Leveraging Open Source Software Platforms towards HIS Implementation in Developing Countries

The Case of DHIS2 in Malawi

Brown Chawanangwa Msiska

PhD Thesis

Department of Informatics

Faculty of Mathematics and Natural Sciences

University of Oslo

September 2018

To Ndagha Hastings Msiska.

Bagga love til di end!

Table of Contents

List of Figures

List of Tables

Acronyms

API	Application Programming Interface
CDC	Center for Disease Control and Prevention
CMED	Central Monitoring and Evaluation Division
CoM	College of Medicine
DHA	Department of HIV and AIDS
DHIS	District Health Information Software
DHIS1	District Health Information Software version 1
DHIS2	District Health Information Software version 2
DHO	District Health Office
DREAM	Drug Resources Enhancement against AIDS and Malnutrition
EHR	Electronic Health Records
EMR	Electronic Medical Records
GIS	Geographic Information System
HIS	Health Information System
HISP	Health Information System Programme
HISP UiO	HISP University of Oslo
HMIS	Health Management Information System
HTTP	Hypertext Transfer Protocol
ICT	Information and Communication Technology
I-TECH	International Training and Education Center for Health
JRE	Java Runtime Engine
NORAD	Norwegian Agency for Development Cooperation
SDK	Software Development Kit
SQL	Structured Query Language
TCO	Total Cost of Ownership
UNICEF	United Nations Children's Fund

Acknowledgements

First, I would like to give praises and express my indebtedness to the Almighty for the gift of life and continued sustenance over the course of this journey.

I am would like to express my heartfelt gratitude towards my supervisors - Associate Professor Petter Nielsen and Professor Jens Kaasboll. Your guidance, encouragement and support has brought me this far.

I am also indebted to professors, Sundeep Sahay, Ole Hanseth, Kristin Braa, Margunn Aanestad and Bendik Bygstad, for knowledge conveyed through your respective courses and your valuable input time and again.

Special thanks to the entire HISP UiO team, especially Ola Titlestad, Knut Starling, Alice Loba and Kjerstin Andreasen for the support rendered during my studies. Thanks, in the same vein to Terje Aksel Sanner and Johan Ivar Sæbø.

I would like to extend special gratitude to all participants in study, including but not limited to staff from the Central Monitoring and Evaluation Division of the Ministry of Health in Malawi, the Baobab Health Trust and HISP Malawi. Special thanks to Simon, Mbongeni, Yamikani, Teddy, Blessings, Maganizo, Matthew, Taona, Grace, Harold and other team members in the DHIS2 reconfiguration project. Further thanks to John Mukulu, John Melin, Tiwonge Manda and Øystein Gammersvik with whom I co-facilitated DHIS2 Web and Android application development workshops in Malawi.

Further gratitude goes to my fellow PhD students, especially Esther, Mikael, Manya, Ellen, Wilfred, Patrick, Flora, Christon, Anna, Roshan, Rangarirai, Arunima and Egil for the time we shared in and outside the 6^{th} *floor PhD Room.*

I would also like to express my sincere gratitude to Mozhdeh, Inna, Michelle and other members of the administrative team at University of Oslo for support rendered during my studies. In the same vain, I extend my thanks to Lånekassen for providing financial support for my studies through the Quota Scheme.

Special thanks go to my family for being an ever-present pillar of strength in this endeavor and in other aspects of my life. Similarly, I would also like to take time and express my gratitude to all the *Malawegians*, permanent and temporary Malawian-Norwegians, too many to mention one by one, for the times we spent together, and Malawian food and drinks we shared. You made Norway a home away from home.

Finally, I would also like to extend my gratitude to all those that have been instrumental on this journey, in Malawi and at University of Oslo, but have not been mentioned here. Not listing your names here does not, in any way, diminish the tremendous gratitude I have for the role you played in getting me this far.

Abstract

Lately, there has been a convergence between open source and software platform approaches in the implementation of health information system solutions for developing countries. Open source software, comprise all software distributed together with its source code under a license that permits the end user to study, modify and redistribute the software. Software platforms, on the other hand, comprise of software artefacts that have an extensible codebase that provides core functionality shared by applications associated with it and an interface through which it interoperates with such applications. The convergence of these two approaches in implementing health information system solutions has given rise to what can be termed as open source health information software platforms. DHIS2 is an example of such open source health information software platforms.

For developing countries, leveraging an existing open source health information software platform and its complementary applications can be less-risky, less-time consuming, and more cost-effective than starting from scratch. However, software platforms come with implicit human capacity requirements necessary to turn them into working solutions within context of use. However, research on open source software and health information systems in developing countries reports of failures attributed to deficiencies in requisite human capacities. Lack of requisite human capacity, if it exists, can constrain efforts by developing countries to leverage open source health information software platforms despite the promises they hold.

Against this background, the main objective of this study is contributing towards a practical and conceptual understanding of developing countries can effectively leverage open source health information software platforms against a backdrop of reported human capacity challenges. For this purpose, a case study involving efforts leveraging the DHIS2 software platform in Malawi, a developing country in southeast Africa, was carried out. Findings from the study relate leveraging open source health information software platforms in developing countries to a range of requisite human capacities, boundary resources and socio-technical generativity in relation to the platform itself, social relationships and generative capacity of actors involved.

With these findings, the study contributes theoretically by advancing socio-technical generativity as a concept to provide a holistic account for generativity exhibited by open source software platforms within their context of use. In addition, drawing on the boundary resources model, the study proposes an extended model to bring to the foreground external generative capacity and capacity building boundary resources as co-factors with software development boundary resources in shaping third-party development. Practically, the study contributes by itemizing and describing requisite human capacities for leveraging open source health information software platforms in developing countries that should guide efforts auditing and building requisite human capacities for open source software platforms in developing countries.

1 Introduction

This thesis relates to a research study on ongoing efforts in developing countries leveraging open source software platforms towards health information systems (HIS) implementation. It is based on a case study of efforts leveraging an open source health information software platform, District Health Information Software version 2 (DHIS2), in Malawi.

This chapter presents a background of the research, the research problem, the research objective, research questions related to the research objective, a summary of research findings, a summary of contributions, and winds up with a structural overview of the thesis.

1.1 Background

Health Information Systems integrate data collection, processing, reporting, and use of information necessary for improving the effectiveness and efficiency of healthcare service (WHO, 2004) and therefore play a critical role in the management of healthcare in both developed and developing countries (Azubuike and Ehiri, 1999). Consequently, health information system strengthening attracts significant effort and research worldwide. However, for health information system implementation initiatives in developing countries, there exists a longstanding challenge in terms of cost-effectiveness and sustainability emanating from the underlying resource constraints and high budgetary deficits (Bakar et al., 2012; Karuri et al., 2014; Kimaro, 2006; Kimaro and Nhampossa, 2005).

One of the cost factors in health information systems implementation is the underlying software. Proprietary software is traditionally distributed under a license which requires end-users to pay annual license fees and restricts them from modifying and redistributing the software (Fogel, 2005). Because of prevailing resource constraints and high budgetary deficits, the total costs of ownership of proprietary software has been a major concern for health information system implementation in developing countries (Sheikh and Bakar, 2012). On the other hand, open source software provides countries with modest resources an opportunity to implement low-cost health information systems and access modern data analysis and visualization tools (Yi et al., 2008). Open source software does not require annual license fees, is made available alongside its source code, and is distributed under a license that permits end-users to study, change, and improve the software (Kandar et al., 2011). This allows end-users to create and extend the software with local

innovations that fit the local context. Consequently, there has been increased usage of open source software as the basis for health information systems implementation in developing countries (Karuri et al., 2014; Sheikh and Bakar, 2012) resulting in proliferation of open source health information software such as Open Medical Record System (OpenMRS), Open Logistics Information System (OpenLMIS) and DHIS2 to mention a few.

However, the freedom to study, change and improve the software presents a unique challenge that has become synonymous with open source software. With open source software there is an inherent possibility of modifications to the software by actors within its context of use resulting in *forks* – new versions of the software that are either incompatible or competing to the original software (Fogel, 2005). Forkability (*ibid.*), especially where it results in incompatible forks, can deny end-users access to subsequent updates to the underlying software.

While this is the case, software ecosystems have emerged as dominant model for software development (Manikas and Hansen, 2013; Tiwana, 2013). Software ecosystems consists of two key elements - a software platform and complementary applications. A software platform is a software-based system that provides core functionality shared by applications that interoperate with it and interfaces through which the applications interoperate with it (Eck et al., 2015; Tiwana, 2013). On the other hand, an application is an add-on software subsystem that connects to the software platform to extend its functionality (Tiwana, 2013). A software ecosystem, therefore, is comprised of a software platform and a collection of applications specific to it (Tiwana, 2013; Tiwana et al., 2010). By decoupling applications from the software platform, the software ecosystem model offers an opportunity to alleviate the forkability challenge that characterize open source software.

The success registered by consumer-oriented software platforms such as Google's Android and Apple's iOS has garnered attention for software platforms in other spheres including health information systems. This has given rise to creation of new open source health information software platforms and platformisation of existing open source health information software (Polak, 2015). This has been a typical trend with contemporary health information software such as OpenMRS and DHIS2, widely used in developing countries. For developing countries, leveraging an existing open source software platform towards health information system implementation is

more cost effective, less risky and quick way than implementing one from scratch (Bansler and Havn, 1994; Yi et al., 2008).

1.2 Research Problem

Software platforms represent a shift from an era where software vendors, in this case platform owners, delivered to end-users, in this case platform consumers, a fully-fledged solution to an era where platform owners deliver software products that must be completed within their context of use by actors working in the platform consumer's community. Noting this trend, Dittrich (2014) labelled software platforms as "half-products" because they defer part of the effort required to compose a complete solution to actors within their context of use. However, deferring part of the development effort to the platform consumer community introduces human capacity requirements with the context of use unlike those demanded by traditional software products. Thus, for developing countries leveraging open source health information software platforms appropriate human capacities must exist within context of use. Otherwise, decay and obsolescence of ICT solutions is common where appropriate human capacities are deficient (Boerma, 1991).

Despite the implicit human capacity requirements introduced by open source health information software platforms, prior research on health information systems projects in developing countries suggests a history of failure and unsustainability due to lack of human capacities to use, develop and maintain health information systems (Kimaro and Nhampossa, 2005). During implementation, donors funding health information system projects in developing countries have usually addressed human capacity gaps by engaging foreign experts at the expense of building local expertise (Kimaro, 2006). Consequently, deficiencies persist leaving developing countries perpetually dependent on external expertise to run and manage their health information systems.

These deficiencies have also been echoed in research focusing on open source software in developing countries. For example, Roets et al. (2007) report that finding skilled developers in developing countries is a struggle due to lack of training and brain drain. It has been further observed that most developers participating in open source projects are predominantly from developed countries (Paudel et al., 2010; Weerawarana and Weeratunge, 2004). Consequently, many developing countries have failed to take full advantage of promises held by open source software.

Drawing on the observation by Dittrich (2014), leveraging open source software platforms towards health information system implementation is an endeavor that entails turning half a solution, delivered by platform owners, to a working solution within context of use. For developing countries, deficiencies in requisite human capacities, where they exist, can limit their ability to leverage open source software health information software platforms. With all this under consideration, it becomes important to investigate and understand how developing countries can effectively leverage open source software platforms, such as DHIS2 and OpenMRS, towards health information system implementation against the backdrop of reported human capacity deficiencies.

1.3 Research Objective and Research Question

Based on the research problem outlined above, the main objective of this study is contributing towards a practical and conceptual understanding of how developing countries can effectively leverage open source health information software platforms against a backdrop of human capacity challenges. To achieve this overarching objective, this thesis addresses three questions:

- **Research Question 1 (RQ1):** What human capacities are needed to leverage open source software platforms as means for implementation of health information systems in developing countries?
- **Research Question 2 (RQ2):** How can gaps in human capacity needed to leverage open source software platforms in developing countries, if any, be assessed and addressed?
- **Research Question 3 (RQ3):** How are efforts aimed at leveraging open source software platforms in implementation of health information systems influenced by the context in a developing country and characteristics of the platform itself?

1.4 Research Approach

To address the research objective and research questions above, an interpretive case study focused on efforts leveraging the DHIS2 software platform in Malawi was carried out from 2015 through to 2017. The case study was carried out under the umbrella of the Health Information Systems Programme (HISP), a global action research project action project championed by the Department of Informatics at University of Oslo, in Norway. Under this project, the University of Oslo in collaboration with several stakeholders in Malawi is carrying out ongoing efforts aimed at strengthening the health information system landscape in Malawi using the DHIS2 software

platform as a vehicle for instituting change. The thesis, in chapter 4, discusses the research approach in more detail.

1.5 Research Findings

The research findings of the study are addressed in the following research papers, appended as part of this thesis:

Table 1.1 Research Papers Appended to the Thesis

1. Msiska, B. & Nielsen, P. (2017), A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries, *Proceedings of the 14th International Conference of IFIP Working Group 9.4*, May 2017, Yogyakarta, Indonesia
2. Msiska, B. (2017), Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries, *Proceedings of IST-Africa 2017 Conference*, May 2017, Windhoek, Namibia
3. Msiska, B. & Nielsen, P (2017*),* Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity, *Information Technology for Development, 24(2), pp 388-421*
4. Msiska, B. (2018), Cultivating Third-Party Development in Platform-Centric Software Ecosystems: Extended Boundary Resources Model, *The African Journal of Information Systems, 10 (4), Article 6, pp 348–365*

In chapter 6, the thesis summarizes each of these papers and tabulates their contributions with respect to the research questions above.

1.6 Research Contributions

The thesis makes both conceptual and practical contributions with respect to leveraging open source software platforms towards HIS implementation in developing countries.

Theoretically, the thesis draws from Zittrain's concept of generative technology (Zittrain, 2008, 2006) and Lane's concept of generative relationships (Lane, 2011), to advance *socio-technical generativity* as a concept aimed at providing a holistic account for generativity exhibited by software platforms within their context of use and argues that it does not make sense to look at technological or social factors that constrain or enable innovation on software platforms in isolation. With respect to third-party application development, the thesis draws from the boundary resources model in Ghazawneh and Henfriddson (2013) and proposes extended boundary

resources model to bring to the foreground external generative capacity and capacity building boundary resources as factors in third-party development alongside software development boundary resources.

Practically, the thesis highlights a set of requisite human capacities for developing countries to effectively leverage open source health information software platforms which call for a departure from the tendency to emphasize on building end-user capacity when implementing software-based systems. The thesis, then argues, that in the absence of the other requisite capacities, the extent to which developing countries can leverage software platforms to derive features demanded by current and prospective end-users would be constrained and possibly lead to the obsolescence of platform instances within their context of use. The thesis also offers practical insights on the impact of software platform release management, backward compatibility, and testing mechanisms on existing instances of software platforms and applications in developing countries and reiterates practices, as suggested by Bosch (2010) for example, to minimize unintended breaks and argues that for stakeholders in developing countries, already battling the scarcity of resources, the cost, effort and time required to fix such unintended breaks might not be as attractive.

1.7 Thesis Organisation

This chapter aimed to set a foundation from which the rest of the thesis can be understood. The rest of the thesis is organization as follows:

- Chapter 2: discusses streams of literature related to the study
- Chapter 3: presents a conceptual framework that underpins the study
- Chapter 4: presents the research methodology employed during the study including among other things data collection techniques used, data analysis, ethical considerations and reflections on the methodology.
- Chapter 5: provides a description of the case this study is based on
- Chapter 6: presents the findings of the study drawing from 4 research papers appended as part of this thesis
- Chapter 7: discusses the findings with respect to the research questions and related literature
- Chapter 8: provides concluding remarks including among other research contributions and areas for further research as well as practice

2 Related Research

This chapter presents a discussion of literature related to the study this thesis reports on. With a focus on developing countries, the chapter reviews literature on health information systems, open source software, and software platforms and ecosystems.

2.1 Health Information Systems

The effectiveness of health care services relies on the availability of information systems that convey timely, accurate and readily available information to health practitioners, policy makers, researchers and the general public (Yi et al., 2008). Generating, analyzing and disseminating such information is the goal of health information systems (AbouZahr and Boerma, 2005; Vital Wave Consulting, 2009). An health information system is a system that integrates data collection, processing, reporting, and use of information necessary for improving effectiveness and efficiency of health care services through better management (WHO, 2004). Health information systems are a source of data addressing, for example, health outcomes and performance statistics such as mortality, morbidity, coverage and so on (AbouZahr and Boerma, 2005) which are critical to management of health care services in both developed and developing countries (Azubuike and Ehiri, 1999). Consequently, significant effort and research is currently being invested in health information systems.

2.1.1 Health Information Systems in Developing Countries

Many developing countries are yet to establish effective health information systems which results in insufficient information for planning and implementation of health programmes (Azubuike and Ehiri, 1999). Effective health information systems are vital for assessing health needs of populations, planning and implementation of health interventions, and monitoring and evaluation of health programmes in terms of their effectiveness and coverage (*ibid.*). In the absence of adequate health information systems, as observed by Azubuike and Ehiri (1999), there is a dearth for relevant, timely and accurate information which constrains important health-related decisions to estimates, sentiments and guesswork.

Besides being confronted by a wide variety of health-related challenges, health systems in developing countries struggle with limited resources which creates a need for health managers to maximize the value of those limited resources and find ways to make the health systems operate as efficiently as possible (Vital Wave Consulting, 2009). Successful strengthening of these health

systems requires relevant, timely, and accurate information on their performance to help measure their efficiency (*ibid.*). Consequently, strengthening health information systems is recognized as one of the key activities required to improve the effectiveness and efficiency of health service in developing countries (Karuri et al., 2014).

2.1.2 Barriers to Health Information Systems in Developing Countries

One barrier to development and maintenance of health information systems in developing countries relates to resource constraints under which the underlying health systems operate (Azubuike and Ehiri, 1999; Oak, 2007). High budget deficits in developing countries culminate in underinvestment towards development and maintenance of health information systems (Karuri et al., 2014; Sheikh and Bakar, 2012). The existence of this constraint should not be seen as a justification for abandoning health information system strengthening initiatives in developing countries but rather a reality that must be taken into account (Clifford et al., 2008). This calls for cost-effective health information system solutions in developing countries.

Another key barrier to health information systems in developing countries concerns inadequate human resource capacity (Azubuike and Ehiri, 1999; Kimaro and Nhampossa, 2005; Oak, 2007). Lack of human resource capacity to use, develop and maintain health information systems is one of the causes of their failure and unsustainability in developing countries (Kimaro and Nhampossa, 2005). Donor-driven health information system strengthening efforts have usually tackled this inadequacy by engaging external consultants, but these cannot be sustained on a long-term basis because of resource constraints as discussed earlier. Thus, scaling health information systems in developing countries entails scaling human resource capacity in terms of numbers and skills of both end users and implementation personnel providing technical support to the end users and the end user organization (Sahay and Walsham, 2006).

Last but not least, the surrounding socio-political landscape can prove to be an even bigger barrier towards development and maintenance of health information systems in developing countries (Littlejohns et al., 2003; Oak, 2007). Conflict of interests among stakeholders arising from the "my baby" syndrome characterized by struggles for control and ownership, for example, can constrain collaboration between various stakeholders surrounding health information systems (Littlejohns et al., 2003). Thus, health information system strengthening efforts in developing countries, other

than focusing on the technology, must embrace a multi-factorial approach if health information systems have to be implemented and maintained successfully (Clifford et al., 2008).

2.2 Open Source Software

Free and Open Source Software (FOSS) includes all software that is distributed alongside its source code and licensed under terms that comply with either the Free Software Definition by the Free Software Foundation or the Open Source Definition by the Open Source Initiative (Chavez et al., 2011). The Free Software Foundation (2018), defines *free software* as software whose license provides its users four essential freedoms: the freedom to run the software for any purpose, freedom to study and modify the software, freedom to distribute copies of the software, and freedom to improve and release improved versions on the software. The Open Source Initiative defines *open source software* as software that meets its 10 software distribution terms, which include the requirement that the software license for an open source software must not restrict other software that interoperate with it to be open source as well (Kandar et al., 2011; Open Source Initiative, 2018; Yildirim and Ansal, 2011). This requirement significantly sets apart open source software from free software, as open source software can make use of and be bundled together with proprietary software whereas free software cannot.

The term open source software has become preferable among many practitioners in the software industry because the term *"free software"* carries an unintended meaning of software that you can get at zero price whereas the intended meaning by the Free Software Foundation is software which guarantees the user certain freedoms (Roets et al., 2007). The Free Software Foundation, however, contends that open source software, although close in meaning, is not identical to free software because the word "open" does not imply freedom (Free Software Foundation, 2018). Furthermore, the unintended meaning of free software as in "*free beer*" holds as well because both free and open source software is usually distributed free of charge (Roets et al., 2007).

In practice, the majority of software that is classified as either free software or open source software complies with both definitions (Chavez et al., 2011). For practical purposes, in this thesis, open source software is defined as software that is made available alongside its source code and distributed under a software license that gives the users the freedom to study, change, and improve the software (Kandar et al., 2011). This definition does not preclude the essential freedoms stipulated in the Free Software Definition but rather establishes clear distinction with proprietary

software. With respect to proprietary software, the source code and copyrights are a preserve of the software vendor (Kandar et al., 2011).

2.2.1 Benefits of Open Source Software for Developing Countries

Open source software holds the promise to solve information system implementation challenges in developing countries that arise from various ills associated with proprietary software, including exorbitant license fees and monopolization by software giants (Câmara and Fonseca, 2007; Jaffry and Kayani, 2005; Weerawarana and Weeratunge, 2004).

The primary benefit open source software offers developing countries is the potential for lower total cost of ownership (TCO) in comparison to proprietary software. The acquisition cost of open source software, minus internet costs required to download the software, is usually zero because the majority open source software is distributed free of charge (Chavez et al., 2011). Moreover, software update costs, which constitute a large part of software TCO, are lower because there is no need to purchase new licenses to access new software features (Kandar et al., 2011; Roets et al., 2007). With proprietary software both software acquisition and access to new versions necessitate paying exorbitant license fees which often are based on the number of users or workstations.

The potential for skills development is another significant benefit that open source software offers developing countries. One of the most effective ways to learn software development is working with an existing codebase (Chavez et al., 2011). Since open source software is distributed alongside its source code, it offers new developers the opportunity to be exposed to the work of done by more experienced developers through which experiential learning can take place and best software development practices can be passed on (Chavez et al., 2011; Roets et al., 2007). Furthermore, the culture of sharing and improving already existing code creates a collaborative environment where people can learn from each other and improve their collective skills (Paudel et al., 2010).

In relation to access to existing source code there are three other significant benefits that can be accrued by developing countries from open source software. First, open software has the potential to break software lock-in and create independence from monopolistic software suppliers triggering further savings in costs (Roets et al., 2007; Weerawarana and Weeratunge, 2004). Second, with access to the source code open source is subject to scrutiny by a large community of developers

which increases the chance for high quality software (Chavez et al., 2011). Third, access to source code guarantees the auditability of the software which can help address global concerns about privacy, public data security and foreign control via proprietary software (Weerawarana and Weeratunge, 2004). The benefits of open source software are summarized in table 2.1 below.

Table 2.1 Benefits of Open Source Software

1. **Low TCO:** total cost of ownership can be low because software is acquired free of charge and updating to new versions does not require purchasing new licenses.
2. **Skills Development:** open source software, by making the source code available, supports learning from the work done by other developers.
3. **Independence from Monopolistic Suppliers:** the availability of source code frees users from software vendor lock-in and monopolistic suppliers.
4. **High Quality Software:** the software is subjected to scrutiny by a large community of developers which potentially increases its quality.
5. **Security:** the exact functions of the software can be audited due to availability of source code thereby allaying concerns about public data security and foreign control via software.

2.2.2 Open Source Software and Health Information Systems in Developing Countries

Due to the barriers discussed earlier, open source software constitutes a strategic choice for implementation of health information systems in developing countries. The cost of traditional proprietary software for analyzing, visualizing and delivering public health data puts them beyond the reach of resource constrained countries (Yi et al., 2008). Due to high financial deficits, exorbitant license fees for proprietary software and concerns over total cost of ownership drive a shift from proprietary software towards open source software in the implementation of health information systems in developing countries (Bakar et al., 2012; Sheikh and Bakar, 2012).

Open source software provides developing countries the opportunity to build low-cost health information systems allowing them access to modern data analysis and visualization tools (Yi et al., 2008). For example, a web-based health information system built with Microsoft Windows, Microsoft SQL Database Server and a GIS package like Esri's ArcGIS can easily cost tens of thousand dollars yet a similar system can be built using open source technologies such as EpiVue and PostgreSQL at a significantly low cost (*ibid.*). In this regard, open source software matches the goal for cost-effective health information systems that exists in developing countries. This

match has been the cause of the shift towards open source software witnessed in developing countries (Bakar et al., 2012).

The benefits open source software holds for health information system implementation in developing countries are not limited to cost alone. Access to source code and freedom to modify open source software emancipates developing countries from vendor lock-in (Bakar et al., 2012). Furthermore, the availability of source code and the freedom to study the software can be a catalyst for health information system integration (Sheikh and Bakar, 2012). However, Sheikh and Bakar (2012), observe that exploiting the benefits open source software offer developing countries towards health information system implementation requires with local readiness in terms of human resource capacity.

2.2.3 Challenges with Open Source Software for Developing Countries

Literature on open source software and developing countries identifies four key challenges. The first challenge relates to internet access. Much of communication and collaboration in open source communities relies on the internet which makes internet access vital for effective participation (Roets et al., 2007). Therefore, challenges with internet connectivity and associated costs can disrupt communication with the open source community at large (Bakar et al., 2012). The second challenge relates to language. English remains the lingua franca in software development and so too in open source communities, creating a potential barrier for would-be participants that are not as skilled in the language (Roets et al., 2007). The third challenge relates to volatility, or the rate of software changes in open source software projects. Open source software projects, characterized by rapid releases, typically have more iterations than proprietary software which creates a management problem as new releases have to be implemented in order extract full benefit from the software (Roets et al., 2007). Lastly, open source software projects have an inherent forkability challenge whereby modification and improvements of the software can creates two or more versions of the software that are potentially incompatible, called *forks* (Fogel, 2005). The emergence of an incompatible local fork can increase the cost of or deny users access to subsequent updates.

Table 2.2 Potential Challenges of Open Source Software for Developing Countries

1. **Internet Access:** challenges and costs associated with internet connectivity in developing countries can constrain participation in the wider open source software community.

2.	**Language:** the necessity of English, as the lingua franca in software development can be an impediment for would-be participants that are not as experienced in the language.
3.	**Volatility:** rapid releases, typical of open source software projects, can create a management problem as new releases must be implemented to extract full benefit from the software.
4.	**Forkability:** local modifications have the potential of generating incompatible software forks which can increase the cost of or deny users to new software updates.

2.3 Software Platforms and Ecosystems

As a consequence of increasing adoption of a software ecosystem approach in software engineering (Bosch, 2010), exemplified by Google's Android, Apple's iOS and Mozilla's Firefox browser, software ecosystems have emerged as a dominant model for software development and software-based services (Tiwana, 2013; Tiwana et al., 2010). A software ecosystem consists of a software platform, a set of internal and external developers, a community of domain experts, and a community of users whose needs are served and satisfied by composing the platform and applications specific to it (Bosch and Bosch-Sijtsema, 2010a). Unlike traditional software development approaches, software ecosystems leverage the expertise of a diverse developer community possessing an appreciation of user needs, which the developers of the software platform might not have, to develop new capabilities unforeseen by the platform developers (Tiwana et al., 2010).

As stated by Tiwana (2013), there are two key elements to a software ecosystem – a software platform and complementary applications. A software platform is a software-based system that provides core functionality shared by applications that interoperate with it and interfaces through which the applications interoperate with it (Ghazawneh and Henfridsson, 2013; Tiwana, 2013; Tiwana et al., 2010). The shared functionality and set of interfaces enable a software platform to serve as a foundation which outside parties can leverage to build complementary products and services that come in the shape of applications. An application, *app* for short, is an add-on software subsystem that connects to the software platform to extend its functionality (Tiwana, 2013). Therefore, in its simplest form, a software ecosystem is a collection of a software platform and apps which interoperate it.

The definition of software ecosystem by Tiwana (2013) and Tiwana et al. (2010) differs slightly from other definitions that take either a technical or a socio-technical perspective. Synthesizing

these definitions, Manikas and Jansen (2013) describe a software ecosystem as interaction of a set of actors on top a common technological platform that results in a number of software solutions or services. This is not unlike the description by Bosch and Bosch-Sijtsema (2009) which describes a software ecosystem in terms of the software platform surrounded by communities of different actors such as internal developers, external developers, domain experts and end users. As rightly observed by Manikas and Jansen (2013), three main elements cut across all these definitions: a common software platform, heterogeneous communities of actors, and relationships connecting them in their attempts to extract value from the software platform.

2.3.1 Benefits and Challenges

The popularity of the software ecosystem approach stems from its economic, strategic and technical advantages it offers (Berger et al., 2014). As identified by Bosch (2009), potential benefits of the software ecosystem approach include:

- **Increased value of the platform to existing users:** through interoperating with and development of a wide range of complementary applications the value of the software platform to its existing users is increased.
- **Increased attractiveness for new users:** by incorporating a range of applications beyond those envisaged by the platform developers the software platform can become attractive to new users.
- **Increased stickiness of the platform:** by interoperating with applications valuable to end users the chance of users switching to alternative solutions is reduced. For example, despite recent privacy concerns and scandals surrounding Facebook its stickiness arising from its interoperability with a multitude of applications has prevented most of its users from taking heed to calls to quit Facebook.
- **Accelerated innovation:** by involving external developers, platform owners can address functionality requirements that cannot be built within reasonable time using inhouse resources.
- **Reduced cost of innovation:** by involving external developers, the overall cost of innovation can be shared with other partners thereby freeing the platform owners from inhouse constraints.

Notwithstanding these benefits, the software ecosystem approach is not without challenges. Software platforms necessitate simplifying contribution and minimizing the effort required by external developers to build complementary applications (Bosch, 2009). Achieving this simplicity can be a challenge. Further to this, the following architectural challenges are identified by Bosch (2010):

- **Interface stability:** there is a need to decouple software platform interfaces from the platform core and evolve them in a predictable fashion to minimize the effects of new platform releases on existing external applications.
- **Evolution management:** the software platform must evolve to incorporate new functionality without inconveniencing current users and breaking existing external applications.
- **Security and reliability:** there is a need to facilitate interoperability with externally developed applications but also minimize opportunities for defective or malicious external code affecting the whole system and establishing mechanisms for auditing and evaluating applications without stifling external developer contribution can be a challenge.

2.3.2 Third-Party Application Development

A software platform is rarely a solution by itself; its attractiveness and stickiness is defined by applications built on top of it (Bosch, 2009). However, the wealth of complementary applications needed is difficult to achieve using inhouse developers alone (Ghazawneh and Henfridsson, 2013). Firstly, inadequate appreciation of user needs by platform developers (Tiwana et al., 2010) constrains their ability to make informed decisions on what applications to develop for the platform (Henfridsson and Lindgren, 2010). Secondly, the turbulence and diversity of information processing requirements in the user community means that the amount of functionality needed to be developed to satisfy the needs of the platform customers exceeds what can be built using inhouse resources (Bosch, 2009; Ghazawneh and Henfridsson, 2013). These reasons, as noted by Ghazawneh and Henfridsson (2013), make third-party application development increasingly attractive for software platform owners.

Third-party application development refers to a type of software development where an actor other than the platform owner, hereby called third-party developer, develops applications satisfying some of the requirements from platform users (Ghazawneh and Henfridsson, 2013). For platform

owners, third-party software development can accelerate innovation (Bosch, 2009) and form the basis for market leadership (Ghazawneh and Henfridsson, 2013). The usurping of market leadership by Apple's iOS ecosystem from Blackberry serves as a good example in this regard (Tiwana, 2013). For end users, third-party application development, fueled by better appreciation of their needs by third-party developers alluded to by Tiwana et al. (2010), addresses a segment of requirements that the platform owner is unable develop by itself (Bosch, 2009).

Third-party application development entails shifting of designing capabilities to external actors (Prügl and Schreier, 2006; von Hippel and Katz, 2002) who, by virtue of proximity, are capable of effectively addressing end user needs through application development (Ghazawneh and Henfridsson, 2013). Thus, provision of resources to facilitate third-party development is an intrinsic part of the software ecosystem approach. Furthermore, software platform owners must aim to simplify contribution by third-party developers by allowing use of generic and popular development environments (Bosch, 2009).

