developers and users, or by bridging design-use knowledge gaps (Balka 2006). PD methods, tools, and techniques, which we hereafter refer to as PD techniques, are practical measures that facilitate user participation in design.

Although the literature we have reviewed here provides rich examples of PD facilitation across distributed settings and within a few very large communities, few scholars have investigated the role of architecture in large scale PD. We hold that architecture is a key feature of both contemporary software products and the ecosystems that surrounds them. In the following subsection, we present concepts that we find relevant to study the role of architecture in large scale PD.

## 2.2 Balancing flexibility for design and use through architecture

As the literature review in section 2.1 highlights, with increasing scale of the PD venture, the distinction between design and use of the software product becomes increasingly blurred. Appropriation by new users and for new uses can take place either out of the box, through customization of generic features, or through the local development of custom extensions. However, this flexibility partly depends on architecture, which can exert a strong influence on software development and design (Baldwin and Clark 2006). In accordance with Van Schewick (2012) we understand architecture as the inner structure of a system, the components, what they do, and how they interact. This description of architecture applies well not only to software products and technical artefacts, but also to the structure of the surrounding ecosystems, including

| *Type of PD* | *Key characteristics* | *Literature examples* |
|---|---|---|
| *Singular PD* | **Politics of design:**<br>Users make feature-oriented design decisions based on concrete work tasks.<br>**Nature of participation:**<br>Situated and iterative face-to-face interaction between developers and users.<br>**Methods, tools and techniques:**<br>Prototypes, mock-ups, role-play, and workshops allow for hands-on evaluation of features by end users. | Braa (1995)<br>Ehn (1988)<br>Muller and Kuhn (1993) |
| *Serial PD* | **Politics of design:**<br>Developers and implementers configure and customise software to fit distributed but relatively similar work practices in collaboration with end users.<br>**Nature of participation:**<br>Developers and implementers work closely with end users and make on-site visits.<br>**Methods, tools and techniques:**<br>Implementers act as design mediators between users and developers. Implementers move from site to site and iteratively contribute to enhance the core system. | Finck et al. (2004)<br>Gumm (2007)<br>Titlestad et al. (2009) |
| *Parallel PD* | **Politics of design:**<br>Power users and local experts play a central role in eliciting requirements. Multiple projects undergo parallel design processes. Generic features receive priority over particular ones.<br>**Nature of participation:**<br>Software developers, customisers and implementers make short field visits and arrange workshops with user representatives.<br>**Methods, tools and techniques:**<br>Interaction between developers and users takes place through meetings, trainings, mailing lists, support calls and participant observation. | Hansson et al. (2006) Karasti (2010)<br>Obendorf et al. (2009) |
| *Community PD* | **Politics of design:**<br>The broader community negotiates generic features. Local adaptations meet specific needs without involvement from the core development team.<br>**Nature of participation:**<br>Power users and domain experts represent end users in meetings and workshops. The focus is on appropriation of the existing software across diverse uses and settings. Core developers make final generic product decisions.<br>**Methods, tools and techniques:**<br>Documented case studies and best practices concerning software appropriation circulate through workshops, newsletters and online resources. Inter-contextual workshops, online forums and social media facilitate feedback from user representatives and the wider community of users. | DiSalvo et al. (2012)<br>Hess et al. (2008) Karasti and Syrjänen (2004)<br>Oostveen and Van den Besselaar (2004) |

Table 1. Typology of PD.

community management, requirements filtering, development processes and use of information and tools to support interaction around design (Markus 2007).

Here we are concerned with how architecture influences key aspects of PD such as the politics of design, the nature of participation and the techniques employed to facilitate participation in design. For instance, as we noted above with reference to the politics of large scale PD, the power to influence the design and use of software tends to transition to a cadre of design mediators such as customisers, implementers and power users who represent end users as projects move from singular to serial and parallel PD. These design mediators understand how local needs can be accommodated through the configurable features of a software product (Titlestad et al. 2009). They are thus in a position to leverage an architectural understanding of the software to operationalise and give priority to design suggestions. Furthermore, end-user participation in development and adaptation of a software product can be enabled or constrained by the level of flexibility with the software itself (Wulf et al. 2008), which largely hinges on its architecture.

Despite the blurred distinction between design and use in practice, we here note a conceptual difference between flexibility for further development, which we refer to as 'design flexibility', and flexibility out of the box for use across a range of different, even unanticipated purposes, which we dub 'use flexibility'. Design and use flexibility have been described as relational in the context of large-scale information systems, as a high level of use flexibility reduces the need for change through design flexibility, and vice versa (Hanseth and Lyytinen 2004; Hanseth et al. 1996). Furthermore, the actual design flexibility of large and complex systems has been linked to modularity (Braa et al. 2007; Edwards et al. 2007). Modularity is the idea that rather than developing one complex system or product where functional interdependencies are hard to discern, one should strive to develop small and reusable parts with clearly defined interfaces between them (Baldwin and Clark 2006; Baldwin and Woodard 2008; MacCormack et al. 2006).

Platforms constitute one prominent example of scalable modular and layered software architecture that can be leveraged to balance design and use flexibility (Olleros 2008; Yoo et al. 2010). Platforms are architectures comprising three key elements: core components with low variability, complementary components with high variability, and interfaces for modularity between core and complementary components (Baldwin and Woodard 2008). Hence, platform architectures may allow PD practitioners to address the age-old challenge of catering for new users that were not part of early design process and allow them to adapt software in unforeseen ways (Ehn 2008). The separation of systems into a stable core and a complementary set of components where variability is high allows for greater design flexibility at the outer layer without compromising the core (Baldwin and Woodard 2008; Olleros 2008). The key purpose with a platform is to facilitate an economy of scale of the lean core, while further development can take place through the core's boundary resources (Ghazawneh and Henfridsson 2013; Yoo et al. 2010), such as standardised Application Programming Interfaces (APIs) and Software Development Kits (SDKs). Standardised interfaces allow platforms to be open for participation, especially through the development of third-party apps (Gawer 2009; Ghazawneh and Henfridsson 2013). In essence, platform architecture defines how innovation is partitioned between app developers and the developers of a platform core, thus shaping the space for participation in local innovation (Tiwana 2013, p. 38).

To develop software product features that are both desirable and feasible, a great deal of knowledge is needed, both about situated contexts of use and about the software architecture.

For instance, as Hess et al. (2008, p. 36) report from a large scale web-based community PD project, "[t]he technical preview, which had been introduced at the beginning of the project, had some unchangeable aspects that restricted the possibilities for innovation." Hence, the initial assumptions and design values of developers and early users both enable and constrain further design flexibility as they lay the foundation for the software architecture (Plantin et al. 2016). Historically, the technical aspects of software development have not been taken sufficiently into consideration in PD literature (Hansson et al. 2006). This, we argue, is particularly true when it comes to architecture. In this paper, we highlight the role of architecture in shaping the politics of design, the nature of participation and the use of PD techniques. We do this by following, over time, the differences in PD approaches along with the growing scale of a PD project. With respect to the implications architecture has on PD, we focus on the emergence of a platform and its surrounding ecosystem, which we in accordance with Tiwana (2013) consider to be inseparably intertwined.

# 3 Method

We report from a longitudinal case study of an ongoing, international development effort that currently involves health information system strengthening initiatives in more than 50 countries. We begin by describing the initial growth of a PD project 20 years ago, which introduces the context and background for our empirical research. Subsequently, we present our research approach, data collection, and data analysis.

## 3.1 Case context and background

Our empirical case concerns the participatory design of the District Health Information Software (DHIS), which primarily supports data collection, data analysis and visualization to inform decision making in public health care organizations. A group of scholars with the Health Information System Programme (HISP) at the University of Oslo in Norway mainly coordinates DHIS development and implementation. The software development efforts are intertwined with research and capacity building in the 'Global South'. The initiative has, from the very beginning, been guided by Scandinavian principles and methods of PD and action research; e.g.; (Braa and Vidgen 1995; Checkland 1991). Throughout the history of DHIS development, university employees have played multiple roles as researchers, software developers, project managers, support staff, trainers and evaluators. Members of the core development team in Oslo hold software engineering positions at the University of Oslo and receive funding primarily through donor grants.

The DHIS software started out as an MS Access-based desktop application to support tasks of public health care management in three pilot districts in Cape Town, South Africa, in 1994, following the fall of apartheid. At the time, emancipation was a central political theme, both in terms of equitable health service provision to all people, irrespective of background and 'race', and in terms of empowering local health authorities to offer services in the ways they saw most

appropriate. Thus the democratizing aims of Scandinavian PD found fertile ground, and DHIS evolved through rapid prototyping with local district staffs (Braa and Hedberg 2002). Over the next few years, DHIS saw a relative success in South Africa, emerging as the de facto national health management information system. DHIS slowly gained a foothold in other countries along with the cultivation of a network of stakeholders including universities, ministries of health, global regulatory agencies like WHO, international development funders, and emergent regional implementation partners such as HISP South Africa and HISP India (Braa et al. 2004).

In 2004, due to technological advances, growing computer literacy, and increasing access to the Internet by many potential DHIS users, work started on DHIS version 2 (DHIS2), a flexible and generic web-based tool developed entirely as free and open source software (FOSS). The new version involved a complete re-engineering and rewriting of the software code. The PD efforts forming the empirical basis for this paper primarily concerns the years after the launch of DHIS2 in 2006. We focus on this period, as this is when the traditional Scandinavian PD approach encountered marked challenges of scale in terms of the number and distribution of heterogeneous settings, developers, users and uses. DHIS2 implementations have scaled to more than 50 countries and different domains such as logistics, patient follow-up and disease surveillance, which in turn has affected the politics of design, the nature of participation and the use of PD techniques.

## 3.2 Research approach

The research reported on in this article has grown out of the four authors' individual analyses and collective discussions and reflections concerning the role of PD across different DHIS2 development and implementation settings. The case study is interpretive (Walsham 1995; 2006) and builds on the authors' experiences from DHIS2 project activities as well as contextual data obtained through participation in implementations, pilot projects, DHIS2 workshops and trainings. We present three vignettes from our longitudinal case study in section 4.1 - 4.3 to highlight how different types of PD have emerged in response to challenges of scale. Our combined research experiences cover multiple heterogeneous implementation sites, user groups and uses that relate to the same software development effort over a relatively long period. This allows us to explore the relationship between architecture and large scale PD, which itself is multi-site and long-term in nature. Our vignette selection represents a continuum of scale by zooming in on different, albeit overlapping, trajectories and settings.

## 3.3 Data collection

Table 2 below summarises the sites and activities that constitute the authors' sources of data for each of the three vignettes. The table also details our role-based access to secondary longitudinal and contextual data. More specifically, Author 4 has collected longitudinal data about the overall scale-up of DHIS2 development and implementation since 1996. This has taken place through research collaborations, participation in workshops, discussions in person and email, and field trips to countries with DHIS implementations.

Primary data collection for Vignette 1 from Sierra Leone constituted six field visits in the period 2008-2011. Author 3 organised two three-week training courses at the time of the initial introduction of DHIS2 to Sierra Leone, for all district monitoring and evaluation (M&E) officers and managers of health districts, as well as six M&E officers working at the Ministry of Health central office. More than 40 people participated in each course. The courses also served as venues for software customisation and requirements elicitation. Four district offices received subsequent visits twice for follow-up observations and interviews. Author 3 engaged in participatory observation and development activities during course sessions and district visits. In addition, Author 3 carried out informal interviews and group discussions with M&E officers and managers. The interviews were concerned with information needs in relation to current work practices, aspects of data quality, and the suitability of the new system including software functionality. Later visits followed up on user experiences with the system, as well as any changes made to software user interfaces, reporting forms, and software functionality. Field notes and photographs recorded observations in Sierra Leone. Extensive email exchanges documented software development and customisation efforts while Author 3 was not in the country.

| | *Primary data generating activities (case vignettes)* | *Secondary longitudinal data (case context and background)* |
|---|---|---|
| *Author 1* | Implementation, customisation, training and software development with DHIS2 'Tracker' in Uganda, 2012-15 (Vignette 2)<br>Participation in DHIS2 Academies 2012-15 (Vignette 3) | PhD student within the HISP and lead development coordinator of DHIS2 mobile phone-based features, 2011-2015 |
| *Author 2* | DHIS2 Academies 2013-15, DHIS-Mobile workshops and strategy meetings 2009-15, DHIS-Mobile implementation, training, evaluation (Vignette 3) | PhD student, Postdoctoral fellow and project coordinator with the HISP 2010-2016 Involved with Implementations of DHIS2 mobile solutions in India, Malawi, Zambia and Uganda |
| *Author 3* | Implementation, customisation, training, management of development in relation to Sierra Leone 2008-10 (Vignette 1)<br>DHIS2 Academies 2010-13 (Vignette 3) | PhD student, Postdoctoral fellow and project coordinator within the HISP 2002-2016, Involved with Implementations of various DHIS2 features in Cuba, Botswana, India, Malawi, West Africa |
| *Author 4* | DHIS2 Academies 2014-15 (Vignette 3) | Implementation evaluation, project advisor, management of global activities in the period 1996-2016, with a focus on South Africa, India, Ethiopia, Tanzania, Rwanda, Zambia |

Table 2. Authors' roles in activities concerning the three case vignettes and the HISP.

Author 1 collected data for Vignette 2 from Uganda during eight field trips of 1-2 weeks' duration between 2012 and 2015. Primary data collection activities constituted 23 interviews, participant observation of three in-country trainings, seven field visits to health clinics, four software developer workshops in Uganda and neighbouring countries, email discussions and phone conferences. The study informants included users at different levels of the health service, such as staff at the ministry of health central office, staff with national or regional responsibility for specific health programs, national and international NGOs working with health programs and software development in Uganda and local health clinic staff, including doctors, midwives and voluntary health workers. Interviews were conducted in the capital area, and during two field visits to rural health facilities in Uganda. The interviews focused on software feature requirements to support patient follow-up and clinical work practice. In one workshop, developers and users acted as patients and clinicians and went through the workflow of the software patient modules to elicit software requirements.

The empirical data concerning the DHIS2 Academies—Vignette 3—was collected through participation in seven workshops each ranging from eight to ten days, as well as the authors' involvement in five other workshops through remote planning and material development and online pre-Academy team-building activities with registered participants. Each author has participated in at least one DHIS2 Academy. Collectively, the authors have attended academies in West Africa, East Africa, Asia, and at the University of Oslo. The academies have ranged from 'basic', for newcomers to the software, 'advanced' for those with prior experience, and 'expert', attended mostly by NGOs, implementation partners and international donors. In most DHIS2 Academies, either the software development roadmap has had a dedicated time slot in the program or discussions have emerged ad hoc during workshop sessions. The authors have had access to material and presentations from the academies in addition to the authors' own notes made during academy participation. Author 1 videotaped one three-hour software requirements discussion during a developer workshop in Mombasa, Kenya.

Finally, we have all contributed to the collection and co-creation of naturally occurring data in the form of documents concerning DHIS2 development and implementation. This material includes reports, meeting memos, project evaluations, implementation budgets, grant proposals, software documentation and training materials. We have also studied requirement specifications and email lists at some level of detail, including a quantitative analysis of the developer mailing list that helped indicate different stages of scale in the history of DHIS2 development. For many of the workshops carried out in countries, relevant data in relation to user participation exists in the form of emails reporting on software bugs or feature requests. The authors, in their roles as facilitators or coordinators of user trainings, have mediated users' requests to the core developers on numerous occasions.

## 3.4 Data analysis

Data analysis has constituted iterative reflections on our long-term engagement with HISP and DHIS2, thus benefiting from overlapping data collection and data analysis (Boland Jr 2005) with blurred stages (Langley 1999). The initial motivation for the study came from two of the authors' observations that an increasing number of DHIS2 stakeholders expressed discontent

with the form and content of user involvement in DHIS2 development. For instance, an NGO working out of the first country to adopt DHIS, complained, during a plenary session at a DHIS2 Academy in Oslo, that submitted feature requests could "be in limbo for more than a year" without any feedback or notification from the core software developers. The authors found this and similar claims paradoxical in the sense that user involvement, through participatory design, had always been the hallmark of DHIS2 development. We considered the observable discontent regarding lack of feedback and transparency with the development process to stem, at least in part, from stretched coordination capacity. In addition, we felt that that stretched capacity had created a gap between global software development efforts and distributed situated implementation processes. Our emergent research interest was consequently to investigate this problem with PD on a large scale.

The empirical data analysed in this study stems from the authors' previous research in the HISP concerning both the scaling of DHIS2 implementations; e.g.; (Braa et al. 2004; Sahay et al. 2013; Sanner et al. 2012) and the application of PD across different country settings; e.g.; (Braa et al. 2004; Kossi et al. 2012). Hence, this study combines data sets, collected by the four authors at different places and different times, to explore the longitudinal scaling of PD. Initially, we drew on this larger composite data set to identify empirical dimensions of scale in relation to PD in HISP. We arrived at dimensions that informed our working definition of scale as the *number* and *heterogeneity* of distributed *users*, *uses* and implementation *settings*. Next, we combined qualitative and quantitative analysis to review a large volume of threads on the DHIS2 developer mailing lists. This allowed us to identify implementation settings and events in our larger data set that were well documented in terms of user involvement in design and that represented scaling of PD in the HISP venture along the aforementioned dimensions.

To explore the interrelationship between scale and PD further, we singled out three case vignettes based on two criteria. First, the authors had extensive experience with the particular empirical settings; i.e.; by having participated in DHIS2 implementation or training. Second, the chosen vignettes illustrated an increase in scale of the overall PD venture over time. Data concerning the case vignettes was analysed with a focus on how participation took place, what kind of tensions emerged regarding design prioritisations, and what participatory techniques were employed. In more detail, this included going through field notes and interview transcripts, videos and audio recordings to identify who participated (developers and users), how and on what issues participation was carried out, modes of communication (face-to-face on-site, email, trainings, etc.), and how user requests were categorised and responded to.

We took Kensing and Blomberg's (1998) three dimensions of PD, the politics of design, the nature of participation and the employment of PD techniques, as a starting point for fleshing out and comparing different forms of PD identified in our empirical material through the exercises mentioned above. This provided us with a framework for reviewing and synthesising extant literature concerned with large scale PD. We went back and forth between literature and empirical findings to define the key characteristics of four types of PD. Individual analysis work in relation to specific case vignettes was guided by meetings where we discussed our analytical categories through figures and tables drawn on whiteboards. A significant theme that emerged from our collective data analysis was the shifting nature of participation in tandem with the shifting architecture of the software itself. The software had gradually adopted what has been described in the literature as a platform architecture (Baldwin and Woodard 2008; Tiwana 2013),

both technically and in terms of the structure of the network of stakeholders that surrounded it. Specifically, DHIS2 was remodelled to separate core functionalities (the software core) from its extensions (called apps). This led us to inquire further to connect aspects of an emergent layered and modular platform architecture with the four types of PD. Finally, we observed that design and use had become conflated and interdependent due to the widespread adoption and local customisation of the software. We identified design and use flexibility as characteristics that could help us describe the role of a platform architecture in shaping different types of PD.

# 4 The case of participatory design in HISP

In this section, we present three vignettes from our longitudinal case study. The timeline in Figure 2 shows important milestones in relation to implementations, capacity building, and technical features throughout DHIS development. The vignettes concern version 2 of the software, called DHIS2, rather than the first version. Although the four types of PD coexist over time, each type has been more prominent in certain eras of DHIS2 development. The yellow colour in Figure 2 indicates the era of singular PD, green is serial PD, blue is parallel PD and pink signifies the emergence of community PD.

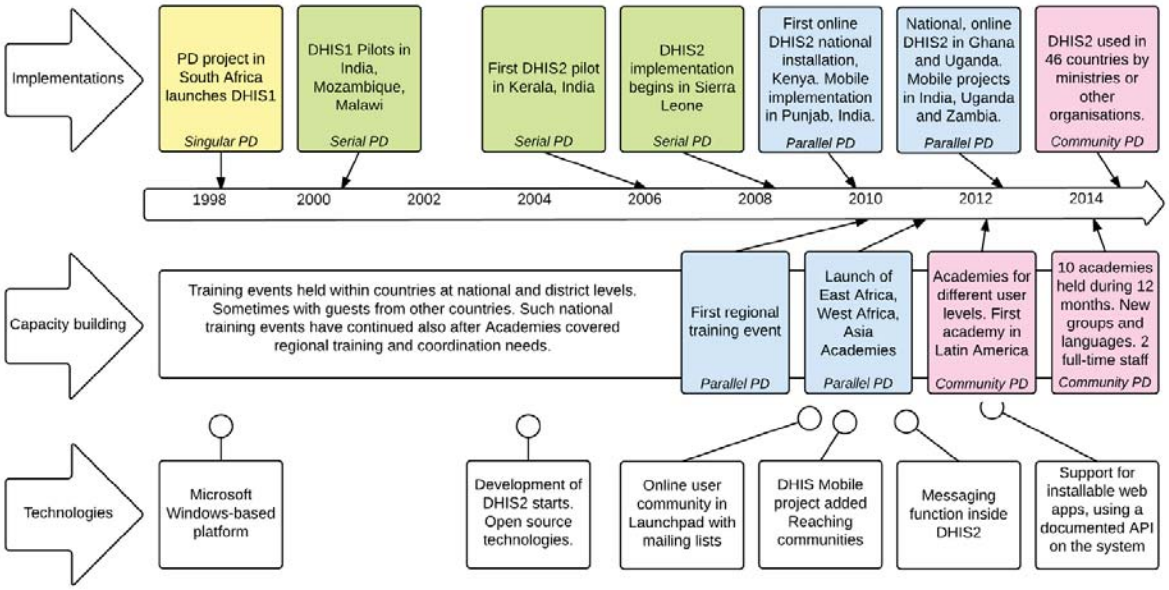


Figure 2. Summary of important milestones through 16 years of DHIS2 development.

## 4.1 Vignette 1: Serial participation (DHIS2 in Sierra Leone, 2008-2011)

While the early period of DHIS development (see Braa and Hedberg 2002) can be classified as classical singular PD, DHIS version 2 was mainly developed through what we have labelled serial PD from around 2005. Every new PD effort thus builds on and extends previous PD efforts at a different time and place. We turn to one particular implementation to show how this was carried out. Our focus is on the interplay between situated PD in Sierra Leone and global core development. The core development team was initially heavily involved in Sierra Leone, but then shifted their attention to new country implementations. The strongest influence from PD in Sierra Leone on the 'global' development of DHIS2 was thus in the early implementation phase.

Sierra Leone was one of the first countries to implement DHIS2. At the time, the development team in Oslo mainly consisted of Master and PhD students, who did software development and implementation work as part of their degrees. The bulk of users were the monitoring and evaluation (M&E) officers at the district level, who were in charge of obtaining approximately 15 different paper reports from each health facility, and entering the data into DHIS2. In each district, there were two M&E officers, as well as the District Medical Officer (DMO), who was in charge of administering health services in the district. In addition, there were different health program officers responsible for different service areas. At the national level, a team of three people led the efforts to customise the national database and keep it synchronised by importing data from the offline district databases. Their roles included supportive supervision of the district staff and communication with DHIS2 implementers and developers. The project lead, with the World Health Organization (WHO) in Geneva, had no prior experience with DHIS2. However, while not an in-country user, the project lead was active in terms of accessing the Sierra Leone database, preparing information products, and conducting analysis. The distribution of various DHIS2 users, in relation to a single-country implementation, points to differences in user involvement even at an early stage of development.

For the most part, user involvement pertained to customisation of DHIS2. Only the more advanced requests made by the project lead at WHO were translated into direct changes in the software code. Inputs from district users mainly concerned local customisation of the database and user interfaces. The core DHIS2 development was, at this time, still based on redressing the functionality of DHIS version 1 into a web-based format. As DHIS2 had very few actual implementations, novel requirements from Sierra Leone received considerable attention. The Sierra Leone rollout started with a three-week intensive course for all district M&E officers and DMOs, many of whom had never used a computer before. Subsequently, core developers and implementers, including Author 3, visited four pilot districts. Each district needed on-site assistance in setting up a local copy of the software and database, as the extant infrastructure did not allow for an online implementation. During this period, user involvement was limited and feedback from users mainly concerned bugs in the software and hardware trouble. The initial focus was limited to data entry and the generation of simple reports.

Following the pilot implementation in four districts, further customisation of the system, such as defining metadata and custom report templates, was done in collaboration between

developers and the national team in Freetown, mediated by the WHO project lead. Soon, however, district users provided feedback on the design of the custom reporting tools. In mid-2008, a core developer of DHIS2 visited Sierra Leone, and found that "health officers were happy with the improved reporting functionality, and many requested new reports and extensions to existing ones." The mentioned reports featured customisable templates, whereby the district users could potentially create new reports and alter report attributes. User involvement in the development of the DHIS2 software core, which required programming skills and deep knowledge of the existing code, was limited, but as the various users got more familiar with the system, ideas of improvement started to be voiced. The project lead, while based in Geneva, made frequent visits to Sierra Leone to work with the team in Freetown and communicated requirements to the core developers.

A major area of interest with the implementation was reporting completeness; i.e.; whether the many health facilities in Sierra Leone had submitted their paper reports to districts and if these reports had then been entered into the database. To monitor this, new functionality had to be developed. This resulted in the design of a 'complete button'. By clicking on the button, the person responsible for data entry could mark a report as submitted; i.e.; complete). Most of the time, a report would contain many empty data entry fields. Smaller clinics provide fewer notifiable health services, and some public health events are just rare. Hence, the label on the button read 'complete' to signal that despite an apparent lack of data, the report was complete. Completeness of reporting could then be calculated by tallying clicks on this button. However, many district users did not click on the complete button. Some forgot to click it, some had not received appropriate training, and some avoided the button because clicking it triggered form validation rules, which pointed out errors, and caused more work. An email exchange between the project lead and the DHIS2 lead developer from early 2009 touches upon the issue:

> An alternative would be for a DHIS2 expert to write some code that will automatically 'press the complete button' when one or more of the compulsory data elements in the form has a non-zero value. Please, please, please. (Project lead)

> We should develop completeness functionality in DHIS2 based on compulsory data elements as discussed before, will do this as soon as I get time (Lead developer)

Later, the possibility to define some fields in forms as compulsory was developed, which in turn allowed for another way of measuring data completeness. The vignette from Sierra Leone shows how user participation took place in relation to the local deployment and the global software development. Getting feedback from end users at the district level was feasible, with Author 3 and the local customisation team traveling from district to district to engage in support activities and requirement discussions. However, with new countries adapting DHIS2 in the years to follow, including whole states in India, user participation soon turned into streams of parallel activities. For a while, the development team in Oslo was able to be involved in and follow up on most implementations. Developers travelled extensively and engaged in requirements elicitation from countries, while long-term follow up was facilitated through communication via phone calls, emails and mailing lists.

## 4.2 Vignette 2: Parallel participation (DHIS2 Tracker in Uganda 2012-2015)

Vignette 2 concerns parallel participation in the development of the DHIS2 Tracker module. By 2012, several countries had implemented DHIS2. Kenya was the first country to deploy a fully online instance in 2011, where a central national office could manage all database and server maintenance. This was to become the dominant mode of DHIS2 deployment in subsequent implementations, testimony of the great mobile infrastructure advances across the developing world at the time. By 2012, a team of developers at HISP India had developed a local module to support clinical follow-up of pregnant women. This new module was a departure from the traditional aggregate statistics-only data model of DHIS2. It was also a departure from the norm where DHIS2 software developers were only situated in Oslo. At first, developers in Oslo considered this module as a hack, but they later re-engineered the highly demanded functionality of the module into the core of DHIS2, as the Tracker module. The DHIS2 Tracker was subsequently developed and implemented in several countries in parallel, and this vignette looks at how this process played out primarily from the point of view of an implementation in Uganda.

The Tracker was a new module in DHIS2 at the time of implementation in Uganda, and it still had design connotations to specific use in India. While a couple of countries started to use Tracker around this time, core designers from Oslo considered the Uganda implementation central in the design process:

> …we have to be part of the configuration of the system [in Uganda], or else we won't know which new features to implement and how to improve it. (Senior developer)

As major changes in the software were required to adapt to implementations in Uganda and other settings, several staff members from the core development team in Oslo were involved over a period of more than two years. The developers considered the implementation in Uganda as a template upon which one could forge generic features. Around this time, DHIS2 experienced massive adoption with the number of client countries growing quickly. Many of the new adopters considered implementing the new Tracker module to support health program follow-up. Consequently, while the developers engaged in situated PD in Uganda, there was an increasing pressure in terms of new requirements from other countries. Developers soon began to be involved in parallel development processes. This created tensions both among developers, and among local users and project stakeholders in Uganda.

The process of distilling generic requirements and developing corresponding software features was recognised as time consuming by local project staff in Uganda, who stated, "It is frustrating to have to wait for a new 'global release' every time a requirement or change has been suggested." The staff in Uganda also felt that their specific requirements were not met because the software had to maintain its generic features. This was brought up in a workshop with the Oslo-based developers, who believed they were largely accommodating local features, while the local staff felt their requirements were sacrificed on the altar of generality. An example of this tension was requirements regarding which start date to use when automatically scheduling events in a health program. The date of a person's enrolment in a health program and the automatic generation of schedules based on this date differ greatly from program to program. For a

pregnancy, a commonly used start date is the often uncertain last menstrual period, leading to a roughly estimated date of delivery. For postnatal and vaccination programs, the start date is given as the birth of the baby, which is comparatively accurate. The logic of assigning dates is also different. Antenatal follow-up visits may be revised based on the condition of the pregnant woman, while vaccinations are predetermined by schedules at the health clinic. When the same generic software was used to manage many different types of programs, the scheduling mechanism, which was initially considered generic, turned out to restrict particular workflows. Over the entire project period in Uganda, no satisfactory solution was implemented to flexibly handle contradictory scheduling requirements.