2.3.3 Software Platforms and Developing Countries

Early efforts to address HIS challenges in developing countries focused on exploiting the benefits held by open source software and gave rise to open source health information systems such as DHIS2, OpenMRS and OpenLMIS. However, the separation of software development from context of use made these solutions susceptible to forking as a result of attempts by local actors to make the solutions fit for local use (Braa and Sahay, 2012a). By separating components that remain stable across different contexts (the platform and its interface) from complementary components that vary across contexts (the applications) (Baldwin and Woodard, 2008), software platforms provide an architecture that can reduce the likelihood of forks emerging. Consequently, open source health information systems targeting developing countries, such as DHIS2 and OpenMRS, have undergone platformisation (Polak, 2015).

The shift towards platformisation of open source health information systems has, however, been accompanied by limited research on how developing countries can effectively amass desired HIS innovations from the emerging platforms. Compared with the research on open source software and developing countries, research on software platforms in developing countries as noted by Nielsen (2017) remains a greenfield warranting further scrutiny. One of the bones of contention, in this regard, is the half-solution nature of software platforms and its implications for the

development of digital HIS innovations in developing countries. It is in this vein that this thesis seeks to make its contributions.

3 Conceptual Framework

This chapter presents a conceptual framework underpinning the study. The conceptual framework has been composed from an attempt to garner three insights towards the study. Firstly, factors that have potential influence on efforts by actors in developing countries leveraging open source software platforms towards HIS implementation. Secondly, resources and capacities necessitated by efforts to leverage open source software platforms in developing countries. Thirdly, avenues for propagating requisite resources and capacities to actors in developing countries. For this purpose, the thesis draws on literature on generativity and boundary resources.

Generativity, with respect to software platforms, embodies literature and concepts pertaining to the extent to which innovations on software platforms by other actors other than the platform owners are enabled and enhanced. The notion of boundary resources, on the other hand, provides insights on requisite resources within a platform-centric software ecosystem to enable creation of innovations by other actors other than the platform owners.

3.1 Generativity

The term generativity has been used in information systems research to denote the overall potential for an existing technology artefact, in this case a software platform, to generate derivative innovations from actors other than its creators. Literally, the term generativity means the ability or capacity to generate or produce something (Avital and Te'eni, 2009). Pertinent to this study are three dimensions to generativity: generativity of the software platform, generativity of relationships between actors involved in efforts to leverage a software platform, and generativity (or generative capacity) of individual or groups of human actors. Each of these dimensions is briefly discussed below.

With respect to the software platform, generativity refers to the overall capacity of the platform to be flexible and malleable by diverse groups of actors and in ways unanticipated by its creators (Eck et al., 2015; Zittrain, 2008, 2006). This denotes the extent to which the software platform stimulates and supports other actors, other than the platform owner, to leverage the platform and create derivative products. In other words, the extent of innovations derived from a software platform depends on its generativity. This, according to Zittrain (2008), is determined by five key characteristics: *leverage*, *adaptability*, *ease of mastery*, *accessibility* and *transferability*. Table 3.1 briefly describes each of these characteristics.

Table 3.1 Characteristics that Determine Software Platform Generativity

1.	**Leverage:** denotes the extent to which productivity is gained by using a technology artefact than not. An artefact with good leverage makes difficult tasks easier.
2.	**Adaptability:** denotes the potential of a technology artefact to be adapted for use in a context different from the one it was designed for.
3.	**Ease of Mastery:** denotes how easy it is for an actor to understand a technology artefact as well as the amount of effort required to adapt it.
4.	**Accessibility:** denotes the easiness with which access to a technology artefact alongside the tools and information necessary for its mastery can be obtained.
5.	**Transferability:** denotes the easiness with which a technology artefact built for one context can be conveyed and re-appropriated in another context.

Without diminishing the importance of the generativity of the software platform, it is important to note that innovation is a human endeavor and is therefore subject to competences of individuals involved. Avital and Te'eni (2009), use the term generative capacity to refer to the generative potential of a person. In this respect, they define *generative capacity* as an attribute of a person that refers to one's ability to produce something in a particular task-driven context (*ibid.*). In other words, generative capacity is the one's ability to engage in acts that lead to innovation or increase of overall value in a given context (Osch and Avital, 2010). With respect to a software platform, generative capacity refers to the ability of a human actor to leverage a software platform as basis for constructing derivative innovations. Therefore, it can be argued that, generativity in a platform-centric software ecosystem depends not only on the generative potential of the software platform but also on the on the generative capacities of individual human actors involved.

Notably, associated with any platform-centric software ecosystem is a collective of human actors that includes, among others, platform owners, third-party developers and end users. Going beyond individuals, such collectives can also be generative or not. In this regard, Osch and Avital (2010), define *generative collectives* as groups of people with shared interests whose mutual rejuvenating, reconfiguring, reframing, and revolutionizing acts drive creativity and innovation. Basing on earlier work by Avital and Te'eni (2009), they define *collective generative capacity* as trait of a collective denoting its ability to engage in generative acts that produce generative outcomes. Thus, as illustrated in figure 3.1, *collective generative capacity,* which stems from the generative capacities of individuals making up a collective, promotes *collective generative acts* which result in *collective generative outcomes*, such as new configurations and possibilities, which in turn

affects the future potential of the collective to be generative, thereby creating a generative cycle (Osch and Avital, 2010). Efforts leveraging open source health information software platforms in developing countries can be perceived as an iterative process spanning such a generative cycle.

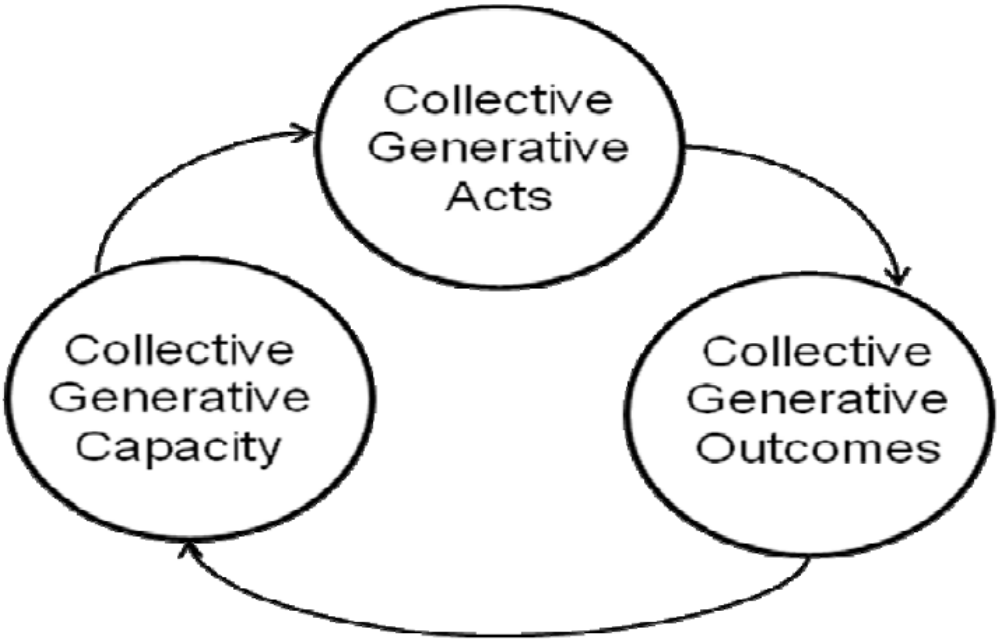


Figure 3.1The Generative Cycle of Collectives(Osch and Avital, 2010, p. 6)

Collective generative capacity is, however, greater than the mere sum of generative capacities of individuals in a collective as it emanates in part from the inherent relationships and interactions prevalent within the collective in question (Osch and Avital, 2010). Therefore, the relationships between various human actors within a software ecosystem can constrain or stimulate innovations atop the software platform in question. In agreement with this observation, Lane (2011) considers innovation as a social activity that is subject to extent of generative relationships among human actors involved. In this regard, the innovative potential of a collective, according to Lane (2011), depends on the existence of generative relationships which in turn depend on five characteristics, briefly described in table 3.2.

Table 3.2 Characteristics of Generative Relationships

1. **Aligned directedness:** denotes the degree of match between interests of different actors within a collective.
2. **Heterogeneity:** denotes the degree of variation in terms of competence, social positions and access to resources which discourages groupthink and encourages innovation.

3. **Mutual Directedness:** denotes the extent to which continued collaboration among actors is desired despite existing and apparent differences in interests or experience.
4. **Appropriate Permissions:** denotes the degree to which what individual actors are allowed to do within a collective is appropriate for desired innovations.
5. **Action Opportunities:** denotes the possibility of actors to engage one another in interactions that bring about change in the collective.

Putting this together, it can be argued that, generativity in a platform-centric software ecosystem depends not only on the platform's generativity but also on the existence of generative relationships and the generative capacities of human actors involved, individually and as a collective. Going further, it can be argued that the generative capacity of the software platform, actors involved, and their social relationships has potential to influence efforts leveraging open source software platforms towards HIS implementation in developing countries. This thesis, therefore, draws from the generativity dimensions discussed above to formulate an underpinning conceptual understanding of generativity as umbrella term that captures several factors that potentially influence leveraging open source health information software platforms in developing countries as summarized in table 3.3 below.

Table 3.3 Generativity Dimensions Related to Software Platforms within Context of Use

Generativity Concept	**Factors**	**Dimension**
Generative Technology	Leverage	Software Platform
	Adaptability	
	Ease of Mastery	
	Accessibility	
	Transferability	
Generative Relationships	Aligned Directedness	Social Relationships between Human Actors
	Heterogeneity	
	Mutual Directedness	
	Appropriate Permissions	
	Action Opportunities	
Generative Capacity	Generative Capacity	Ability of an individual human actor
Generative Collectives	Collective Generative Capacity	Ability of a collective of human actors

3.2 Boundaries in Collectives

Hierarchically, Osch and Avital (2010) observe that collectives encompass a small esoteric community and a larger exoteric community, each consisting of members of the collective with shared traits and interests. With respect to software ecosystems, the internal esoteric community is represented by the platform owners and the external exoteric community is represented by platform consumers at large. Separating these two communities is a boundary. Boundaries are a separation of two groups of people arising from differences in interests and identity (Wenger, 2000). Such boundaries constitute channels through which competences, experiences and resources are exchanged resulting in co-learning across the communities involved which enriches generative capacities on either side. The ensuing exchange is facilitated by three bridges: *boundary objects*, *boundary interactions*, and *brokers* (Wenger, 2000). Table 3.4 briefly describes each of these bridges.

Table 3.4 Boundary Bridges

1.	**Boundary Objects:** artefacts, including tools and documents for example, that link or are shared by communities across a boundary
2.	**Boundary Interactions:** events or encounters, for example visits and meetings, that provide direct exposure to members of another community
3.	**Brokers:** human actors operating between communities and engaged in the import and export of competences, knowledge and resources

Since software platforms necessitate the shifting of generative capacity between platform owners and platform consumers (Prügl and Schreier, 2006; von Hippel and Katz, 2002), the bridges between communities within a collective - boundary objects, boundary interactions and brokers - can be instrumental in deriving useful insights with respect to this desired shift and help shape our understanding of potential implications for leveraging open source health information software platforms in developing countries.

3.3 Boundary Resources Model

Besides the propagation of generative capacity, software platforms necessitate the existence of resources at the boundary between platform owners and platform consumers to enable actors outside the platform owner community create derivative innovations through third-party development. In this respect, Ghazawneh and Henfridsson (2010) defined *boundary resources* as

software tools and regulations, such as application programming interfaces (APIs) and software development kits (SDKs), that serve as an interface between platform owners and application developers to facilitate the development of third-party applications. Drawing from this definition, Ghazawneh and Henfridsson (2013) created the *boundary resources model,* presented in figure 3.1, to provide a theoretical account for the critical role played by boundary resources in stimulating generativity in platform-centric software ecosystems.

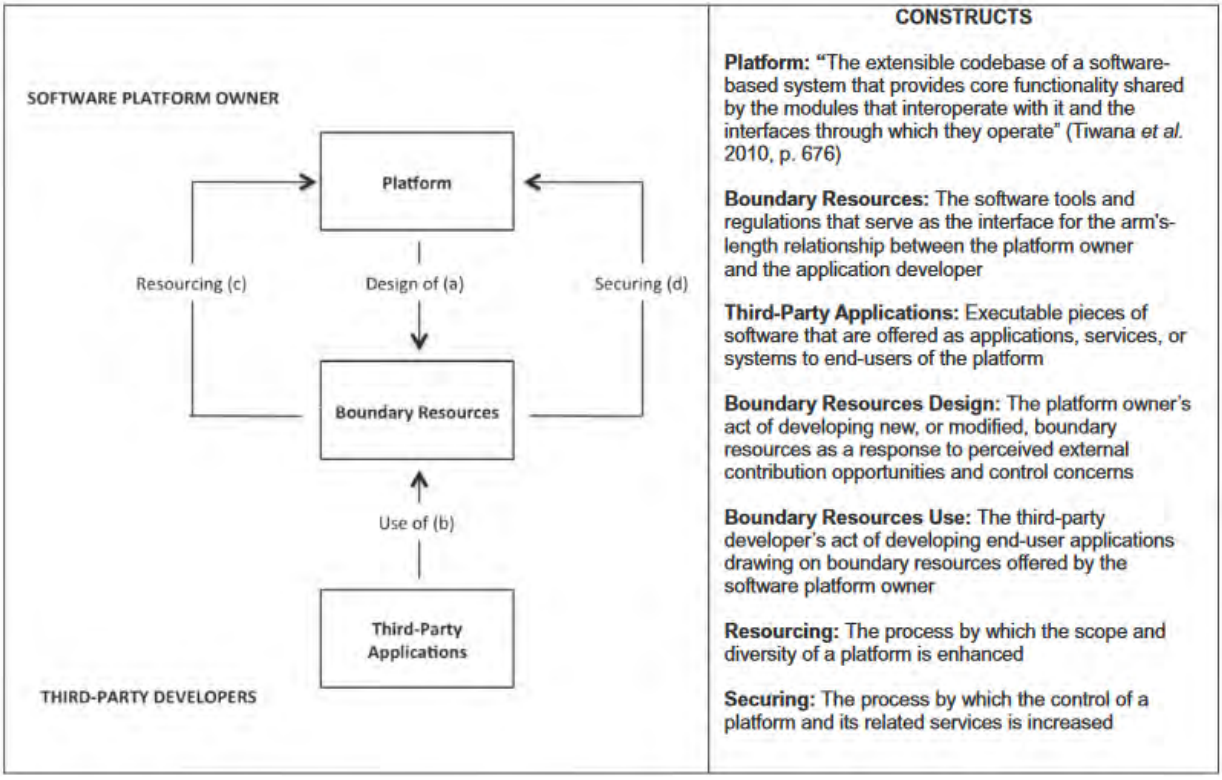


Figure 3.2 The Boundary Resources Model (Ghazawneh and Henfridsson, 2013)

The left-hand side of the model illustrates the relationships between the software platform and third-party applications as facilitated boundary resources. The right-hand side of the model briefly describes each of the constructs in the model. Platform owners *design* (or *redesign*) boundary resources to facilitate external contributions and address control concerns if the integrity of the platform is threatened by actions by application developers. Application developers make *use of* boundary resources to create derivative applications from the software platform.

Thus, boundary resources have a dual role: *resourcing* a software platform to enable third-party development and *securing* a software platform to address platform owner's control concerns

(Ghazawneh and Henfridsson, 2013). They use "jail-breaking" as witnessed in the Apple's iOS software platform as a typical control concern that might trigger platform securing. Through these two processes boundary resources determine what application developers can do with a software platform. Thus, as observed by Ghazawneh and Henfridsson (2013), boundary resources have the power to either stimulate or constrain generativity. While the model captures *resources design* as the act of the platform owner developing new or modified boundary resources, Ghazawneh and Henfridsson (2013) also talk of *self-resourcing* as the act of act third-party developers developing their own boundary resources as a response to perceived limitations with existing boundary resources provided by the platform owner.

The success and value of a software platform to its prospective end users is partly established by the range of third-party applications available in its ecosystem (Bosch, 2009; Tiwana, 2013). As illustrated in figure 3.2, boundary resources are an essential ingredient in the development of third-party applications. While leveraging open source software platforms towards HIS implementation in developing countries it might become necessary to develop third-party applications that address requirements not readily addressed by the software platform and its existing applications. In this regard, boundary resources could be serve as an important element in leveraging open source health information software platforms in developing countries.

3.4 The Conceptual Framework in Summary

The thesis draws from literature on *generativity* and *boundary resources* to formulate a conceptual framework that underpin subsequent discussions in later chapters and research papers appended as part of this thesis. Perspectives on generativity as summarized in table 3.3, provide insights on a myriad of factors that have potential influence on efforts by actors in developing countries leveraging open source software platforms towards HIS implementation. The boundary resources model, illustrated in figure 3.2, serves a basis for reflecting on and understanding boundary resources necessitated by efforts to leverage open source software platforms towards HIS implementation in developing countries. Boundaries in general, provide bridging constructs, described in table 3.4, as avenues for understanding the propagation of requisite resources and capacities to the platform consumer community involved in the leveraging open source health information software platforms in developing countries.

4 Methodology

This chapter provides an account of the underlying research methodology pertaining to this study. The subsequent sections present the research paradigm and approach, the research strategy, data collection techniques and data analysis techniques, respectively. The chapter ends with an account of ethical considerations that were undertaken as part of the study.

4.1 Research Paradigm and Approach

A research paradigm is a shared set of assumptions, beliefs and values about aspects of the world that guides how to do research and gain or create knowledge (Oates, 2005; Schwandt, 2001). Broadly, pertaining to research in the field of information systems, which this thesis belongs to, there are three philosophical paradigms: positivism, interpretivism and critical research (Oates, 2005; Orlikowski and Baroudi, 1991). Each of these philosophical paradigms holds and is characterized by a particular view about the nature of the world or what is reality (*ontology*), and how knowledge about it is acquired (*epistemology*) (Guba, 1990; Oates, 2005).

Positivism centers on the belief that our world is ordered and regular, not random, and all true knowledge we may obtain is based on the observation or experience of real phenomena in an objective and real world (Cornford and Smithson, 2005; Oates, 2005). Research, therefore, aims to unearth value-free and undisputable facts. Subsequently, these facts are used to determine laws, causal relationships and patterns that are independent of human cognition – personal feelings, thoughts and values (Oates, 2005). In other words, the researcher plays the role of a neutral observer and knowledge obtained does not depend on who carried out the research.

Interpretivism, also referred to as constructionism, is based on the belief that the world we see around us is the creation of the mind (Walliman, 2011). Therefore, unlike positivist research, interpretivist research accepts to be value-laden and subjective. What we call ‘data’ or ‘knowledge’ is a construction of our minds as individuals or as a group (Oates, 2005; Walsham, 2006). The role of research is, therefore, to reveal different interpretations of the world as made by people participating in social processes and how these interpretations help constitute social action (Orlikowski and Baroudi, 1991; Walliman, 2011).

Lastly, critical research agrees with interpretivism in that social reality is constructed by people but also agrees with positivism in that social reality possesses objective properties that tend to

dominate our experiences and ways of seeing the world (Oates, 2005). Critical research focuses on power relations, identifying and critiquing conflicts and contradictions in our modern world and helping to eliminate them as causes of alienation and domination (Oates, 2005; Orlikowski and Baroudi, 1991). So, while the other two paradigms are content with predicting or explaining the status quo the critical research paradigm is concerned with critiquing and challenging it.

Going beyond the research paradigms, research can be further categorized according to the underlying research approach which could be quantitative or qualitative (Cornford and Smithson, 2005; Creswell, 2009). This is not purely about quantitative versus qualitative but finding where research practice lies on a continuum between the two (Creswell, 2009). On this continuum we also find, mixed methods, which uses both qualitative and quantitative approaches. Qualitative research is based on measurements (numbers) that describe the phenomena under study (Cornford and Smithson, 2005), whereas qualitative research is based on words – including pictures and video (Miles and Huberman, 1994). Finally, mixed methods research is based on both quantitative and qualitative data.

Information systems, and by extension health information systems, are social systems in which technology is just one of many components (Angell and Smithson, 1991; Cornford and Smithson, 2005; Lippeveld et al., 2000; Walsham et al., 1988). The success of information system projects is heavily influenced by the surrounding social context as much as the technology artefact (Cornford and Smithson, 2005). Thus, leveraging open source software platforms to build health information systems in developing countries is subject to influence by the social context surrounding the software platform. This calls for an empirical approach that can unearth in-depth understanding from human interpretations and meanings and drawing from Walsham (2006, 1995), interpretive research was found suitable for carrying out the study. In the same vein, the study employed a qualitative approach.

4.2 Research Strategy

Practically, research can be distinguished by the underlying research strategy (also referred to as *strategy of inquiry* or *research method*) (Creswell, 2009). For qualitative research strategies of inquiry include ethnography, grounded theory, case study and action research (Creswell, 2009; Myers, 1997). Amongst these strategies, case study is the most commonly used strategy in interpretive research on information systems (Myers, 1997; Walsham, 1995). A case study

involves in-depth investigation of a contemporary phenomenon – for example, an organization, a department, an information system, a development project, and so on – within its real life context over an extended period of time (Creswell, 2009; Myers, 1997; Oates, 2005; Walsham, 1995).

In this study, the case study strategy was employed to investigate various aspects relating to efforts to leverage the DHIS2 software platform towards implementing the national health management information system in Malawi. As observed by Myers (1997), case studies are particularly well-suited for information systems research because when studying information systems organizational issues, rather than technical issues, are of interest. Furthermore, by investigating a phenomenon within it real-life context, case studies possess the ability to generate rich and detailed insights into the life, complex relationships and processes idiosyncratic to the phenomenon under study (Oates, 2005).

Although case study is the predominant research strategy for this study, the study itself was wedged within the scope of a global action research project, health information system programme (HISP), involving the University of Oslo and various stakeholders across the world. HISP is driven by a network of action (Braa et al., 2004) comprising of national and regional nodes in developing countries worldwide. Examples of these nodes include HISP India, HISP South Africa, HISP Tanzania, HISP Uganda, HISP Kenya, HISP Malawi, HISP West Africa and HISP East Africa. Under this global action research project, several interventions around DHIS2 were carried out in Malawi during the period of the study which I was part of.

Action research is a collaborative research strategy where the researcher forsakes their traditional observer role and takes part with the subjects to solve a problematic situation (Cornford and Smithson, 2005). It is an iterative plan-act-reflect process which involves diagnosing the source of the problem, planning remedial actions, implementing the actions, and reflecting on actions taken, their outcomes and lessons learnt (Oates, 2005). The aim of the researcher is to conduct research while effecting change (Benbasat et al., 1987; Cornford and Smithson, 2005). With respect to HISP, the overarching aim is to strengthen health information systems in developing countries by using a participatory approach (University of Oslo, 2016). During the study, I participated in various DHIS2 related interventions undertaken to strengthen the national health management information system in Malawi. These interventions included, among others, the DHIS2

reconfiguration and migration project, DHIS2 application development workshops and formulation of standard working procedures.

As a result of the my involvement in these interventions, I played two different roles in study: an outside observer and an involved researcher (Walsham, 1995). This, in some ways, facilitated the case study. First, my field visits were often scheduled to coincide with scheduled interventions in Malawi. Second, my involvement in the various interventions also enabled me to purposively sample respondents to be interviewed as part of the case study and provided an avenue for participant observation. A summary of field visits is provided in table 4.1 below.

Table 4.1 Summary of Field Visits

Period	**Activity**
January 2015 to May 2015	Field visit for preliminary data collection on DHIS2 implementation in Malawi – background, current status, local stakeholders and their roles, and challenges – Lilongwe and Zomba, Malawi
August 2015 to January 2016	Field visit – plans for DHIS2 reconfiguration and data migration, plans for DHIS2 web applications development workshop – Lilongwe and Zomba, Malawi
March 2016	DHIS2 web applications development workshop, Zomba, Malawi
April 2016 to July 2016	DHIS2 reconfiguration and data migration project, Ministry of Health, Lilongwe, Malawi
November 2016 to January 2017	Follow up DHIS2 reconfiguration and data migration project, Ministry of Health, Lilongwe, Malawi
October 2017	DHIS2 android applications development workshop, Zomba, Malawi

4.3 Data Collection

Each research strategy has one or more data collection methods (Oates, 2005). In case studies, interviews are the primary data collection technique as they allow the researcher to access interpretations participants attach to actions and events which have taken place or are taking place and views held by other participants (Walsham, 1995). Other qualitative data collection methods that may be used to complement interviews include observations, document reviews, and web-based data sources such as emails, mailing lists and websites (Creswell, 2009; Myers, 1997; Walsham, 2006). In this study data was collected using interviews, participant observations, document analysis and web-based data sources.

4.3.1 Interviews and Group Discussions

An interview is a conversation between people whereby one party seeks to gain information from the other (Oates, 2005). Interviews can be categorized into three types: structured, unstructured and semi-structured interviews (Cornford and Smithson, 2005; Oates, 2005). An interview is regarded as *structured* if it follows a predefined sequence of pre-planned questions and *unstructured* if it is not based on pre-planned questions but explores in-depth topics emerging from the conversation with the interviewee. *Semi-structured* interviews follow a predefined interview guide that specifies key questions and allow the interviewer to explore in-depth emergent topics just like unstructured interviews. In this study, semi-structured interviews were used as the primary data collection technique.

Interviews involved staff in the Central Monitoring and Evaluation Division (CMED) of the Ministry of Health in Malawi and staff from different stakeholder organisations involved in the use, implementation, and day to day running of DHIS2 in Malawi. Such stakeholder organisations include, among others, HISP Malawi, Baobab Health Trust (BHT), University of Malawi, International Training and Education Center for Health (I-TECH), Luke International Norway (LIN) and D-tree International. Purposeful sampling (Creswell, 2009) was used to identify respondents for the interviews depending on their involvement with DHIS2 in Malawi. Table 4.2 presents a summarized account of interviews conducted during the study.

Table 4.2 Tabulation of Interviews Conducted in the Study

Organisation	**No. Respondents**	**Designation**
Ministry of Health – CMED	6	Director Chief Technical Assistant Chief Statistician DHIS2 Technical Assistants (3)
Ministry of Health – IT Department	2	System Developers (2)
HISP Malawi	3	Board members (3)
Baobab Health Trust	4	Executive Director Software Development Manager DHIS2 Integration Team Members (2)
University of Malawi	5	HISP Malawi Representative Software Developer/DHIS2 Technical Assistant (4)
I-TECH	2	Technical Assistant, Systems Administration

		Technical Assistant, Statistics
D-tree International	1	Project Manager
Luke International Norway	2	Software Developers
GIZ/EPOS Health Management	2	Senior Technical Advisor DHIS2 Technical Assistant

Interviews, although usually one-to-one, can be undertaken with a group of respondents, referred to as focus group discussions (Oates, 2005). In addition to using semi-structured interviews further data was collected using focus group discussions during the study. As part of the DHIS2 reconfiguration and data migration project, focus group discussions and weekly progress review meetings were conducted with health program coordinators and other ministry of health officials. The focus group discussions and weekly review meetings offered useful insights that enriched the data collected from semi-structured interviews. This included, discussions aimed at resolving challenges and conflicts related to DHIS2 by various concerned parties.

Over the period of the study, I attended a DHIS2 Web Applications Development workshop in Kampala, Uganda and a DHIS2 Web Applications Development Academy in Dar es Salaam, Tanzania. Through these events, I interfaced with and interviewed participants from HISP Uganda, HISP Rwanda, HISP Tanzania, HISP South Africa and HISP Kenya. Although the interviews were not directly related to the implementation of DHIS2 in Malawi, they helped shape my understanding of HISP operations in other developing countries and provided a platform sharing and learning from each other's experience.

4.3.2 Participant Observation

Participant observation is a data collection technique in which the researcher participates in the situation under study to experience it from the perspective of other participants in the setting (Oates, 2005). This enables the researcher to find out what participants actually do instead of what they say they do or did. During this study I participated in several DHIS2 related activities in Malawi alongside various participants. These activities, briefly described below, allowed me to appreciate first-hand experiences, challenges and strategies involved in the implementation of DHIS2 in Malawi.

- *Setting Up a New DHIS2 Instance:* In 2016 the Ministry embarked on a DHIS2 reconfiguration and data migration project to migrate data from the old DHIS2 instance

hosted and managed by HISP Malawi to a new instance hosted by the ministry's HIV/AIDS department. As part of this project, I worked with a team comprising of staff drawn from University of Malawi, HISP Malawi and I-TECH to deploy a new DHIS2 instance at the offices of the HIV/AIDS Department in Lilongwe, Malawi.

- *Configuring New DHIS2 Instance:* Following the deployment of the new instance I further worked with a team comprising of staff drawn from University of Malawi, HISP Malawi, Baobab Health Trust, Ministry of Health ICT department and GIZ/EPOS Health Management creating customs forms, defining data elements and indicators for the new instance.
- *Software Development for Data Migration:* I worked with two software developers – a DHIS2 technical assistant from HISP Malawi and a software developer from University of Malawi – developing scripts and applications to automate migration of data from the old instance to the new instance. This was necessary because the data elements definitions in the two instances were structurally different which made reusing the database in the old instance impossible.
- *Data Migration:* Working in pairs, the team used the scripts and applications developed to migrate data from the old instance for each health program running over the period 2009 to 2016. During this phase I was paired with one of the staff from Baobab Health Trust who was part of the DHIS2 reconfiguration and data migration team.
- *Progress Review Meetings for DHIS2 Reconfiguration and Data Migration*: Every Thursday noon the DHIS2 reconfiguration and data migration team prepared a progress report circulated on a DHIS2 Malawi mailing list comprising of various stakeholders within and outside the ministry of health. Every Friday noon there was a progress review meeting based on the report circulated the previous day.
- *Content Development for DHIS2 Application Development Workshops:* To build capacity towards application development on top of the DHIS2 two workshops were carried out in Malawi – a DHIS2 Web Applications Development workshop in March 2016 and a DHIS2 Android Applications Development workshop in October 2017. The workshops involved participants from Kenya, Zambia, Ethiopia, Mozambique and Malawi. I worked with staff from HISP University of Oslo, HISP Tanzania and University of Malawi developing content for the workshops.

- *Facilitation of DHIS2 Application Development Workshops:* Together with staff from HISP University of Oslo, HISP Tanzania and University of Malawi I facilitated the application development workshops in Malawi.
- *mHealth4Afrika Project:* The mHealth4Afrika project is a joint project run by the University of Oslo in Norway, University of Gondar in Ethiopia, Strathclyde University in Kenya, University of Malawi in Malawi and Nelson Mandela Metropolitan University in South Africa. The aim of the project is to develop an open source maternal and child health electronic medical records (EMR) system on top of the DHIS2 software platform. I have been participating in the project as a software developer representing University of Malawi in the project.

Figure 4.1 Part of the DHIS2 Reconfiguration and Data Migration Team (Top Left). Setting Up New DHIS2 Instance (Top Right). DHIS2 Reconfiguration and Data Migration Progress Review Meeting (Bottom Left). Content Development for DHIS2 Web Application Development Worskshop (Bottom Right).

4.3.3 Document Reviews

The term "*documents*" extends beyond just written materials; includes any symbolic representation that can be recorded and retrieved for analysis (Oates, 2005). In this regard documents include

textual documents (e.g. reports, memos and academic publications), audiovisual data (e.g. photographs, diagrams, animations, video and sounds) and electronic data (e.g. screenshots, computer games and archives) (*ibid*). During the study, a wide range of documents related to HISP, DHIS2 and the implementation of DHIS2 in Malawi were collected and reviewed. Such documents included, among many others, policy documents, reports, manuals and research publications by other researchers. Some of the documents were obtained through the interviews conducted and others were downloaded from websites of stakeholder organizations.