For the core developers in Oslo, the emergent parallel development processes also had implications. The graphical user interface and features that had been carefully modelled in Uganda were subsequently picked up by design teams in other countries to fit other requirements. Because all countries used the same open source code base, developers working for one country could make adaptations that affected everyone else. One particular user interface changed back and forth several times, due to teams in different countries each adapting it to their needs. Some of the core central developers were caught in the middle, constantly changing the software and increasing complexity.

During this particular implementation period, DHIS2 went through a major architectural change and key features were revised to fit a new modular architecture. When a team of central developers rewrote the code base for the Tracker module to fit the new platform concept, a large part of the embedded local requirements from Uganda were lost. This led to heated discussions within the central development team, whereupon one of the central developers expressed his sentiments by saying, "You cannot expect your requirements from one case in one country to be leading the requirements for this generic software." The comment illustrates how the overall DHIS2 development had moved from a serial PD-oriented process to a process that encompassed parallel cases, sometimes with conflicting requirements. It was no longer enough to engage in participation on the ground, since 'the ground' constituted many different realities.

The design experiences from Uganda also became central during the development of a new Android client meant to extend the DHIS2 Tracker module. A key focus with the Android development was to make a software development kit (SDK) that would allow different projects to make their own mobile applications with specific user interfaces, rather than a configurable but non-extensible product. The vignette concerning the Tracker in Uganda is representative of a transition in the conduct of PD in the HISP. First, it shows an emerging *parallel* mode of PD, with some developers active in one country while others were active in another. Tensions ensued and led to the adoption of a more pronounced strategy for the development of generic software. The motivation for PD would then shift towards feeding inputs to this software development process, while at the same time trying to customise generic software modules to meet local needs. Later, the mounting tensions between the parallel projects informed the development of a more platform-oriented software architecture. This phase was also characterised by a transition where the core team started using the platform's capabilities themselves to make applications using open APIs, rather than the earlier mode of implementing functionality inside the core and just exposing APIs for others to use. While generic functionality remained hidden in the software core, a customisation layer and a support structure emerged to support the development of custom applications as extensions of the DHIS2 core. Vignette 3 in the following sub-section

details the emergence of a suite of workshops, called DHIS2 Academies. The DHIS2 Academies have played a key role in balancing software development with growing community expectations and needs.

## 4.3 Vignette 3: Community participation (DHIS Academies, 2010-2015)

The third and last vignette illustrates Community PD and concerns the DHIS2 Academies. Since 2010, the DHIS2 community, with coordination from Oslo, has arranged a number of 1-2 week workshops targeting mainly national and regional implementation teams. Only a few academies were held the first few years. However, more recently, academies are held regularly in Africa, Asia and Latin America, and have diversified to cater for implementers and software customisers at different experience levels. The academies mainly consist of training, but they also provide an arena for local implementers to keep up to date with recent software developments, share use cases and experiences, build professional networks, and give feedback to developers directly.

In the period covered by this study, core developers would regularly participate at DHIS2 Academies. This ensured close communication between them and implementers, who represented local users. The academies included sessions where developers presented new and upcoming features of the software. Participants could then discuss this and suggest further requirements. The DHIS2 Academies thus offered a better alternative, logistically and financially, than distributed discussions confined to individual countries. In addition, participants expressed that they valued coming together to share experiences through mingling and plenary discussions about software use, feature requests, technical challenges, and the organising of the DHIS2 community itself. During a 12-month period from 2014-2015, ten DHIS2 Academies were held with the number of fee-paying attendees in each workshop exceeding 50, which illustrates the significance of these workshops.

With a community of users and user representatives from different countries, the Academies provide a venue for distilling requirements and avoiding the coordination hurdles encountered with parallel PD. During a DHIS2 Academy in Mombasa, Kenya in 2012, a group of 12 specialists from different countries in east Africa spent three hours discussing requirements for mobile phone-oriented features in DHIS2. During a discussion concerning a feature that had been difficult to implement across multiple contexts, one of the core DHIS2 developers shared his positive assessment of the experience, saying,

> In order for us to create new functionality, we really need a proper use case, a real one, so we can communicate. We cannot just [gesture showing something coming out of the air]… We have been thinking about this [feature] for the longest time, from the very beginning, but we never found good examples of what would be beneficial for the health worker as we've discussed today.

In particular, generic functions that have local variations in workflow and local improvisation have been difficult to develop without discussions concerning specific use cases, with one example being integrated disease surveillance and response (IDSR). IDSR is a process for discover-

ing and managing disease outbreaks. IDSR has a number of global WHO guidelines, but still requires local adaptation to succeed in practice. During an Academy session, a discussion on usage of SMS alerts for IDSR opened the requirement scope to include many other non-IDSR functions that could benefit from SMS alerts, such as SMS reminders to health workers to improve data quality in routine health data reporting. One of the DHIS2 Academy coordinators thanked the group by saying,

> This is very good. We have been thinking about some of these cases, but very narrow. Now it's clear that it's a whole area with different […] it has more than one use case.

The academy discussions mentioned above helped clarify design features that despite international guidelines had local variations and idiosyncrasies. During the same session, one of the coordinators tried to uncover local variations by saying,

> I'm wondering about IDSR. When there is an outbreak, to which extent is the process based on these structured processes versus people picking up the phone, coordinating on the phone, and breaking out of the process…

A long discussion followed, where representatives shared their views on how IDSR was managed in their respective countries.

In addition to providing valuable requirement input to the core development team, participants stated after the session that it had been useful to discuss use across national boundaries. Since the Mombasa Academy, several of the attendees have been active in arranging new DHIS2 academies and have been teaching and implementing DHIS2 in various countries. The sessions were videotaped so that other community members such as developers and implementers who did not attend the workshop could learn from the discussions. The workshop was an attempt to bring developers from the other side of the globe closer to users, or at least their local representatives. Interestingly, despite the consensus during the session on a number of features for IDSR, these features were not implemented in the software due to resource limitations, until 2014 when a large international NGO paid for the development of the features as part of their DHIS2 adoption. This brings up another aspect with increased scale of PD projects: the prioritisation of design proposals.

At an Expert Academy in Oslo in 2015, with more than 80 participants from international NGOs, individual consultants, and some prominent country implementation representatives, a session was held on the software development process and how various stakeholders could interact with it. Although prioritisations and final decisions about standard releases are made by the core development team in Oslo, the development coordinator listed three different ways for community members to propose new features: 1) write on the developer mailing list, 2) participate at DHIS2 Academies, or 3) write and upload a well-defined specification. However, these modes of participation are not equally accessible to all users and organisations. Interestingly, the lead developer also presented a list of six qualitative evaluation criteria employed to rank functional requests. The list constituted the following criteria: perceived usefulness to the wider community of users, level of software development effort, perceived benefit beyond what is already possible with the software, interdependencies with other functions and modules, expected level of participation and feedback from the owner of the request, and availability of funds. Based on these criteria, it is not surprising that late adopters such as well-funded international

NGOs have become increasingly influential in shaping the DHIS2 software development agenda. The DHIS2 Academies also serve as venues to vent frustrations and discuss concerns, such as potential domination from resourceful international NGOs over early adopters of DHIS2 with limited technical capacity and funds, like ministries of health in less developed economies.

International NGOs with relatively deep pockets are able to travel and sit down with the core software developers in Oslo, and therefore have the opportunity to invest in more face time to make sure their requirements are well understood and recognised in future development of the software. These stakeholders are also in a position where they can fund or hire developers directly to implement their specific requirements, for instance as third party apps, and their wishes are more likely to make it into production at an earlier stage. A recent trend with DHIS2 development has been the development of third-party apps by international NGOs to meet the needs of users in their own organisations. Many of these apps could potentially be useful to other organisations such as ministries of health. However, most custom apps have not been shared with the broader software community. One possible explanation for this is that sharing generates additional responsibilities and costs associated with making solutions more generic and maintaining them over time.

The vignette from the DHIS2 Academies shows how community PD has emerged as an additional type of PD in the HISP over the last few years. While some developers continue to be engaged in specific projects and parallel PD, there are just too many activities for the global development team to be engaged everywhere. The DHIS2 Academies, which target users and community members at different levels, have emerged as occasions for community filtering of requirements and getting feedback from a variety of users.

# 5 Discussion: Architectures for participation

In section two, we reviewed extant literature and identified four types of PD with respect to scale, namely: singular, serial, parallel, and community PD. From our longitudinal case study, we find that all four types of PD have emerged along with the scale of the project. This can be seen as a consequence of key developers and project managers trying to uphold political and pragmatic ideals of user participation, while grappling with a rapid increase in the number and distribution of heterogeneous user needs and requirements (Braa and Sahay 2012; Shaw and Braa 2014). Rather than one type of PD replacing another entirely, we find that different types of PD coexist and interplay in the development of DHIS2. In parallel with the emergence of different types of PD, we also note the emergence of a generic software product, increasingly structured as a platform. We now turn to our discussion on how the emergent modular and layered architecture enables and constrains PD along three key dimensions (Kensing and Blomberg 1998): the nature of participation, the politics of design, and techniques for participation.

## 5.1 The nature of participation

One could argue that DHIS2 has been designed with a layered architecture from the very beginning, the backend with data models and core functional modules, and the frontend with customisable features and user interfaces. While the backend would be invisible to most users, the frontend would allow for some customisation of both the backend content and frontend views. This included configuration of what data to collect, process, and present to users. Early implementations of DHIS2 thus applied PD in frontend customisation, administered by implementers together with local users. As Vignette 1 from Sierra Leone illustrates, there were occasional PD processes whereby core developers made changes to the software code. This primarily took place when the scope for customisation was exhausted, and additional features needed to be developed.

Over time, DHIS2 was adopted by more diverse users and organisations, in distributed settings (Gumm 2006; Obendorf et al. 2009). We understand this as increasing scale. The modular development of generic functionality allowed for localisation and appropriation through the customisation of a frontend layer. In turn, this allowed for parallel PD processes across implementations. As the core developers were increasingly unable to be directly involved in implementations, they came to rely more on communication with DHIS2 *implementers*, who were skilled mediators of software requests and issues. This, however, necessitated the availability of qualified mediators who knew the software well and could translate requirements to core developers.

Tensions between design and use were frequent with the release of new features that were derived from particular use cases in a particular context. With new features derived from situated PD, the flexibility for use across a range of tasks may be low because alternative areas of use have not been explored. Even if parallel PD is employed to develop more generic solutions, increasingly distributed adoption and use of the feature places constraints on how well this can be achieved without compromising particular needs (Hansson et al. 2006). Vignette 2 concerning the Tracker implementation in Uganda exemplifies this tension between generic design and local use.

The emergent platform architecture of the software offered a partial remedy to tensions between the generic and the specific. In addition to the development of standard bundled apps, the division into a stable core and boundary resources, such as APIs and SDKs, allows for the development of custom apps to meet particular user needs. In short, a platform architecture allows innovation to take place locally without compromising any of the software components used more widely. The application of PD is then uncontroversial in terms of conflicting user needs and the logistics of PD. On the downside, useful functionality developed as custom apps by local developers often remains unknown beyond the local implementation. Hence, a software platform architecture alone is not a strong vehicle for very large scale PD, unless it receives support from its surrounding ecosystem. This can for instance be facilitated through inter-contextual workshops, such as DHIS2 Academies, and community circulation of power users' case studies (Obendorf et al. 2009), or video material from discussions and presentations as we have also seen in the case of DHIS2.

The organisation of capacity building and community management efforts in HISP has mirrored the growth and scale of DHIS2 software adoption. The DHIS2 Academies emerged

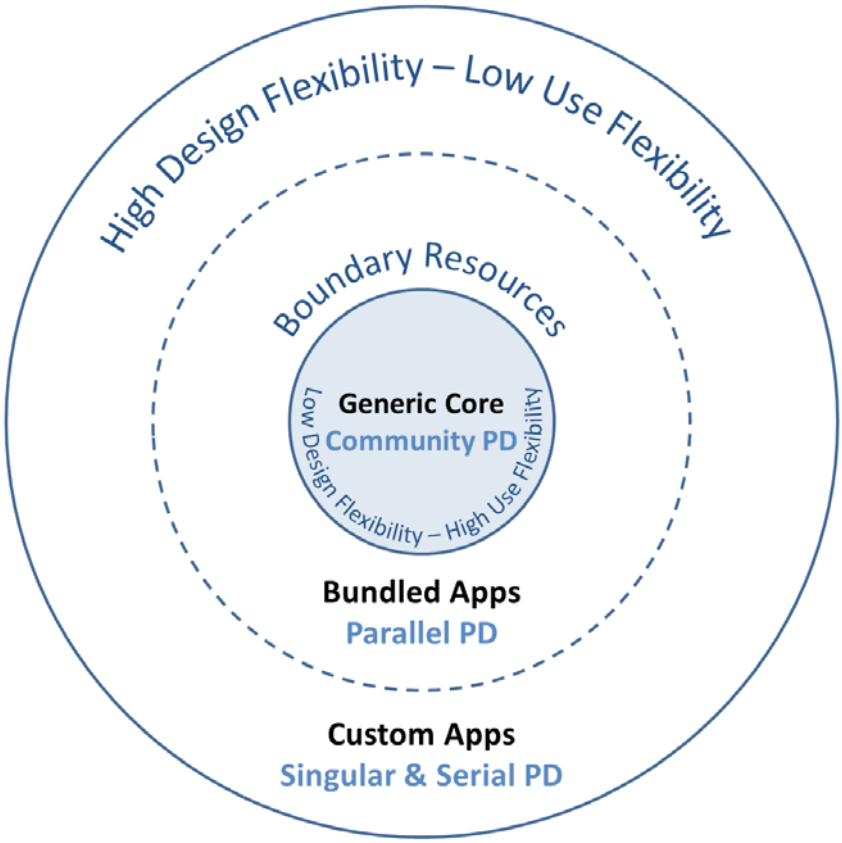


Figure 3. A platform architecture for Participatory Design.

as a more cost-effective and efficient answer to the need for distributed training and experience sharing and could be seen as integral to the DHIS2 platform ecosystem. Hence, workshops play a key role in shaping the nature of participation in that they offer a venue for community PD, where participants negotiate design on behalf of different user groups. The latter is important in that it both smoothens out differences between various groups' design requirements, through community filtering (see; e.g.; Pollock and Williams 2007), and that it allows core developers to identify potential resolutions to conflicting user needs by altering design at different levels of the software architecture.