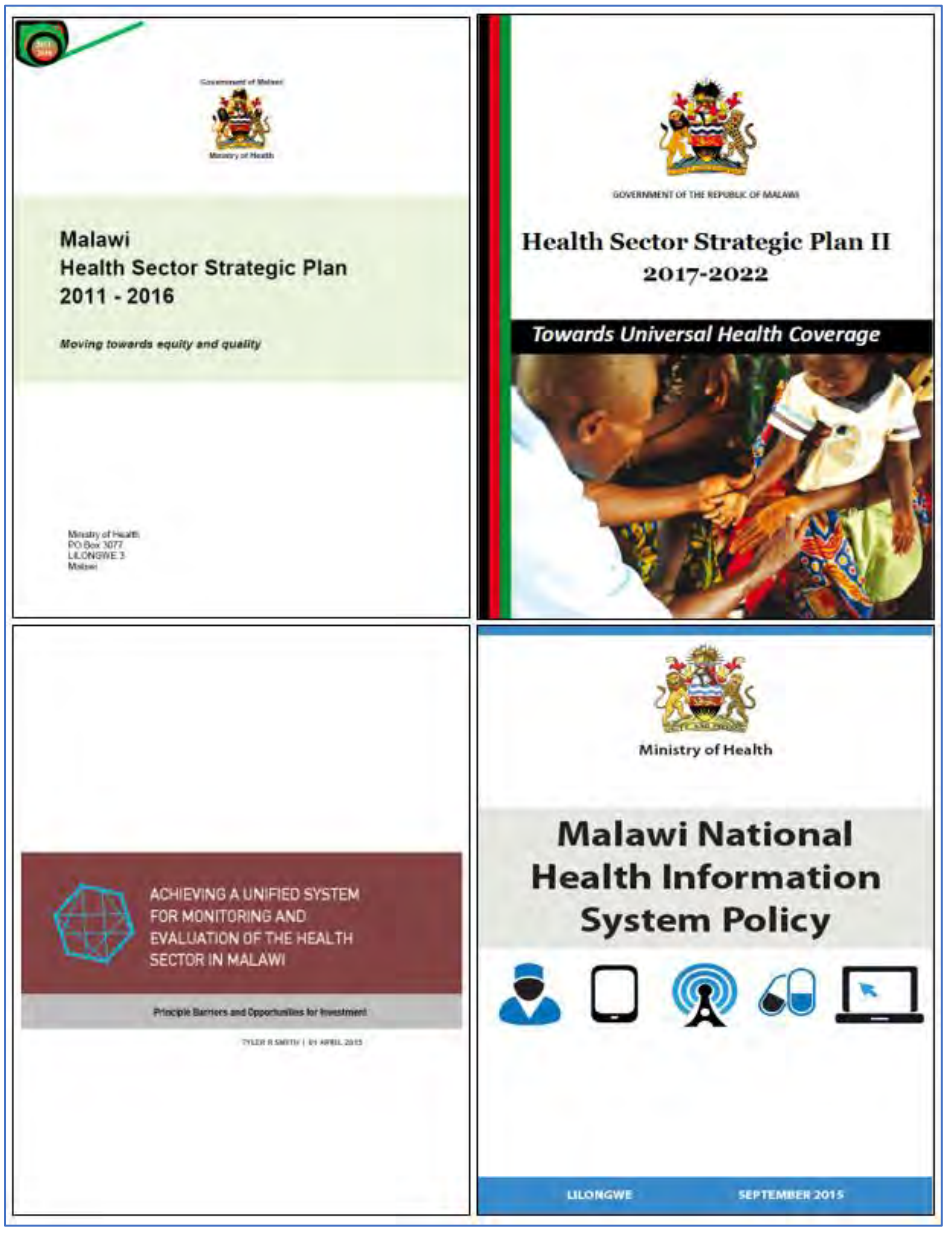


Figure 4.2 Snapshot of Sample Documents Reviewed

4.3.4 Web-based Data Sources

Further data was collected using web data sources such as mailing lists, emails and websites. In Malawi a mailing list bringing together various stakeholders was setup to act as a platform for discussion, and dissemination of information pertaining to DHIS2. Besides acting as a source for reports and a platform for discussions around DHIS2 in Malawi, the mailing list allowed me to maintained contact with the team in Malawi the times when I was in Oslo.

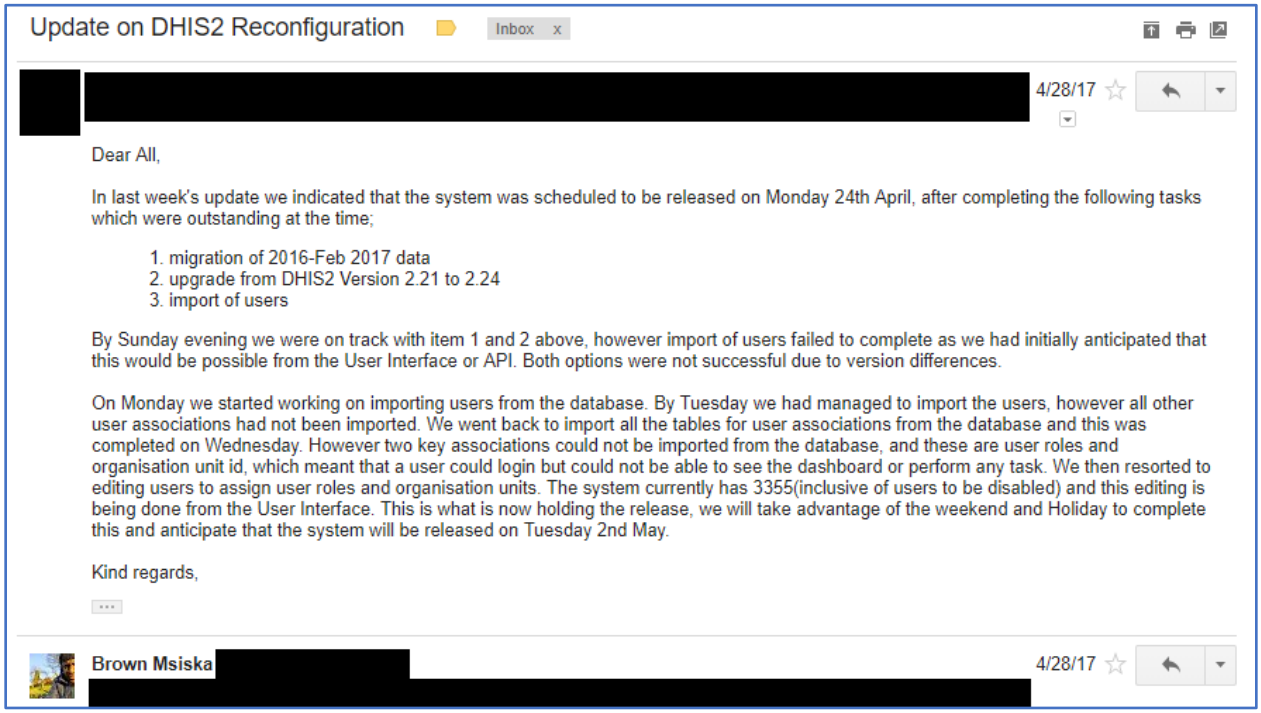


Figure 4.3 An extract from DHIS2 Malawi Mailing List

Further information was collected by visiting various pages on the DHIS2 website, HISP University of Oslo website, Malawi Ministry of Health Website and HISP Malawi website. Collectively, these websites provided a starting point in establishing a basis for understanding HISP and DHIS2 related activities in Malawi. For example, the Ministry of Health website was key in understanding the underlying context surrounding DHIS2 in Malawi. From the website, policy documents and reports related the health information system in Malawi were downloaded. The HISP Malawi website provided background information to the introduction and eventual roll out of DHIS2 in Malawi. It was also provided documentation related to trainings, for example the training of trainers, that were undertaken in Malawi to aid the national roll out of DHIS2. HISP UiO website provided the global context for the DHIS2 software platform.

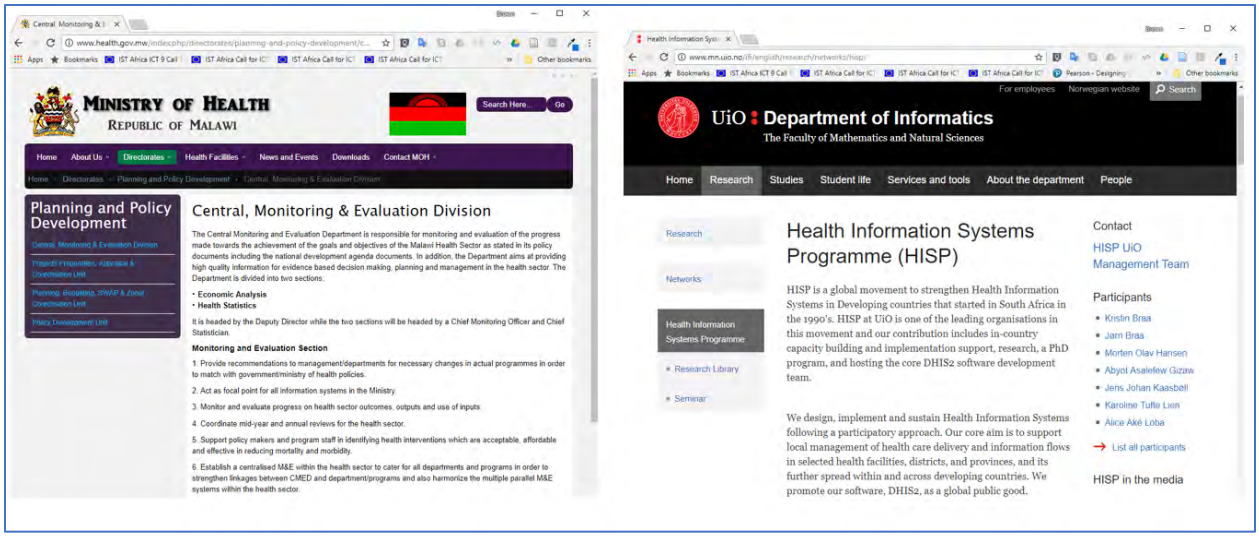


Figure 4.4 HISP UiO and Malawi Ministry of Health Websites

4.4 Data Analysis

Data analysis involved an iterative review of textual and audiovisual data collected throughout the period of the study. No qualitative software tools were used during the analysis – often paper, pencils and pens were the only tools used to aid the process. The process started with organizing and preparing of data collected for analysis. This involved creating transcripts from tape-recorded interviews and expanding field notes to capture the full essence of what was discussed or observed. This was usually done immediately after every field visit. Gaps in the data were noted and appropriate follow ups were made before further analysis was done.

The primary analytical technique used during the study was thematic analysis (Braun and Clarke, 2006; Maguire and Delahunt, 2017; Vaismoradi et al., 2013). This involved going through transcripts and field notes assigning codes to segments of data collected as means to identify patterns and themes that would form the basis for interpretation. *Codes* are labels (words or phrases) used to allocate units of meaning to segments of data (Creswell, 2009; Walliman, 2011). Coding was partly *in vivo* – using participants own words (Creswell, 2009) and partly guided by concepts in the conceptual framework used. Resulting codes were then grouped (reduced or categorised) into themes. *Themes* pull together coded information into smaller and meaningful groupings which allow the development of a more integrated understanding of the situation (Walliman, 2011). The outcome of this process was a corresponding coding structure from which

interpretations could be elicited. Visualisations, for example tables, timelines and diagrams, were created to help disseminate interpretations constructed or support further analysis.

Table 4.3 presents a brief illustration of the analytical process followed using the following extract from the data collected:

> *"...besides being able to use the system we need to get to a point whereby we have local staff that can deploy an instance of the system, customize it and keep the system up to date and secure ..."* [CMED Official]

Table 4.3 A Short Illustration of Thematic Analysis

Statement Chunk	**Code**	**Theme**
"… *besides being* ***able to use the system*** *..."*	able to use system	System Use
"…*we need to get to a point whereby we have local staff that* ***can deploy an instance of the system*** *...*"	able to deploy system	System Implementation
"…*we need to get to a point whereby we have local staff that ...* ***customize it*** *..."*	able to customize system	System Implementation
"…*we need to get to a point whereby we have local staff that can ...* ***keep the system up to date*** *...*"	able to upgrade system	System Maintenance
"…*we need to get to a point whereby we have local staff that can ...* ***keep the system … secure*** *...*"	able to maintain system security	System Implementation; System Maintenance

The analytical strategy described above was followed during the writing of this thesis as well as the four research papers published over the course of the study. In preparing these publications, feedback and ideas from co-authors, peers and other reviewers was also instrumental in the shaping the data analysis process.

4.5 Ethical Considerations

Issues of ethics are important in research especially in qualitative research which often relies on opinions and views expressed by respondents. Research ethics concerns the rights of research participants and the responsibilities of the researcher. The rights of research participants include: *the right not to participate*, *the right to withdraw*, *right to give informed consent*, *right to anonymity* and *right to confidentiality* (Oates, 2005). At the same time, researchers are expected

to: *act with integrity by recording data accurately and fully*, *ensure no unnecessary intrusion into research participants' activities*, and *ensure that they do not plagiarize by giving full credit to original authors for any borrowed ideas* (*ibid*.).

To ensure compliance with these ethical requirements an informed consent form was developed and issued to all prospective research participants. The informed consent form outlined what the study was all about and assured the participants of their rights to choose not to participate, to give informed consent, to withdraw, to anonymity and to confidentiality. Before consenting or declining to participate in the study prospective participants could seek clarification regarding any issue not adequately addressed by the informed consent form. In addition, consent was sought for any photographs and audio recordings taken in the study. Going further, care has been taken in the thesis and related publications in order not to violate the rights of participants to anonymity and to confidentiality. In addition, the publications provide detailed references to give full credit to original authors of any borrowed ideas that have been utilized and to allow the reader to find the corresponding published material.

4.6 Reflections on Research Methodology

The fact that the case study was carried out within the confines of an ongoing action research project, HISP, meant that my involvement in the field was both that of an outside observer and an involved researcher (Walsham, 1995). An outside observer collects data without participating directly in the phenomena under study whereas an inside (or involved) researcher participates in the situation under study to experience it from the perspective of other participants in the setting (Oates, 2005).

My engagement in various interventions, as described in section 4.3.2, offered me two significant advantages. First, I was able to appreciate first-hand the experiences of various actors involved in leveraging the DHIS2 software platform towards putting in place an integrated national health information system instead of relying merely on claims from research participants. Secondly, it allowed me easy access to the stakeholder organisations and the data required. At the same time, it afforded me an opportunity to "give back" by availing my expertise where required.

Where a researcher actively participates in the situation under study there is usually a likelihood of the researcher's actions influencing other participants and eventual outcomes as well as introducing bias in the data collected. However, in my case I was not directly involved in setting

the agenda in the various interventions I participated in. Therefore, my level of influence was largely limited towards the implementation end of agenda set by other stakeholders. At the same time, participant observation was not the only data collection method used. The triangulation of data collection methods allowed for the alleviation of elements of bias that could emanate from directly participating in the phenomena understudy.

5 Research Context and Case Description

This chapter presents the research context and case description in relation to efforts to leverage the DHIS2 software platform towards the implementation of the health management information system in Malawi. Section 5.1 provides a description of research setting, the country Malawi, to give a picture of the social context surrounding DHIS2 in the country. Section 5.2 presents a summary of the background and evolution of DHIS2. Lastly, section 5.3 describes the case including the background, timeline and description of activities related to the DHIS2 in Malawi.

5.1 Research Context

The research setting for this study is Malawi, a landlocked developing country located in southeast Africa, bordered by Zambia on the northwest, Tanzania on the northeast and Mozambique on the southeast and southwest (see figure 5.1 below). The country has an estimated population density of 139 persons per square kilometer resulting from an estimated total population of 16.3 million people over an area covering 11.85 million hectares of which 2.43 million hectares is covered by water bodies (National Statistics Office, 2016).

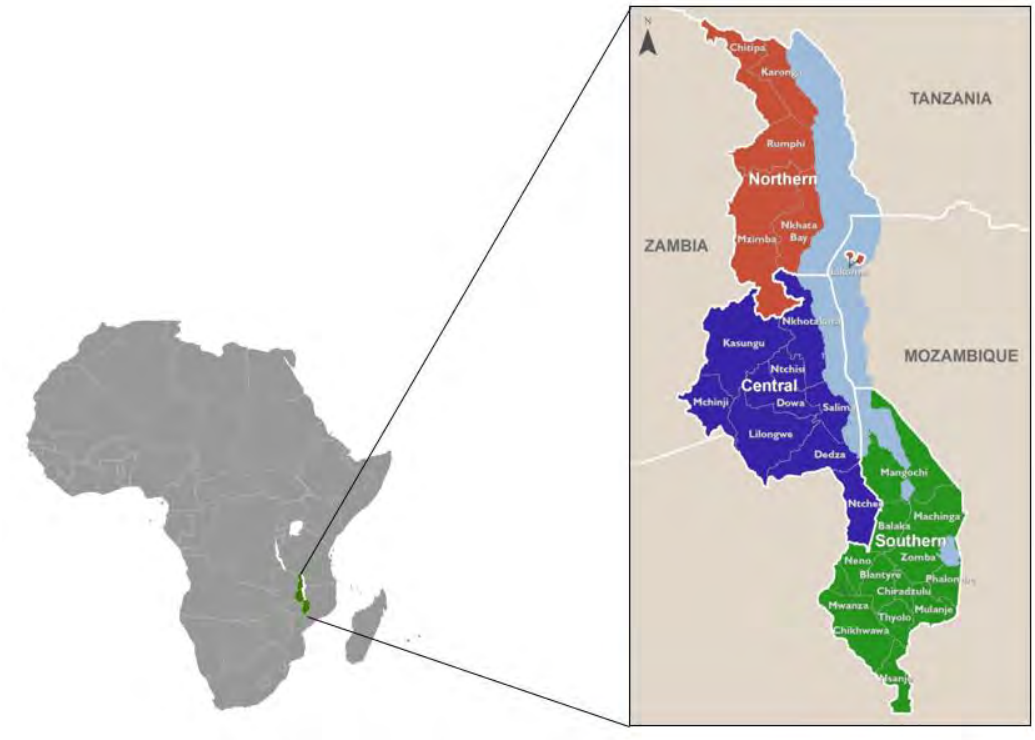


Figure 5.1 A Projection of Malawi from Africa Map

Malawi is divided into three administrative regions (or *provinces*): Northern Region, Central Region and Southern Region. The regions are further divided into a total of 28 districts: 13 districts in Southern Region, 9 districts in Central Region, and 6 districts in Northern Region. The administrative capital of Malawi is Lilongwe district located in the Central Region and it is here where you find the headquarters of the Malawi Government at a place called Capital Hill. Located at Capital Hill are head offices of the various ministries under the Malawi Government. Amongst these ministries is the Ministry of Health (MoH) - the major stakeholder in provision and management of health services in Malawi.

5.1.1 Ministry of Health in Malawi

The Ministry of Health is both a major provider as well as a regulator of health services in Malawi. As a health services provider, the ministry is responsible for the running of public health facilities which account for over 60 percent of the total number of health facilities. Public health facilities in Malawi are organized into a three-tier referral structure: *tertiary* facilities, *secondary* facilities and *primary* facilities. A health condition cannot be addressed at a primary health facility is referred to a secondary level health facility; and if the condition cannot be handled at a secondary level facility the patient is referred to a tertiary level facility. The tertiary level comprises of central hospitals, the secondary level comprises of district hospitals, and the primary level comprises of rural hospitals and other smaller health facility deployments.

As a regulator of health services in Malawi, the Ministry of health is responsible for developing, reviewing and enforcing health and related policies for the health sector; spearheading sector reforms; developing and reviewing standards, norms and management protocols for service delivery and ensuring that these are communicated to lower level institutions; planning and mobilizing health resources for the health sector including allocation and management; advising other ministries, departments and agencies on health related issues; coordinating research; and monitoring and evaluation of health related services and programs (Ministry of Health, 2018).

The ministry has fourteen directorates through which it executes its mandate: administration, finance, human resources, clinical services, nursing and midwifery services, preventative health services, public health, health technical support services, planning and policy development, health research, reproductive health, safe motherhood, nutrition, and quality management. Under the planning and policy development directorate the ministry has the Central Monitoring and

Evaluation Division (CMED) which is tasked with the responsibility of monitoring and evaluating progress made towards the achievement of the goals and objectives of the Malawian Health Sector (Ministry of Health, 2018). CMED is the major stakeholder with respect to implementation and use of DHIS2 in Malawi.

5.1.2 Health Information Systems in Malawi

The health information system landscape in Malawi is characterized by a multiplicity of information systems. Various information systems have been put in place to aid the regulatory and service-provision roles served by the Ministry of Health. At the top of information system hierarchy is the national health management information system (HMIS) which is supposed to provide an integrated (unified) view of the health sector in Malawi to facilitate monitoring and evaluation activities, and evidence-based decision making. Under the national HMIS are various subsystems with varying levels of automation. The subsystems include: health services information system (HSIS), logistics management information system (LMIS), human resource management information system (HRMIS), financial management information system (FMIS), integrated disease surveillance (IDS), and physical assets management information system (PAMIS).

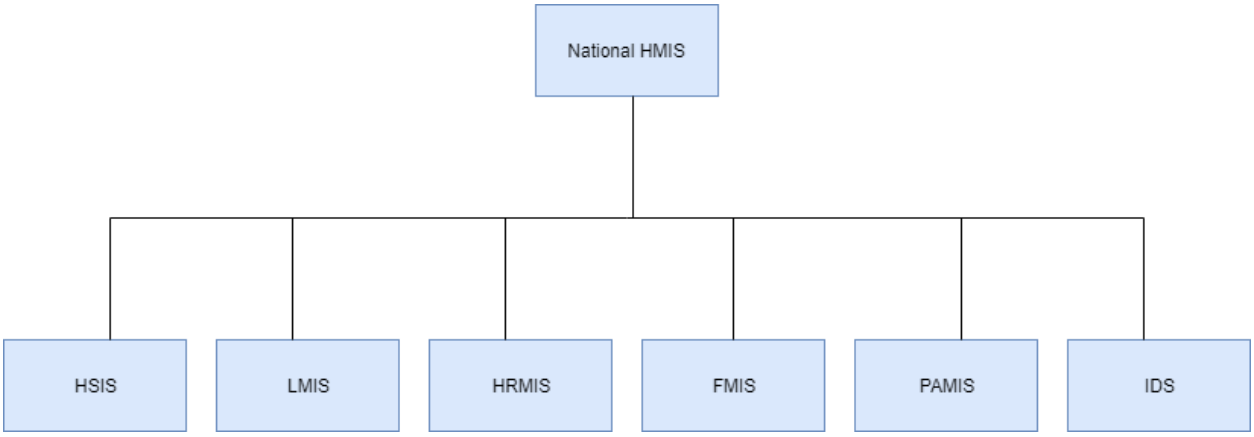


Figure 5.2 Overview of the Health Information System Landscape in Malawi

The HSIS subsystem comprises of point of care information systems and program specific management information systems. Depending on their data needs, some health programs, for example the National TB program and the HIV/AIDS program, run inhouse management information systems. Point of care information systems consist of a blend that includes a paper-based patient records system and electronic patient record (EMR) systems. There are two dominant

EMRs in Malawi: Baobab EMR developed by, *Baobab Health Trust*, a local non-governmental organisation, and the Drug Resources Enhancement against AIDS and Malnutrition Electronic Health Records (DREAM EHR) developed by an Italian non-governmental organisation, *Community of Sant'Egidio*. Of these two, the Baobab EMR is the most widely used with deployments in over 70 health facilities.

DHIS2 in Malawi sits at the top of the hierarchy presented in figure 5.2 as the backbone for the national HMIS. Because of its position in the health information system hierarchy, "*besides serving as an M&E system it also serves as a national health data warehouse, a one stop shop, facilitating all kinds of decisions by different stakeholders in the health sector*" (CMED, Official). Thus, interest in DHIS2 extends beyond the confines of CMED. The health information system in Malawi comes from a fragmented background and DHIS2 is seen as a vehicle towards achieving an integrated national HMIS (Smith, 2015).

Beside the Ministry of Health itself, there are multiple local and international stakeholders playing different roles around the health information system setup in Malawi. The stakeholders include local and international end-users of the information generated by the health information systems, and local and international organisations providing technical and financial assistance towards the implementation and maintenance of various information systems. The implementation of DHIS2 in Malawi involves several local and international stakeholders in terms of technical and financial assistance (see figure 5.3 and table 5.1).

Figure 5.3 Cropped Screenshot of Key Financial Partners for DHIS2 in Malawi

Table 5.1 Key Technical Assistance Stakeholders for DHIS2 in Malawi

Organisation	Role
HISP Malawi	Technical assistance to CMED on DHIS2
University of Malawi	Member of HISP Malawi, Hosted DHIS2 between 2009 and 2017/Member of HISP Malawi; Technical assistance to CMED on DHIS2
Baobab Health Trust	Deployment and Maintenance of Baobab EMRs, South and Central Regions; Integration with DHIS2
Luke International Norway	Deployment and Maintenance of Baobab EMRs, Northern Region; Integration with DHIS2
International Training and Education Center for Health (I-TECH)	Technical assistance to CMED on DHIS2 and Department of HIV/AIDS Management Information System (DHA MIS); Server Management
HISP University of Oslo	Development of DHIS2, Technical assistance to CMED/HISP Malawi on DHIS2

5.2 The DHIS2 Software Platform – Background and Evolution

This subsection provides a description of DHIS2, from its origin to becoming an open source software platform. DHIS2 is an open source health information software platform developed by HISP University of Oslo in Norway. It is primarily used by ministries of health and other stakeholders in developing countries to implement health information systems that facilitate collection, management, aggregation, analysis and visualization of routine data for evidence-based decision making. In addition, DHIS2 has a tracker component, often referred to as *DHIS2 Tracker*, which can be used to capture data specific to an entity instance, for example a patient, within a given health programme. Although DHIS2 is primarily a health information software platform, it has demonstrated capacity for leverage beyond the health sector and is increasingly being used in other domains such as agriculture and education. DHIS2 is a free and open source web-based software distributed under the BSD (Berkeley Software Distribution) license which gives end users the right to modify and redistribute the software. In addition, DHIS2 is now a software platform which comes with a web API that allows third-party developers to build applications on top of it.

5.2.1 The Origin of DHIS2

The origin of DHIS2 can be traced to the Reconstruction and Development program instituted in 1995 in post-apartheid South Africa under which HISP was first conceived and initiated. One of

the aims of HISP in South Africa was creating a unified health information system across the country. As a vehicle towards fulfilling this aim, a pilot project to develop a district-based health information system was proposed in the Western Cape province. The pilot project received financial backing from the Norwegian Agency for Development Cooperation (NORAD) and resulted in version 1 of District Health Information Software (DHIS1) released as a prototype in 1998.

Following successful implementations in the Western Cape province, DHIS1 was adopted by the Eastern Cape province in the same year. The number of provinces in South Africa using DHIS1 had significantly grew by the year 2000, prompting interest from other developing countries. As a result, DHIS1 was introduced in other developing countries including Mozambique, Malawi and India. DHIS1 was a desktop application built using Microsoft Access as the underlying database management system. The use of Microsoft Access necessitated payment of Microsoft Office license fees for every workstation on which DHIS1 was installed.

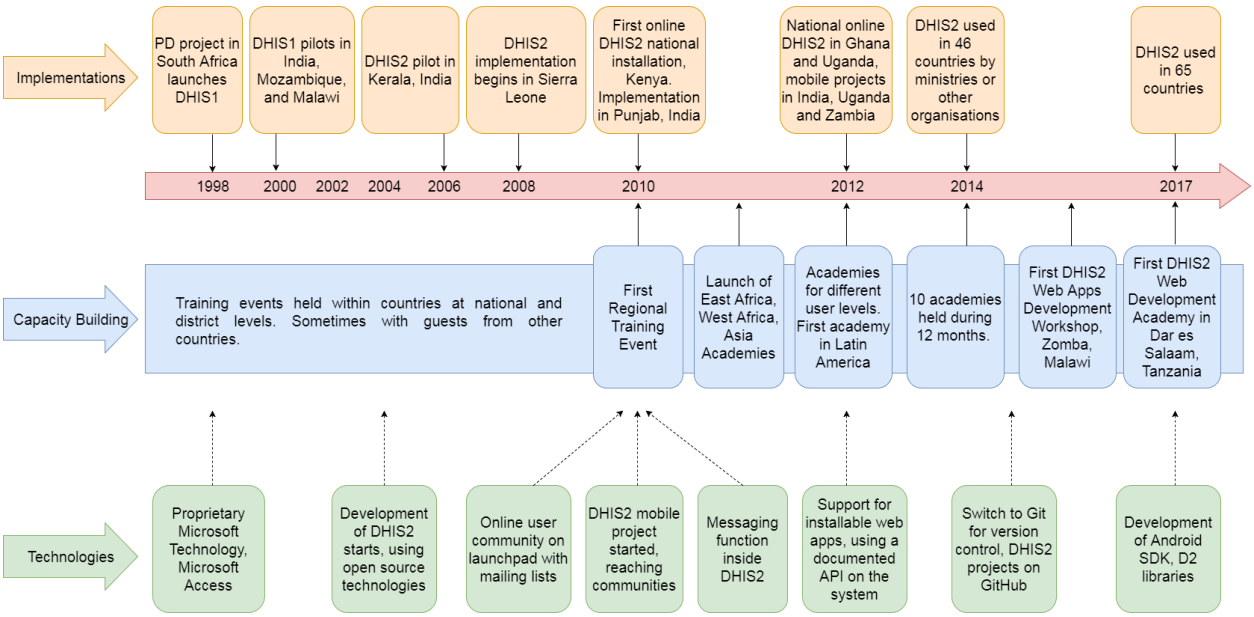


Figure 5.4 Timeline for DHIS2 (adapted from a slide in the DHIS2 Online Academy)

5.2.2 Becoming an Open Source and Web-based Software

Adopters of DHIS1 faced two key challenges. First, the use of Microsoft Access as the underlying database management system made scaling the solution costly as rolling out entailed acquiring Microsoft Office licenses for workstations running DHIS1. Furthermore, being a desktop

application supporting multiple deployments of DHIS1 required travel and other incidental expenses for support personnel. For developing countries, the cost of implementation and supporting DHIS1 deployments, made the solution unattractive and unsustainable. Second, being a desktop application DHIS1 lacked support for sharing data between geographically distant stakeholders.

In response to these challenges, HISP UiO, with financial support from NORAD and various international stakeholders embarked on a project to develop an open source and web-based version of the software. Adopting open source technologies would obviate the need to pay license fees for each deployment of the software. On the other hand, making the software web-based would obviate the need to support multiple installations of the software and provide support for sharing data between geographically distant stakeholders. The project commenced in 2004 and culminated in the development and release of District Health Information Software version 2 (DHIS2) in 2006. Since 2006 several versions of DHIS2 have been released. Currently, there are three DHIS2 versions released each year.

5.2.3 Platformisation of DHIS2

With the scalability challenges that worked against DHIS1 obviated, the number of ministries of health and non-governmental organisations leveraging DHIS2 to implement health information systems in developing countries has significantly risen over the past decade. By mid-2017, DHIS2 was in use by ministries of health in more than 60 countries. This ushered in some new challenges. First, the growth in size of the user community meant an increase in number of user requirements for DHIS2. At the same time, differences between the various end-user contexts where DHIS2 was being used meant that the user requirements were often very different and diverging.

For HISP UiO, prioritizing and addressing a growing number of divergent end user requirements became a challenge. Subsequently, there have been complaints and queries from members of the DHIS2 end user community regarding how user requirements are prioritized and addressed. The prioritization and addressing of end user requirements has, for example, been one of the key topics of debate at the annual DHIS2 experts academy held by HISP UiO. Sharing the software development burden with developers within the end-user community is part of the remedy and since DHIS2 is open source, ideally, the burden of addressing end user requirements doesn't rest on HISP UiO alone.

However, as the case is with all open source software, DHIS2 faces the challenge of *forkability* – the inherent risk for emergence of multiple and incompatible versions (*forks*) of a software product. Furthermore, the core of DHIS2 was developed using Java and a stack of related software development frameworks. For some external developers, developing for DHIS2 meant learning a new programming language as well as a range of associated technologies and therefore not as attractive. To effectively and timely address growing and divergent end user requirements there was a need to devise a way to allow external contributions without requiring external developers master Java and at the same time circumvent forkability of the software. This has resulted in DHIS2 evolving from a mere open source application to now an open source software platform.