In sum, we find that a *platform,* as one type of architecture, constitutes an enabler of large-scale PD (Figure 3). The concept of a platform, much like the notion of architecture, can denote both a model of the software itself and the ecosystem around it. The platformisation of both the software and the surrounding ecosystem allows PD to scale. With reference to our definition of scale, seen as the number and distribution of heterogeneous settings, developers, users and uses, we find that a platform supports singular PD, predominantly at the outer layer, as shown in Figure 3. The challenges related to logistics of participation and design priority setting has been mitigated by the platform architecture, both because more singular PD can be carried out on specific apps, and because community PD plays a harmonising and standardising role in managing the platform core. However, the technical architecture of a platform and its surrounding ecosystem are complementary. A modular and loosely coupled software architecture will itself not necessarily support scaling of *participation in design*. Likewise, structures that support the logistics of participation and negotiation of requirements will not necessarily succeed without

a flexible software architecture that allows heterogeneous user inputs to be accommodated in design (Tiwana 2013; Wulf et al. 2008).

The configurable middle layer of the platform allows implementers to customise the software to meet end user needs within the scope of the pre-built apps. These apps are generally part of the standard software releases, labelled as 'bundled apps' in Figure 3. Recall that in section 2.2 we differentiated between flexibility for further development, which we refer to as design flexibility, and flexibility for use across a range of different purposes, which we dub use flexibility (Hanseth and Lyytinen 2004; Hanseth et al. 1996). In general, the bundled apps have a high degree of use flexibility through customisation, but low design flexibility due to software interdependencies between bundled modules and widespread adoption and use. As is currently the case with DHIS2, only senior developers, employed in full-time positions, work on the bundled apps, which provide the most widely used functionality of the platform. The dashed line in the outer layer in Figure 3 illustrates that although bundled apps need to interface with the platform core, through boundary resources, in the same way as custom apps, they differ in how they are developed and maintained. In particular, the core development team coordinated the development of bundled apps through standard software releases. The outer layer in Figure 3 represents custom or third party apps—often with low flexibility for use across tasks, due to specific design requirements, but high flexibility for further design improvements, due to fewer dependencies.

## 5.2 The politics of design

Figure 3 above illustrates how we consider a platform architecture to support different types of PD at different layers. This in turn has implications for the politics of design, in the sense that different power struggles may take place in relation to different platform layers and through involvement of different stakeholders. As we have pointed out, the core of a platform is relatively stable, and its developers seek generic solutions. As is the case with most software platforms, the core developers, such as the DHIS2 senior developers in Oslo, make final decisions regarding feature roadmaps and revisions of the software core. Generic core development steers the general movement of the software by allowing for new functional possibilities and opening up for use in new domains, based on assessments of community needs.

The development and refinement of a platform core is far removed from traditional politically motivated PD projects of empowering situated blue-collar workers. Instead, the political project of core development may be to focus on eliciting requirements from disempowered organisations, such as ministries of health, in low and middle-income countries. In particular, the political agenda could be to assist governments in their struggles with proprietary software vendors and large international organisations who seek to steer development for their own reasons and on their own terms. Through Community PD, the target for empowerment has shifted from the situated individual worker towards whole organisations and large cadres of users. Building capacity, both technical and organisational, is key to empowering organisations to design and configure the information systems they need, in the fashion they deem most relevant.

Classical situated PD is more pronounced at the outer layers, but will occasionally lead to changes in the core, especially at points of major revisions, such as with the initial implementation of DHIS version 2 in Sierra Leone, where many design decisions were still up for grabs.

Similarly, the Indian Tracker module, which was initially seen as a hack, was later incorporated as a core DHIS2 module. As the core software modules have become more stable, there is now an informal set of evaluation criteria applied by the core developers to determine how, when and for what purpose significant revisions should be made. For instance, core developers may get involved in the redesign of custom apps, developed through outer-layer innovation, to ensure that new functionality can be bundled with standard releases and adopted by more users in different contexts. Common requirements can potentially be accommodated in the platform core, but a main activity is to provide a flexible data model and stable boundary resources for the outer layers to leverage. New modules or apps can then be developed loosely coupled to the platform core, through the involvement of either core developers themselves or local third-party software developers. However, the appreciation for and employment of PD principles by third-party software developers at the fringes of a platform can only be encouraged, not ascertained, by the core developers and political agents of the platform.

Similar to the platformisation of the DHIS2 software, support structures, such as the DHIS2 Academies have been divided into basic, advanced and expert levels. In turn, these levels have been thematically modularised to cater for different user roles and interests. Increasingly, core developers are no longer present at DHIS2 Academies held around the world. The formation of an annual Expert Academy in Oslo, with decision makers, funders, and a few country experts, maintains at least one venue where core developers and a heterogeneous group of user representatives and other stakeholders meet. However, the attendants at these Academies are of course not the typical DHIS2 users, most of which are data entry clerks and health facility and district managers using basic functionality, but implementers and national core team members who act as intermediaries. The level of end-user involvement in domestic requirement development processes vary, and is largely unknown to the authors.

## 5.3 Participatory design techniques

An important architectural development of a platform is the stabilisation of boundary resources such as a web-API and an SDK, which in theory allows any third-party developer to make custom apps based on local requirements (Tiwana 2013). As such, platform boundary resources serve as potential means to facilitate situated PD at the fringes of the platform. To date, DHIS2 apps have mainly been developed in Oslo. However, the development of boundary resources has made it possible to stabilise the core and at the same time allow for customisation, prototyping, experimentation, and local development. This has two important implications for PD. First, it allows situated PD to be carried out directly in relation to apps. PD can take place either through custom configuration of the widely used, generic apps bundled with standard releases, such as data entry modules and general visualisation tools, or more specific apps developed with specific work practices in mind. Second, people other than the core developers can do such development, since software programmers will only communicate with the software core through boundary resources like the web-API and the SDK without the need to comprehend the internal dynamics of the core itself.

The organising of the Health Information System Program (HISP) has followed a similar trajectory towards platformisation. User involvement has become layered too. In addition to

direct *in situ* involvement, the DHIS2 Academy and corresponding user community channels such as email lists, forums, and online development resources allow for distributed and layered user participation. The Academies were begun in response to a growing number of implementation sites, to be able to offer training in customisation of DHIS2 bundled apps to more than one country at a time. As stated above, the DHIS2 Academies have shaped the nature of participation, but they are also important techniques for large-scale PD. Academies or workshops bring various users and developers together. They serve as arenas for sharing new requirements and development agendas, eliciting feedback from users, and for various users to learn from each other. Since the vast majority of implementations are in the field of health, the various stakeholders have some shared understanding through common needs. Health program management has to a certain degree been standardised through the efforts of large international agencies, such as WHO, though there will generally be local variations. Requirements have been synthesised and filtered through community discussions and negotiations at DHIS2 Academies. Common values and a mutual understanding within the community informs the search for generic solutions similarly to the community-filtered distillation of generic requirements observed by Pollock and Williams (2007) in relation to global ERP development.

In addition to the Academies, the email lists, online development resources, and other online channels such as pre-Academy training programs and an emerging online learning platform offering standardised courses, constitute the platform ecosystem. These tools and structures offer distributed users and developers an arena for communication. In general, techniques associated with traditional singular PD tend to transition towards the outer layer in Figure 3 above. New software implementations, such as when a country adopts DHIS2 to be a national health information system, typically focus on developing the system within the frame of extant functionalities. A growing number of experts and implementers can facilitate this customisation in close collaboration with local stakeholders. Hence, features that were once developed through someone's situated PD effort, has become incorporated and generalised into someone else's off-the-shelf software package.

# 6 Conclusions

Based on a narrative literature review, we identified four types of PD along the dimension of scale. We then applied these four types to our empirical case in order to address our research question: What role does architecture play in large scale PD? We do this by discussing the role of architecture in relation to key aspects of PD, namely the nature of participation, the politics of design and techniques for PD. In this paper, we have exemplified the application of PD through more than one decade of development and use of an increasingly complex software product. The main characteristic of this journey has been the emergence of a platform for PD. Specifically, we have considered PD in relation to a platform architecture and how the four different types of PD may play different roles in relation to the emergence of different layers of a platform and its surrounding ecosystem.

We derive three key points from this story. First, our model of a layered architecture shows that PD may take place even with a large base of heterogeneous users and settings. Layering,

both in the software development and in the conduct of PD, can reduce the logistical challenges of large-scale PD. This helps resolve, through design and use flexibility, some of the conflicting demands that a growing number of heterogeneous users and uses can create.

Second, user participation in design is more direct and open ended at the fringes, where standardisation and generic features are less of a concern and dependencies are fewer. For instance, PD takes place in parallel through the customisation of bundled apps. The room for classical PD is particularly elaborate in the early stages of design of new custom apps. In these processes, boundary resources such as APIs and SDKs are both design enablers and constraints. Community PD can play a role in the developments of the software core and the more mature apps, but not as pronounced as new development projects around custom apps. More broadly, community PD plays a new political role in giving voice to different user groups and organisations and in allowing for transparent negotiations of requirements, interests and values that set the general direction for further software development.

Third, a platform for PD, as we see it, depends on support from both a technical architecture and a surrounding ecosystem. The modularisation of DHIS2 and the growth of the academies and various other community mobilisation channels have developed in tandem. This is not coincidental. Both these measures have been responses to growing scale, and both have allowed users to participate in design. We thus see the software architecture and its surrounding ecosystem as constitutive of a platform for large scale PD.

# References


Baldwin, C. Y., and Clark, K. B., (2006). The architecture of participation: Does code architecture mitigate free riding in the open source development model? *Management Science*, (10:7): 1116–1127.

Baldwin, C. Y., and Woodard, C. J., (2008). The architecture of platforms: a unified view. In: *Platforms, Markets and Innovation*, A. Gawer (ed.), Edward Elgar Publishing, Northampton, MA, USA, pp. 19–41.

Balka, E., (2006). Inside the belly of the beast: the challenges and successes of a reformist participatory agenda. In *Proceedings of the ninth conference on Participatory design: Expanding boundaries in design-Volume 1*, G. Jacucci, F. Kensing (eds.), ACM Press, New York, pp. 134–143.

Balka, E., (2010). Broadening discussion about participatory design: A reply to Kyng. *Scandinavian Journal of Information Systems,* (22:1): 77–84.

Bjerknes, G., and Bratteteig, T., (1995). User Participation and Democracy: A Discussion of Scandinavian Research. *Scandinavian Journal of Information Systems*, (7:1): 73–98.

Bjerknes, G., Ehn, P., and Kyng, M., eds., (1987). *Computers and Democracy - A Scandinavian Challenge*, Avebury, Aldershot.

Boland Jr, R. J., (2005). Interpretation and theory in qualitative research. In: *The art of science,* S. Tengblad, R. Solli, B. Czarniawska (eds.), Liber & Copenhagen Business School Press, Malmö, pp. 219–236.

Braa, J., Hanseth, O., Heywood, A., Mohammed, W., and Shaw, V., (2007). Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy. *MIS Quarterly*, (31:2): 381-402.

Braa, J., and Hedberg, C., (2002). The struggle for district-based health information systems in South Africa. *The Information Society*, (*18:*2): 113–127.

Braa, J., Monteiro, E., and Sahay, S., (2004). Networks of action: Sustainable health information systems across developing countries. *MIS Quarterly*, (*28:*3): 337–362.

Braa, J., and Sahay, S., (2012). Participatory Design within the HISP network. In: *Routledge International Handbook of Participatory Design*, J. Simonsen, T. Robertson (eds.), Routledge, New York, pp.: 235-256.

Braa, J., Titlestad, O. H., and Sæbø, J., (2004). Participatory health information systems development in Cuba: the challenge of addressing multiple levels in a centralized setting. In: *Proceedings of the eighth conference on Participatory design: Artful integration: interweaving media, materials and practices-Volume 1*, A. Clement, P. Besselaar (eds.), ACM Press, New York, pp. 53–64.

Braa, K., (1995). Priority workshops: springboard for user participation in redesign activities. In: *Proceedings of conference on Organizational computing systems*, N. Comstock, C. Ellis (eds.), ACM, New York, pp. 258–267.

Braa, K., and Vidgen, R., (1995). Action case: exploring the middle kingdom in information system research methods. In: *Proceedings of 3rd Decennial Conference Computers in context: Joining Forces in Design*, Århus, pp. 50-60.

Brigham, M., and Introna, L. D., (2007). Invoking politics and ethics in the design of information technology: undesigning the design. *Ethics and Information Technology*, (9:1): 1–10.

Bødker, S., Ehn, P., Kammersgaard, J., Kyng, M., and Sundblad, Y., (1987). A Utopian experience. In: *Computers and Democracy - A Scandinavian Challenge*, G. Bjerknes, P. Ehn, and M. Kyng, (eds.), Avebury, Aldershot, pp. 251–278.

Bødker, S., and Pekkola, S., (2010). A short review to the past and present of participatory design. *Scandinavian Journal of Information Systems*, (22:1): 45–48.

Checkland, P., (1991). From framework through experience to learning: the essential nature of action research. In: *Information Systems Research: Contemporary Approaches and Emergent Traditions*, H.E. Nissen, H.K. Klein, R.A. Hirschheim (eds). North-Holland, Amsterdam, pp 397–403.

Clement, A., and Van den Besselaar, P., (1993). A retrospective look at PD projects. *Communications of the ACM*, (36:6): 29–37.

DiSalvo, C., Clement, A., and Pipek, V., (2012). Participatory design for, with, and by communities. In: *Routledge International Handbook of Participatory Design*, J. Simonsen, T. Robertson (eds.), Routledge, New York, pp. 182–209.

Edwards, P. N., Jackson, S. J., Bowker, G. C., and Knobel, C., (2007). Understanding infrastructure: Dynamics, tensions, and design. In: *Workshop on History & Theory of Infrastructure: Lessons for New Scientific Cyberinfrastructur*, National Science Foundation, Alexandria, Virginia, USA.

Ehn, P., (1988). *Work-oriented design of computer artifacts*, Lawrence Erlbaum Associates, Hillsdale, N.J.

Ehn, P., (2008). Participation in design things. In: *PDC2008: Proceedings of the Tenth Anniversary Conference on Participatory Design,* J. Simonsen, T. Robertson and D. Hakken (eds.), ACM Press, New York, pp. 92–101.

Finck, M., Gumm, D. C., and Pape, B., (2004). Using groupware for mediated feedback. In: *PDC-04 Proceedings of the Participatory Design Conference Vol 2*, A. Bond, A. Clement, F. Cindio, D. Schuler, P. Besselaar (eds.), CPSR, Toronto, pp. 45–48.

Gawer, A., (2009). Platform dynamics and strategies: from products to services. In: *Platforms, Markets and Innovation*, A. Gawer (eds.), Edward Elgar Publishing Inc, Northampton, Massachusetts, USA, pp. 45-76.

Ghazawneh, A., and Henfridsson, O., (2013). Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal*, (23:2): 173–192.

Gumm, D.C. (2006). Distributed participatory design: An inherent paradoxon. In: *Proceedings of 29th Information Systems Research Seminars in Scandinavia*, Y. Dittrich, D.L. Strand, J. Nørbjerg, O.W. Bertelsen, W.G. Bleek (eds.), AIS, København.