DHIS2, as a software platform, provides a RESTful web API as a boundary resource to enable third-party developers create applications on top of it. The API has undergone a series of revisions with the aim of offering better support to third-party developers. A newer version of the API is bundled into the release of each new version of DHIS2. The provision of the API eliminates the need to learn Java and a stack of associated frameworks to develop applications for DHIS2. Developers are free to use whatever programming language they are familiar with as long as it has features to pull and push data across a web API. Most modern programming languages have such features. Applications built on top of DHIS2 can exist outside the platform or installed on an instance of the platform. Installable DHIS2 applications are written using JavaScript, CSS and HTML5. This lowers the learning curve for developing DHIS2 applications.

On top of the web API, HISP UiO is also developing auxiliary boundary resources to further lessen the effort required to develop various kinds of DHIS2 applications. The main auxiliary boundary resource currently available is the d2 library, a JavaScript library, which provides a level of abstraction above the API and allows third-party developers to develop applications without requiring in depth knowledge of the API. In addition, there are efforts to build a DHIS2 Android SDK, currently in beta, as an abstraction layer for building DHIS2 Android applications. Dedicated DHIS2 app stores have been created: one for distribution of web applications and another for distribution of Android applications.

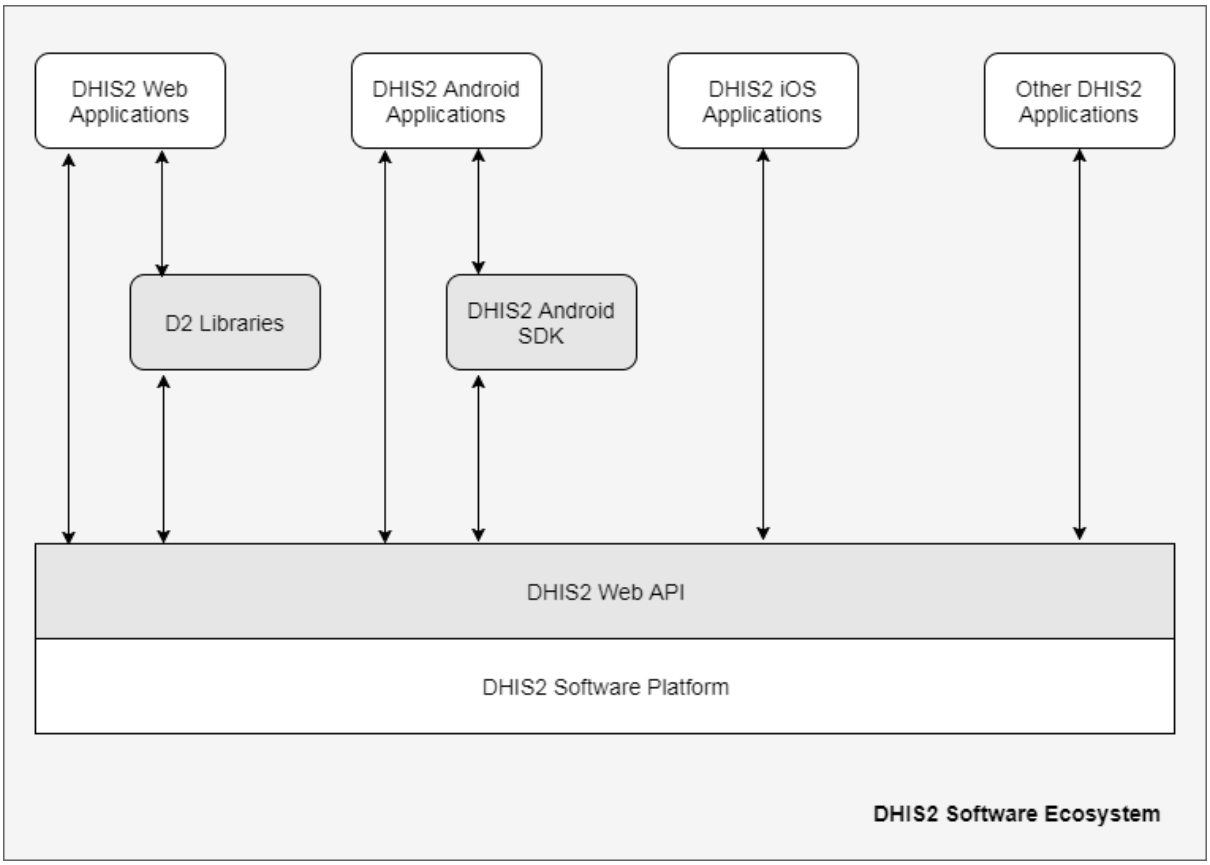


Figure 5.5 An Overview of the DHIS2 Software Ecosystem

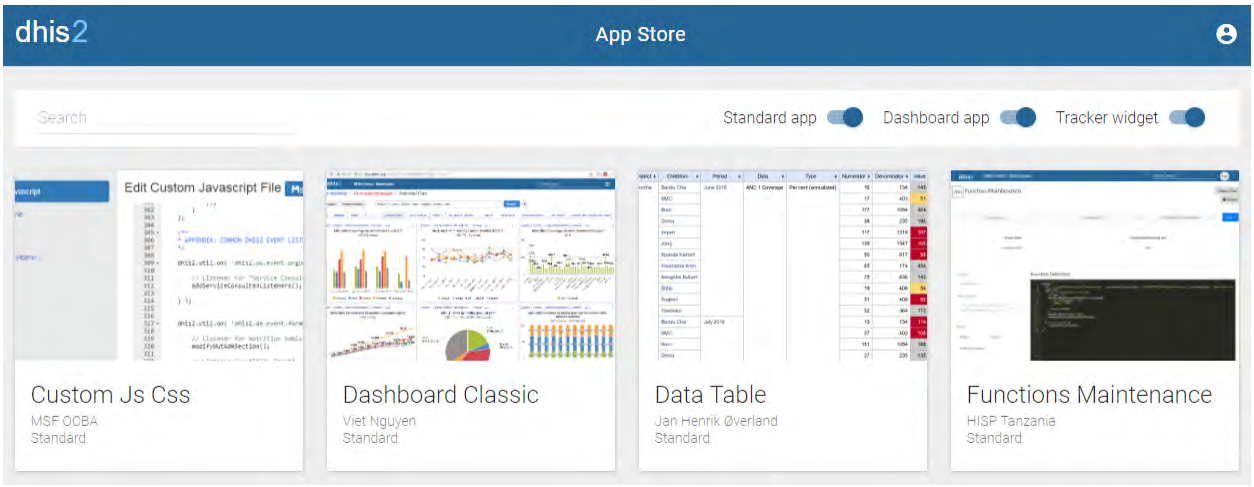


Figure 5.6 Cropped Screenshot of the DHIS2 Web App Store

5.3 DHIS2 in Malawi

Efforts to leverage DHIS2 in Malawi fall under the health management information system strengthening program which commenced in 1999. A situation analysis of the health information

system landscape in Malawi was carried out to identify weaknesses and strengths and map strategies towards improvement. The situation analysis found out that the health information system landscape in Malawi was heavily fragmented. Several information systems specific to vertical health programs existed resulting in lack of reliable information for health services planning and management. Due to the fragmentation sharing of data across programs and geographical boundaries was difficult which resulted in duplication of data collection efforts. This in turn affected the quality of data as different information systems often had conflicting versions of the same data. Limited data quality negatively affected its usefulness and subsequent use for decision making.

Consequently, an integrated national health management information system was proposed. With funding from the Dutch Government, a review of the health management information system was carried out between 1999 and 2002 which culminated in the adoption and implementation of DHIS1, a desktop-based antecedent of DHIS2 based on Microsoft Access, in January 2002. With DHIS1 in place, stakeholders in Malawi were confronted with similar challenges to other adopters as described in section 5.2.2. *"Sharing data to geographically distant stakeholders was still a challenge*" (CMED Official) and as a result fragmentation persisted. Rolling out and supporting DHIS1 deployments across the country also proved costly thereby negatively affecting its attractiveness and sustainability. The challenges experienced in Malawi and other adopters of DHIS1 resulted in the development and subsequent release of DHIS2 by HISP UiO in 2005.

5.3.1 Implementation of DHIS2 in Malawi

Efforts to replace DHIS1 with DHIS2 in Malawi commenced in 2009 as a pilot project in three districts. This involved CMED as a major stakeholder under the Ministry of Health, HISP UiO and the University of Malawi represented by its constituent colleges: College of Medicine, Chancellor College and Polytechnic. The staff establishment in CMED comprises of economists and statisticians and no ICT personnel. At the same time, the focus of the ICT department within the Ministry of Health was largely on keeping the ICT equipment running and supporting the processing of salaries rather than the health information system needs of the ministry. Therefore, on its own, CMED did not have the requisite technical capacity to implement and rollout DHIS2. This necessitated the establishment of a strategic relationship with HISP UiO and University of Malawi to mitigate the capacity gaps in CMED. Drawing its members from CMED and the

University of Malawi, a local node of HISP, HISP Malawi, was established to provide technical assistance to CMED towards the implementation and maintenance of DHIS2.

Being a web-based application, DHIS2 required a stable hosting space with sufficient internet bandwidth to handle data traffic from prospective users. For this purpose, the Government Wide Area Network (GWAN) through which CMED gets its internet connectivity was found lacking as it was deemed *"... very slow ... inadequate and unstable to run DHIS2*" (DHIS2 Technical Assistant). Comparatively, the University of Malawi had better internet connectivity and infrastructure. Furthermore, the Ministry of Health was not in favor of hosting DHIS2 on the cloud. Consequently, a decision was made to deploy and host DHIS2 at University of Malawi's College of Medicine.

Funding for the pilot project was secured through the University of Oslo following which two members of staff were recruited under HISP Malawi and seconded to CMED as DHIS2 technical assistants. In addition, a Chief DHIS2 Technical Assistant was recruited under I-TECH and seconded to CMED as well. After three years of intermittent piloting, DHIS2, still hosted by the College of Medicine, was rolled out to all the 28 districts in 2012 with support from NORAD, United Nations Children's Fund (UNICEF), I-TECH, Center for Disease Control and Prevention (CDC), HISP UiO, HISP Malawi and University of Malawi. With DHIS2 rolled out countrywide aggregated health data is transmitted electronically from the district health office (DHO) to the national level. Data collection from health facilities to the DHO is through paper-based forms completed at the health facilities and sent to the DHO where the data is entered DHIS2. Where possible data entry from health facilities can also be done through the DHIS2 mobile (DHIS2m) interface.

5.3.2 Capacity Requirements and Capacity Building

In relation to DHIS2, pre-implementation and post-implementation human capacity requirements can be identified. In Malawi, a number of strategies have been and are being employed to meet the various human capacity requirements that come with DHIS2. Prior to the piloting of DHIS2 in Malawi, some staffs from University of Malawi and the Ministry of Health were enrolled into the master's in informatics and PhD programs at the University of Oslo, between 2005 and 2007, bringing them into the HISP fold and offering them a first exposure to DHIS2. By 2009, all the master's students had returned to Malawi following their graduation and the pilot project

commenced with them working as part-time DHIS2 technical assistants. The chief DHIS2 technical assistant and the two full-time DHIS2 technical assistants, recruited under HISP Malawi, underwent a series of trainings in and outside Malawi to enable them carry out various customization and configuration tasks with respect to DHIS2. Implementing DHIS2 involved setting up a webserver, installing DHIS2 on the web server and defining metadata. In addition, custom forms and reports were created to match corresponding paper versions as a way of preserving familiarity and ensuring a smooth transition from paper based to electronic data entry.

Besides training targeting technical personnel, a series of end user training workshops were carried out during the pilot phase as well as after the roll out. The aim of these workshops was to create requisite capacity for all prospective end users in all 28 districts to enable them use DHIS2. As a strategy, "*end user training employed a cascade approach*" (Director, CMED). To kick start these workshops, a training of trainers (TOT) workshop was carried out in August 2012. Trainers of trainers (TOTs) identified at the national level were the first to be trained on DHIS2. The TOTs, in turn, trained other trainers at the district level who later undertook training of other end users in their respective districts. However, due to budgetary constraints among other things, end user training has not done regularly. As a result, there is a backlog of untrained staff. This constrains their capacity to use DHIS2. Furthermore, because of changes introduced in subsequent versions of DHIS2 a good number of end users trained earlier need re-training.

Globally, the HISP community uses DHIS2 academies as a vehicle for building various DHIS2 related capacities at regional and national level. Various members of the technical team in Malawi have attended several regional academies. These academies afforded them an opportunity to catch up with and take advantage of various advances in DHIS2. Equally, there have been academies targeting end users but Malawi's participation in some academies has been limited due to logistical constraints. As a result, the Ministry of Health in collaboration with HISP Malawi has at times organized local workshops aimed at building end user capacity. A good example in this regard, are nationwide DHIS2M trainings conducted to bring end user up to speed with mobile-based data entry into DHIS2.

DHIS2 requires a stack of open source software tools as part of its running environment. The stack includes a Java Runtime Environment (JRE), a Java servlet container which provides an HTTP (or web-based) web server environment on which to run Java code, and a database management

system. A typical DHIS2 instance uses the JRE distributed by Oracle, Tomcat – an open source servlet release by the Apache Foundation, and PostgreSQL as the database management system. While it is possible to deploy DHIS2 on a Windows server environment most DHIS2 deployments make use Linux, typically Ubuntu, as the underlying operating system for the server environment. This necessitates the existence of system administration capacity to configure and manage the underlying tools in the DHIS2 environment. Newer versions of DHIS2 often require newer versions of these tools. So, transition to a newer version of DHIS2 often demands updating the underlying software tools as well. If not done properly such updates can lead to unintended results, as exemplified by "*the corruption of database for a DHIS2 tracker pilot in 2014*" (DHIS2 Technical Assistant). As a result, system administration capacity is a critical component for continued leverage of DHIS2. With the absence of ICT personnel in CMED, system administration tasks have largely been carried out by personnel from HISP Malawi or University of Malawi and at times external experts, including PhD and master's students, engaged by HISP UiO.

Increased use DHIS2, following the national roll out, enabled stakeholders to identify gaps and new requirements for the software. Some of the emerging requirements required developing third party applications or modifying some aspect of the DHIS2 core. Responding to such requirements demanded software development capacity in relation to DHIS2. Over the years, such requirements have usually been reported to HISP UiO for possible implementation by DHIS2 core developers. Sometimes these requirements have been promptly addressed whereas on other occasions such has not been the case. To ensure promptness in handling some of these requirements, it became imperative to share the software development burden with HISP UiO instead of leaving everything up to DHIS2 core developers. This necessitated building local capacity for third-party application development on DHIS2.

Consequently, with support from HISP UiO and UNICEF, University of Malawi and HISP Malawi organized a couple of DHIS2 third-party application development workshops – a DHIS2 web applications workshop in March 2016 and a DHIS2 Android applications development workshop in October 2017. Within Malawi, the workshops attracted participants from the Ministry of Health ICT department, HISP Malawi, University of Malawi, Baobab Health Trust and Luke International Norway. In addition to Malawian participants, there were participants from Zambia, Kenya, Ethiopia and Mozambique. Subsequently, a similar web applications development workshop was

held in Kampala, Uganda in January 2017 and a fully-fledged DHIS2 web applications development academy was held in Dar es Salaam, Tanzania in November 2017.

Recognizing the various capacity needs around DHIS2, HISP Malawi is undergoing a restructuring process, establishing different directorates which will be responsible for different tasks that leveraging DHIS2 necessitates. Five directorates have been proposed: software development directorate, training and capacity development directorate, infrastructure management and support directorate, monitoring and evaluation directorate, and research directorate. Figure 5.7 shows the proposed organogram for restructured HISP Malawi. Currently, HISP Malawi simply has a board of directors and one or more full-time DHIS2 technical assistants.

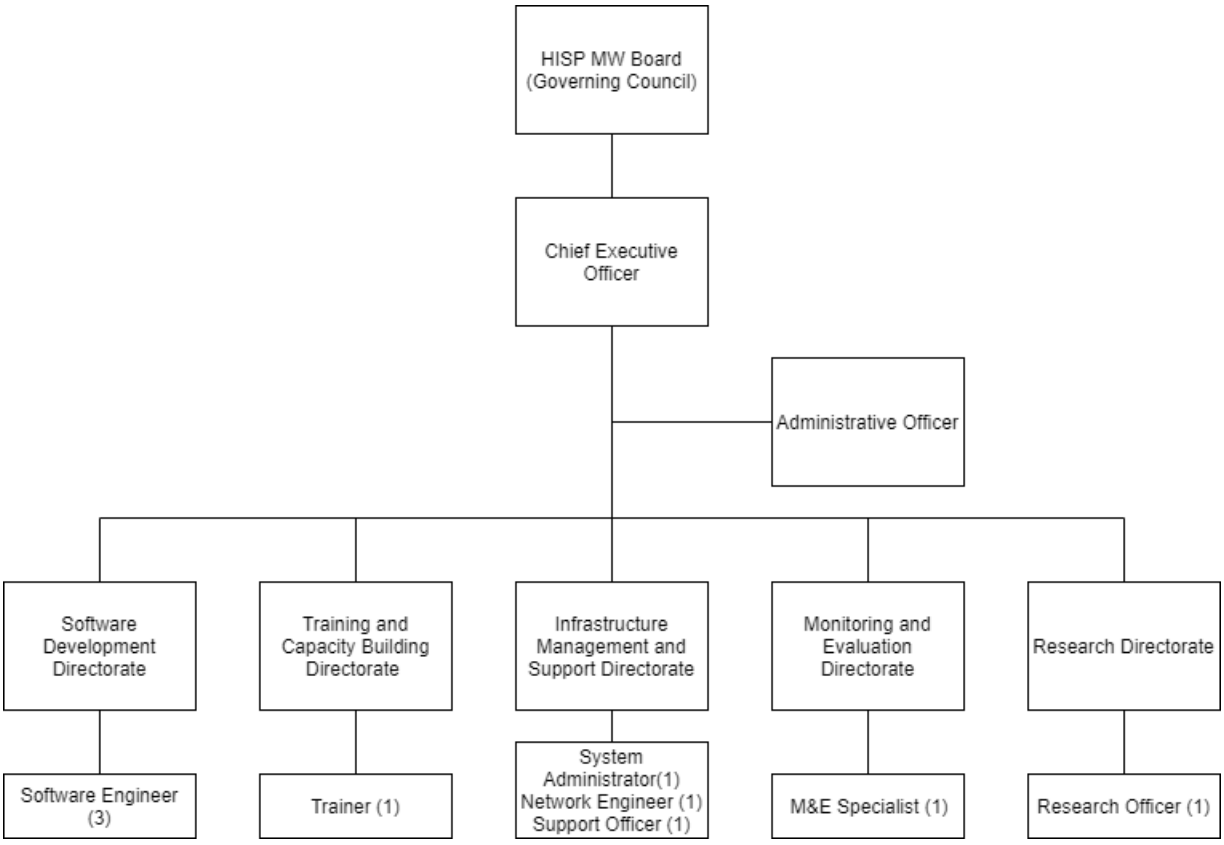


Figure 5.7 HISP Malawi Proposed Organogram

5.3.3 DHIS2 Reconfiguration and Data Migration

In 2015 a decision was made to migrate the DHIS2 instance in Malawi from College of Medicine (CoM) to the MoH server room located at the Department of HIV/AIDS (DHA). This was partly a result of tensions between HISP Malawi, CMED and other stakeholders with respect to control

and ownership of the DHIS2 instance. Hosting the instance within the ministry would therefore help establish a clear position regarding the ownership of the DHIS2 instance and the data within. In addition, a memorandum of understanding articulating the terms of agreement between HISP Malawi and the MoH was drafted to formalize and add transparency to matters of control, ownership and access to DHIS2. As part of the migration, there was also a requirement to reconfigure DHIS2 to iron out problems in the instance, such as data duplication across datasets, and take advantage of new configuration features available in the latest version of DHIS2.

The DHIS2 reconfiguration and data migration project was carried by a pool of personnel drawn from CMED/HISP Malawi, MoH ICT department, Baobab Health Trust, GIZ/EPOS Health Management and University of Malawi in consultation with MoH health program coordinators with support from HISP UiO, the Global Fund and UNICEF. Prior to the project, there was no collaboration between CMED and MoH ICT department on DHIS2. To remedy the situation, consultation was made with head of the ICT department and two members of staff from the department were assigned to work on the project. To kick start the project, standard working procedures were drafted to guide the collaboration between stakeholders and define permission and control structures around the new DHIS2 instance. HISP Malawi ceded administrative rights to the instance at CoM to the reconfiguration and migration team to facilitate data migration.

DHIS2 reconfiguration commenced with the deployment of a DHIS2 instance on the CMED server at the DHA in April 2016. This was followed by the redefinition of data elements to take advantage of new configuration features in DHIS2. An example in this respect is the use of the category combinations feature to define the data elements on the new instance as compared to flat data elements in the CoM instance. By leveraging the category combination feature one data element on the new instance would map to several data elements in the CoM instance. This reduced the data element definition effort but made the two instances structurally different even though they were semantically equivalent. So, data migration could not be carried out by simply copying over the database from the CoM instance.

During the project, consultations were made with program coordinators to gain clarity on their datasets and corresponding data elements. Further to that, joint meetings were held with respective program coordinators whenever data duplication across their programs was identified. For example, there were cases of data duplication between the TB program and the HIV/AIDS program

with respect to coinfection data. A meeting involving the two program coordinators resolved that the data be collected by the TB program only and reused by the HIV/AIDS program. Resolving the cases of data duplication further reduced the data element definition effort.

Because of structural differences between data element definitions on the new instance and those on the CoM instance, there was need to map the data elements across the two instances before data could be migrated. The mapping involved extracting 3 corresponding IDs per data element: *data element ID*, *category combo ID* and *attribute combo ID*. Therefore, mapping a pair of data elements across the two instances required extracting a total of 6 IDs. To facilitate the mapping, the reconfiguration and data migration team developed a web application called *dataset details lister* which created a table listing all data elements in a selected dataset alongside their corresponding IDs. This information was then copied into a spreadsheet mapping corresponding data elements across the two instances and saved as a comma separated values (CSV) file. The CSV file then became input into two other applications, *data migrator* and *data migration validator*, developed by the team to migrate data and validate data migrated between the two instances.

An agile-like approach was used to prioritize, configure and migrate different health programs in different timeboxes. Summing up the timeboxes, the entire project was scheduled to run up to August 2017, with the first set of reconfigured and migrated programs delivered in June 2016. At this point, the health programs that were in the first timebox were expected to make a switch to the new instance but the switch was pended because of two issues. First, for the convenience of end users the IP-address-based URL of the instance (*http://<<ipaddress>>/dhis/*) was supposed be mapped to a subdomain of the MoH website, *dhis2.health.gov.mw*. Second, the new instance shared internet connectivity with other applications hosted in the server room at the DHA. So, there was need to upgrade the internet bandwidth to handle the extra data traffic. The issues were tabled at a stakeholders meeting, the MoH ICT department commenced arrangements for URL mapping and a cost-sharing agreement was made for the upgrading of the internet bandwidth. The reconfiguration and data migration was completed four months ahead of schedule, in April 2017 and programs switched to the new instance in May 2017.

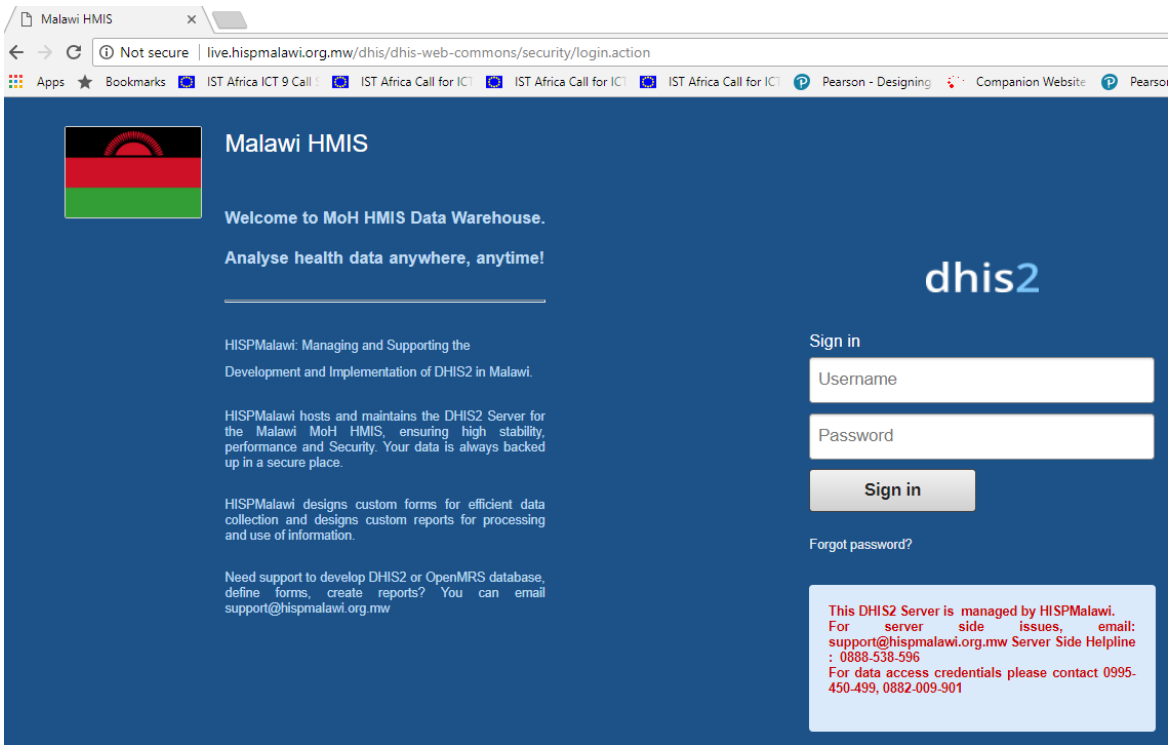


Figure 5.8 Cropped Screenshot of the DHIS2 Instance at CoM

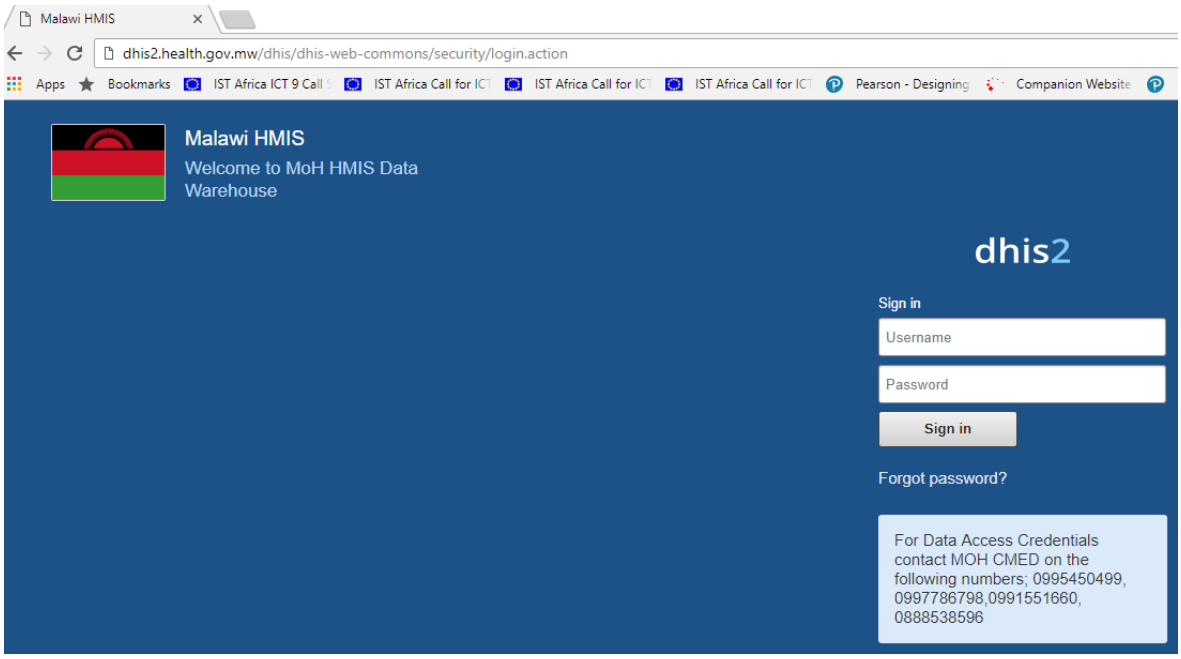


Figure 5.9 Cropped Screenshot of the DHIS2 Instance at DHA

5.3.4 Integration

DHIS2 in Malawi sits atop an HIS landscape embattled with fragmentation, dating back as far as 1999 as reported by Chaulagai et al. (2005). Besides serving as a monitoring and evaluation tool, DHIS2 in Malawi is regarded as a point of integration – a data warehouse bringing together data from disparate data sources to facilitate decision making for a multiplicity of stakeholders. The quest for integration is driven by three key objectives: to facilitate sharing data across geographically distributed stakeholders, to automate the transfer of data from auxiliary systems to DHIS2 and to eliminate cases of conflicting data arising from data collection overlaps between two or more data sources.

Around DHIS2 are several auxiliary information systems that are either paper-based or desktop applications with no internet connection or web-based applications. Such systems include, among others, a human resource management information system based on iHRIS, a logistics management information system based on a desktop application supply chain manager (SCMgr) scheduled to be replaced by openLMIS, DHA MIS, Baobab and DREAM EMRs, and Epi Info used by the TB program. Data in these systems is manually aggregated then sent for entry into DHIS2. As a result, data is entered twice: onto paper-based forms then later into DHIS2. Integration, where possible, would streamline the data entry efforts. Thus, the call for integrating DHIS2 with some of the auxiliary system exists.

However, in the absence of personnel to actively pursue integration efforts, the situation has not changed that much (Smith, 2015). To address this anomaly, in the last quarter of 2015, a DHIS2 technical assistant was recruited by HISP Malawi and seconded to CMED to drive collaborative efforts with other stakeholders on integration. Applications or scripts built to facilitate integration need to leverage the DHIS2 Web API. Thus, the application development workshops conducted in Malawi, by propagating knowledge of the API to participants from various stakeholders, are part and parcel of efforts towards achieving integration.

One of the stakeholders that has been actively pursuing integration with DHIS2 is the Baobab Health Trust. Besides the Baobab EMR, Baobab Health Trust has, in collaboration with the National Registration Bureau (NRB) and MoH, been piloting an electronic birth registration (EBR) system that allows babies to be registered at birth. In health facilities where Baobab EMR and EBR are being used data is first electronically captured into the systems, and later manually aggregated

for entry into DHIS2. Integrating Baobab systems with DHIS2 can free up staff time at the facilities to focus on data validation and quality rather than manual data entry, and would further reduce costs related to printing and transferring paper forms between sites (Smith, 2015). However, integration efforts by Baobab stalled because of failure "*to secure appropriate access rights to DHIS2 from HISP Malawi to allow us to test our solution*" (Staff, Baobab Health Trust). With the migration of DHIS2 to DHA, these efforts have been revived and Baobab Health Trust deployed two programmers to be part of the reconfiguration and data migration team. In the new standard working procedures, "*to ensure fairness in granting access rights to DHIS2 collaborating partners, HISP Malawi surrendered administrative rights to a panel of three people, the Director of CMED, Director MoH ICT department and the chief DHIS2 technical assistant*." (Team Representative, DHIS2 Reconfiguration and Data Migration Review Meeting).

Serendipitously, the applications developed for data migration during the DHIS2 reconfiguration and migration project served as a model for integration between DHIS2 and some of its auxiliary systems in Malawi. First, they serve to demonstrate the mapping required to transfer data between a DHIS2 instance and another system. Second, they serve to demonstrate what applications might have to be created to achieve integration. Lastly, portions of the code can be reused to build integration applications between DHIS2 and its auxiliary systems in Malawi. Depending on the level of connectivity between DHIS2 and the auxiliary systems, integration can be fully app-driven or file-based.

5.3.5 DHIS2 Application Development

Through the DHIS2 web and android application development workshops held in Malawi, a total of 17 Malawian developers, 12 males and 5 females, acquired requisite capacity for developing third-party applications atop the DHIS2 software platform. The developers were drawn from 6 organizations: University of Malawi, MoH ICT department, CMED/HISP Malawi, Baobab Health Trust, Luke International Norway, and Jhpiego. Further to that, three developers from University of Malawi participating on the mHealth4Afrika project attended another web application development workshop in Kampala, Uganda and a workshop/hackathon in Port Elizabeth, South Africa. Subsequently, some of the developers in Malawi have taken up DHIS2 web application development tasks commissioned by University of Oslo and UNICEF. Other applications were developed under the DHIS2 reconfiguration and data migration project (*see Table 5.2 and Figure*

5.11). Under the mHealth4Afrika project, developers at the University of Malawi, have been participating in the development of a maternal health records application by leveraging DHIS2 tracker.

Figure 5.10 Cropped Screenshot of mHealth4Afrika App login page

Table 5.2 Summary of DHIS2 Data Migration Applications in Malawi

Application	**Description**
Dataset Details Lister	a DHIS2 web app that lists each data element in a selected dataset alongside three IDs required to uniquely map it to another data element on another DHIS2 instance. Facilitated creation of dataset maps used as input when migrating data across two instances.
Data Migrator	a PHP application that takes as input a dataset map (*from the Dataset Details Lister*), connects to two DHIS2 instances and migrates data from one instance to the other.
Data Migration Validator	a PHP application that compares data in a pair of corresponding datasets across two DHIS2 instances to ensure the integrity of the migrated data. Same data elements with different values are flagged to allow the migration team catch discrepancies and fix them

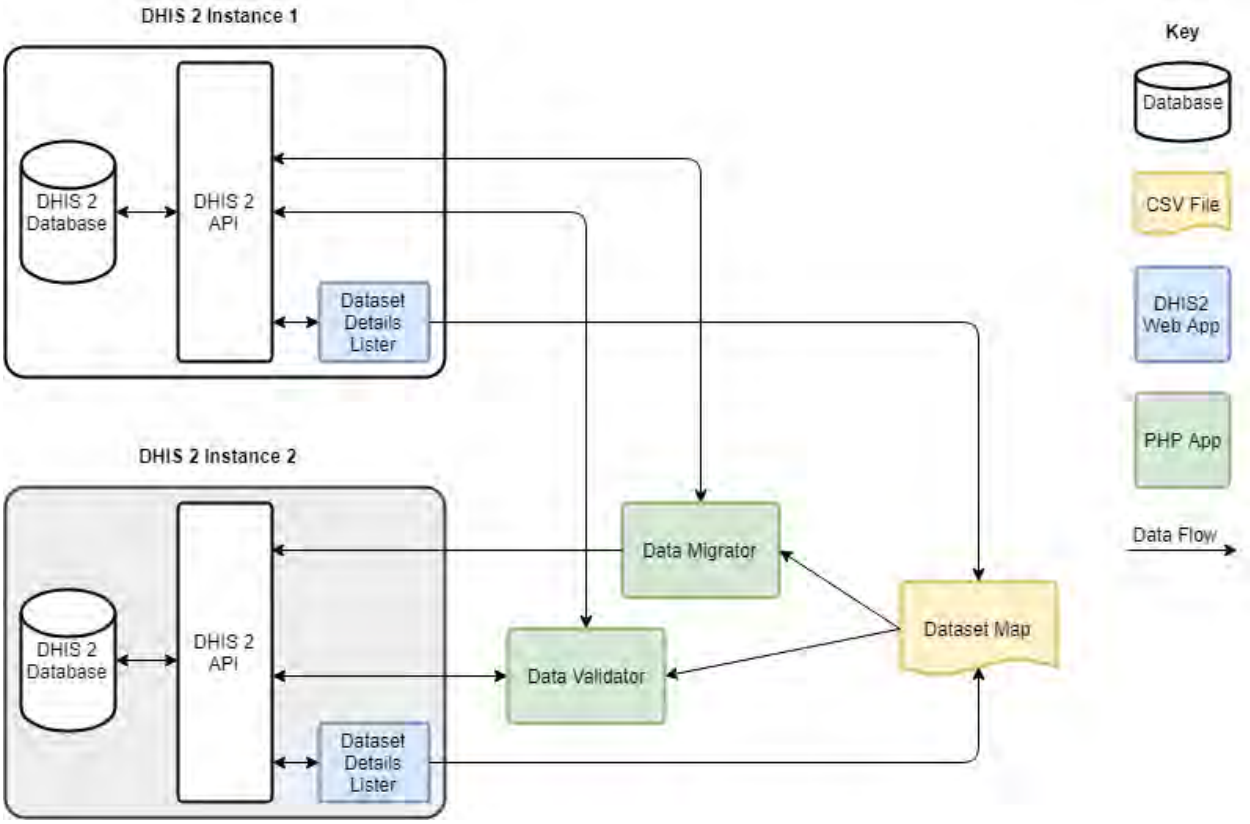


Figure 5.11 DHIS2 Data Migration Apps in Malawi

5.3.6 Challenges

Efforts leveraging the DHIS2 software platform in Malawi have faced some challenges. One of the key challenges is lack of ICT personnel in CMED coupled with limited collaboration with the ICT department on DHIS2. This made CMED overly dependent on external expertise to address technical challenges on DHIS2. Strategic partnerships with local and international organizations have granted CMED access to a pool of human resources with requisite technical capacity for implementation and management of DHIS2. While a proposal was submitted to the government to adjust the staff establishment in CMED to include ICT personnel, strengthening collaboration with the ICT department on DHIS2 is regarded as a better option going forward. This was done during the DHIS2 reconfiguration and data migration project.

A related challenge has been staff turnover with respect to DHIS2 technical assistants recruited by HISP Malawi and seconded to CMED. All technical assistants recruited by HISP Malawi between 2009 and 2015 have left for various reasons thereby threatening the day to day operations of DHIS2. New technical assistants have had to be recruited and trained to the level of competence

necessary to drive DHIS2 operations. It is hoped that the ongoing restructuring of HISP Malawi will bring forth staff retention mechanisms that will help address the problem of staff turnover and ensure continuity of day to day technical support on DHIS2.

With respect to end users, another challenge is the need for retraining resulting from staff transfers, recruitment of new staff, changes in DHIS2 features, and so on. "*There is a backlog of prospective end users without any DHIS2 training*" (Representative, HISP Malawi). Self-study solutions such as the DHIS2 online academy can help with this challenge. However, the cost of internet connectivity gets in the way. Therefore, "*offering offline versions of the study material can go a long way in ensuring continuity of training in place of a regular academy or training workshop*" (Representative, HISP Malawi).

The frequency of new releases of DHIS2 presents a challenge towards system administration. Currently, there are three releases of DHIS2 per year. For system administrators, the dilemma becomes when to update DHIS2 instances. To stay up to date you would need to update your DHIS2 instances immediately after the release of a new version but that increases the update burden and makes you susceptible to bugs in the latest version. Alternatively, updating can be pended but this creates an update lag whereby an instance is one or more version behind the latest release of DHIS2. Update lags deny end users access to new features and can be a source of security concerns. Furthermore, update lags create a need for several incremental updates with respect to intermediate versions. If not properly carried out updates result in corruption of the DHIS2 instance as was the case with a DHIS2 tracker pilot project in Malawi.

Besides system administration challenges, the rapid releases of DHIS2 present potential challenges to software developers. With each new version of DHIS2 comes a new version of the API. This has the potential to break already existing applications if the API resources they are using are changed. To remedy the situation, DHIS2 introduced API versioning whereby the last three API versions are supported. As an example, DHIS version 2.29 supports API version 29, 28 and 27. With DHIS2 released three times a year, this means that, unless updated, an application released the previous year could potentially break by the end of the current year due to API changes. Such has been the experience in the mHealth4Afrika project whereby attempts to leverage features provided in the latest DHIS2 release would introduce API related bugs in the mHealth4Afrika application.

6 Findings

This chapter presents the findings of the study based on four research papers appended as part of this thesis. It provides a summary of each paper and a tabulated summary of findings in the papers in relation to research questions posed earlier. The four research papers, listed in Table 6.1, include two conference papers and two journal papers.

Table 6.1 Research Papers Appended to the Thesis

1. Msiska, B. & Nielsen, P. (2017), A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries, *Proceedings of the 14th International Conference of IFIP Working Group 9.4*, May 2017, Yogyakarta, Indonesia
2. Msiska, B. (2017), Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries, *Proceedings of IST-Africa 2017 Conference*, May 2017, Windhoek, Namibia
3. Msiska, B. & Nielsen, P (2017*),* Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity, *Information Technology for Development, 24(2), pp 388-421*
4. Msiska, B. (2018), Cultivating Third-Party Development in Platform-Centric Software Ecosystems: Extended Boundary Resources Model, *The African Journal of Information Systems, 10 (4), Article 6, pp 348–365*

6.1 Paper 1: A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries

Msiska, B. & Nielsen, P. (2017), Proceedings of the 14th International Conference of IFIP Working Group 9.4, May 2017, Yogyakarta, Indonesia

Drawing inspiration from the work of Dittrich (2014), this paper observes that software platforms are half-products that require extra from actors within their context of implementation and use to be made into working solutions. Turning half-products into desired solutions demands local human capacities beyond those of traditional vendor-driven software. The paper argues that to effectively leverage open source health information software platforms in developing countries stakeholders must be aware of the human capacity requirements associated with such platforms. The aim of the paper is to identify requisite human capacities for leveraging open source health information software platforms to provide a framework for assessing and addressing their inadequacy in developing countries.

Figure 6.1 Software System Timeline within Context of Use

Base on the work of Pigoski (1997), Carzaniga et al. (1998), Van Vliet (2007) and Sommerville (2011), the paper delineates between software platform deployment phase and operation (or productive use) phase. A case study of DHIS2 in Malawi was carried out, with respect to the two phases, to identify key activities carried out and map out corresponding human capacity requirements. For a start, deployment capacity is needed to setup the underlying hardware and software environment and followed by the installation of the software platform. Once the software platform is installed, it must be configured and customized in line with the local needs in it context of use which entails customisation capacity. Specific end user requirements within the context of use not readily addressed by the software platform or its existing applications might necessitate modification of existing applications or development of new ones. This entails the availability of application development capacity. Periodically, the software platform must be upgraded to newer versions and its operating environment must be adjusted to keep it in good running condition which creates a need for system administration capacity. Lastly, during its deployment and operation, a software platform is subject to use by end users for testing purposes as well as productive use which demands existence of usage capacity.

Table 6.2 Key Activities and Human Capacity Requirements

Phase	**Key Activities**	**Human Capacity Categories**
Deployment	Setting up hardware and software environment	Deployment Capacity/System Administration Capacity
	Installation of the software platform	Deployment Capacity
	Configuration and customisation for local use	Customisation Capacity
	Testing against user requirements	Usage Capacity
	Developing or modifying applications to add missing functionality	Application Development Capacity

Operation	Upgrading the hardware and software environment	System Administration Capacity
	Deploying new versions of the software platform	Deployment Capacity
	Configuration and customisation new platform versions for local use	Customisation Capacity
	Testing new versions against user requirements and using the platform productively	Usage Capacity
	Developing or modifying applications to add missing functionality	Application Development Capacity

As depicted in table 6.2, the extent to which developing countries can leverage open source software platforms in implementation of health information systems is subject to the availability of human capacities related to deployment, customisation, use, application development and system administration. In contexts where deficiencies of such capacities exist leveraging open source health information software platforms could be a challenge. Table 6.3 itemizes and briefly describe each of these key requisite capacities as a contribution towards informing efforts assessing and addressing human capacity inadequacies in developing countries leveraging open source health information software platforms.

Table 6.3 Human Capacity Requirements for Software Platforms within Context of Use

1. **Deployment capacity:** the capacity to set up the required hardware and software environment and deploy the software platform.
2. **Customization capacity:** the capacity to configure the software platform to match the needs in the context of use.
3. **Usage capacity:** the capacity of end users to use the system for testing purposes or in production.
4. **System administration capacity:** the capacity to keep the system up to date and in good running condition to ensure its reliability and availability.
5. **Application development capacity:** the capacity to develop complementary applications addressing needs not readily addressed by the platform.

Based on these findings, the paper argues that the extent to which developing countries can leverage open source health information software platforms is subject to the existence of a range of different capacities. The itemization of these capacities acts as a framework informing

stakeholders leveraging open source health information software platforms in developing countries what capacities to put in place in order to fully leverage the software platforms. Furthermore, it informs efforts aimed at assessing and addressing inadequacies in human capacities where open source health information software platforms have been implemented. In Malawi, compared with usage capacity there were significant deficiencies with respect to the other capacities in relation to DHIS2.

6.2 Paper 2: Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries

Msiska, B (2017), Proceedings of IST-Africa 2017 Conference, May 2017, Windhoek, Namibia

One of the major causes of failure and unsustainability of health information systems in developing countries has been the lack of appropriate human capacities for their use, development and maintenance (Kimaro and Nhampossa, 2005). Thus, there is potential for deficiencies in human capacities constraining efforts to leverage open source health information software platforms in developing countries. Developing countries are characterized by multiple health information system projects involving several stakeholder organisations. Taking advantage of this, creating a shared pool of personnel with requisite capacity drawn from multiple stakeholder organisations has been suggested as a possible strategy for alleviating human capacity challenges faced by open source health information system projects in developing countries (Staring and Titlestad, 2008).

Motivated by this suggestion, this paper uses a case study of DHIS2 in Malawi to explore the effectiveness, challenges and factors for pooling human resources across organizational boundaries to address gaps in requisite human capacities for leveraging open source health information software platforms in developing countries.

The paper reports that the Central Monitoring and Evaluation Division (CMED) of the Ministry of Health in Malawi, under whose custody DHIS2 falls, lacks inhouse human resources to support implementation and operation of the platform. Still, CMED has been successful in leveraging a pool of personnel drawn from various stakeholder organisations to drive the implementation and operation of DHIS2. This renders credence to the suggestion by Staring and Titlestad (2008).

Figure 6.2, for example, illustrates the pooling of personnel across organizational boundaries during the DHIS2 reconfiguration and data migration project.

Information dependencies, mutual interests, and the cost-effectiveness of using a common platform other than several inhouse solutions attracts stakeholders to pool personnel to collaboratively leverage an open source health information software platform. However, fluctuation in the availability of pooled personnel due to pressing commitments in their respective organisations has the potential to constrain the speed with which mutually beneficial tasks are accomplished. Thus, the paper concludes that pooling of personnel is viable as a stopgap not as a replacement for long-term capacity building initiatives.

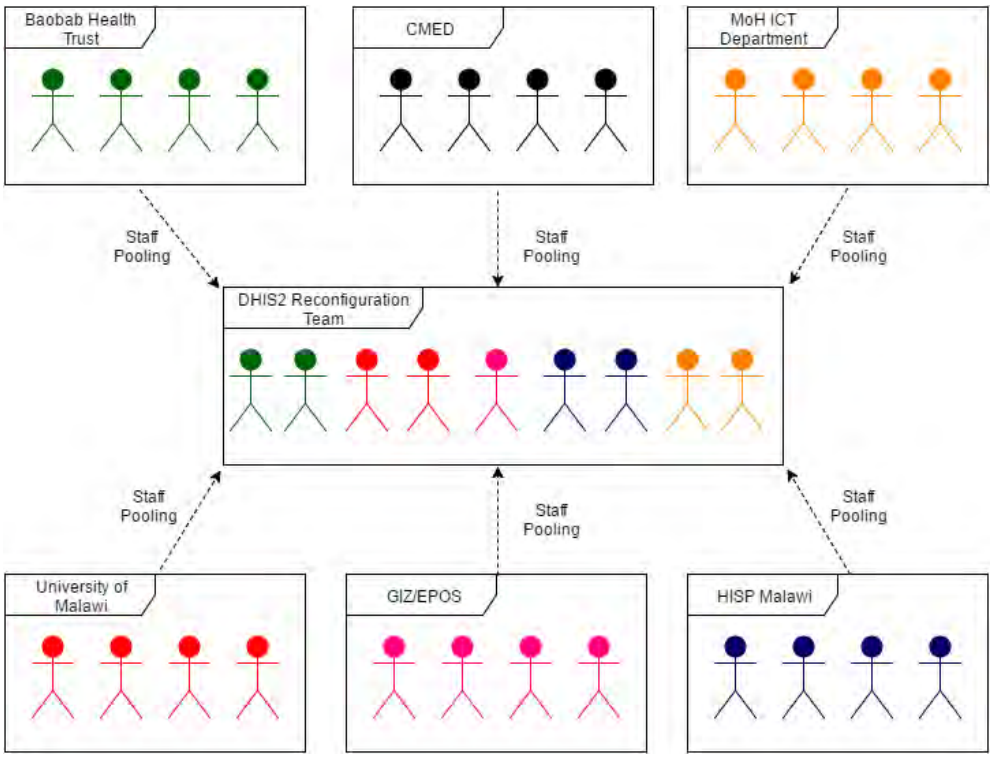


Figure 6.2 Staff Pooling during the DHIS2 Reconfiguration and Data Migration Project

6.3 Paper 3: Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity

Msiska, B. & Nielsen, P (2017), Information Technology for Development

Understanding the role of different contributors is central to ongoing research on platform-centric software ecosystems and innovation. Enshrined in this discourse is the concept of generativity,

which focuses on how attributes of software platforms influence who can innovate on top of them and the extent to which innovation is possible within their respective ecosystems that also include people and complementary technologies. Unfortunately, the discourse has paid limited attention to the unfolding of innovation in the *fringes* - contexts, typically represented by developing countries, which are peripheral to and disconnected from the context where the software platform is developed, and often characterized by a scarcity of resources necessary for digital innovation.

The paper draws from Zittrain (2008) and Lane (2011) to formulate a socio-technical perspective on innovation and generativity in the fringes of a platform-centric software ecosystem. In relation to this, the paper addresses the question: *how can our understanding of generativity be framed to provide a holistic account of both technological and social factors that constrain and enable innovation in the fringes of software ecosystems*? To address this question, the paper employs the case study of DHIS2 in Malawi, focusing on the implementation of the software platform and subsequent "innovation episodes".

The paper shows that it does not make sense to look at technological or social factors that constrain or enable innovation in isolation. Using the DHIS2 reconfiguration project in Malawi, the paper notes that even though the software platform through its open and flexible Web API offered opportunities for innovation, it would have accomplished little without alignment, action opportunities and appropriate permissions being accorded to a network of experts drawn from various stakeholder organizations.

At the same time, as demonstrated by challenges experienced in Malawi with an earlier version of DHIS which was based proprietary technology, the existence of alignment, action opportunities and appropriate permissions between different actors cannot independently foster desired innovations where the underlying technology is not permitting. Thus, generativity is socio-technical – technical attributes of a software platform work in concert with existing human relationships to constrain or foster possible innovations.

Practically, adopting the socio-technical generativity perspective has implications for both design and subsequent implementation of software platforms in developing countries. First, the software platform must be designed to offer required accessibility, adaptability and ease of mastery corresponding to the contexts in which it will be used. Second, necessary local and global relationships should be nurtured and sustained over time by, among other things, offering

appropriate permissions for innovation to a wide audience, affording them action opportunities for reconstructing their collaborative networks, and implementing policies to establish a regime of openness.

6.4 Paper 4: Cultivating Third Party Development in Platform-Centric Software Ecosystems: Extended Boundary Resources Model

Msiska, B. (*2018*), The African Journal of Information Systems

The value of a software ecosystem to end-users is partly a product of the range of complementary applications built on top of the software platform specific to that ecosystem. Challenges in making informed decisions regarding what applications to build for the platform to satisfy user requirements and adequately addressing turbulent and divergent user requirements make third-party development increasingly attractive to software platform owners. Through their *boundary resources model*, Ghazawneh and Henfridsson (2013) aim to provide a theoretical account for cultivating third-party development in platform-centric software ecosystems.

Third-party development relies on both boundary resources (Ghazawneh and Henfridsson, 2010) and the generative capacity (Avital and Te'eni, 2009) of third-party developers. However, while denoting the critical role boundary resources play in shaping third-party development, the boundary resources model (Ghazawneh and Henfridsson, 2013) relegates to the background the complementary role played by the generative capacity of third-party developers involved. Furthermore, by focusing on software artefacts, such as APIs and SDKs, the model emphasizes the role played by *boundary objects* (Bergman et al., 2007; Carlile, 2002; Star, 2010; Star and Griesemer, 1989; Wenger, 2000) but pays little attention to the role played by other boundary bridges that exist between communities of practice – *boundary interactions* and *brokers* (Wenger, 2000) – in shaping third-party development.

Based on a case study of the DHIS2 software ecosystem, this paper focuses on the efforts to cultivate third-party development in Malawi and other developing countries that are leveraging the DHIS2 software platform. The case study reveals that third-party development unfolds where boundary resources are accompanied by third-party developers' generative capacity. The paper, then, makes a distinction between *internal generative capacity* possessed by developers in the platform owner community and *external generative capacity* possessed by developers outside the

platform owner community. The paper, further, argues that providing boundary resources without building corresponding external generative capacity is not adequate to cultivate third-party development.

Going further, the paper uses boundary bridges constructs, mentioned as part of the conceptual framework and as defined by Wenger (2000), to note that boundary resources employed to cultivate third-party development are socio-technical artefacts comprising of not only boundary objects but also boundary interactions and brokers. This allows making a distinction between *software development boundary resources* and *capacity building boundary resources* both of which are important in cultivating third-party development. Figure 6.3 presents an *extended boundary resources model*, proposed by the paper, to bring to the foreground *external generative capacity* and *capacity building boundary resources* as co-factors with *software development boundary resources* in shaping third-party development.

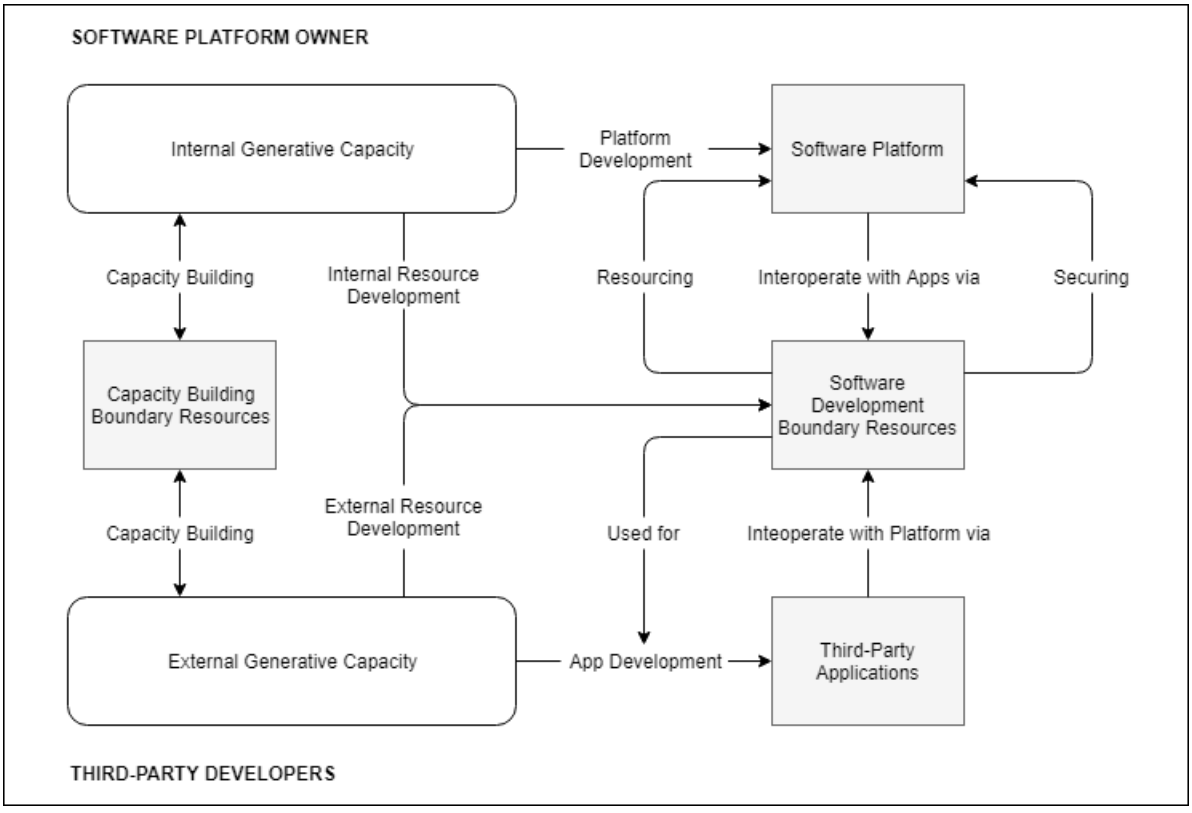


Figure 6.3 Extended Boundary Resources Model

In the model, *internal generative capacity* refers to the platform owners' software development and related competences used to develop the software platform and software development

boundary resources. *External generative capacity* refers to third-party developers' software development and related competencies used to develop third-party applications and their own software development boundary resources, in an act which Ghazawneh and Henfridsson (2013) refer to as *self-resourcing*.

Software development boundary resources comprise of boundary objects, in form of software artefacts such as SDKs and APIs, that regulate and facilitate development of third party applications. By either resourcing or securing (Ghazawneh and Henfridsson, 2013) the software platform, software development boundary resources play an enabling or constraining role with respect to what third-party developers can do with the platform. Whereas, *capacity building boundary resources* comprise of boundary objects, boundary interactions and brokers deployed within the software ecosystem to facilitate propagation of generative capacity within the software ecosystem. In the case of the DHIS2 software ecosystem, capacity building boundary resources include: artefacts such as DHIS2 documentation, events such as DHIS2 academies, and DHIS2 academy facilitators as brokers of various forms of human capacities required in developing countries making use of the platform.

6.5 Summary of Findings in Research Papers in Relation to Research Questions

To recap, the main objective of this study is contributing towards a practical and conceptual understanding of how developing countries can effectively leverage open source health information software platforms against a backdrop of human capacity challenges. To address this overarching objective, the study posed three research questions:

- **Research Question 1 (RQ1):** What human capacities are needed to leverage open source software platforms as means for implementation of health information systems in developing countries?
- **Research Question 2 (RQ2):** How can gaps in human capacity needed to leverage open source software platforms in developing countries, if any, be assessed and addressed?
- **Research Question 3 (RQ3):** How are efforts aimed at leveraging open source software platforms in implementation of health information systems influenced by the context in a developing country and characteristics of the platform itself?

Table 6.4, below, presents a tabulated summary of findings in the appended research papers in relation to these three questions.

Table 6.4 Research Paper Findings in Relation to Research Questions

Paper	**Contributions in Relation to Research Questions**
Paper 1: *A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries*	RQ1: The paper identifies five key requisite human capacities for leveraging open source health information software platforms in developing countries: • Deployment capacity • Customization capacity • Use capacity • System administration capacity • Application development capacity RQ2: The itemization of the human capacities serves as a framework to facilitate assessing and addressing of inadequacies where they exist. The paper further recommends that capacity building must be a continuous process to cope with volatility of open source software platforms characterized by rapid innovation and evolution.
Paper 2: *Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries*	RQ2: The paper provides empirical evidence for pooling of expertise across organizational boundaries as a strategy for mitigating gaps in human capacities requisite for leveraging open source health information software platforms in developing countries. It also presents a corresponding model for achieving this. RQ3: The paper highlights inter-organizational challenges that might constrain pooling expertise across organizational boundaries as a strategy to mitigate gaps in requisite human capacities requisite for leveraging open source health information software platforms. Thus, concludes that if used it is likely to be a stopgap and not a replacement for long-term capacity building requirements in relation to open source health information software platforms.

Paper 3: *Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity*	RQ3: The paper provides empirical evidence regarding the roles played software platform attributes such as accessibility, adaptability, transferability and ease of mastery in shaping possible innovations on open source health information software platforms in developing countries. However, the paper shows that, in shaping innovations, technical characteristics of the software platform work in concert with the nature of human relationships established around the software platform in terms of appropriate permissions, action opportunities and mutual alignment of actors. Deficiencies on either end can constrain innovation in the fringes of a software ecosystem. The paper indirectly deals with RQ1 and RQ2 by linking innovation to the availability of a range of expertise – domain expertise and software platform expertise for example - and highlighting a need to have mechanisms for building capacity to cover gaps with respect to such expertise.
Paper 4: *Cultivating Third Party Development in Platform-Centric Software Ecosystems: Extended Boundary Resources Model*	RQ2: Focusing on third-party development, the paper highlights the role of capacity building boundary resources in the form of boundary objects, boundary interactions and brokers in building requisite external generative capacity for leveraging open source health information software platforms in developing countries. By segregating boundary resources into capacity building and software development boundary resources the paper brings to the foreground capacity building as an important element in leveraging open source health information software platforms in developing countries. RQ3: The *resourcing* and *securing* processes involving a software platform and software development boundary resources have potential implications on efforts to leverage an open source health information software platform. Through these two processes possible innovation efforts are enabled or constrained.

7 Discussion

The main objective behind this thesis is contributing towards a practical and conceptual understanding of how developing countries can effectively leverage open source health information software platforms against a backdrop of human capacity challenges. To achieve this objective, the thesis sought to address three questions:

- *What human capacities are needed to leverage open source software platforms as means for implementation of health information systems in developing countries?*
- *How can gaps in human capacity needed to leverage open source software platforms in developing countries, if any, be assessed and addressed?*
- *How are efforts aimed at leveraging open source software platforms in implementation of health information systems influenced by the context in a developing country and characteristics of the platform itself?*

In the subsections that follow, this chapter presents a discussion in line with these research questions.

7.1 Human Capacities Needed to Leverage Open Source Software Platforms

The implementation of a software-based system, in this case a software platform, within its context of use involves a series of phases. In this thesis, the lifecycle of a software platform within its context use has been delineated into two phases: deployment phase and operation phase. The deployment phase encompass all activities carried out in order to make a software platform available for use (Carzaniga et al., 1998). Activities in this phase may include setting up the required hardware and software environment, installing the software platform, piloting and adapting it for local use, and testing it against functional and nonfunctional requirements to determine its readiness for use. Once deemed ready for use, it is rolled out and enters the operation (or productive use) phase. During productive use, anomalies are discovered, operating environments change, and new user requirements emerge necessitating corresponding changes to the software platform (Pigoski, 1997; Sommerville, 2011). Changing of a software-based system or its components to correct anomalies, improve performance or other attributes, adapt to changing environment and requirements during the operation phase is referred to as software maintenance

(Sommerville, 2011; Van Vliet, 2007) which accounts for about 70% of the cost and effort expended during its lifetime (Ogheneovo, 2014).

Within the lifecycle of software platform in its context of use, a range of different human capacities is required to support various key activities and lack of appropriate human capacities in respective phases can constrain their deployment and subsequent operation. Therefore, understanding what human capacities are required within each of these phases can be instrumental in ensuring the effectiveness of health information system implementation projects in developing countries that leverage open source software platforms. Using the case study of DHIS2 in Malawi, the thesis, in section 6.1, identified five categories of human capacities needed to leverage open source software platforms (*repeated for reference sake in table 7.1*). While it could be argued that similar human capacities might be required for other software-based systems, the half-solution nature of software platforms makes the existence of these human capacities within context of use quite critical.

Table 7.1 Human Capacity Requirements for Software Platforms within Context of Use

1. **Deployment capacity:** the capacity to set up the required hardware and software environment and deploy the software platform.
2. **Customization capacity:** the capacity to configure the software platform to match the needs in the context of use.
3. **Usage capacity:** the capacity of end users to use the software platform and associated applications for testing purposes or in production.
4. **System administration capacity:** the capacity to keep the software platform up to date and in good running condition to ensure its reliability and availability.
5. **Application development capacity:** the capacity to develop complementary applications addressing needs not readily addressed by the platform.

This multiplicity of requisite human capacities has significant implications for efforts leveraging open source software platforms towards health information system implementation in developing countries. For a start, there is need to part ways with historical approaches in dealing with requisite human capacities. Most health information system projects in developing countries have ended up as failures or unsustainable in the operation phase because of human capacity inadequacies left behind by their respective donors (Kimaro and Nhampossa, 2005). In the deployment phase, donors and their agents traditionally fill human capacity gaps by engaging foreign experts at the expense of building local expertise (Kimaro, 2006). Such charity works well in the short-term

deployment phase but creates challenges with maintenance and sustainability in the long-term operation phase. As observed by Yunus and Jolis (2003) and reiterated by Qureshi (2013), charity only serves to avoid recognizing an underlying problem and finding a lasting solution to it. Left without requisite human capacity, locals fail to support information systems in question thereby perpetuating dependency on external support and subsequently, leading to gradual decay and obsolescence of the systems (Boerma, 1991) in the operation phase. Therefore, the need to build local expertise around software platforms cannot be overemphasized.