Hanseth, O., and Lyytinen, K., (2004). Theorizing about the design of Information Infrastructures: design kernel theories and principles. *Sprouts: Working Papers on Information Environments, Systems and Organizations*, (4:4): 207–241.

Hanseth, O., Monteiro, E., and Hatling, M., (1996). Developing Information Infrastructure: The Tension Between Standardization and Flexibility. *Science, Technology and Human Values*, (21:4): 407–426.

Hansson, C., Dittrich, Y., and Randall, D., (2006). How to include users in the development of off-the-shelf software: a case for complementing participatory design with agile development. In: *HICSS '06 Proceedings of the 39th Annual Hawaii International Conference on System Sciences - Volume 08*, IEEE, pp. 175c-175c.

Hess, J., Offenberg, S., and Pipek, V., (2008). Community driven development as participation?: involving user communities in a software design process. In: *Proceedings of the Tenth Anniversary Conference on Participatory Design 2008*, Indiana University, pp. 31–40.

Karasti, H., (2010). Taking PD to Multiple Contexts: A reply to Kyng. *Scandinavian Journal of Information Systems*, (22:1): 85–92.

Karasti, H., (2014). Infrastructuring in participatory design. In: *Proceedings of the 13th Participatory Design Conference: Research Papers-Volume 1*, O.S. Iversen, H. Winschiers-Theophilus, V. D'Andrea, C. Bossen, M. Teli, K. Bødker (eds.), ACM Press, New York, pp. 141–150.

Karasti, H., and Syrjänen, A.-L., (2004). Artful infrastructuring in two cases of community PD. In: *Proceedings of the eighth conference on Participatory design: Artful integration: interweaving media, materials and practices-Volume 1,* A. Clement, F. Cindio, A.-M. Oostveen, D. Schuler, P. Besselaar (eds.), ACM Press, New York, pp. 20–30.

Kensing, F., (1987). Generation of visions in systems development: a supplement to the tool box. In: *The IFIP TC 9/WG 9.1 Working Conference on system design for human development and productivity: participation and beyond on System design for human development and productivity: participation and beyond,* North-Holland Publishing Co, Amsterdam, pp. 285–301.

Kensing, F., and Blomberg, J., (1998). Participatory design: Issues and concerns. *Computer Supported Cooperative Work (CSCW)*, (7:3–4): 167–185.

Kossi, E. K., Sæbø, J. I., Braa, J., Jalloh, M. M., and Manya, A., (2012). Developing decentralised health information systems in developing countries – cases from Sierra Leone and Kenya. *The Journal of Community Informatics*, (9: 2).

Kyng, M., (2010). Bridging the Gap Between Politics and Techniques: On the next practices of participatory design. *Scandinavian Journal of Information Systems*, (22:1): 49–68.

Langley, A., (1999). Strategies for theorizing from process data. *Academy of Management Review*, (24:4): 691–710.

MacCormack, A., Rusnak, J., and Baldwin, C. Y., (2006). Exploring the structure of complex software designs: An empirical study of open source and proprietary code. *Management Science*, (52:7): 1015–1030.

Markus, M. L., (2007). The governance of free/open source software projects: monolithic, multidimensional, or configurational? *Journal of Management & Governance*, (11:2): 151–163.

Monteiro, E., Pollock, N., Hanseth, O., and Williams, R., (2012). From artefacts to infrastructures. *Computer Supported Cooperative Work (CSCW)*, (22:4-6): 575–607.

Muller, M. J., and Kuhn, S., (1993). Participatory design. *Communications of the ACM*, (36:6): 24–28.

Neumann, L. J., and Star, S. L., (1996). Making infrastructure: The dream of a common language. In *PDC'96 Proceedings of the Participatory Design Conference*, J. Blomberg, F. Kensing, and E.A. DykstraErickson (eds.). CPSR, Cambridge, MA USA, pp. 231–240.

Obendorf, H., Janneck, M., and Finck, M., (2009). Inter-contextual distributed participatory design. *Scandinavian Journal of Information Systems*, (21:1): 51-76.

Olleros, X., (2008). The lean core in digital platforms. *Technovation*, (28:5): 266–276.

Oostveen, A.-M., and Van den Besselaar, P., (2004). From small scale to large scale user participation: a case study of participatory design in e-government systems. In: *Proceedings of the eighth conference on Participatory design: Artful integration: interweaving media, materials and practices-Volume 1*, A. Clement, F. Cindio, A.-M. Oostveen, D. Schuler, P. Besselaar (eds.), ACM Press, New York, pp. 173–182.

Pilemalm, S., and Timpka, T., (2008). Third generation participatory design in health informatics—Making user participation applicable to large-scale information system projects. *Journal of Biomedical Informatics*, (41:2): 327–339.

Plantin, J.-C., Lagoze, C., Edwards, P. N., and Sandvig, C., (2016). Infrastructure studies meet platform studies in the age of Google and Facebook. *New Media & Society*.

Pollock, N., Williams, R., and D'Adderio, L., (2007). Global Software and Its Provenance: Generification Work in the Production of Organizational Software Packages. *Social Studies of Science,* (37:2): 254–80.

Sahay, S., Sæbø, J., and Braa, J., (2013). Scaling of HIS in a global context: Same, same, but different. *Information and Organization*, (23:4): 294–323.

Sandberg, Å. (ed.), (1979). *Computers dividing man and work*. Swedish Center for Working Life, Demos Project Report no 13, Utbildningsproduktion, Malmö.

Sanner, T. A., Roland, L. K., and Braa, K., (2012). From pilot to scale: Towards an mHealth typology for low-resource contexts. *Health Policy and Technology*, (1:3): 155–164.

Shapiro, D., (2005). Participatory design: the will to succeed. In: *Proceedings of the 4th decennial conference on Critical computing: between sense and sensibility,* O. W. Bertelsen, N. O. Bouvin, P. G. Krogh, M. Kyng (eds.), ACM Press, New York, pp. 29–38.

Shapiro, D., (2010). A Modernised Participatory Design? A reply to Kyng. *Scandinavian Journal of Information Systems*, (22:1):69–76.

Shaw, V., and Braa, J., (2014). Approaches to participatory design in Africa in the age of cloud computing. In: *Proceedings of the 13th Participatory Design Conference - Volume 2*, O.S. Iversen, H. Winschiers-Theophilus, V. D'Andrea, C. Bossen, M. Teli, K. Bødker (eds.), ACM Press, New York, pp. 45–48.

Titlestad, O. H., Staring, K., and Braa, J., (2009). Distributed development to enable user participation: Multilevel design in the HISP network. *Scandinavian Journal of Information Systems*, (21:1): 27-50.

Tiwana, A., (2013). *Platform ecosystems: aligning architecture, governance, and strategy,* Morgan Kaufmann, Waltham, MA.

Van Schewick, B., (2012). *Internet architecture and innovation*. MIT Press, Cambridge, MA.

Walsham, G., (1995). The emergence of interpretivism in IS research. *Information Systems Research*, (6:4): 376–394.

Walsham, G., (2006). Doing interpretive research. *European Journal of Information Systems*, (15:3): 320–330.

Wulf, V., Pipek, V., and Won, M., (2008). Component-based tailorability: Enabling highly flexible software applications. *International Journal of Human-Computer Studies*, (66:1): 1–22.

Yoo, Y., Henfridsson, O., and Lyytinen, K., (2010). Research commentary-The new organizing logic of digital innovation: An agenda for information systems research. *Information Systems Research*, (21:4): 724–735.

III

IV

# Flexibility in EHR ecosystems: five integration strategies and their trade-offs

Authors: Lars Kristian Roland, Terje A. Sanner, Margunn Aanestad, Department of Informatics, University of Oslo. Emails: lars@roland.bz; terjeasa@ifi.uio.no; margunn@ifi.uio.no

**Abstract**

*In this paper, we address various strategies to integrate Electronic Health Record systems with other software features and information sources. We identify a critical tension between a) the end users' desire for a seamless EHR workflow, and b) the system architects' desire for a loose and modular integration, which favors change and evolution of various applications and components over time. Through a qualitative study from Norway, we examined the integration strategies of an EHR system with i) a specialized patient record solution for childbirth, ii) an ePrescription system, and iii) an integrated Summary Care Record system. We complement these empirical studies with related research concerning one additional EHR integration strategy identified in the health informatics literature. We illustrate how four of the strategies compromise either end-users' need for seamless workflow or system architects' desire for a changeable and loosely integrated portfolio of information systems. However, one particular strategy, albeit experimental, shows promise in resolving the identified tension to a more substantial degree than the other four.*

Keywords: eHealth architecture, EHR, integration, seamless workflows, use-flexibility, change-flexibility.

## 1 THE PROBLEM OF INTEGRATING APPLICATIONS INTO ELECTRONIC HEALTH REGISTERS (EHR)

Healthcare workers are dependent on the availability of comprehensive information in their work. The Electronic Health Record system (EHR) is a core application in hospital departments, specialists' and general practitioner offices as well as in nursing homes. Despite its centrality, EHR design still leaves much to be desired in many settings. Friedberg et al. (2013) claim, based on a US study that the IT industry has failed to provide appropriate IT tools for doctors and that the introduction of systems such as EHRs have worsened their professional satisfaction, rather than improved it. Although the study indicates that doctors have an overall positive attitude to EHR introduction, poor user interfaces with slow and overly templated data entry, lack of cross-institutional information access, yet increasing information overload are pointed out as major issues with current solutions (ibid, Payne et al. 2015). In this paper we address one aspect of this complex problem; the need for the EHR workspace to be integrated with other information sources to support clinical practice.

If a health worker's workflow associated with treating a patient involves interacting with multiple applications besides the EHR, the user who encounters lack of integration between such applications may resort to workarounds (Damsleth 2013). A common workaround would, for example, be the use of Windows clipboard to copy data between applications. The term seamlessness denotes the perceived support for workflow across applications and varies inversely with the number of workarounds required to perform the work tasks. Across hospitals, some use EHR systems that are monolithic ("suite" applications) and others use composite or "best of breed" systems (Koppel and Lehman 2015). In either case, there is a need for some degree of interoperability with additional systems. In the Norwegian health sector, there are thousands of applications at multiple sites that require various levels of integration with the EHR (Heimly et al. 2011, Bygstad and Hanseth 2016). These systems include both local applications (e.g., specialized systems for medical charts, imaging, equipment), or external, inter-organizational systems and national solutions that span the whole sector (such as national solutions for ePrescription or nationally shared EHRs). The approach for integrating

Please cite as Roland, L.K., Sanner, T.A. & Aanestad, M..: Flexibility in EHR ecosystems: five integration strategies and their trade-offs (2017). Paper presented at NOKOBIT 2017, Oslo, 27-29 Nov. NOKOBIT, vol. 25, no. 1, Bibsys Open Journal Systems, ISSN 1894-7719.

these systems has not been systematically discussed in the healthcare IS or medical informatics literature, and we, therefore, offer a theoretically informed empirical investigation into five different EHR integrations strategies.

A starting point for our conceptualization of EHR integration is the inherent tension involved with providing the health worker with a high level of workflow support (understood in this context as seamless cross-application integration). This seamlessness requires tight integration of relevant information sources in an individual's workspace (Humphreys 2000, Ellingsen and Monteiro 2006, Weng et al. 2012, Krist et al. 2014). However, from a strategic and architectural perspective, there is a need to provide the solution cost-effectively to a broad population of end-users, who will have different information needs and use a multitude of different information systems. In this article, we use the word *scaling* to describe this need for large-scale solution adoption, expansion, and growth. The tight coupling of two systems that exhibit a high level of workflow seamlessness may negatively impact the efficiency with which IT professionals can meet future needs and provide improved functions to the user population (Ellingsen and Monteiro 2006). Also, tight integration makes future changes more problematic than if the coupling between systems is looser. Our interest here is in creating a better understanding of the issues involved when dealing with this dilemma in the context of the EHR workspace. We contribute to knowledge about how local EHR workflow support can be balanced against cost-efficient governance of the health information infrastructure as a whole. Health professionals need useful systems, and at the same time, managers and architects need their portfolio of information systems to be governable and changeable over time, which ultimately also is in the best interest of end-user clinicians.

Methodologically we proceeded as follows. We first examine relevant literature to identify integration strategies (or options/styles) that are feasible for integrating the EHR with third-party systems and applications. We then conduct an expert survey to validate our initial analysis, followed by a case study of three Norwegian integration projects. With each of the five integration strategies we identified, we examined their potential to support seamless integration and flexibility, and we compare the experiences regarding the trade-offs between seamless local integration and large-scale standardized infrastructures. We find that four of the strategies prioritize either seamless integration or change-flexibility and compromise the other dimension. Interestingly, one integration strategy appears to resolve the tensions between the two equally desirable goals but remains experimental in practice. Our conclusion supports the view that EHR integration can help reduce the costs and frustrations of IT in clinical practice and help promote the goals of increased quality, accessibility, and efficiency of the provision of health services (Eysenbach 2001), while keeping in mind the professional satisfaction of clinicians (Friedberg et al. 2013, Payne et al. 2015).

# 2 FIVE INTEGRATION STRATEGIES IN THE EHR WORKSPACE

This article concerns integration related to a single user dividing a task between multiple systems, rather than issues arising when multiple different users access one or more systems as shown in the figure. The term workflow in this paper therefore addresses a single users' perspective of a sequence of tasks, rather than multiple health workers' tasks in an integrated patient pathway.

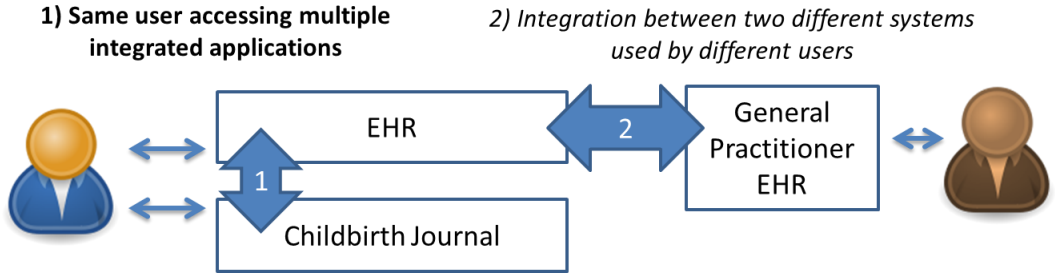


*Figure 1 shows two of several possible dimensions in which integration can be considered. Integration can concern communication between two applications that are used by the same health worker (1), for example, an EHR and a connected application. Integration can also concern communication between two systems that are*

*used by different health workers for treating a single patient (2), such as the communication between two different EHRs. This paper is limited to scenario #1, and we expect the more complex scenario #2 to be shaped by additional dimensions.*

The extant literature points to several ways in which third-party functionality and data sources can be integrated into an EHR workspace. The following five strategies, named A-E, for EHR-application integration vary for example with regards to the degree of separation and independence, mode of data exchange and sharing of patient context, user log-in and usage pattern (semantic and concept equality), and EHR vendor control and ownership. Other integration strategies may exist but have not been covered in this study and are to our knowledge not widely adopted in practice.