While donor-driven health information system projects in developing countries are often accompanied by short-term training focusing on building usage capacity (Kimaro, 2006), leveraging open source software platforms demands a shift from this approach. As depicted in Table 7.1, software platforms come with an inherent demand for human capacity extending beyond mere usage to include deployment, customisation, system administration and application development. Therefore, scaling of human capacity in terms of numbers and skills of both users and technical personnel (Sahay and Walsham, 2006) ought to be an integral part of leveraging open source software platforms in implementation of health information systems in developing countries. Adequate usage capacity alone is not sufficient to turn the half-solutions delivered by platform owners into working solutions within context of use.

Open source software projects, and by extension open source software platforms, are characterized by rapid releases and typically have more iterations than their proprietary counterparts (Roets et al., 2007). This inherent volatility means that scaling of human capacity needed to leverage open source health information software platforms cannot be a once-off endeavor. As demonstrated by the existence of a backlog of personnel in Malawi needing retraining around DHIS2, to cope with rapid innovations and evolution associated with open source health information software platforms capacity building ought to be a continuous process. This echoes the observation by Kimaro (2006), that capacity building for health information systems in developing countries is a continuous process.

7.2 Addressing Gaps in Requisite Human Capacity

From the preceding discussion it is apparent that leveraging open source software platforms in implementation of health information systems entails the existence of a range of related human capacities. This is the case because software platforms are not usually complete solutions but rather

half products (Dittrich, 2014) that constitute a departure from an era where software vendors delivered fully-fledged solutions to one where solutions delivered must completed within the context of use by other actors working with the end-user community. The deferring of implementation of certain aspects of software platforms to external actors is attractive to platform owners (Bosch, 2009; Boudreau and Lakhani, 2009) as it helps them overcome the challenge of dealing with turbulent and divergent user requirements. However, this introduces unique human capacity requirements in the context where the software platforms are used. Because of this, open source health information software platforms cannot deliver expected benefits unless they are supported by appropriate human capacity locally (Kimaro, 2006). Thus, to effectively leverage open source health information software platforms in developing countries any gaps in requisite human capacity must be addressed.

7.2.1 Pooling Human Resources

Longstanding deficiencies and challenges with ICT-related human capacity (Kimaro and Nhampossa, 2005; Mutula and Van Brakel, 2007; Paudel et al., 2010; Roets et al., 2007; Weerawarana and Weeratunge, 2004) are a potential constraint towards efforts leveraging open source health information software platforms in developing countries. The health sector in developing countries is characterized by a multiplicity of health information system projects involving different stakeholders. Creating a shared pool of talent across such projects by bringing together human resources from different stakeholder organizations has been suggested by Staring and Titlestad (2008) as one possible strategy for addressing human capacity gaps faced by open source health information system projects in developing countries. Enshrined in this suggestion is the networks of action approach (Braa et al., 2004; Braa and Nielsen, 2015) which has long been a hallmark of the HISP community.

Empirical evidence from the deployment and operation phases of DHIS2 in Malawi renders credence to the efficacy of this strategy. Figure 7.1 presents a model, adapted from Msiska (2017), for pooling human resources required to leverage an open source health information software platform. With respect to the temporality of collectives as described by Osh and Avital (2010), the transient nature of staff pools can be instrumental in overcoming fixed routines and formal rules within organizational boundaries and result in higher collective generativity. On the other hand,

the marked transiency also means that staff pooling is more of a stopgap arrangement suitable for projects than a long-term arrangement.

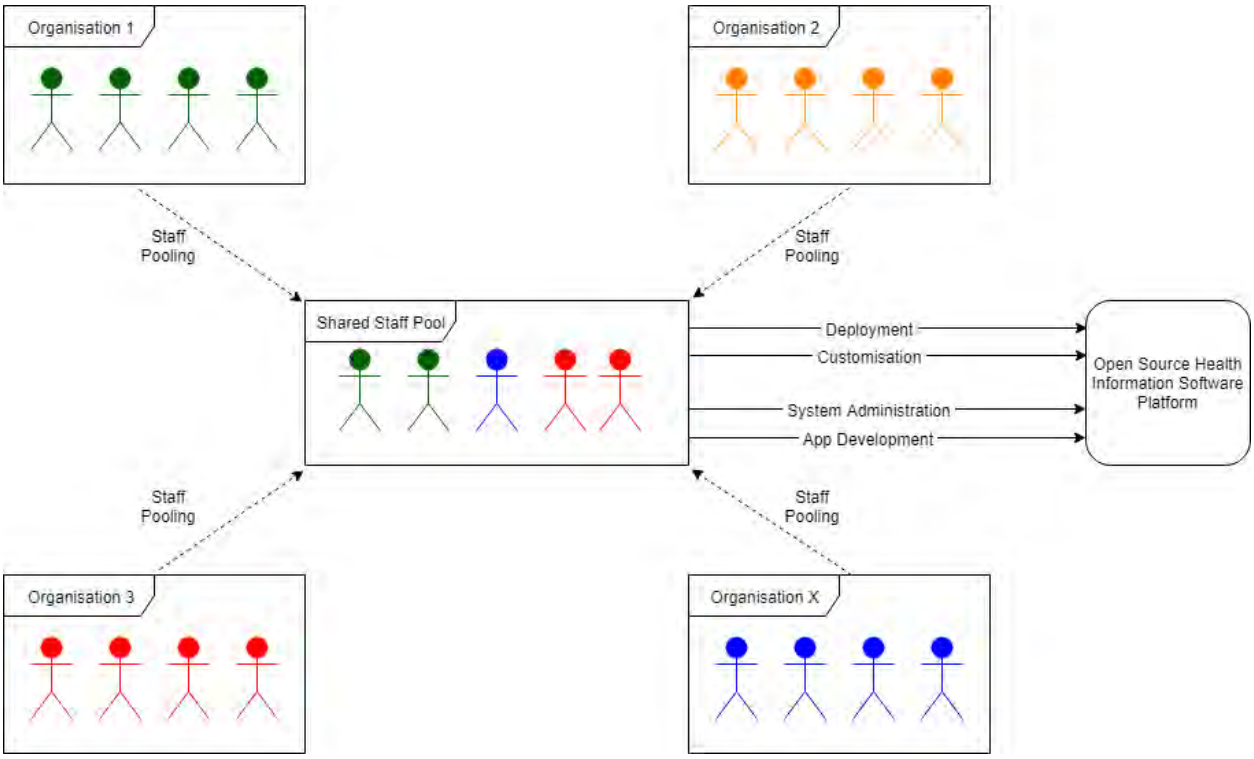


Figure 7.1 A model for pooling human resources needed to leverage an open source health information software platform

As illustrated in figure 7.1, by pooling human resources across organizational boundaries, gaps in human capacity related to deployment, customization, system administration and application development can be alleviated and allow stakeholder organizations in developing countries collaboratively leverage open source health information software platforms. Such has been the case between CMED and various stakeholder organisations around DHIS2 in Malawi. Mutual interests, inter-organizational information dependencies and the cost-effectiveness of using a common platform other than several inhouse solutions serve as attractors for stakeholder organizations to pool personnel to collaboratively leverage an open source health information software platform (Msiska, 2017). However, drawing on the description of transient generative collectives by Osch and Avital (2010), staff pools can be rather fortuitous, forming and dissolving at any moment, and should therefore not be a replacement but rather a complement of other capacity-building initiatives. In such initiatives capacity building boundary resources as discussed below play a critical role.

7.2.2 Capacity Building Boundary Resources

The definition of a software ecosystem by Bosch and Bosch-Sijstema (2010b) and the synthesis of several software ecosystem definitions by Manikas and Hansen (2013) allude to the existence of different communities around a common software platform. Communities within a software ecosystem include for example platform owners, domain experts, third-party developers and end users. The boundaries separating these communities constitute channels through which competences, experiences and resources are exchanged resulting in co-learning which enriches generative capacities of community members on either side of the boundaries. Wenger (2000) identifies three bridges through which the ensuing exchange happens: *boundary objects*, *boundary interactions* and *brokers*.

In their boundary resources model, Ghazawneh and Henfridsson (2013) defined *boundary resources,* basing on the boundary object construct (Bergman et al., 2007; Carlile, 2002; Star, 2010; Star and Griesemer, 1989; Wenger, 2000), as software tools and regulations, such as application programming interfaces (APIs) and software development kits (SDKs), that serve as an interface between platform owners and application developers facilitating the development of apps on top of a software platform. The model highlights the critical role played by such boundary resources in enabling third-party application development and stimulating generativity within a software ecosystem. While agreeing with the model on this, the case study of DHIS2 in Malawi also reveals a complementary role played by the generative capacity (Avital and Te'eni, 2009) of application developers in third-party application development. This thesis, in section 6.4, makes a distinction between *internal generative capacity* in platform owners community and *external generative capacity* in the community of third-party application developers.

The thesis, in section 6.4, makes a further distinction is made between *software development boundary resources* and *capacity building boundary resources*. The implicit need for a design capability shift from platform owners to application developers (Prügl and Schreier, 2006; von Hippel and Katz, 2002) in software ecosystems necessitates the existence of capacity building boundary resources as channels for building external generative capacity. Since third-party application development demands external generative capacity, both software development boundary resources and capacity building boundary resources are instrumental in fostering the leveraging of open source software platforms through third-party application development.

Going further, it is possible to generalize capacity building boundary resources beyond application development capacity to cover human capacities, listed in table 7.1, needed to leverage open source software platforms in developing countries. A generalized capacity building boundary resources model can be constructed by relating capacity building boundary resources to the platform owner community on one end and the platform consumer community on the other whereby a platform consumer refers to any external actor involved in the leveraging of an open source software platform. This is depicted in Figure 7.2.

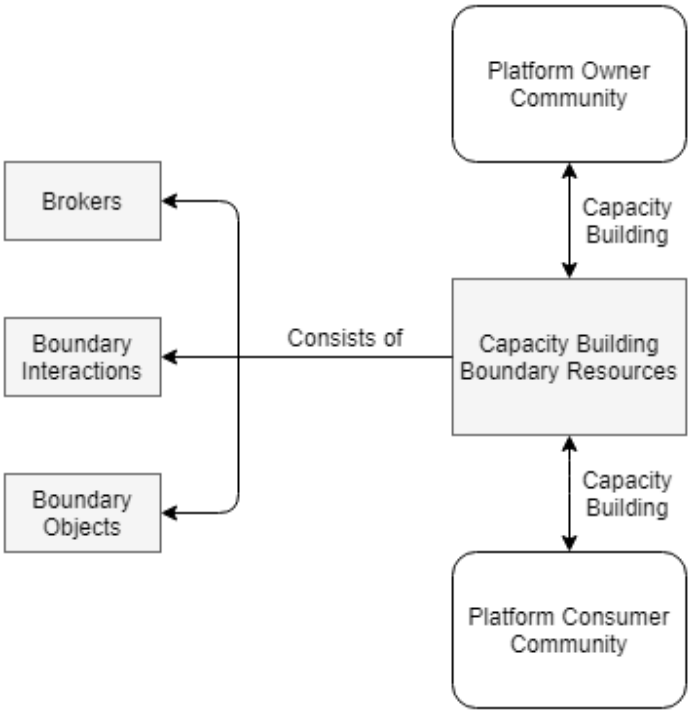


Figure 7.2 Capacity Building Boundary Resources Model

In the model capacity building boundary objects consists of artefacts, for example documentation put in place to build capacities of different actors in the platform consumer community. In the case of DHIS2 this includes implementers documentation, end-user documentation and so on. Capacity building boundary interactions constitute different forms of boundary encounters between members of different communities aimed at building capacities. In the case of DHIS2 these include academies, workshops and mentorship or internship programmes, for example. Lastly, capacity building brokers would be people facilitating the import and export of competences between the communities. Facilitators at academies or similar capacity building events are a typical example.

7.3 Socio-Technical Factors for Leveraging Open Source Software Platforms

Turning attention to the third research question, the term generativity has been used with respect to software platforms to denote extent to which a software platform stimulates and supports actors, other than the platform owner, to leverage the platform and create derivative products. Chapter 3 provides three dimensions to generativity: generative technology (Eck et al., 2015; Zittrain, 2009, 2008, 2006), generative relationships (Lane, 2011) and generative capacity (Avital and Te'eni, 2009). The discussion in section 7.2.2 has shown how generative capacity works in tandem with boundary resources to shape third-party application development, one of the key activities that are part and parcel of leveraging a software platform. This section shifts the focus towards the constraining or enabling role played by software platform itself and/or relationships between stakeholders within its context of use.

7.3.1 Software Platform Attributes

Drawing on the work of Zittrain (2008), chapter 3 introduces five attributes of a technology artefact, in this case a software platform, that define its generativity: leverage, adaptability, ease of mastery, accessibility and transferability. The first factor in determining whether a software platform is generative regards its *leverage* which denotes the extent to which productivity is gained by using a technology artefact as compared to not using it (Zittrain, 2008). In the case of DHIS2 in Malawi, the software platform offers the Ministry of Health, CMED and other stakeholders a ready-made software tool for health data management and visualization whose web-based features offer an opportunity to address longstanding fragmentation as reported by Chaulagai et al. (2005) and Smith (2015). By leveraging the DHIS2 software platform, CMED and other stakeholders have means to reduce the time, effort, cost and risk associated with putting in place an integrated national health management information system.

Since DHIS2 is metadata-driven (Braa and Sahay, 2012b) it is highly configurable and adaptable to suit a particular context of use. This level of adaptability allowed the Ministry of Health, CMED and other stakeholders in Malawi to configure and customize DHIS2 to fit underlying data requirements in Malawi. Through its Web API, DHIS2 enables its features to be extended beyond what comes out of the box. In Malawi, this was instrumental during the DHIS2 reconfiguration and data migration project by allowing the project team to develop applications to facilitate and fast-track the data migration process. Leveraging on the adaptability of DHIS2, several other

application development initiatives have been undertaken on top of DHIS2 in Malawi including the League Table application and the mHealth4Afrika application.

According to Zittrain (2008), ease of mastery denotes the extent of easiness for a new actor to understand a technology artefact and the amount of effort required to adapt it. For each version of DHIS2, HISP University of Oslo maintains an extensive collection of documentation for the software platforms targeting different actors including developers, implementers and end users, for example. Such documentation provides these sets of actors means to learn and master various aspects related to leveraging the DHIS2 software platform. Notwithstanding, this key role played by the documentation, the case study in Malawi revealed that sometimes it can be a source of problems when it is vague and out of sync with corresponding DHIS2 versions. Going further, the adoption of JavaScript, CSS and HTML as default technologies for DHIS2 application development agrees with the suggestion by Bosch (2009) that platform owners must aim to simplify contribution by third-party developers by allowing use of generic and popular development environments.

Accessibility is used, in the context of generativity, to reflect the easiness with which access to a technology artefact and the tools and information necessary for its mastery can obtained (Zittrain, 2008). In this regard, HISP UiO fulfills this requirement first by offering DHIS2 as an open source software platform available for download free any license fees and second by providing extensive documentation to help various actors master and leverage the platform. In the face of budgetary constraints, the absence of license fees as an accessibility barrier rendered DHIS2 highly attractive to CMED and its stakeholders in Malawi (Msiska and Nielsen, 2018). This also works in favor of other developing countries, that share traits with Malawi.

The last of these attributes, transferability, concerns the easiness with which a technology artefact can be conveyed and re-appropriated for use in another context. One of the strategies employed at the core of DHIS2 development is the “open generification” (Gizaw et al., 2017) which enables the software platform to be transferred, configured and customized to support diverse needs in diverse use contexts. The transferability ensuing from this strategy has seen DHIS2 being adopted and adapted in over 67 developing countries including Malawi. This transferability extends towards its complementary applications as well. A good example in this regard is the league table application which was first developed as a pilot project in Malawi (Moyo et al., 2016, 2015) and

now adopted and adapted by HISP Tanzania. To further facilitate the transferability of complementary applications HISP UiO has established DHIS2 app stores for both Android and Web applications.

When perceived through the five attributes of generativity outlined by Zittrain (2008) the DHIS2 software platform embodies characteristics of a generative technology. Unlike its desktop antecedent, DHIS2 offers significant leverage for developing country contexts, possesses adaptability required to align it with local needs, eliminates classical accessibility barriers associated with proprietary technology, is accompanied by mechanisms that cater for ease of mastery and has transferability enshrined in its development strategies (Msiska and Nielsen, 2018).

7.3.2 Release Management and Stability

The terms release management and volatility are synonymous with the discourse on open source software (Fogel, 2005; Roets et al., 2007). In the software platform discourse, stability - loosely used as the opposite of volatility - of the platform core and interfaces is more commonly used (Bosch, 2010; Silsand and Ellingsen, 2014). Behind these terms, lies a need to address challenges relating to rapid changes and release cycles that are characteristic of open source software platforms and open source software in general. Often complementary applications break due to updates to the underlying platforms (Bosch, 2010) and consumers become wary of new software versions causing dilemmas regarding whether and when to upgrade to new versions (Fogel, 2005; Roets et al., 2007).

The case study of the DHIS2 software platform in Malawi reflects similar concerns. Currently, there are three major releases of DHIS2 in a year. Immediate upgrade after a release makes users susceptible to bugs in new versions while pended version upgrade create update lags that deny users access to new features and create security concerns. Addressing such concerns might necessitate revisiting release management mechanisms related to DHIS2 and similar software platforms. Learning from other open source software platforms could provide a way forward. For example, Android has one major release in a year and Ubuntu has two major releases a year of which one is a long-term support (LTS) version. This is echoed in the findings of the PATH (2016) report on DHIS2 which carries the following quote:

> *Maybe it's a good idea to distinguish bi-annual major releases (which might contain technology breaks with a migration path) and trimestral minor releases (which only*

> *contain bug-fixes and new features which don't require a migration path) [INTERVIEW COMMENT]* (PATH, 2016, p. 20)

To minimize effects of API updates on existing applications, API versioning was introduced in DHIS2 to offer backward compatibility with each release supporting up to last three API versions. However, the extent of backward compatibility offered is somehow limited as it means, with three DHIS2 releases per year, an application released the previous year could potentially break by the end of the current year due to API updates. Drawing on the Bygstad (2017), applications occupy the lightweight end and software platforms occupy the heavyweight end of the IT spectrum. As argued by Bygstad (2017), generativity unfolds where there is loose coupling between lightweight and heavyweight IT. Limited API backward compatibility results in unintended tight coupling between specific versions of software platforms and existing applications which potentially constrains generativity. Consequently, longer periods of backward compatibility might be more desirable.

Going further, Bosch (2010) suggests that interfaces changes need to announced long before the actual release takes place to allow third-party developers the opportunity to adjust their applications and exploit new functionality. Bosch and Bosch-Sijtsema (2010b) extends this argument further by suggesting that, to minimize unintended breaks in software platform functionality when rolling out new releases, new versions should be first validated in close collaboration with end users and external developers not just published. This call for thorough testing agrees with the following quote in PATH (2016) report on DHIS2:

> *"Testing—I will prefer less features and know that every release can be trusted as a fully working version, bug free (or at least almost bug free)." [SURVEY RESPONSE]* (PATH, 2016, p. 20)

7.3.3 Social Relationships

Lane (2011) outlines five attributes of social relationships, described in chapter 3 of this thesis, that influence generativity: aligned directedness, heterogeneity, mutual directedness, appropriate permissions and action opportunities. According to Lane (2011), aligned directedness denotes the degree of match between interests of different actors within a collective. The leveraging of the DHIS2 software platform in Malawi brings together people from different organizations and domains: CMED in the Ministry of Health, researchers and technical experts from University of

Malawi and the University of Oslo, for example, whose efforts are aligned towards the same direction – strengthening the national HMIS in Malawi. Resulting from this aligned directedness between actors from different organizations is heterogeneity in terms of the actors' competences, social positions and access to resources that works more in favor of innovation than groupthink in addressing prevalent challenges related to leveraging DHIS2 in Malawi. This is exemplified by the initial hosting of DHIS2 at College of Medicine to get around infrastructure hurdles in CMED and establishing HISP Malawi to mitigate the deficiencies in requisite human capacity within CMED.

Mutual directedness, according to Lane (2011), denotes the extent to which continued collaboration among actors is desired despite existing and apparent differences in interests or experience. While leveraging the DHIS2 software platform in Malawi collaboration between stakeholder organizations has at times been threatened by apparent conflicts on access rights and ownership for example. However, the level of mutual directedness has been a gluing factor persuading actors involved to reshape and reorganize the collective for continued collaboration. For example, standard working procedures were introduced and a memorandum of understanding between MoH and HISP Malawi was drafted. The coming together of various actors to put in place these governance tools also demonstrates the possibility of actors to engage one another in interactions that bring about change in the collective which Lane (2011) refers to as action opportunities. Furthermore, these governance tools serve to accord appropriate permissions to actors outside CMED, for example researchers and students from University of Oslo, to render their expertise in various innovative endeavors from which CMED and other stakeholders have subsequently benefitted. A good example in this regard is the development of league table application.

7.3.4 Technology Attributes and Social Relationships Working in Concert

Drawing from Zittrain (2008) and Lane (2011), the DHIS2 software platform and the established social relationships around it in Malawi can be deemed as generative. This generativity, however, was not given but emerged over time as the software platform and the surrounding social relationships evolved. A further observation that emerges from this discussion is the socio-technical nature of generativity. Illustratively, the various applications developed for DHIS2 in Malawi during the reconfiguration and data migration project hinged on the enabling role played

by the platform's Web API. This is in sharp contrast to the desktop version of DHIS which provided limited avenues for extension thereby constraining the end users to use it as it is.

At the same time, the generative attributes of DHIS2 ascribed to its Web API would have been immaterial in the absence appropriate permissions to the two DHIS2 instances being accorded to the members of the DHIS2 reconfiguration and data migration team. Thus, the case illustrates that the generativity of a software platform within its context of use is shaped by technical attributes of the platform working in concert with existing social relationships. To this end, Msiska and Nielsen (2018) argue for a more holistic perspective towards technological or social factors that either constrain or enable innovation which leads them to coin the term *socio-technical generativity* in an attempt to provide a holistic account for the generativity software platforms within context of use.

The value of socio-technical generativity lies in its ability to account for the interplay between software platform attributes and the social context in which the software platforms get embedded. For software platforms, such as DHIS2, that get embedded in a developing country social context, generativity is not merely an intrinsic attribute but emerges extrinsically with respect to prevalent social relationships. For example, considering accessibility as defined by Zittrain (2008) without taking into consideration the existence of appropriate permissions as defined by Lane (2011) cannot fully account for the easiness with which access to a technology artefact can obtained within the context of use.

Similarly, talking about software platform adaptability and ease of mastery (Zittrain, 2008) without considering the existence mutual and aligned directedness (Lane, 2011) in the collective of actors cannot adequately account for the capacity of leverage a software platform possesses within its context of use. Therefore, the potential of a software platform to be generative within a given context of use is an emergent trait arising from the interplay between intrinsic attributes of the software platform and extrinsic social relationships that govern a collective of human actors around it. Socio-technical generativity, as a concept, embraces and makes explicit this envisaged interplay without limiting out focus to one or the other.

8 Conclusion

The thesis sets out to contribute towards a practical and conceptual understanding of how developing countries can effectively leverage open source health information software platforms against a backdrop of reported human capacity challenges. This chapter, therefore, concludes the thesis by presenting conceptual contributions, practical contributions, and concluding remarks which include limitations of the study and possible areas of further research.

8.1 Conceptual Contributions

For some time, there existed a deficiency in literature on software platforms with a focus on developing countries. This deficiency warranted a call in Nielsen (2017) inviting research interest towards digital innovations on software platforms as a research agenda for information systems in developing countries. Despite not setting out as a direct response to Nielsen (2017), this thesis alongside its associated research papers somehow contributes towards alleviating the alleged deficiency noted in the said call. Presented briefly in subsequent paragraphs in this section are some conceptual contributions made by this thesis and its related research papers.

The first conceptual contribution made by the thesis is the concept of *socio-technical generativity*. The concept of generativity, defined as the capacity of a technology or a system to be malleable by diverse groups of actors besides it producers (Zittrain, 2006), has recently gained considerable traction in information systems research in relation to software platforms and software ecosystems (Eck et al., 2015). In this regard, the extent to which external actors, beside the owners, can leverage a software platform to derive desired innovations depends on several factors that influence generativity. Zittrain (2008) views these factors of generativity in terms of the extent of *leverage*, *adaptability*, *ease of mastery*, *accessibility* and *transferability* that a technology artefact possesses. Lane (2011), on the other hand, views generativity in terms of aligned *directedness*, *heterogeneity*, *mutual directedness*, *appropriate permissions* and *action opportunities* exhibited by the social-relationships involving a collective of actors around a technology artefact. Using the case of DHIS2 software platform in Malawi, it has been argued in this thesis that it makes more sense to look at technological or social factors that constrain or enable innovation on software platforms holistically rather than in isolation. Therefore, drawing on the concept of generative technology (Zittrain, 2008, 2006) and the concept of generative relationships (Lane, 2011), the

thesis advances the concept of *socio-technical generativity* in an attempt to provide a holistic account for generativity exhibited by software platforms within their context of use.

One aspect in leveraging open source software platforms involves third-party development of complementary applications. In relation to this, the second contribution made by the thesis is drawing on the boundary resources model by Ghazawneh and Henfridsson (2013) to delineate *capacity building boundary resources* from *software development boundary resources* and also bring *generative capacity* to the foreground alongside boundary resources as factors necessary for third-party development and, by extension, leveraging open source software platforms in developing countries. While agreeing with Ghazawneh and Henfridsson (2013) on the critical role played by software development boundary resources, the thesis uses the case of DHIS2 in Malawi to emphasize the complementary role played by capacity building boundary resources as well as generative capacity, as defined by Avital and Te'eni (2009), in shaping third-party development. To this end, the thesis makes a distinction between *internal generative capacity* involving core developers inside the platform owner's community and *external generative capacity* involving third-party developers outside the platform owner's community. Going further, the thesis presents a generalized capacity building boundary resources model in section 7.2 to highlight the role of *boundary interactions*, *boundary objects* and *brokers* as avenues for capacity building in platform-centric software ecosystems.

Furthermore, the thesis introduced the concept of *fringes* of software ecosystems to describe contexts, typically represented by developing countries, which are peripheral to and disconnected from the context where the software platform is developed, and often characterized by a scarcity of resources necessary for digital innovation. While significant research effort has been expended on software platforms and their respective ecosystems, limited attention to the unfolding of innovation in the fringes. While this is the case, it is usually difficult for software platform owners to adequately address contextual requirements in the fringes because of limited contextual appreciation resulting from the separation of software development from the context of use. Using this concept calls our attention to the importance of mechanisms for propagation of boundary resources and requisite human capacities that stimulate innovation in the fringes of software ecosystems.

8.2 Practical Contributions

From a practical point of view, embracing the socio-technical generativity perspective has implications on the qualities the software platform must possess as well as activities that ought to complement the software platform to actualize the generative potential of the software platform within its context of use. The half-product nature of software platforms demands that they offer a good degree of accessibility, adaptability and ease of mastery to lessen the effort, time and cost required to compose a desired solution within the context of use. This entails, among other things, having in place adequate documentation and mechanisms to facilitate local capacity building. However, the focus should not be on the software platform alone but also on cultivating enabling social relationships between various actors surrounding the software platform, in terms of action opportunities and appropriate permissions for example, within its context of use. Unless there is a generative fit between attributes of the software platform and the prevalent social relationships within context of use desired innovations will remain constrained.

In relation to human capacities needed to leverage open source software platforms in implementation of health information systems in developing countries, the thesis notes that extent to which developing countries can leverage open source software platforms in implementation of health information systems is subject to the availability of human capacities related to *deployment*, *customization*, *use*, *application development* and *system administration*. This calls for a departure from the tendency to emphasize on building end-user capacity when implementing software-based systems in developing countries. In the absence of the other requisite capacities, the extent to which developing countries can leverage software platforms to derive features demanded by current and prospective end-users would be constrained and possibly lead to the obsolescence of platform instances within their context of use. This itemization could, therefore, be instrumental in guiding stakeholders auditing and building requisite human capacities for leveraging open source software platforms in developing countries.

With respect to the said requisite human capacities, drawing on Staring and Titlestad (2008), the thesis goes further to highlight the effectiveness, factors and challenges for *pooling human resources* across organizational boundaries as a short-term strategy to mitigate gaps in human capacity related to deployment, customization, system administration and application development to allow stakeholder organizations in developing countries collaboratively leverage open source

health information software platforms. Where it is possible, pooling human resources should not be employed as a replacement but rather a complement of other capacity-building initiatives

The thesis also offers practical insights on the impact of release management, testing and backward compatibility mechanisms with respect existing instances of software platforms and their associated applications. While software platforms and their corresponding interfaces must evolve to address emerging needs, there is need to minimize the migration burden within the context of use and application breaks due to changes or bugs in new versions of software platforms. In this regard, the thesis reiterates the need for platform owners to announce changes to the platform or its interfaces earlier than they are released and validate the changes in close collaboration with end-users and external application developers to minimize unintended breaks (Bosch, 2010; Bosch and Bosch-Sijtsema, 2010b). For stakeholders in developing countries, already battling the scarcity of resources, the cost, effort and time required to fix such unintended breaks might not be as attractive.

Lastly, the DHIS2 reconfiguration and data migration project described in the case description offers a model architecture for data migration, integration and interoperability between homogeneous or heterogeneous software platform instances. In Malawi, it has been adapted for integration and interoperability between DHIS2 and other systems within the country's health sector.

8.3 Concluding Remarks

For developing countries, leveraging existing open source health information software platforms and their associated applications can be less-risky, less-time consuming, and more cost-effective than implementing inhouse health information systems from scratch. However, by their nature, software platforms are usually not complete solutions but rather provide a foundation on which desired solutions can be constructed. Implementing a desired health information system solution from an open source health information software platform may require extra effort in terms of customization and application development which in turn requires the existence of certain human capacities. Therefore, lack of requisite human capacity, if it exists, can constrain efforts by developing countries to leverage open source health information software platforms despite the promises they hold.

Against this background, the main objective of this thesis has been contributing towards a practical and conceptual understanding of how developing countries can effectively leverage open source health information software platforms against a backdrop of human capacity challenges. For this purpose, a case study involving efforts leveraging the DHIS2 software platform in Malawi was carried out. Empirically, the thesis has discussed leveraging open source health information software platforms in developing countries in relation to a range of requisite human capacities, boundary resources and socio-technical generativity. By leveraging the DHIS2 software platform, CMED and other stakeholders in Malawi, both local and global, have been able to reduce the time, effort, cost and risk associated with putting in place an integrated national health management information system. In this endeavor, software platform attributes, software development and capacity building boundary resources, social relationships amongst various actors, and adequacy of requisite human capacities, or lack thereof, have been pivotal.

8.3.1 Limitations of Study

As typical of most studies, there are recognizable limitations to this study. First of which is the fact that the study has been limited to a single open source software platform, DHIS2. At the same time, the study has been limited to a single developing country, Malawi. However, the efforts on DHIS2 in Malawi bring together multinational actors working with other software platforms as well and there are potential similarities between DHIS2 and other emerging open source software platforms. Nevertheless, similar studies with respect to other health information software platforms and other developing countries can potentially enhance our perspectives and insights on how developing countries can effectively leverage open source software platforms in implementing health information systems.

Furthermore, the study has been limited to the leveraging of software platforms towards health information systems in developing countries but software platforms and indeed information systems challenges in developing countries are not limited to the health sector alone. DHIS2, itself, as a software platform has seen increased adoption in other sectors such as education for example. Therefore, a more general account for the leveraging of open source software platforms in developing countries could even be more beneficial.

With respect to third-party development, to some extent the range of applications available for a software platform depends on how attractive and motivating it is for third-party developers to

develop applications for the software platform in question. The study, this far, has paid limited attention towards factors that would attract and motivate third-party developers to develop applications for an open source software platform whose primary end-user base consists of developing countries.

Lastly, to a large extent the study has focused on perspectives from various actors around an open source software platform within its context of use. It would also be interesting to marry these perspectives with those obtained by a complementary focus on the roles and perspectives held by various actors within the platform owner's community.

8.3.2 Areas of Further Research

Going forward, an area of further research that readily comes to mind relates to the socio-technical generativity concept. Further research might be required to explore in depth the relationships between the highlighted social and technical attributes. Although, an attempt has been made to relate these attributes in section 7.3.4, further research focusing on how these attributes potentially reinforce or counteract each other and their cumulative implications towards generativity in general might be worthwhile.