A. A separate application that is distributed and managed independently from the EHR. The application may use EHR interfaces for sharing login, the patient context and data between the EHR and the application. The user works with (i.e., enters and reads information from) each application separately. The EHR vendor is not involved in distributing or managing the external application (Berger and Baba 2009, Skorve and Aanestad 2010).
B. The application is a web application embedded into the EHR. The application is downloaded from a separate server, allowing external parties to update the applications without interaction with the EHR vendor. The EHR may provide interfaces for identity, context, and data sharing for tighter coupling between the application and EHR (Mandel et al. 2016, Mandl et al. 2015, Mandl and Kohane 2009, Mandl and Kohane 2012).
C. The third-party application (e.g., a module) is integrated into the EHR by the EHR vendor. The modules are bundled with the EHR and do not work on their own (Mandl and Kohane 2009, Bernstein et al. 2005, Koppel and Lehman 2015, Pesaljevic 2016, Hanseth and Bygstad 2017).
D. The EHR vendor implements the whole user interface but uses standardized interfaces towards third-party backend services, which can be reused by multiple EHR systems. These central components can have shared storage and/or services (Noumeir 2011, Koppel and Lehman 2015, Pesaljevic 2016).
E. The EHR vendor implements the whole user interface but includes configurable intercept points (sometimes called "hooks") to external services to fetch and display external information as part of the workflow (Warner et al. 2016).

# 3 ANALYTICAL PERSPECTIVE: SEAMLESSNESS AND FLEXIBILITY IN HEALTHCARE INFORMATION INFRASTRUCTURES

Studies of information infrastructures constitute the exploration of large, complex and evolving information system architectures, where immediate and local needs are continuously contested and negotiated against large-scale efforts to harmonize and standardize across local settings (Rolland and Monteiro 2002, Monteiro et al. 2003). This tension has been expressed as inherent with generic solutions to particular needs (Ellingsen and Monteiro 2003), or as a tension between standardization and flexibility (Hanseth et al. 1996, Ellingsen and Monteiro 2006). In particular, Ellingsen and Monteiro (2006, p 443) argue, in the context of health information system integration, that "[e]nforcing order in the form of standards across multiple local settings, seemingly a prerequisite for tight integration, simultaneously produces disorder or additional work in other locations for other users." Here, we draw on an information infrastructure perspective to explore how the five EHR integration strategies outlined above balance tensions between "local" end-user needs for seamless workflows and "global" concerns for long-term configurability and change of national and inter-organizational eHealth architectures, of which EHR systems are essential components. Specifically, comprehensive and integrated EHRs have been proposed as a means for integrating heterogeneous systems in healthcare information infrastructures, but too comprehensive standardization and consolidation efforts may also lead to stagnation, curb innovation, and negatively affect evolving clinical practice (Ellingsen and Monteiro 2003, Ellingsen and Monteiro 2006).

With traditional information systems development, it has been noted that system flexibility can be realized both at the time of implementation/change and at the time of use (Orlikowski 1992). An information infrastructure, with a high level of interdependence between actors (e.g., client organizations, vendors, IT-departments and governance bodies) and system components, typically has low flexibility for further change (Hanseth et al. 1996, Hanseth and Lyytinen 2004, Roland et al. 2017, Sanner et al 2012), due to high technical complexity, diverse interests among heterogeneous stakeholders and challenges with coordinated actions across systems and actors. Further, the flexibility for change of large and complex systems has been linked to modularity, in the form of architectural layers and "lean" modules that are loosely coupled (Braa et al. 2007, Edwards et al. 2007, Hanseth et al. 1996). Modularity is a design principle that emphasizes the development of small and reusable components with clearly defined interfaces between them, rather than developing one comprehensive system or product where functional interdependencies are hard to discern (Baldwin and Clark 2006, Baldwin and Woodard 2008, MacCormack et al. 2006).

Gebaur and Schober (2006) build on Hanseth et al. (1996) to discern two types of information system flexibility: (i) flexibility in the pattern of use (short: use-flexibility) and (ii) flexibility for further changes (change-flexibility). Gebauer and Schober define use-flexibility as the range of process requirements supported without major changes, where a major change constitutes "adjustments and modifications that require a fresh system setup, including re-installation and re-testing" (ibid, page 128). Use-flexibility thus refers to flexibility "out of the box" for use across a range of different, even unanticipated purposes and tasks. Design and use-flexibility have been described as relational in the context of information infrastructures because a high level of use-flexibility reduces the need for change-flexibility, and vice versa (Hanseth and Lyytinen 2004, Hanseth et al. 1996). In relation to the current study, we further specify that use-flexibility is closely tied to the choices and activities of end users or IT support functions close to and responsive to end users' needs, while change-flexibility is more closely associated with design decisions made by system architects, IT governance bodies and external vendors of "off the shelf" software packages such as EHR solutions. We further note that different actors in an information infrastructure can tap into a varying degree of use-flexibility and change-flexibility, depending on their roles, skills, ownership, and interdependencies.

Change-flexibility in the context of healthcare information infrastructures is shaped by both social and technical arrangements. These factors include IT personnel's skills and mandates, contractual arrangements and regulations, the choice of standards and interfaces between integrated data sources and applications in the overall architecture, and modularity, for instance, provided by re-usable software modules and vendor-independent database connectivity. In essence, change-flexibility constitutes different actors' ability to provide new functionalities, recombine and reorganize access to various data sources and to allow for modifications of the user interface for instance to support more seamless EHR workflows. Hanseth et al. (1996) emphasize the importance of exploring the relative degrees of flexibility with different solution types to maximize flexibility for future change. In this study, we examine the flexibility to change associated with five EHR integration strategies and juxtapose it with the ability of the integrated EHR solution to support clinical tasks through seamless workflows. Hence, we use the term seamlessness as a quality of EHR workspace integration, indicated by the accomplishment of a task that traditionally spans multiple applications and data sources.

When users attempt to accomplish tasks and encounter lack of seamlessness (i.e., lack of integration between data sources and applications), they fill gaps by introducing their own workarounds (Damsleth 2013). A workaround can, for example, be the use of Windows clipboard to copy patient data between applications or manually edit and manipulate text files generated by one system before they are uploaded to another system for sharing with other professionals. Lack of design-time integration is compensated by use-flexibility workarounds (ibid). As such, the necessity for workarounds can be seen as an indication of how well applications and data sources are integrated. We understand the total flexibility of an integrated EHR solution as constitutive of use-flexibility and change-flexibility, in addition to manual workarounds that may be both intended and unintended from the system architects' perspective to meet the limitations with EHR integration (Gebaur and Schober 2006). Beyond seamless workflow to support tasks, we note some prerequisites to any integrated EHR solution including security, certification, data protection and following laws and standards that are

mandated within the region/country. These requirements will most likely affect the choice of EHR integration strategy and, as we show with the five strategies identified in this paper, will, in turn, shape the balance between change-flexibility and the seamlessness of integrated EHR workflows.

# 4 RESEARCH APPROACH AND METHODS

## 4.1 Research context

Norway had an early adoption of EHRs at all levels of the health service. The first rollout started in 1985, and the penetration had reached 90% by 2000 for general practitioners and 2005 for hospitals (EPJ Monitor 2008, Heimly et al. 2011). In 2008 the last of the public hospitals implemented an EHR system. The early and distributed introduction of EHRs led to a fragmented EHR landscape. One implication of this fragmentation is that making common EHR changes across the whole healthcare sector is difficult (EIEJ 2015). There are multiple EHR vendors for the hospitals (>2), municipalities (>3) and general practitioners (>3). While there is a national program in place to consolidate EHR systems into fewer systems (EIEJ 2015), the current situation is that rollout of any national services requires a tremendous amount of integration work with each of the EHR vendors.

Two of the national solutions targeting health workers in Norway are ePrescription and Summary Care Record (SCR). Both these applications are integrated into the EHRs using Norway-specific mechanisms that include several of the integration types described in this paper. The hospitals in Norway currently integrate a large number of local solutions covering vertical usage needs into their EHR system. One of these applications is a solution that offers a structured user interface documenting data around pregnancy and childbirth.

## 4.2 Qualitative case study

This research is based on a qualitative study of three Norwegian cases that highlight different aspects of the dilemmas associated with the integration of EHRs and other applications. The three cases represent five different integration strategies and their outcomes. The first case (Childbirth record) represents type A (a separate application operated in parallel with the EHR), the second case (ePrescription) represent types C and D, and the third case (Summary Care Record/SCR) represents type B, but is moving towards type D. We could not identify any Norwegian implementations of category E and had to rely on the limited experiences reported in the literature. The SCR and ePrescription solution are national solutions that have been integrated into a large number of EHR systems in Norway and are by many considered templates for future similar projects. Learning more about how these templates perform in sustained use is therefore useful for future EHR integration projects. The study was designed to collect information about the experiences with these different integration strategies, explicitly concerning the trade-off related to workflow seamlessness and change-flexibility.

## 4.3 Data collection

The primary author is a part-time Ph.D. student and works at the Norwegian Directorate of eHealth as a solution architect. In this role, he has engaged in projects that include architecting application integration with EHRs. All interviews and other data collection were conducted by the primary author, while the two other authors have participated in framing the study, analyzing data and have given feedback on the study and manuscript. The data collection covered several data sources: 1) participation by the primary author in meetings and user forums regarding EHR and application integration over a period of 2 years, 2) informal discussions with project members and colleagues, 3) review of documents from the Norwegian Directorates of eHealth and Health, including user survey reports, 4) review of previously published case studies concerning the Norwegian Summary Care Record and ePrescription (Hanseth and Bygstad 2017, Pesaljevic 2016, Larsen and Mydske 2013), 5) formal, semi-structured interviews with 20 managers, architects, product managers, developers and end users. The formal, semi-structured interviews followed an interview guide that started with a general discussion about application integration into EHRs, followed by the interviewees being introduced to the five types of integration as described in Figure 1, which prompted a discussion about the five integration types. The description and comparative evaluation of the types evolved during the

study, based on feedback from interviews. As examples, strategy A, which exemplifies separate applications, was initially portrayed as an undesirable alternative, but based on interview feedback this strategy was considered to be both important and sometimes necessary. Many interviews led to a discussion on how real-life implementations used a combination of the different integration strategies over time. The five integration strategies were initially founded in extant literature, backed by empirical data from data collection points 1-4 above, but were then refined and validated through the semi-structured interviews.

### 4.4 Data analysis

Interviews were recorded and transcribed. In the analysis phase, keywords, concepts, and quotes illustrating characteristics of the types and dynamics between them were highlighted. Contradictions between assumptions in the categorization and the interview observations led to iterative changes of the integration strategy typology. Data were further analyzed using data displays (Miles and Huberman 1994) to exemplify contrasts between the different integration types, some of which were refined and included in the discussion part of this paper. For example, the weighting of the different integration strategies regarding tradeoffs related to seamlessness and flexibility were a result of data display analysis and repeated iterations of data display revisions after discussions between the three authors and with informants during interviews.

The three empirical cases served slightly different purposes in the analysis process. The childbirth journal case served to explore the aspect of seamlessness (in cross-application integration). The four seamlessness characteristics derived from the first case were then used to analyze the second and third case in more detail. For these two cases, we collected more data and focused on the implications of the chosen integration solution for both the initial implementation and for further change.

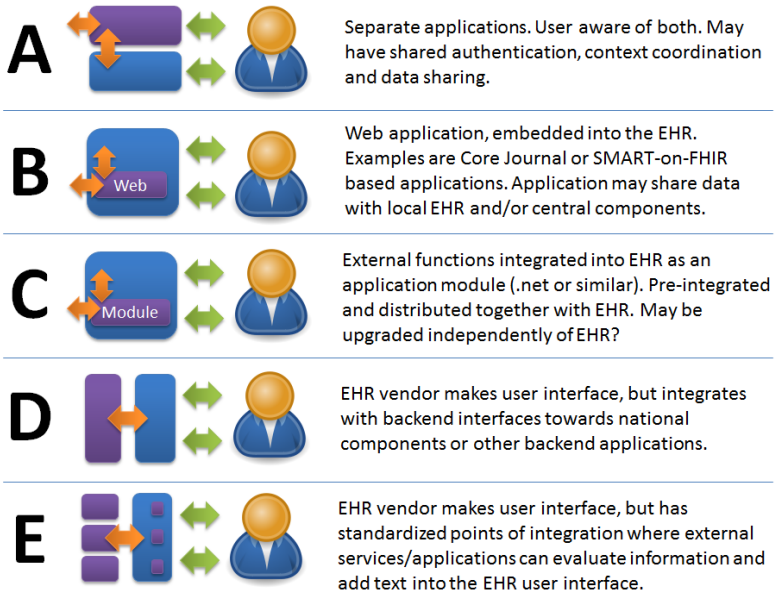


*Figure 2 shows the final version of the figure that was used during interviews to get feedback on the categorization of integration types. The figure was revised from interview to interview.*

## 5 FINDINGS

### 5.1 Childbirth record solution (Type A: separate applications, minimal linkages)

Clinicians use a structured user interface in the childbirth record application to document observations and treatment, some of which is exported to the core EHR as documents. The clinicians also use the EHR directly to view other relevant information about the patient, including data entered by other departments and historical data. The childbirth record and EHR system are accessed as separate applications, and the user must log in separately to each of them. The systems are linked so that

selecting a patient in one application also synchronizes the patient context with the other. Patient safety is the primary reason for this context synchronization: "In a heterogeneous environment, maintaining context is the main problem … there are many applications in a hospital" (source). Patient safety is not the only reason: "This is primarily about patient security, but it is also a matter of saving time for clinicians" (architect). The two systems are also linked on the data-level by exporting documents from the childbirth record to the EHR (the level of integration and type of integration interfaces varies between hospitals and software versions). The EHR and childbirth record exhibit different conceptual and semantic approaches, as the childbirth record application requires more structured data and the EHR uses more unstructured free-text documents. This difference between structure and free-text was perceived to be a change in documentation "culture," and was "a barrier for the doctors who were more used to the [EHR]. Some doctors felt that the narrative and some subtle nuances were lost in the structured data." (Architect).

From this case we identified four aspects that relate to the integration of workflows between the applications: 1) Is there a shared user identity and login? If yes, this lowers the hurdle of having to log in multiple times. 2) Do the applications share patient and treatment context? If yes, this prevents having to find the patient and consultation in each application. 3) Do the applications share relevant data? If yes, this prevents double-entry and inconsistency between data. 4) Is there visual, semantic and concept equality among the systems? If yes, this prevents having to switch documentation and reading style/cognitive style. In the following we thus conceptualize systems integration as related to 1) user's identity; 2) patient/treatment context; 3) data exchange; and 4) visual, semantic and concept equality.

## 5.2 National ePrescription solution (Types C and D)

Two different integration patterns emerged in the evolution of ePrescription. Overall, the Norwegian national solution for ePrescription uses a message-oriented, client-server architecture, with a national server and distributed clients at pharmacies, general practitioners, hospitals, and municipalities.

The ePrescription project started in 2003, with the first successful pilot beginning in 2009. The initial plans were to reuse the prescription modules that existed in the general practitioners' EHRs and integrate these with a national prescription database (integration type D). The integration strategy required the vendors to undertake development, and this route turned out to be a lengthy and challenging process. The central project decided to develop and freely offer an alternative, external prescription module, to support EHRs that lacked ePrescription support. This module required lower investment from the EHR vendors and resolved the problem of the delayed roll-out. The external prescription module comes with its own ePrescription user interface, logic and local storage, pre-integrated towards the central component. However, the module is not usable on its own, so we classify the external module as an instance of integration type C. Thus, the resulting architecture allowed for two different ways to integrate the EHR with the national ePrescription infrastructure as is shown in figure 2. These two approaches use the same standardized messages towards the central prescription server.

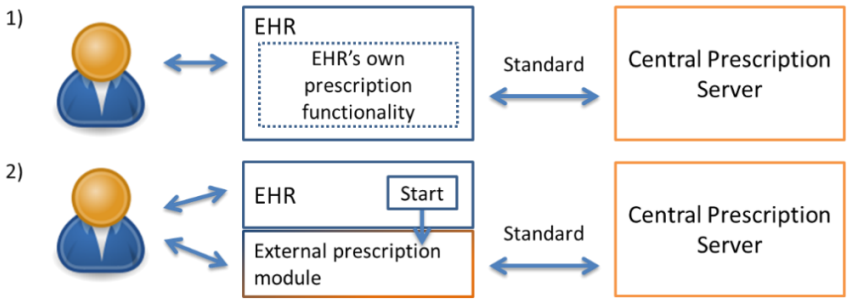


*Figure 3 shows the two ePrescription implementation alternatives, with or without the external prescription module.*

Integrating the external prescription module into an existing EHR is considerably cheaper than developing a vendor-specific prescription module from scratch. However, when EHR vendors develop

own modules, these are typically more seamless with the rest of the EHR workspace and have a similar look-and-feel.

In contrast, the visual appearance of the external prescription module looks different from the rest of the EHR, though there are plans to allow vendors to adapt the external module's appearance. If vendors are allowed to adapt the external module's appearance to fit their EHR, this integration model also approaches the type D category, because the EHR vendor will have stronger control over the development.

Whether the user interface was consistent with the rest of the EHR was found to be significant for the user experience. During the rollout of ePrescription, it became evident that many clinicians are sensitive to the look-and-feel and semantics of their workspace. Even minor differences in views seemed to make a difference, as stated by a project member: "The clou is work process. You must ensure a consistent work process for the doctor. During quality surveys of ePrescription, we have seen that just filling in a simple checkbox can be a huge barrier. To enable consistent work processes and to ensure that this is an efficient tool that supports the user's needs, it is required to be an integrated solution. If not, it will not be used."

Let us consider the change-flexibility of the ePrescription solution. Introducing new functions into the ePrescription infrastructure has proven difficult, given a large number of independent actors and a distributed architecture that requires all involved actors to be part of implementing new features. No actors have end-to-end control. Although the implementation of ePrescription is widely considered a success, there are therefore challenges with introducing changes and new functions into the existing infrastructure. Even adding a single attribute to the standard messages requires work in all EHRs. However, there is a difference between the external and internal implementations related to the ability to change, because the external module can be installed and updated independently of the EHR vendor's roadmap.

The following example illustrates this difference: A function that has been planned since the beginning of the project is the support for multidose prescriptions, a necessary feature for patients who use multiple drugs that should be prescribed and dispensed together. The current method for handling multidose is a paper card that is faxed between the general practitioner and pharmacies. Introducing an electronic version of multidose is therefore considered a priority in the Norwegian health care sector. The external prescription module has supported multidose functionality since 2014, but EHR vendors have not yet implemented this feature into their own prescription modules. Hence, EHRs using the external module can already support multidose.

## 5.3 The Norwegian Summary Care Record (Type B, migrating towards type D)

The SCR is a national solution that collects information about patients (including recently dispensed medication, visits to health facilities, allergies, etc.) in a national database that health workers can access when necessary. The data available is not a complete shared record, and availability of some data depends on clinicians entering data manually. To access the SCR, the health worker starts from the standard patient view in their local EHR. If the integrated SCR-button in the patient view is red, this indicates that there is critical information stored in the national database. Clicking the button brings up the national web application in an embedded browser. The patient context passes from the EHR to the central web application, but no data integration is available yet. The clinician must enter data manually or use Windows clipboard to transfer data between the EHR and the web application, i.e., requiring the use of workarounds. The lack of seamless data integration with other views of the EHR has been pointed out by users as a significant barrier to adoption. The SCR provider is currently implementing standardized interfaces that can allow structured data to be integrated seamlessly into the EHR by the EHR vendor, as an alternative or complement to the web application (e.g., prescriptions and allergies). Both the web application and information exchange through tightly integrated standardized interfaces are planned to operate in parallel, to fit different usage and solution scaling needs.

The choice of an embedded web application was deliberate, prioritizing the need for national scaling of the solution. "The strategic choice was considered carefully and still feels correct. With the limits

the project had at the time, within the first year, it was impossible to integrate tightly with all EHRs for all the necessary content. There were many EHR vendors, and the pilot had to cover the whole chain. This complexity was the reason for the decision to use web integration" (project member). This strategy required an initial investment in aligning actors, setting standards and implementing the technology in the EHRs. However, the investment was rewarded by the simpler deployment of new applications since the web application can be updated centrally without making changes to the local EHRs. "With the web application, updates become super flexible. We can fill the portal with lots of new content and have it spread overnight to the whole health service. This is one of the greatest advantages. It works in practice. We deliver four times every year, and the new functionality is available when we have finished deploying." (Project member). However, the same project member acknowledges that the tight integration could have come earlier: "to reduce the noise we should perhaps have implemented the service-oriented interfaces and tight integration earlier, to avoid double registration of data." Establishing the web platform capability in the EHRs has been a painful process. The initial rollout had to deal with delays in EHR version rollout, differences in web browser vendors and versions, infrastructure connectivity, the need to introduce electronic signatures and the resolution of other required infrastructure issues, all contributed to slow down the implementation.

The US initiative called "SMART-on-FHIR" represents a more widely standardized, but similar web-based integration strategy as that used in SCR. "SMART-on-FHIR" is an open framework that enables embedding web applications into EHRs, with standardized mechanisms for authentication, authorization, context and data sharing (Mandel et al. 2016).

### 5.4 Strategy E – Allowing external "intercept points" to engage with the workflow

We could not identify any Norwegian implementations of category E and had to rely on limited experiences reported in the literature, although one informant argued that the red SCR button is a standardized EHR intercept point. An open initiative for embedding intercept points and micro-services into EHRs is the Clinical Decision Support hooks (CDS-hooks), which connects small external services as part of an EHR-supported workflow (Warner et al. 2016). CDS-hooks is currently being included in the HL7-FHIR standard.

## 6 ANALYSIS

### 6.1 Summary of findings

The various cases illustrate different trade-offs between seamlessly integrated workflows across applications (tight integration) and change-flexibility (loose integration).

*Table 1 shows a comparison of the five integration types, along the dimensions of seamlessness and flexibility. The rating is based on interview data and specific feedback from interviewees when asked about the seamlessness and change-flexibility of each integration strategy. These judgments have been assessed and supplemented by the researchers' analysis of the other empirical material (e.g., documents and surveys).*

| Integration strategy | Cases | Seamless workflow | Change-flexibility |
|---|---|---|---|
| A - Separate applications | Childbirth Journal | Low | Medium |
| B - Embedded web | Summary Care Record (SCR) | Medium | High |
| C - Native modules | ePrescription using the external module | Medium | Medium |
| D - All UI implemented by EHR vendor | ePrescription using the EHR system's internal prescription module | High | Low |
| E - Standardized intercept points into workflow | No Norwegian case. CDS hooks is an example, but not yet widely deployed. | High | High |

The score in the column "Seamless workflow" relates to how well four dimensions of seamlessness are supported. These were: 1) shared user identity, to prevent having to log in separately; 2) sharing

patient, encounter and treatment context, to prevent having to find the patient and encounter in each application; 3) sharing of relevant data between applications, to prevent double-entry and inconsistency between data; and 4) semantic and visual concept equality/similarity.

*A - Separate applications: Low seamlessness + Medium change-flexibility*

The category of separate applications rarely achieves full workflow seamlessness according to the four criteria. The childbirth journal case met some criteria since the patient context and data was shared between the applications, but users did have to log in twice. Data integration was limited, and the structural concepts were very different between the two applications. Type A has therefore been rated as low on context seamlessness. However, support for richer context APIs in the EHR would improve context seamlessness. Other instances of this integration type may therefore score differently. It scores higher on change-flexibility, as the separate applications can be updated without mutual adjustment between the vendors. We classify the change-flexibility to medium since this type of local application still requires updating a large number of installations to deploy new features nationally.

*B - Embedded web applications: Medium seamlessness + High change-flexibility*

Embedded web applications have varying degrees of integration with the local EHR workspace. The Norwegian SCR web application is currently integrated at step 1 and 2 of context seamlessness, while sharing of data is achieved through the use of the Windows clipboard. The introduction of full SMART-on-FHIR support or other structured interfaces would allow for better sharing of data between the application and EHR. It is however unlikely that step 4 could be reached with a web application in the short term. Step 4 would require that the two applications appeared as a single application to the end user, sharing user interface concepts and semantics. The change-flexibility of this solution is very high, allowing over-night updates of the core solution and full availability of new features immediately without any local changes. However, this positive flexibility rating may underestimate the organizational aspects such as training required for new and updated functions.

*C - Bundled third party application modules: Medium seamlessness + Medium change-flexibility*

Because the EHR vendor is responsible for integrating the module in type C, this type can provide better context seamlessness than both A and B, though dependent on the module's integration capabilities. The visual appearance and semantic interoperability of applications are critical to clinicians, and the considered ePrescription feature of allowing adaptation of the module's user interface appearance would further improve the level of context seamlessness, possibly addressing level 4. Adapting the user interface per EHR would, however, lower the change-flexibility dimension of the solution. In respect to change-flexibility, type C is classified as medium, somewhere between A/B and D, because implementing new features may require mutual adjustment between stakeholders. The implementation of automated update tools in the ePrescription case improved the change-flexibility of the solution.

*D - User interface entirely developed by EHR vendor: High seamlessness + Low change-flexibility*

When the EHR vendor implements the whole user interface, they are better equipped to provide consistent views that flow well between different tasks, even if some of the functions require backend calls to external components. The level of context seamlessness for this integration type can therefore be high, as the feedback of the ePrescription project shows. However, we saw from the ePrescription project that integration type D scores poorly on change-flexibility. For example, the multidose functionality that was quickly introduced in the external ePrescription module has still not been introduced by EHR vendors using type D, even after several years.

*E - EHR workflow with standardized hooks for apps: High seamlessness + High change-flexibility*

Type E has not been treated in any vignettes, and it remains to be seen how well this type supports context seamlessness and change-flexibility in practice. The vision of those behind initiatives such as CDS Hooks, one implementation of type E, seems to be that it would be a win-win for both dimensions. The ability to plug in external functions into various parts of an otherwise seamless workflow may be a dream worth pursuing, perhaps combining the best parts of type E and B. The

success of this type would be highly dependent on a successful establishment of the EHR as a platform for external services, a vision that not all informants of this study considered realistic.

## 6.2 Change-flexibility of EHR ecosystems

The integration scenarios described in the vignettes above engage and link the stakeholders in the EHR ecosystem differently. These different stakeholder relationships have significant implications for the overall change-flexibility of the ecosystem, understood as the ability to evolve. With evolve, we, for example, mean adding new functionality and recombine different data sources. Choosing an architecture for integration, will among other things shape and impact the innovation dynamics in the EHR ecosystem as a whole. We illustrate two different dependency variants in figure 3 below.

1) For the integration strategies A, B, and C, where the EHR plays a central role as an ecosystem enabler, the project first goes through an ecosystem enablement phase and secondly enters an innovation phase. In the first phase, the standardization, creating central enablers for platform services and implementing platform support in the EHRs are prerequisites for open application development. During this enablement phase of the ecosystem, we see reciprocal interdependence between the involved actors as the core platform functions are forged. Subsequently, we see an innovation phase where external parties can develop applications with limited involvement from the ecosystem enablers. In this phase, new applications using standards and central infrastructure can be developed without changes to the EHR platform enabler. Evidence of this platform creation was seen in the SCR case and ePrescription when using the external prescription module.

2) For integration strategy D, the sequence of events and interaction between players is different. The EHR vendor must be involved for each new application since the full user interface and logic are implemented by the EHR vendor and must be planned as part of the EHR roadmap and deployment rollout. The central standards for the applications must be developed ahead of EHR development of the user interface. After standardization and creation of central components, the actors are reciprocally interdependent and require mutual adjustment during development. For each new application and function to be deployed, one may need to return to the first step of standardization and re-engage with EHR vendors. This strong interdependence was seen in the ePrescription case when adding multidose functionality. The feasibility of this scenario for implementing new functions will depend highly on the number of EHR vendors and the complexity in upgrading on-premise EHR installations. The outcome may be favorable regarding workflow seamlessness through context sharing.