Another dimension warranting further research relates to motivational factors for third-party development on open source software platforms in developing countries and how this feeds back into the generativity discourse. Already, there have been studies regarding developer motivation factors in the open source community at large, but the developing country context adds a potentially interesting dimension.

References

AbouZahr, C., Boerma, T., 2005. Health information systems: the foundations of public health. Bulletin of the World Health Organization 83, 578–583.

Angell, I.O., Smithson, S., 1991. Information Systems Management: Opportunities and Risks, Information Systems Series. Palgrave Macmillan UK.

Avital, M., Te'eni, D., 2009. From generative fit to generative capacity: exploring an emerging dimension of information systems design and task performance. Information Systems Journal 19, 345–367.

Azubuike, M.C., Ehiri, J.E., 1999. Health information systems in developing countries: benefits, problems, and prospects. The journal of the Royal Society for the Promotion of Health 119, 180–184. https://doi.org/10.1177/146642409911900309

Bakar, A., Sheikh, Y., Sultan, B., 2012. Opportunities and Challenges of Open Source Software Integration in Developing Countries: Case of Zanzibar Health Sector. Journal of Health Informatics in Developing Countries 6.

Baldwin, C.Y., Woodard, C.J., 2008. The Architecture of Platforms: A Unified View (SSRN Scholarly Paper No. ID 1265155). Social Science Research Network, Rochester, NY.

Bansler, J.P., Havn, E.C., 1994. Information systems development with generic systems., in: ECIS. pp. 707–718.

Benbasat, I., Goldstein, D.K., Mead, M., 1987. The Case Research Strategy in Studies of Information Systems. MIS Quarterly 11, 369–386. https://doi.org/10.2307/248684

Berger, T., Pfeiffer, R.-H., Tartler, R., Dienst, S., Czarnecki, K., Wąsowski, A., She, S., 2014. Variability mechanisms in software ecosystems. Information and Software Technology 56, 1520–1535. https://doi.org/10.1016/j.infsof.2014.05.005

Bergman, M., Lyytinen, K., Mark, G., 2007. Boundary objects in design: An ecological view of design artifacts. Journal of the Association for Information Systems 8, 546.

Boerma, J.T., 1991. Health Information for Primary Health Care. African Medical and Research Foundation.

Bosch, J., 2010. Architecture Challenges for Software Ecosystems, in: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume, ECSA '10. ACM, New York, NY, USA, pp. 93–95. https://doi.org/10.1145/1842752.1842776

Bosch, J., 2009. From software product lines to software ecosystems, in: Proceedings of the 13th International Software Product Line Conference. Carnegie Mellon University, pp. 111–119.

Bosch, J., Bosch-Sijtsema, P., 2010a. From integration to composition: On the impact of software product lines, global development and ecosystems. Journal of Systems and Software, SI: Top Scholars 83, 67–76. https://doi.org/10.1016/j.jss.2009.06.051

Bosch, J., Bosch-Sijtsema, P.M., 2010b. Softwares Product Lines, Global Development and Ecosystems: Collaboration in Software Engineering, in: Collaborative Software Engineering. Springer, Berlin, Heidelberg, pp. 77–92. https://doi.org/10.1007/978-3-642-10294-3_4

Boudreau, K.J., Lakhani, K.R., 2009. How to manage outside innovation. MIT Sloan Management Review 50, 69–75.

Braa, J., Monteiro, E., Sahay, S., 2004. Networks of action: sustainable health information systems across developing countries. Mis Quarterly 337–362.

Braa, J., Sahay, S., 2012a. Integrated Health Information Architecture: Power to the Users : Design, Development, and Use. Matrix Publishers.

Braa, J., Sahay, S., 2012b. How to Set-up DHIS2 in a New Context?, in: Integrated Health Information Architecture: Power to the Users : Design, Development, and Use. Matrix Publishers, p. 362.

Braa, K., Nielsen, P., 2015. Sustainable Action Research: The Networks of Actions Approach, in: 13th International Conference on Social Implications of Computers in Developing Countries. Presented at the 13th International Conference on Social Implications of Computers in Developing Countries, Sri Lanka.

Braun, V., Clarke, V., 2006. Using thematic analysis in psychology. Qualitative Research in Psychology 3, 77–101. https://doi.org/10.1191/1478088706qp063oa

Bygstad, B., 2017. Generative innovation: a comparison of lightweight and heavyweight IT. J Inf Technol 32, 180–193. https://doi.org/10.1057/jit.2016.15

Câmara, G., Fonseca, F., 2007. Information policies and open source software in developing countries. Journal of the American Society for Information Science and Technology 58, 121–132. https://doi.org/10.1002/asi.20444

Carlile, P.R., 2002. A pragmatic view of knowledge and boundaries: Boundary objects in new product development. Organization science 13, 442–455.

Carzaniga, A., Fuggetta, A., Hall, R.S., Heimbigner, D., Van Der Hoek, A., Wolf, A.L., 1998. A characterization framework for software deployment technologies. DTIC Document.

Chaulagai, C.N., Moyo, C.M., Koot, J., Moyo, H.B., Sambakunsi, T.C., Khunga, F.M., Naphini, P.D., 2005. Design and implementation of a health management information system in Malawi: issues, innovations and results. Health Policy Plan. 20, 375–384. https://doi.org/10.1093/heapol/czi044

Chavez, C., Terceiro, A., Meirelles, P., Jr, C.S., Kon, F., 2011. Free/Libre/Open Source Software Development in Software Engineering Education: Opportunities and Experiences, in: Fórum de Educação Em Engenharia de Software. Presented at the CBSoft 2011-SBES-FEES, Sao Paulo, Brazil, p. 8.

Clifford, G.D., Blaya, J.A., Hall-Clifford, R., Fraser, H.S., 2008. Medical information systems: A foundation for healthcare technologies in developing countries. BioMedical Engineering OnLine 7, 18. https://doi.org/10.1186/1475-925X-7-18

Cornford, T., Smithson, S., 2005. Project Research in Information Systems: A Student's Guide. Macmillan International Higher Education.

Creswell, J.W., 2009. Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. Sage.

Dittrich, Y., 2014. Software engineering beyond the project – Sustaining software ecosystems. Information and Software Technology 56, 1436–1456. https://doi.org/10.1016/j.infsof.2014.02.012

Eck, A., Uebernickel, F., Brenner, W., 2015. The Generative Capacity of Digital Artifacts: A Mapping of the Field.

Fogel, K., 2005. Producing Open Source Software, 1st ed. O'Reilly.

Free Software Foundation, 2018. What is free software? [WWW Document]. GNU Project - Free Software Foundation. URL http://www.gnu.org/philosophy/free-sw.html (accessed 5.17.18).

Ghazawneh, A., Henfridsson, O., 2013. Balancing platform control and external contribution in third-party development: the boundary resources model. Information Systems Journal 23, 173–192. https://doi.org/10.1111/j.1365-2575.2012.00406.x

Ghazawneh, A., Henfridsson, O., 2010. Governing third-party development through platform boundary resources, in: The International Conference on Information Systems (ICIS). AIS Electronic Library (AISeL), pp. 1–18.

Gizaw, A.A., Bygstad, B., Nielsen, P., 2017. Open generification. Information Systems Journal 27, 619–642. https://doi.org/10.1111/isj.12112

Guba, E.G., 1990. The Paradigm Dialog. SAGE Publications.

Henfridsson, O., Lindgren, R., 2010. User involvement in developing mobile and temporarily interconnected systems. Information Systems Journal 20, 119–135.

Jaffry, S.W. u Q., Kayani, U.R., 2005. FOSS Localization: A Solution for the ICT Dilemma of Developing Countries, in: 2005 Pakistan Section Multitopic Conference. Presented at the 2005 Pakistan Section Multitopic Conference, pp. 1–5. https://doi.org/10.1109/INMIC.2005.334488

Kandar, S., Mondal, S., Ray, P., 2011. A review of Open Source Software and Open Source Movement in Developing Countries. International Journal of Computer Science & Informatics 1, 89–93.

Karuri, J., Waiganjo, P., Orwa, D., Manya, A., 2014. DHIS2: The Tool to Improve Health Data Demand and Use in Kenya. Journal of Health Informatics in Developing Countries 8.

Kimaro, H.C., 2006. Strategies for Developing Human Resource Capacity to Support Sustainability of ICT Based Health Information Systems: A Case Study from Tanzania. The Electronic Journal of Information Systems in Developing Countries 26.

Kimaro, H.C., Nhampossa, J.L., 2005. Analyzing the problem of unsustainable health information systems in less-developed economies: Case studies from Tanzania and Mozambique. Information Technology for Development 11, 273–298. https://doi.org/10.1002/itdj.20016

Lane, D.A., 2011. Complexity and Innovation Dynamics. Handbook on the economic complexity of technological change 63.

Lippeveld, T., Sauerborn, R., Bodart, C., 2000. Design and Implementation of Health Information Systems. World Health Organisation, Geneva.

Littlejohns, P., Wyatt, J.C., Garvican, L., 2003. Evaluating computerised health information systems: hard lessons still to be learnt. BMJ 326, 860–863. https://doi.org/10.1136/bmj.326.7394.860

Maguire, M., Delahunt, B., 2017. Doing a thematic analysis: A practical, step-by-step guide for learning and teaching scholars. AISHE-J: The All Ireland Journal of Teaching and Learning in Higher Education 9.

Manikas, K., Hansen, K.M., 2013. Software ecosystems – A systematic literature review. Journal of Systems and Software 86, 1294–1306.

Miles, M.B., Huberman, A.M., 1994. Qualitative Data Analysis: An Expanded Sourcebook. SAGE.

Ministry of Health, 2018. About the Ministry [WWW Document]. Republic of Malawi Ministry of Health. URL http://www.health.gov.mw/index.php/explore/management (accessed 4.24.18).

Moyo, C., Frøyen, M.H., Sæbø, J.I., Kaasbøll, J.J., 2015. Using Performance League Tables To Promote Accountability And Feedback In Health Management In Malawi, in: Proceedings of the 13th International Conference on Social Implications of Computers in Developing

Countries. Presented at the 13th International Conference on Social Implications of Computers in Developing Countries, IFIP WG 9.4, Negombo, Sri Lanka.
Moyo, C., Kaasbøll, J., Nielsen, P., Sæbø, J., 2016. The Information Transparency Effects of Introducing League Tables in the Health System in Malawi. The Electronic Journal of Information Systems in Developing Countries 75.
Msiska, B., 2017. Pooling human resources needed to leverage open source health information software platforms in developing countries, in: 2017 IST-Africa Week Conference (IST-Africa). Presented at the 2017 IST-Africa Week Conference (IST-Africa), pp. 1–8. https://doi.org/10.23919/ISTAFRICA.2017.8102370
Msiska, B., Nielsen, P., 2018. Innovation in the fringes of software ecosystems: the role of socio-technical generativity. Information Technology for Development 24, 398–421. https://doi.org/10.1080/02681102.2017.1400939
Mutula, S.M., Van Brakel, P., 2007. ICT skills readiness for the emerging global digital economy among small businesses in developing countries: Case study of Botswana. Library Hi Tech 25, 231–245. https://doi.org/10.1108/07378830710754992
Myers, M., 1997. Qualitative Research in Information Systems. Management Information Systems Quarterly 21.
National Statistics Office, 2016. Statistical Yearbook. National Statistics Office, Zomba, Malawi.
Nielsen, P., 2017. Digital Innovation: A Research Agenda for Information Systems Research in Developing Countries. Presented at the 14th International Conference on Social Implications of Computers in Developing Countries, Yogyakarta, Indonesia.
Oak, M., 2007. A review on barriers to implementing health informatics in developing countries. Journal of Health Informatics in Developing Countries 1.
Oates, B.J., 2005. Researching Information Systems and Computing. SAGE.
Ogheneovo, E.E., 2014. On the Relationship between Software Complexity and Maintenance Costs. Journal of Computer and Communications 02, 1–16. https://doi.org/10.4236/jcc.2014.214001
Open Source Initiative, 2018. The Open Source Definition [WWW Document]. Open Source Initiative. URL https://opensource.org/osd (accessed 5.16.18).
Orlikowski, W.J., Baroudi, J.J., 1991. Studying Information Technology in Organizations: Research Approaches and Assumptions. Information Systems Research 2, 1–28. https://doi.org/10.1287/isre.2.1.1
Osch, W., Avital, M., 2010. Generative Collectives. ICIS 2010 Proceedings.
PATH, 2016. An Interim Review of the Health Information Systems Programme - University of Oslo - with Recommendations for Future Action. Seattle.
Paudel, B., Harlalka, J., Shrestha, J., 2010. Open Technologies and Developing Economies, in: Proceedings of CAN InfoTech (IT) Conference 2010.
Pigoski, T.M., 1997. Practical software maintenance: best practices for managing your software investment. Wiley Computer Pub.
Polak, M., 2015. Platformisation of an Open Source Software Product: Growing up to be a generative software platform. University of Oslo, Oslo, Norway.
Prügl, R., Schreier, M., 2006. Learning from leading-edge customers at The Sims: opening up the innovation process using toolkits. R&D Management 36, 237–250.
Qureshi, S., 2013. Networks of change, shifting power from institutions to people: how are innovations in the use of information and communication technology transforming development? Information Technology for Development 19, 97–99.

Roets, R., Minnaar, M., Wright, K., 2007. Open source: towards successful systems development projects in developing countries, in: Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries. Sao Paulo, Brazil.

Sahay, S., Walsham, G., 2006. Scaling of health information systems in India: Challenges and approaches. Information Technology for Development 12, 185–200. https://doi.org/10.1002/itdj.20041

Schwandt, T.A., 2001. Dictionary of Qualitative Inquiry. SAGE Publications.

Sheikh, Y.H., Bakar, A.D., 2012. Open Source Software Solution for Healthcare: The Case of Health Information System in Zanzibar, in: Popescu-Zeletin, R., Jonas, K., Rai, I.A., Glitho, R., Villafiorita, A. (Eds.), E-Infrastructure and e-Services for Developing Countries. Springer Berlin Heidelberg, pp. 146–155.

Silsand, L., Ellingsen, G., 2014. Generification by Translation: Designing Generic Systems in Context of the Local. Journal of the Association for Information Systems 15.

Smith, T.R., 2015. Achieving a Unified System for Monitoring and Evaluation of the Health Sector in Malawi: Principle Barriers and Opportunities for Investment. The Global Fund.

Sommerville, I., 2011. Software Engineering, 9th ed. Pearson Education.

Star, S.L., 2010. This is Not a Boundary Object: Reflections on the Origin of a Concept. Science, Technology, & Human Values 35, 601–617.

Star, S.L., Griesemer, J.R., 1989. Institutional Ecology, "Translations" and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. Social Studies of Science 19, 387–420. https://doi.org/10.2307/285080

Staring, K., Titlestad, O.H., 2008. Development as a Free Software: Extending Commons Based Peer Production to the South. ICIS 2008 Proceedings 50.

Tiwana, A., 2013. Platform Ecosystems: Aligning Architecture, Governance, and Strategy, 1 edition. ed. Morgan Kaufmann, Amsterdam; Waltham, MA.

Tiwana, A., Konsynski, B., Bush, A.A., 2010. Research Commentary—Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics. Information Systems Research 21, 675–687.

University of Oslo, 2016. HISP - Health Information Systems Project, Department of Informatics [WWW Document]. URL http://www.mn.uio.no/ifi/english/research/networks/hisp/ (accessed 4.10.15).

Vaismoradi, M., Turunen, H., Bondas, T., 2013. Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. Nurs Health Sci 15, 398–405. https://doi.org/10.1111/nhs.12048

Van Vliet, H., 2007. Software engineering: principles and practice. Wiley New York.

Vital Wave Consulting, 2009. Health Information Systems in Developing Countries: A Landscape Analysis.

von Hippel, E., Katz, R., 2002. Shifting Innovation to Users via Toolkits. Management Science 48, 821–833. https://doi.org/10.1287/mnsc.48.7.821.2817

Walliman, N., 2011. Research Methods: The Basics. Routledge.

Walsham, G., 2006. Doing interpretive research. European Journal of Information Systems 15, 320–330. https://doi.org/10.1057/palgrave.ejis.3000589

Walsham, G., 1995. Interpretive case studies in IS research: nature and method. Eur J Inf Syst 4, 74–81. https://doi.org/10.1057/ejis.1995.9

Walsham, G., Symons, V., Waema, T., 1988. Information Systems as Social Systems: Implications for Developing Countries. Information Technology for Development 3, 189–204. https://doi.org/10.1080/02681102.1988.9627126

Weerawarana, S., Weeratunge, J., 2004. Open source in developing countries. Sida, Stockholm.

Wenger, E., 2011. Communities of practice: A brief introduction.

Wenger, E., 2000. Communities of practice and social learning systems. Organization 7, 225–246.

WHO, 2004. Developing Health Management Information Systems: A Practical Guide For Developing Countries. World Health Organisation, Geneva.

Yi, Q., Hoskins, R.E., Hillringhouse, E.A., Sorensen, S.S., Oberle, M.W., Fuller, S.S., Wallace, J.C., 2008. Integrating open-source technologies to build low-cost information systems for improved access to public health data. International Journal of Health Geographics 7, 29. https://doi.org/10.1186/1476-072X-7-29

Yildirim, N., Ansal, H., 2011. Foresighting FLOSS (free/libre/open source software) from a developing country perspective: The case of Turkey. Technovation 31, 666–678. https://doi.org/10.1016/j.technovation.2011.07.004

Yunus, M., Jolis, A., 2003. Banker To The Poor: Micro-Lending and the Battle Against World Poverty. PublicAffairs.

Zittrain, J., 2009. Law and Technology: The End Of The Generative Internet. Communications of the ACM 52, 18. https://doi.org/10.1145/1435417.1435426

Zittrain, J., 2008. The future of the Internet and how to stop it. Yale University Press, New Haven, [Conn.].

Zittrain, J., 2006. The Generative Internet. Harvard Law Review 119, 1974–2040.

Appendices

Appendix 1: List of Papers

1. Msiska, B. & Nielsen, P. (2017), A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries, *Proceedings of the 14th International Conference of IFIP Working Group 9.4*, May 2017, Yogyakarta, Indonesia
2. Msiska, B. (2017), Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries, *Proceedings of IST-Africa 2017 Conference*, May 2017, Windhoek, Namibia
3. Msiska, B. & Nielsen, P (2017*),* Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity, *Information Technology for Development, 24(2), pp 388-421*
4. Msiska, B., 2018. Cultivating Third Party Development in Platform-centric Software Ecosystems: Extended Boundary Resources Model. *The African Journal of Information Systems*, *10 (4), Article 6, 348–365*

Appendix 2: Paper 1

Msiska, B. & Nielsen, P. (2017), A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries, *Proceedings of the 14th International Conference of IFIP Working Group 9.4*, May 2017, Yogyakarta, Indonesia

A Framework to Assess and Address Human Capacities Needed to Leverage Open Source Software Platforms in Developing Countries

Brown Msiska(✉) and Petter Nielsen

Department of Informatics, University of Oslo, Oslo, Norway
bmsiska@gmail.com, pnielsen@ifi.uio.no

Abstract. While open source health information software platforms provide developing countries a low-cost, quick and less risky way to build health information systems as compared to in-house solutions, human resource capacity challenges can limit their ability to leverage such platforms. Drawing from a case study focusing on the deployment and operation phases of the DHIS2 platform in Malawi, we observe open source software platforms require a range of human resource capacities that go beyond capacity to use the platform. To fully leverage open source health information software platforms entails the availability of platform usage capacity, platform deployment capacity, platform customisation capacity and platform module development capacity. Most capacity building initiatives for information systems in developing countries have been short-term efforts focused on initial end user capacity to use such systems. However, to cope with rapid innovations and evolution associated with open source software platforms, capacity building ought to be a continuous process encompassing a range of human resource capacities not only use of the platform.

Keywords: Health information systems · Open source software · Software platforms · Software ecosystems · Developing countries

1 Introduction

Health Information System (HIS) integrates data collection, processing, reporting, and use of the information necessary for improving health service effectiveness and efficiency through better management at all levels of the health services [1]. In most developing countries, there are major problems with the quality of health information which is often incomplete, inaccurate and not available at the right time for the right people. Consequently, strengthening HIS is recognised as one of the key activities required to improve the effectiveness and efficiency of health services in developing countries [2].

Resource constraints that characterise most developing countries form a key barrier to global HIS strengthening efforts. Due to high budget deficits, there is underinvestment towards HIS strengthening efforts which limits their ability to acquire appropriate software to drive their HIS operations [2, 3]. To mitigate this, a common strategy in developing countries is to adopt and adapt a generic health information system for use

J. Choudrie et al. (Eds.): ICT4D 2017, IFIP AICT 504, pp. 81–92, 2017.
DOI: 10.1007/978-3-319-59111-7_8

in the local context. This is different than starting from scratch. Adopting generic systems is cheaper, quicker and less risky compared to in-house software development [4]. By adopting generic health information systems, developing countries reduce the cost and more importantly the development time of the systems. Generic systems contain a standardised core made up of common components that remain stable across different contexts and customisable components that vary across contexts and over time [5]. Developing countries leverage these customizable components of generic health information systems to make them fit for use in the local context.

The extent of customisation afforded by generic systems varies. Some generic systems provide resources for developing third party applications (or components) which allow the end user community to extend the functionality of the systems in order to meet specific needs within context use. Such generic systems act as software platforms on which other applications required within context of use can be built by local developers or other third parties. A software platform is a software-based system with an extensible codebase that provides shared core functionality and resources with which derivative application can be created [6, 7]. In this paper, the term health information software platform is used to refer to an extensible generic health information system that provides resources with which derivative applications can be built. This is typical of some contemporary generic health information systems being used in developing countries; such as Open Medical Record System (OpenMRS), District Health Information Software (DHIS2), and Open Logistics Information System (OpenLMIS) to mention a few.

Traditionally, software is distributed under a proprietary license which requires clients to pay annual license fees for using the software and prevents them from modifying and redistributing the software [8]. With the prevailing resource constraints, the cost of ownership and freedom to modify software to fit local settings are major concerns for developing countries [3]. Open-source software provides an opportunity to build low-cost health information systems allowing countries with modest resources access to modern data analysis and visualization tools [9]. Open-source software, abbreviated as OSS, is software that is made available with source code and is provided under a software license that permits users to study, change, and improve the software [10]. By granting the user community access to the source code, open source software allows them to create and extend the software with local innovations that fit the local context. Furthermore, open source software is viewed as a tool for technical self-sufficiency as it eliminates the need for outside consultants and reduces costs of ownership of software [11]. Consequently, the use of generic open source health information systems in developing countries is common [2, 3]. The convergence of open source software and software platform approaches in the design and development of health information systems has given rise to open source health information software platforms with DHIS and OpenMRS as popular examples.

HIS strengthening initiatives in developing countries have a history of failure and unsustainability resulting from a number of factors. One important contributing factor is the lack of human resource capacity to use, develop and maintain the systems [12]. Furthermore, finding skilled developers in developing countries is a struggle due to lack of training and brain drain [13]. Other studies also show that developers participating in and contributing to open source projects are predominantly from developed

countries [14, 15]. However, HIS cannot deliver expected benefits unless they are supported by appropriate human resource capacity locally [12]. Thus, to fully leverage open source health information software platforms appropriate human resource capacity must exist.

Software platforms are to some extent "half products" which have to be customised and/or extended before they can be fit for use in a particular context [16]. They constitute a departure from an era where end-users got a fully-fledged solution from a software vendor to an era where software vendors deliver solutions that must be completed by the end-user community within the context of use. Because of this, the capacity requirements for leveraging open source health information platforms vary from those of traditional HIS software. Platforms do not only allow for, support and encourage, but mandates local initiatives and innovations. The purpose of this paper is therefore to empirically address the question: what human resource capacities are required to leverage open source health information software platforms within context of use? By addressing this question the paper aims to strengthen our understanding of the implications open source health information software platforms have on HIS capacity building initiatives in developing countries. Beyond identifying concrete capacities in the case of HIS in Malawi, we also contribute with a general framework to assess and address human capacities needed to leverage open source software platforms in developing countries.

2 Software Platforms and Ecosystems

A software platform is an extensible codebase of a software based system that provides core functionality as well as an interface shared by the modules that interoperate with it [17]. Building on this definition others have gone further to describe a software platform as a software based system with extensible codebase that provides shared core functionality and resources from which derivative applications are generated [6, 7]. A software ecosystem is the software platform and the collection of modules specific to that platform [17]. Thus, the software platform is just one among other parts that make up a software ecosystem. Within the ecosystem, the software platform is complemented by modules (or applications). Modules are add-on software subsystems that connect and add new functionality to the software platform [17]. Associated with software platforms and software ecosystems is a shift from a closed software product-line based development approach to an ecosystem based development approach [16].

With the ecosystem based development approach, the software platform is rarely a solution itself; its functionality is exposed to end users through applications running on top of it. This shift in development approach comes with its own challenges. A key challenge emerges from the separation between the platform developer and platform's context of use, as well as the diversity of user requirements. Challenges faced by platform developers include making an informed decision on what applications or services to develop [18] and how to effectively respond to turbulent information processing requirements within context of use [7]. These challenges make the involvement of third party application developers close to context of use increasingly attractive for software platform developers [19, 20] as it allows them to focus on the core extensible

base of the platform and defer satisfying specific user needs to third party developers. The proximity of third party developers to context of use enables them to make informed decisions on add-on modules and to develop them in response to particular needs. At the same time, deferring development to third parties creates new local capacity requirements within contexts of use. This makes the human resources capacity challenge prominent in initiatives aimed at assuring sustainability of health information system platforms in developing countries. This is the key challenge addressed in this paper.

3 Software Platforms and Human Resource Capacity Requirements

Once commissioned, a software system undergoes a number of phases within its context of use. For the purpose of this paper, distinction is made between two major phases: deployment phase, and operation phase. Software deployment is a process comprising all activities carried out in order to make a software system available for use [21]. This includes among other things setting up the required hardware and software environment; installing the software system in question; piloting and adapting it for local use; and testing it against functional and nonfunctional requirements to determine its readiness for use. Once deemed ready for use, the software system is rolled out into operation; ushering it into the operation (or productive use) phase (Fig. 1).

Fig. 1. Software system timeline within context of use

Once the software system is put into operation, anomalies are discovered, operating environments change, and new user requirements emerge and the software system must change accordingly [22]. The process of changing a software system or its component to correct faults, improve performance or other attributes, or adapt to a changed environment after it has been put into operation is called software maintenance [23, 24]. Software maintenance activities span a system's productive life cycle and 70% of all effort on a software system is estimated to be expended on maintenance alone [25]. A range of human resource capacities are therefore required during both the deployment and operation phases of a software system. For software platforms, in particular, the lack of appropriate human resource capacities in the deployment and operation phases can constrain their implementation, maintenance and hence sustainability within context of use. Understanding what human resource capacities are required for open source software platforms within each of these phases can therefore be instrumental in ensuring their success and sustainability in developing countries.

4 Methodology

This paper is based on a qualitative case study carried out in Malawi with DHIS2 as the focal software platform under study. DHIS2 is a web based, free, generic and open source health information software platform. It is currently the leading solution for aggregate health data and is being used in more than 50 developing countries by government ministries, donor agencies and NGOs [26]. DHIS2 is a flexible metadata-driven software that affords implementers the flexibility to customize its data model to fit the data needs of particular end-users. Besides the flexible data model, DHIS2 also allows customisation of data entry forms and reports. In addition, DHIS2 comes with a RESTful Web API that allows extension of its functionality through third party innovations built using common web technologies such as JavaScript, CSS and HTML5. In addition to applications running on top of it through the Web API, DHIS2 also allows development of custom software modules that sit side by side with its core modules. The introduction of such extensibility features into DHIS2 has seen it evolve from a traditional health information system to a software platform supporting open innovation through extensions by other developers within its ecosystem other than the core software development team at the University of Oslo [26, 27].

DHIS2, in Malawi, falls under the custody of the Central Monitoring and Evaluation Division (CMED) in the Ministry of Health (MoH). CMED is in charge of collecting and analyzing aggregate data used to monitor and evaluate various health programmes run by MoH. The DHIS2 platform is used by CMED and other stakeholders to collect, store, analyse and visualize aggregated data for decision making purposes. Therefore, the case study largely focused on CMED and key stakeholders involved in the deployment and operation of DHIS2 in Malawi. The aim of the case study was to address the question what human resource capacities are required to leverage open source health information software platforms within context of use. To address this question, we started by purposively sampling [28] key personnel from CMED and other stakeholders involved in the deployment and operation phases of DHIS2 as respondents for the study. The respondents were drawn from Ministry of Health and CMED, HISP Malawi, University of Malawi (UNIMA) and Baobab Health Trust. We then conducted semi-structured interviews with the selected respondents in order to establish prevailing human resource capacity requirements and challenges during the deployment phase and the subsequent operation phase of DHIS2 (Table 1).

Table 1. Respondents

Organisation	No. respondents	Designation
Ministry of Health/CMED	5	Director, CMED Chief Technical Assistant, HMIS DHIS2 Technical Assistants (3)
HISP Malawi	3	Board Members (3)
Baobab Health Trust (BHT)	4	Executive Director Software Development Manager DHIS2 Integration Team Members (2)
UNIMA	1	HISP Malawi Representative

Further data was collected through participatory observation which involved attending stakeholder meetings, training workshops, and working with the DHIS2 implementation team in Malawi comprising of staff from CMED/HISP Malawi and other strategic local partners. As part of these observations, one of the researchers worked as part of the DHIS2 implementation team in Malawi in addition to being a member of the team of trainers in a DHIS2 Application Development Workshop that took place in March 2016. In addition, data was also collected through document reviews comprising of reports on DHIS2 in Malawi by practitioners and other researchers.

The data yielded from the interviews, participatory observations and document reviews was thematically analysed [28] resulting in grouping of human resource capacities and challenges identified according to the two phases: deployment and operation. The capacities and challenges were further analysed with respect to the kind of human resource capacity; resulting in a framework which we present in our discussion. In the next section, we describe the human resource capacity requirements and challenges during the deployment and operation of DHIS2 in Malawi identified during the case study. This is followed by the discussion and later on, concluding remarks.

5 DHIS2 Software Platform in Malawi

A case study was carried out in Malawi focusing on the deployment and operation phases on the DHIS2 software platform. The aim of the case study was to establish prevailing human resource requirements and challenges in each of the phases. The results of the case study are presented in Sects. 5.1 and 5.2.

5.1 Human Resource Capacity Requirements in the Deployment Phase of the DHIS2 Platform in Malawi

DHIS2 deployment in Malawi commenced in 2009 as a pilot involving three districts; Blantyre, Zomba and Lilongwe. The pilot project ran for a period of three years and resulted in DHIS2 being rolled out to all districts and all health programmes in Malawi in 2012. The deployment of DHIS2 in Malawi involved: setting up a web server on which to run the platform; installing the DHIS2 platform on the web server; defining metadata for the platform in terms organizational units, users and user roles, data elements and indicators for data sets selected for the pilot. The deployment of DHIS2 therefore required appropriate human resources capacity to set up the web server, installing DHIS2, and define required metadata for selected data sets. Once the platform was deployed the default data entry forms and reports generated by DHIS2 turned out significantly different from the paper based forms and reports that were currently in use by end users. To preserve familiarity and ease the transition towards using DHIS2, custom reports and forms had to be designed and implemented. Implementing such custom forms and reports required the availability of human resource capacity to create custom forms and reports in DHIS2.

The human resource capacity requirements mentioned above came with a challenge: the lack of ICT personnel in CMED to carry out the deployment and customization tasks

required for the deployment of DHIS2. To make ICT human resources available to CMED for the deployment of DHIS2 a local HISP node, HISP Malawi, was established. With funding acquired through the University of Oslo, HISP Malawi recruited two programmers on contract and placed them on secondment to CMED. These were joined by ICT staff from the University of Malawi constituent colleges: Chancellor College, College of Medicine and the Malawi Polytechnic. The pool of ICT personnel from HISP Malawi and University of Malawi underwent training on DHIS2 to enable them implement the deployment and customization tasks related to the pilot. In addition to the technical human resource capacities, there was a need for end users participating in the pilot study to be able to use DHIS2. Therefore, end users participating in the pilot study underwent training to enable them use DHIS2.