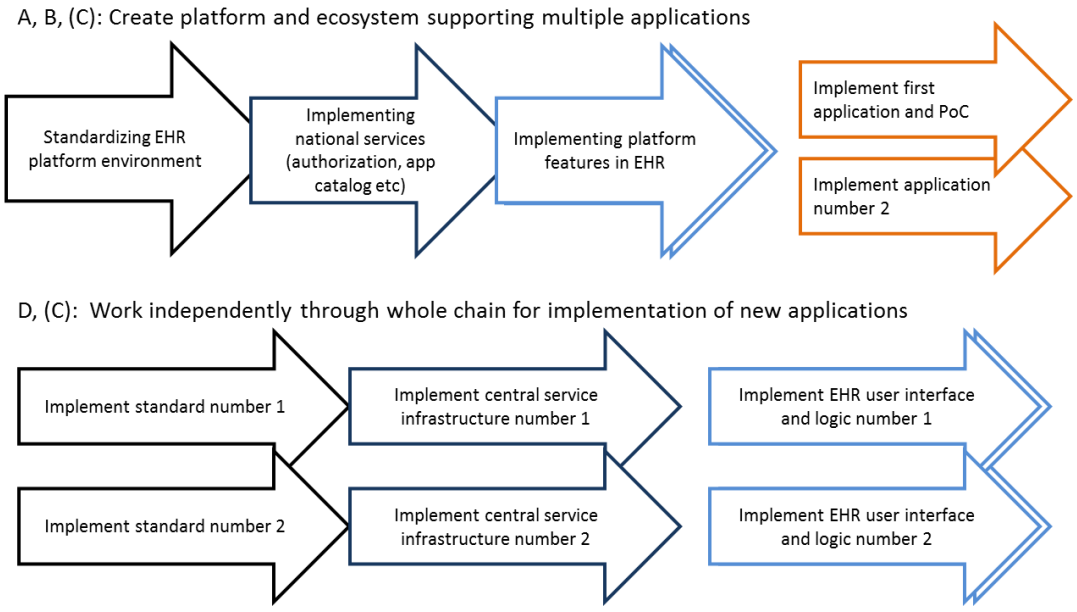


*Figure 4 shows how an initial investment of a platform may enable the development of several parallel applications after the platform has been established.*

### 6.3 Trade-offs in EHR ecosystems

Figure 4 maps out how the various EHR integration strategies balance the trade-off between workflow seamlessness and change-flexibility:

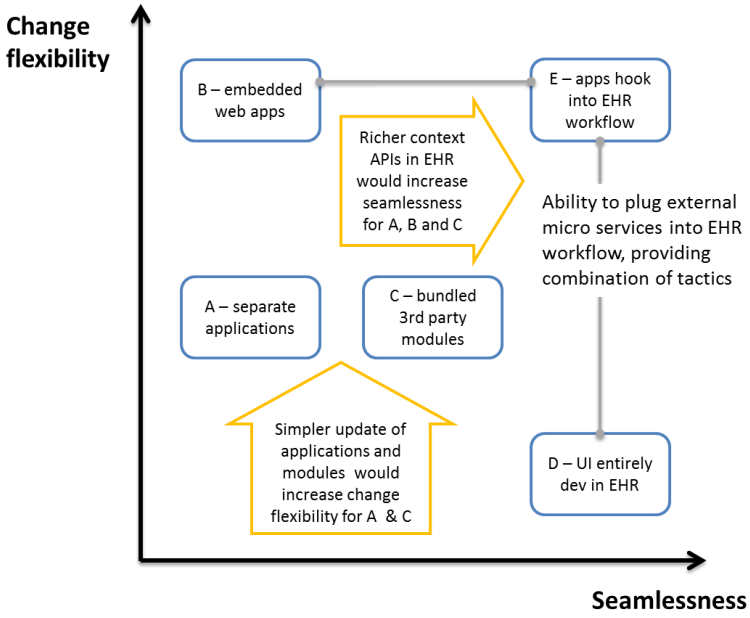


*Figure 5 shows how change and design flexibility contrast with seamlessness.*

The vignettes indicate that EHR integration efforts may target multiple integration strategies within the same project, drawing benefits from different strategies for separate use cases. Two of the vignettes show that one may also decide to leverage different integration strategies over time to fit the scaling requirements of the initiative. In both the SCR and ePrescription projects, types B and C that favored change-flexibility were chosen early on to help scale the solution across the national base. Accessing all users in the country was more important than optimizing seamlessness for specific users. These early strategies were followed by type D that improved workflow seamlessness for clinicians. An emphasis on usefulness and simplicity for clinicians (e.g., through close user involvement) may initially favor integration strategy D since this is experienced as more seamless by the end user and reuses existing EHR functions. However, such initiatives could encounter challenges of scaling and change-flexibility in the EHR ecosystem. In the childbirth record case seamlessness could be prioritized over reaching many users. In this case, the project could benefit from giving priority to measures that improve seamlessness for type A or choose D instead. The real world challenge that faces implementers is, however, a careful balance of multiple types of integration and related concerns, governed by conflicting project goals.

## 7 CONCLUSION

In this paper, we have analyzed the relationship between seamlessness and flexibility when integrating applications with the EHR workspace. Seamlessness was found to include 4 characteristics: *1) shared user identity, to prevent having to log in separately; 2) sharing patient, encounter and treatment context, to prevent having to find the patient and encounter in each application; 3) sharing of relevant data between applications, to prevent double-entry and inconsistency between data; 4) semantic and visual concept equality*. Five different types of integration strategies were found in extant literature, and their characteristics were refined through an empirical case study. We compared the five integration strategies with respect to change-flexibility and seamlessness. Some integration types exhibited higher change-flexibility (A, B, C) while one (D) showed high potential for seamlessness. Future empirical research is required to assess whether the fifth type (E) indeed lives up to the expectations to deliver both change-flexibility and workflow seamlessness.

The empirical cases from Norway showed that IT professionals can and do choose different strategies based on their requirements for reaching a large number of users, or optimize workflow seamlessness of the solution, and that a single project can plan to employ more than one integration strategy throughout the project lifecycle to optimize for example national scaling in one phase and workflow seamlessness in the next.

The more change-flexible types (B, C, and D) require an enablement phase where platform features are developed. These types need open interfaces in the EHR to be able to deploy, as well as a functioning ecosystem where the EHR vendor plays an important role. We can see signs of such platform enablement of EHRs in the healthcare industry (Mandel et al. 2016), but it remains to be seen how successful these initiatives will be. A shift may occur when the EHR vendors start basing their application development on their own open interfaces (Roland et al. 2017).

The study is based on an examination of three Norwegian projects, and a more comprehensive study, including an international perspective, might have yielded other insights. We also wish to clarify that there are additional factors (beyond change-flexibility and workflow seamlessness) that play a role in making decisions on the types of integration. Some other factors include whether the solution was located on-premise or in the cloud, or if the integration efforts were implemented by one of the vendors, the healthcare organization itself or a third party. We believe that our study demonstrates the need to pay careful attention to the balance between workflow seamlessness and change-flexibility to succeed with visions an integrated EHR ecosystem.

# 8 REFERENCES


Baldwin, C. Y., & Clark, K. B. (2006). The Architecture of Participation: Does Code Architecture Mitigate Free Riding in the Open Source Development Model? *Management Science*, *52*(7), 1116–1127. https://doi.org/10.1287/mnsc.1060.0546

Baldwin, C. Y., & Woodard, C. J. (2008). The architecture of platforms: A unified view. SSRN Scholarly Paper, ID 1265155. Rochester, NY: Social Science Research Network. https://papers.ssrn.com/abstract=1265155, accessed October 9, 2017.

Berger, R. G., & Baba, J. (2009). The realities of implementation of Clinical Context Object Workgroup (CCOW) standards for integration of vendor disparate clinical software in a large medical center. *International Journal of Medical Informatics*, *78*(6), 386–390. https://doi.org/10.1016/j.ijmedinf.2008.12.002

Bernstein, K., Bruun-Rasmussen, M., Vingtoft, S., Andersen, S. K., & Nøhr, C. (2005). Modelling and implementing electronic health records in Denmark. *International Journal of Medical Informatics*, *74*(2), 213–220. https://doi.org/10.1016/j.ijmedinf.2004.07.007

Braa, J., Hanseth, O., Heywood, A., Mohammed, W., & Shaw, V. (2007). Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy. *MIS Quarterly*, *31*(2), 381–402.

Bygstad, B., & Hanseth, O. (2016). Governing e-Health Infrastructures: Dealing with Tensions. *ICIS 2016 Proceedings*. Retrieved from http://aisel.aisnet.org/icis2016/ISHealthcare/Presentations/2

Damsleth, W. A. S. (2013). Filling the Holes with Workarounds: Watching Maps Work in the Terrain. Retrieved from https://www.duo.uio.no/handle/10852/37420

Edwards, P. N., Jackson, S. J., Bowker, G. C., & Knobel, C. P. (2007). Understanding infrastructure: Dynamics, tensions, and design.

EIEJ (2015). Én Innbygger – Én Journal. *Ehelse.No*. https://ehelse.no/strategi/n-innbygger-n-journal, accessed October 3, 2017.

Ellingsen, G., & Monteiro, E. (2003). A Patchwork Planet Integration and Cooperation in Hospitals. *Computer Supported Cooperative Work (CSCW)*, *12*(1), 71–95. https://doi.org/10.1023/A:1022469522932

Ellingsen, G., & Monteiro, E. (2006). Seamless Integration: Standardisation across Multiple Local Settings. *Computer Supported Cooperative Work (CSCW)*, *15*(5–6), 443–466. https://doi.org/10.1007/s10606-006-9033-0

*EPJ Monitor Årsrapport 2008*. (2008). Retrieved from hiwiki.idi.ntnu.no/images/7/79/EPJ-monitor-2008-hovedrapport.pdf

Eysenbach, G. (2001). What is e-health? *Journal of Medical Internet Research*, *3*(2), e20. https://doi.org/10.2196/jmir.3.2.e20

Friedberg, M. W., Chen, P. G., Van Busum, K. R., Aunon, F., Pham, C., Caloyeras, J., … Tutty, M. (2013). Factors Affecting Physician Professional Satisfaction and Their Implications for Patient Care, Health Systems, and Health Policy. Retrieved March 7, 2017, from http://www.rand.org/pubs/research_reports/RR439.html

Gebauer, J., & Schober, F. (2006). Information system flexibility and the cost efficiency of business processes. *Journal of the Association for Information Systems*, *7*(3), 8.

Hanseth, O., & Bygstad, B. (2017). The ePrescription Initiative and Information Infrastructure in Norway. In *Information Infrastructures within European Health Care* (pp. 73–87). Springer, Cham. https://doi.org/10.1007/978-3-319-51020-0_6

Hanseth, O., & Lyytinen, K. (2004). Theorizing about the design of Information Infrastructures: design kernel theories and principles. Retrieved from http://sprouts.aisnet.org/124/

Hanseth, O., Monteiro, E., & Hatling, M. (1996). Developing information infrastructure: The tension between standardization and flexibility. *Science, Technology & Human Values*, *21*(4), 407–426.

Heimly, V., Grimsmo, A., Faxvaag, A., & others. (2011). Diffusion of Electronic Health Records and electronic communication in Norway. *Applied Clinical Informatics*, *2*(3), 355–364.

Humphreys, B. L. (2000). Electronic Health Record Meets Digital LibraryA New Environment for Achieving an Old Goal. *Journal of the American Medical Informatics Association*, *7*(5), 444–452. https://doi.org/10.1136/jamia.2000.0070444

Koppel, R., & Lehmann, C. U. (2015). Implications of an emerging EHR monoculture for hospitals and healthcare systems. *Journal of the American Medical Informatics Association*, *22*(2), 465–471. https://doi.org/10.1136/amiajnl-2014-003023

Krist, A. H., Beasley, J. W., Crosson, J. C., Kibbe, D. C., Klinkman, M. S., Lehmann, C. U., … Waldren, S. E. (2014). Electronic health record functionality needed to better support primary care. *Journal of the American Medical Informatics Association*, *21*(5), 764–771. https://doi.org/10.1136/amiajnl-2013-002229

Larsen, E., & Mydske, P. K. (2013). IJMR-Developing Electronic Cooperation Tools: A Case From Norwegian Health Care | Larsen | Interactive Journal of Medical Research. Retrieved March 13, 2017, from http://www.i-jmr.org/2013/1/e9/

MacCormack, A., Rusnak, J., & Baldwin, C. Y. (2006). Exploring the structure of complex software designs: An empirical study of open source and proprietary code. *Management Science*, *52*(7), 1015–1030.

Mandel, J. C., Kreda, D. A., Mandl, K. D., Kohane, I. S., & Ramoni, R. B. (2016). SMART on FHIR: a standards-based, interoperable apps platform for electronic health records. *Journal of the American Medical Informatics Association*, ocv189. https://doi.org/10.1093/jamia/ocv189

Mandl, K. D., & Kohane, I. S. (2009). No Small Change for the Health Information Economy. *New England Journal of Medicine*, *360*(13), 1278–1281. https://doi.org/10.1056/NEJMp0900411

Mandl, K. D., & Kohane, I. S. (2012). Escaping the EHR Trap — The Future of Health IT. *New England Journal of Medicine*, *366*(24), 2240–2242. https://doi.org/10.1056/NEJMp1203102

Mandl, K. D., & Kohane, I. S. (2015). Federalist principles for healthcare data networks. *Nature Biotechnology*, *33*(4), 360–363. https://doi.org/10.1038/nbt.3180

Mandl, K. D., Mandel, J. C., & Kohane, I. S. (2015). Driving Innovation in Health Systems through an Apps-Based Information Economy. *Cell Systems*, *1*(1), 8–13. https://doi.org/10.1016/j.cels.2015.05.001

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. Sage Publications, Incorporated.

Monteiro, E., & others. (2003). Integrating health information systems: a critical appraisal. *Methods of Information in Medicine*, *42*(4), 428–432.

Noumeir, R. (2011). Sharing Medical Records: The XDS Architecture and Communication Infrastructure. *IT Professional*, *13*(4), 46–52. https://doi.org/10.1109/MITP.2010.123

Orlikowski, W. J. (1992). The Duality of Technology: Rethinking the Concept of Technology in Organizations. *Organization Science*, *3*(3), 398–427. https://doi.org/10.1287/orsc.3.3.398

Payne, T. H., Corley, S., Cullen, T. A., Gandhi, T. K., Harrington, L., Kuperman, G. J., … Zaroukian, M. H. (2015). Report of the AMIA EHR-2020 Task Force on the status and future direction of

EHRs. *Journal of the American Medical Informatics Association*, *22*(5), 1102–1110. https://doi.org/10.1093/jamia/ocv066

Pesaljevic, A. (2016). e-Prescription Embeddedness in the Norwegian Health Sector. Retrieved from https://www.duo.uio.no/handle/10852/54597

Rolland, K. H., & Monteiro, E. (2002). Balancing the Local and the Global in Infrastructural Information Systems. *The Information Society*, *18*(2), 87–100. https://doi.org/10.1080/01972240290075020

Roland, Lars Kristian, Terje Aksel Sanner, Johan Ivar Sæbø, and Eric Monteiro (2017). P for Platform. *Scandinavian Journal of Information Systems 30(2)*.

Sanner, T. A., Roland, L. K., & Braa, K. (2012). From pilot to scale: Towards an mHealth typology for low-resource contexts. *Health Policy and Technology*, *1*(3), 155–164. https://doi.org/10.1016/j.hlpt.2012.07.009

Skorve, E., & Aanestad, M. (2010). Bootstrapping Revisited: Opening the Black Box of Organizational Implementation. In *Scandinavian Information Systems Research* (pp. 111–126). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14874-3_8

Warner, J. L., Jain, S. K., & Levy, M. A. (2016). Integrating cancer genomic data into electronic health records. *Genome Medicine*, *8*, 113. https://doi.org/10.1186/s13073-016-0371-3

Weng, C., Appelbaum, P., Hripcsak, G., Kronish, I., Busacca, L., Davidson, K. W., & Bigger, J. T. (2012). Using EHRs to integrate research with patient care: promises and challenges. *Journal of the American Medical Informatics Association*, *19*(5), 684–687. https://doi.org/10.1136/amiajnl-2012-000878