5.2 Human Resource Capacity Requirements in the Operation Phase of the DHIS2 in Malawi

Once a decision was made to roll out DHIS2 to all districts in Malawi, the demand for human resource capacity to use the software platform grew. As a result, a series of training workshops have been conducted targeting end-users of the software platform. Such trainings included, among others, a training of trainers workshop in August 2012 and a series of DHIS2 mobile trainings [29]. "The end user training employed a cascade approach" (Director, CMED). Trainer of Trainers (TOTs) at national level were identified and trained and the TOTs in turn trained district trainers who undertook training of health workers in their respective districts. However, we found out during the case study that end-user training has not been done regularly. As a result, there is still a backlog on untrained staff. Furthermore, changes in newer versions of DHIS2 have left a number of end-users requiring re-training.

At the same time, rolling out DHIS2 to all health programmes under the ministry of health required further deployment and customisation tasks in terms of metadata and implementation of custom forms and reports for the programmes that were not part of the pilot. Furthermore, frequent releases of newer versions of DHIS2 necessitated deploying and transitioning to newer versions of DHIS2. As a result, the demand for human resource capacity to deploy and customize the software platform grew as well. To catch up with advances in DHIS2 that came with each release members of the DHIS2 technical team have attended a number of regional DHIS2 Academies. Through the academies they acquired a range of knowledge required to install and customize DHIS2.

During the period DHIS2 has been in operation there have been a number human resource capacity challenges. First of all, staff turnover involving technical staff at HISP Malawi has threatened the day to day operations of DHIS2. All technical assistants recruited by HISP Malawi and placed on secondment to CMED between 2012 and 2015 have left for various reasons including funding for their retention. Currently, HISP Malawi has two technical assistants recruited towards the end of 2015. These were fresh graduates from the University of Malawi and have had to attend a series of DHIS2 academies to bring them up to speed with DHIS2.

In addition, the lack of ICT staff in CMED has left it dependent on external expertise, both local and foreign, to keep DHIS2 in operation. The MoH, like all other

ministries in the Malawi Government, has an ICT department which has an allocation of ICT staff from the Department of e-Government in the Office of the President and Cabinet. However, until recently none of the ICT staff in the ICT department have been working with CMED on DHIS2. CMED has therefore relied on HISP Malawi and other external expertise to keep DHIS2 in operation. CMED sees the continued reliance on external expertise as a threat to the sustainability of DHIS2 and has been lobbying government to adjust its staff establishment to include ICT personnel. "We came up with a position paper requesting for ICT personnel under CMED … but things take time in government so we are not sure when that will happen" (Chief Technical Assistant, CMED). The decision to adjust CMED's staff establishment is however still outstanding.

Furthermore, some problems and requirements CMED has had with respect to DHIS2 required changing or developing new DHIS2 modules. Handling such requirements and problems required human resource capacity to develop or modify DHIS2 modules. These have usually been reported to DHIS2 core developers in Norway because of lack of necessary human resource capacity locally. As part of efforts to address this gap, In March 2016, a DHIS2 Application Development workshop took place at University of Malawi, Chancellor College from 7th March to 16th March 2016 [30]. The training introduced participants to the DHIS2 API and how to develop DHIS2 applications using the API. The training was funded by University of Oslo with support from UNICEF. It attracted participants from Kenya, Ethiopia, Zambia and Malawi including the two technical assistants at CMED.

6 Discussion

Analysing the deployment and operation phases of DHIS2 in Malawi reveals four categories of human resource capacity needed to leverage open source software platforms. Before a software platform can be put into operation it must be deployed first and this requires the availability of platform deployment capacity which includes ability to setup the platform operating environment and install the software platform. Once the software platform is installed, its customisable components have to be customised to fit local needs. This entails platform customisation capacity. Historically, donor driven HIS projects in developing have used foreign experts to mitigate the gap in deployment and customisation capacity in the deployment phase.

Once the system is put into operation it requires human capacity for its maintenance, including the development of new platform modules. This entails platform module development capacity. Such capacity is instrumental to third party innovations which respond to specific needs within context of use. Maintenance often happens at a time when donor and external support is no longer available leading to sustainability challenges where local capacity is lacking. Both during deployment and maintenance the software system is subject to use by end users. In the deployment phase this happens as a result of piloting and testing of the system against end user requirements. Therefore in both phases there is a requirement for human capacity to use the software platform, hereby referred to as platform usage capacity. Furthermore, during the

Table 2. Human capacity requirement for software platforms within context of use

Phase	Key activities	Human capacities required
Deployment	– Setting up hardware and software environment – Installation of the software platform – Customisation – Testing/Piloting system	– Platform deployment capacity – Platform customisation capacity – Platform usage capacity
Operation	– Correcting system faults – Upgrading to newer software versions – Customising new versions – Develop platform modules – Test implemented changes	– Platform deployment capacity – Platform customisation capacity – Platform module development capacity – Platform usage capacity

maintenance phase there are episodes of deployment where the software system is upgraded to a newer version. These capacity needs are summarized in Table 2.

The extent to which developing countries are able to leverage open source software platforms is, therefore, subject to the availability of human capacities to deploy, customize and use the platforms; complemented by capacity to develop platform modules is response to new requirements or those not adequately addressed by platform developers. This we summarise in the model in Fig. 2, showing a ladder of human resource capacities required to leverage software platforms within context of use. The model can act as a framework informing stakeholders implementing of open source health information software platforms in developing countries what human resource capacities to put in place in order to fully leverage the software platforms. At the same time, it could be used to inform efforts to assess and address gaps in human resource capacities needed to leverage open source platforms where they have been implemented.

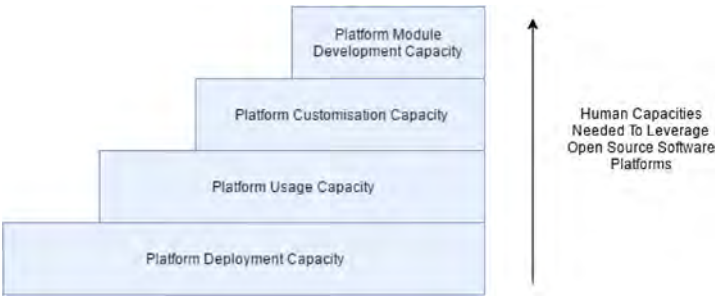


Fig. 2. A model for human capacities needed to leverage open source software platforms

Many of donor-driven HIS in developing countries have ended up as unsustainable and/or failures due to a lack of local human resource capacity left behind after implementation [31]. During implementation, donors and their agents have traditionally filled the human resources gap by engaging foreign experts at the expense of building local expertise [12]. While this works well in the short term it creates long

term challenges with maintenance and sustainability. Without requisite capacity, locals fail to support and maintain the solutions leading to gradual decay and obsolescence of the solutions as they fail to respond to emerging needs within context of use. As shown in the model above, open source software platforms come with an implicit demand for local capacity not only in terms of use but also in terms of customizing and extending the platform to meet local needs. This supports the argument made by Sahay and Walsham [32] who state that health information systems need to be accompanied by the scaling of local human resources capacity at least two levels: level of end users and level of the implementation team.

The introduction of health information systems in developing countries is often accompanied by short-term training aimed at building the capacity of end-users to use the systems. This is important, but not enough to address long-term learning needs of end users and the implementation and maintenance team to stay up to date as the health information system evolves overtime. Capacity building is a continuous process [12]. To cope with rapid innovations and evolution associated with health information software platforms and open source software platforms in general, capacity building ought to be a continuous process and include more than the platform usage capacity. Associated with increasing complexity of health information systems is the need to scale the technical competence of users and that of the implementation team responsible for providing technical support to the users and the user organization [32]. With the shift towards health information software platforms, the issue of human resources capacity building in developing countries involves not only building capacity of local end-users and local implementation team but also ensuring that there is continuous learning to enable them cope with the rapid evolution and innovations that characterise software platforms.

7 Conclusion

Open source health information software platforms provide developing countries a low-cost, quick and less risky way to build health information systems compared to developing in-house solutions. However, software platforms also place new capacity requirements within context of use as compared to traditional off-the-selves or commissioned software systems. The availability of human resource capacity to deploy, use and maintain a health information software platform can have an influence on the extent to which developing countries can leverage the potential of these platforms. Thus, capacity building initiatives around health information software platforms should strive to build this range of capacities. Short-term training aimed at building the capacity of end-users to use the systems is important, but not enough to address long-term learning needs of end users and the implementation and maintenance team to stay up to date as the health information system evolves overtime. To cope with rapid innovations and evolution associated with open source software platforms, capacity building ought to be a continuous process covering a range of human resource capacities not only use of the platform.

References

1. WHO: Developing Health Management Information Systems: A Practical Guide For Developing Countries. World Health Organisation, Geneva (2004)
2. Karuri, J., Waiganjo, P., Orwa, D., Manya, A.: DHIS2: the tool to improve health data demand and use in Kenya. J. Health Inform. Dev. Countries **8** (2014)
3. Sheikh, Y.H., Bakar, A.D.: Open source software solution for healthcare: the case of health information system in Zanzibar. In: Popescu-Zeletin, R., Jonas, K., Rai, Idris A., Glitho, R., Villafiorita, A. (eds.) AFRICOMM 2011. LNICSSITE, vol. 92, pp. 146–155. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29093-0_14
4. Bansler, J.P., Havn, E.C.: Information systems development with generic systems. In: ECIS 1994, pp. 707–718 (1994)
5. Silsand, L., Ellingsen, G.: Generification by translation: designing generic systems in context of the local. J. Assoc. Inf. Syst. **15**, 177–196 (2014)
6. Eck, A., Uebernickel, F., Brenner, W.: The generative capacity of digital artifacts: a mapping of the field. In: Proceedings of PACIS 2015 (2015)
7. Ghazawneh, A., Henfridsson, O.: Balancing platform control and external contribution in third-party development: the boundary resources model. Inf. Syst. J. **23**, 173–192 (2013)
8. Fogel, K.: Producing Open Source Software, 1st edn. O'Reilly, Sebastopol (2005)
9. Yi, Q., Hoskins, R.E., Hillringhouse, E.A., Sorensen, S.S., Oberle, M.W., Fuller, S.S., et al.: Integrating open-source technologies to build low-cost information systems for improved access to public health data. Int. J. Health Geogr. **7**, 29 (2008)
10. Kandar, S., Mondal, S., Ray, P.: A review of open source software and open source movement in developing countries. Int. J. Comput. Sci. Inform. **1**, 89–93 (2011)
11. Vital Wave Consulting: mHealth for Development: The Opportunity of Mobile Technology for Healthcare in the Developing World, Washington, D.C. (2009)
12. Kimaro, H.C.: Strategies for developing human resource capacity to support sustainability of ICT based health information systems: a case study from Tanzania. Electron. J. Inf. Syst. Dev. Countries **26**, 1–23 (2006)
13. Roets, R., Minnaar, M., Wright, K.: Open source: towards successful systems development projects in developing countries. In: Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries, Sao Paulo, Brazil (2007)
14. Paudel, B., Harlalka, J., Shrestha, J.: Open technologies and developing economies. In: Proceedings of CAN InfoTech (IT) Conference (2010)
15. Weerawarana, S., Weeratunge, J.: Open Source in Developing Countries. Sida, Stockholm (2004)
16. Dittrich, Y.: Software engineering beyond the project – sustaining software ecosystems. Inf. Softw. Technol. **56**, 1436–1456 (2014)
17. Tiwana, A., Konsynski, B., Bush, A.A.: Research commentary—platform evolution: coevolution of platform architecture, governance, and environmental dynamics. Inf. Syst. Res. **21**, 675–687 (2010)
18. Henfridsson, O., Lindgren, R.: User involvement in developing mobile and temporarily interconnected systems. Inf. Syst. J. **20**, 119–135 (2010)
19. Bosch, J.: From software product lines to software ecosystems. In: Proceedings of the 13th International Software Product Line Conference, pp. 111–119. Carnegie Mellon University (2009)
20. Boudreau, K.J., Lakhani, K.R.: How to manage outside innovation. MIT Sloan Manag. Rev. **50**, 69–75 (2009)

21. Carzaniga, A., Fuggetta, A., Hall, R.S., Heimbigner, D., Van Der Hoek, A., Wolf, A.L.: A characterization framework for software deployment technologies. DTIC Document (1998)
22. Pigoski, T.M.: Practical Software Maintenance: Best Practices for Managing Your Software Investment. Wiley, New York (1997)
23. Sommerville, I.: Software Engineering, 9th edn. Pearson Education, Boston (2011)
24. Van Vliet, H.: Software Engineering: Principles and Practice. Wiley, New York (2007)
25. Ogheneovo, E.E.: On the relationship between software complexity and maintenance costs. J. Comput. Commun. **02**, 1–16 (2014)
26. PATH: An Interim Review of the Health Information Systems Programme - University of Oslo - with Recommendations for Future Action, Seattle, September 2016
27. Polak, M.: Platformisation of an Open Source Software Product: Growing up to be a Generative Software Platform. University of Oslo, Oslo (2015)
28. Creswell, J.W.: Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. Sage, Los Angeles (2009)
29. HISP Malawi: DHIS2 TOT Training [Internet]. DHIS2 TOT Training. http://hispmalawi.org.mw/index.php?page=pages&pid=39. Accessed 7 Nov 2016
30. University of Malawi: Computer Science Department holds DHIS2 Workshop - University of Malawi|Chancellor College [Internet]. Computer Science Department holds DHIS2 Workshop (2016). http://cc.ac.mw/news/computer-science-department-holds-dhis2-workshop-17-03-2016
31. Kimaro, H.C., Nhampossa, J.L.: Analyzing the problem of unsustainable health information systems in less-developed economies: case studies from Tanzania and Mozambique. Inf. Technol. Dev. **11**, 273–298 (2005)
32. Sahay, S., Walsham, G.: Scaling of health information systems in India: challenges and approaches. Inf. Technol. Dev. **12**, 185–200 (2006)

Appendix 3: Paper 2

Msiska, B. (2017), Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries, *Proceedings of IST-Africa 2017 Conference*, May 2017, Windhoek, Namibia

IST-Africa 2017 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2017
ISBN: 978-1-905824-56-4

Pooling Human Resources Needed to Leverage Open Source Health Information Software Platforms in Developing Countries

Brown MSISKA
Department of Informatics, University of Oslo, Gaustadalléen 23 B, 0373 Oslo, Norway
Tel: +4748635839, Email: bmsiska@gmail.com

Abstract: While open source software platforms have emerged as viable options for building health information systems in developing countries, challenges in terms of requisite human resource capacities in some developing countries limit their ability to fully leverage the opportunities such platforms offer. Creating a shared pool of talent across multiple stakeholders has been suggested as a possible strategy to mitigate challenges with human resource capacities in developing countries. However, there is a dearth of HIS or open source literature on the efficacy of pooling human resources capacities needed to leverage open source health information software platforms. This paper contributes by identifying occurrences of staff pooling during the implementation and reconfiguration of the DHIS2 health information software platform in Malawi and assessing the efficacy, challenges and factors for staff pooling as a strategy to mitigate human resource challenges in developing countries.

Keywords: Health Information Systems, Open Source Software, Software Platforms, Developing Countries, Human Resource Capacity, Staff Pooling.

1 Introduction

Software platforms have emerged as a viable option for building different information systems. A software platform is a customisable software based system with an extensible codebase that provides shared core functionality and resources with which derivative applications can be created [1, 2]. Successes registered by consumer-oriented software platforms such as Android and iOS have garnered attention for software platforms in other spheres including in the area of health information systems (HIS). Health information software platforms are particularly attractive for developing countries where high budget deficits affect HIS strengthening efforts [3, 4]. For such countries, leveraging software platforms provide a cheaper, quicker and less risky option as compared to building in-house HIS solutions from scratch.

The term open source software refers to software that is made available with source code and is provided under a software license that permits users to study, change, and improve the software [5]. For developing countries, open source software provides an opportunity to build low-cost health information systems allowing them access to modern data analysis and visualization tools [6]. The convergence of open source software and software platform approaches in the design and development of health information systems has given rise to open source health information software platforms. Currently, there are a number of open source health information software platforms being used in developing countries with DHIS2 and OpenMRS as leading examples

One of the major causes of failure and unsustainability of HIS in developing countries has been lack of appropriate human capacities for their use, development and maintenance [7, 8]. However, to fully leverage open source software platforms human resource

capacities beyond mere use are required. This is the case because software platforms are to some extent "half products" which have to be customised and/or extended before they can be fit for use in a particular context [9].

HIS in developing countries are characterised by a multiplicity of stakeholders. Such stakeholders are likely to have varying levels of human resource capacities required to leverage open source health information software platforms. Creating a shared pool of talent across multiple stakeholders has been suggested as a possible strategy to mitigate challenges with human resource capacities needed to leverage the platforms in question [10]. Notwithstanding this suggestion, there is a gap in HIS and open source software literature addressing the effectiveness, challenges and factors for staff pooling as means to address the aforementioned capacity gaps. This paper, therefore, explores the effectiveness, challenges and factors for staff pooling as means for addressing gaps in human resource capacities required to leverage open source health information software platforms in developing countries. This is done by tracing and assessing occurrences of staff pooling around the DHIS2 software platform in Malawi.

2 Software Platforms and Human Resources in Developing Countries

Software platforms are a kind of generic software. Generic software systems contain a standardised core made up of common components that remain stable across different contexts and customisable components that vary across contexts and over time [11]. The customisable components make the software system malleable by other actors other than the software vendor; allowing them to configure the system to fit the needs of a particular context of use. A software platform is an extensible software based system that provides core functionality as well as an interface shared by applications that interoperate with it [12]. Associated with a software platform is a software ecosystem and applications specific to that platform. A software ecosystem comprises of the software platform and the collection of application specific to that platform [12]. Applications or modules, on the other hand, are add-on software subsystems that connect and add new functionality to the software platform.

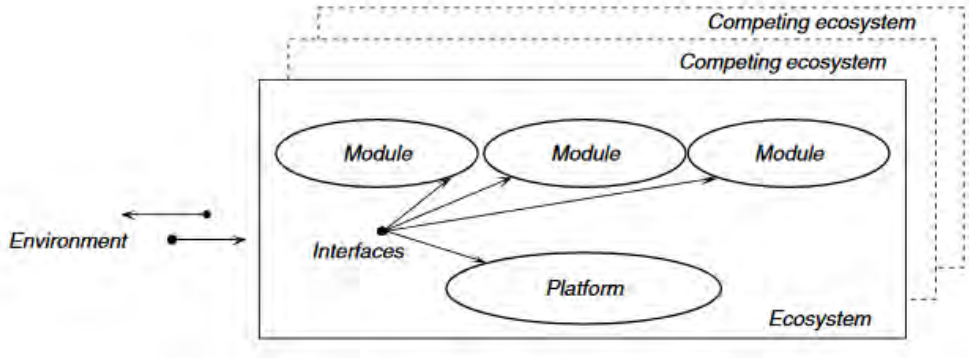


Figure 1: Elements of a Software Platform-Centric Ecosystem (source[12])

Software platforms represent a shift from an era where end-users got a fully-fledged solution from a software vendor to one where software vendors deliver solutions that must be completed by end-users through modules (applications) developed by local developers or other third parties. However, deferring part of the software development to end-users entails the availability of other human resource capacities beyond those required by traditional software. Thus, for developing countries to fully leverage open source health information software platforms, appropriate human resource capacity must exist since lack of appropriately trained staff can lead to the decay and obsolescence of ICT solutions [13]. Therefore, ensuring the existence of requisite human resource capacities is one of the

challenges stakeholders leveraging open source health information software platforms in developing countries have to deal with.

Creating a large shared pool of talent across multiple stakeholders has been suggested as a possible strategy to mitigating challenges with human resource capacities needed to leverage open source solutions in developing countries [10]. With respect to human resources, the term pool refers to a group of people available for work when required or considered as a resource [14]. Pooling is the process of putting together and sharing resources for the benefit of all involved. HIS initiatives in developing countries bring together multiple stakeholders with varying human resource capacities. Going by this suggestion, stakeholders in developing countries can pool and collectively exploit human resource capacities at their disposal to enable them better leverage open source health information software platform they mutually benefit from. However, there is a dearth of HIS and open source software literature addressing the effectiveness, challenges and factors for staff pooling as means to address human resource capacity gaps around open source health information software platforms in developing countries. There is therefore a need to ascertain the efficacy of staff pooling in this regard, factors that work in its favour and potential challenges. Responding to this need is the objective and envisaged contribution of this paper.

3 Methodology

This paper is based on a case study [15] of the DHIS2 software platform in Malawi, tracing and assessing occurrences of staff pooling by various stakeholders in order to collaboratively address human resource capacity gaps during the implementation and subsequent productive life of the software platform. The study contributes to a global action research initiative, Health Information Systems Programme (HISP), coordinated by the Department of Informatics at University of Oslo, Norway. HISP is a collaborative global network with regional and country nodes worldwide. Within each node are several stakeholders, including health administration units (community, sub-district, district, provincial, and national), researchers, universities, NGOs, and funding providers. HISP aims to enable and support developing countries strengthen their health information systems in order to improve management and delivery of health services.

Central to the aim of HISP is DHIS2, an open source health information software platform developed by the HISP team at University of Oslo. Malawi is one of several developing countries in sub Saharan Africa that are using DHIS2. The Central Monitoring and Evaluation Division (CMED) in the Ministry of Health (MoH) in Malawi is the custodian of DHIS2. The case study therefore involved CMED and other key stakeholders around DHIS2 in Malawi. Data was collected through semi-structured interviews and participatory observations. Semi-structured interviews were conducted with personnel from CMED and other key stakeholders involved in the implementation and maintenance of the DHIS2 software platform in Malawi. Participatory observations involved the researcher joining and working with a team of technical personnel working on project aimed at reconfiguring DHIS2 in Malawi over a period of 3 months. Notes were taken to capture important issues emerging from interviews. Field notes were written with respect to participatory observations. Thematic analysis, a qualitative data analysis technique involving the identification of patterns (or themes) within data [15], followed after the data collection.

4 Case Description

DHIS2 was introduced in Malawi in 2009 through a series of pilots involving three districts: Blantyre, Zomba and Lilongwe. The DHIS2 pilot project ran from 2009 to 2012.

In 2012, DHIS2 was rolled out to all districts as the backbone of National Health Management Information System (HMIS). The roll out attracted support from various local and international stakeholders including: Norwegian Agency for Development (Norad), Support for Service Delivery Integration (SSDI), United Nations Children's Fund (UNICEF), Centre for Disease Control and Prevention (CDC), International Training and Education Centre for Health (I-TECH), HISP Malawi, University of Malawi and the University of Oslo. Since 2009 DHIS2 has been hosted by the College of Medicine, a constituent college of the University of Malawi. Currently, there is an ongoing DHIS2 reconfiguration project. As part of this project, a new instance of DHIS2 is being implemented and is being hosted in the MoH server room located at Department of HIV and AIDS (DHA) offices. Data is being migrated from the old DHIS2 instance at College of Medicine to the new instance at DHA. Once the project is completed the DHIS2 instance at College of Medicine will serve as a backup. The timeline in Figure 2 offers a snapshot of key DHIS2 events in Malawi. Occurrences of staff pooling identified across these key events are described in sections 4.1 and 4.2.

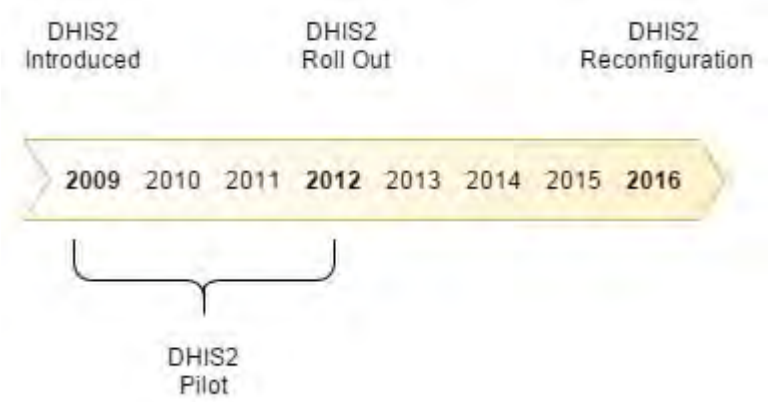


Figure 2: Timeline of DHIS2 in Malawi

4.1 Pooling Human Resource Capacities for DHIS2 Piloting and Implementation

One of the challenges that was identified during the piloting and implementation of DHIS2 was the absence of ICT personnel in CMED. The staff establishment of CMED consists of statisticians and economists. As a result, there were no ICT personnel within the division to drive the implementation of DHIS2. MoH has an ICT department but members of staff in the department mainly focus on maintenance of ICT equipment and running the ministries payroll system. Until recently none of the staff in the MoH ICT department had been working with CMED on DHIS2. Consequently, there was lack of requisite human resource capacity needed to deploy and leverage the DHIS2 software platform.

To address the lack of technical capacity in CMED and MoH a local HISP node, HISP Malawi was established. HISP Malawi recruited two programmers and placed them on secondment to CMED as technical assistants on DHIS2. These were joined by an IT expert from I-TECH seconded to CMED as the chief technical assistant. Additional IT staff came from the University of Malawi through its constituent colleges; Chancellor College, Malawi Polytechnic and College of Medicine. Furthermore, PhD and master students from the University of Oslo as well as foreign experts from other HISP nodes were at different times engaged to assist with the piloting and implementation of DHIS2 in Malawi. With a pool of IT personnel from different stakeholders, CMED was able to implement DHIS2 and rolled it out to all districts in Malawi in 2012. Below, Figure 3 summarises the pooling of human resource capacities by various stakeholders during the implementation of DHIS2 in Malawi.

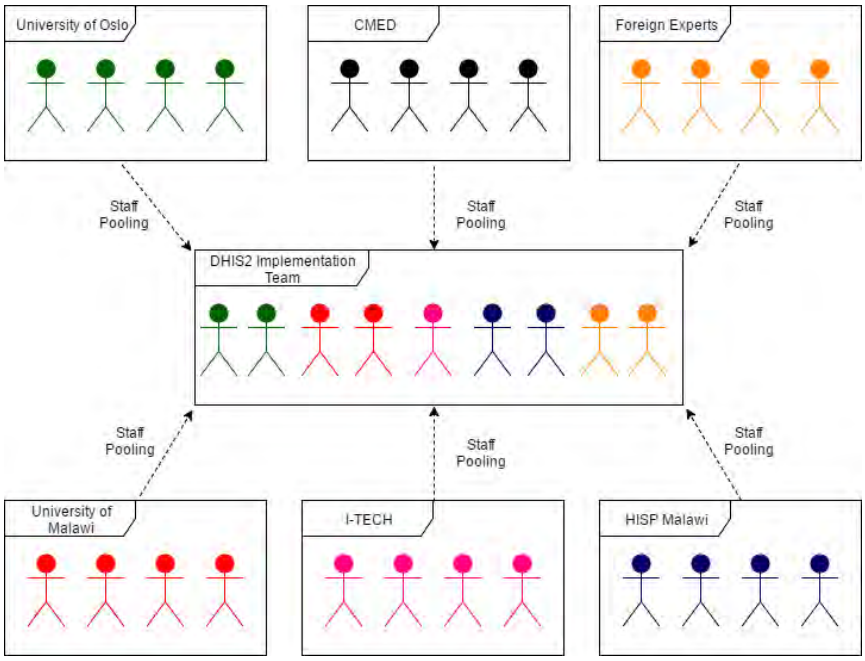


Figure 3 Staff Pooling During DHIS2 Implementation

4.2 Pooling Human Resource Capacities for DHIS2 Reconfiguration

DHIS2 has been in productive use in Malawi since its national roll out in 2012. Over the said period CMED has been lobbying the Malawi government to adjust its staff establishment to include ICT personnel to allow it better leverage the platform. However, the matter is still outstanding. Through its use a number of shortfalls with the current DHIS2 setup were discovered that needed to be ironed out. For example, the way the majority of data elements are defined in the DHIS2 instance at College of Medicine prevents certain high level data analysis and reporting required by some health programmes and stakeholders. As a result of shortfalls in the current setup, some stakeholders and health programmes have been running their own parallel systems which worked against the goal for an integrated national HIS. A DHIS2 reconfiguration project was initiated in order to address the anomalies in question and bring more health programmes and stakeholders on board.

The absence of IT personnel in CMED, once more, necessitated pooling human resource capacities from other stakeholders. A collective of IT personnel drawn from various local and international organisations is currently working together on the DHIS2 reconfiguration project. The team comprises of 2 members from HISP Malawi, 2 members from MoH ICT department, 2 members from Baobab Health Trust, 2 members from University of Malawi and 1 member from GIZ/EPOS Health Management. Data dependencies between stakeholders, improving the quality of reported data, the drive for an integrated national HIS and streamlining of HIS costs were stated as some of reasons why various stakeholders contributed staffs to the DHIS2 reconfiguration team. Altogether, there are nine members in the DHIS2 reconfiguration technical team drawn from 5 organisations. Figure 4, below, illustrates the pooling of IT personnel by different stakeholders towards DHIS configuration.

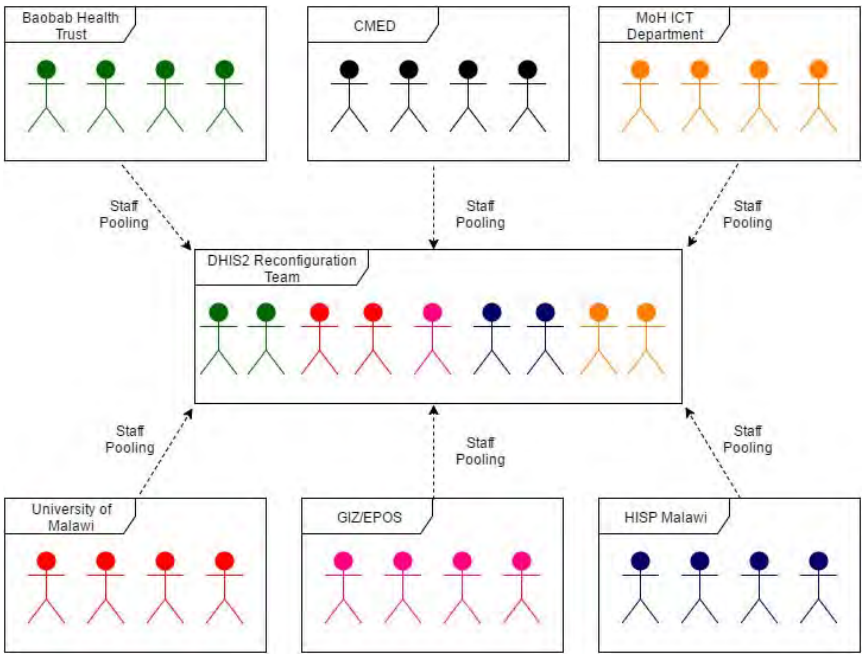


Figure 4 Staff Pooling During DHIS2 Reconfiguration

The team has since succeeded in setting up a new instance of DHIS2 which is hosted at DHA and written applications that run on top of DHIS2 to help map and migrate data from the old instance. The first set of reconfigured and migrated data was delivered in June 2016. At this point, it was expected that corresponding stakeholders would start using the new DHIS2 instance at DHA. However, a challenge emerged with respect to Internet bandwidth which led to the switch being delayed. This is the case because the new DHIS2 instance shares internet connection with other web-based systems also hosted at DHA. Therefore, switching was postponed pending upgrades to the Internet bandwidth.

5 Discussion

5.1 Effectiveness of Pooling Human Resource Capacities across HIS Stakeholders

One of the objectives of the study was to establish the efficacy of staff pooling as a means to address gaps in human resource capacities needed to leverage open source health information software platforms. The implementation and reconfiguration of the DHIS2 software platform in Malawi provides support towards the effectiveness of staff pooling as a strategy to mitigate challenges with human capacities needed leverage open source health information software platforms in developing countries. CMED, under whose custody DHIS2 falls, lacked in-house human resources to implement and later on reconfigure the platform when the need arose. However, by pooling human resources with other stakeholders collaborating with it on DHIS2, CMED garnered requisite human resource capacity to drive the implementation and reconfiguration of the software platform in Malawi. The fact that CMED was able to successfully implement and reconfigure the DHIS2 by leveraging a pool of staff drawn from other stakeholders renders credence to the suggestion by Staring and Titlestad [10]. However, CMED's quest for in-house ICT personnel as demonstrated by the request to have its staff establishment adjusted means staff pooling is only viable as stopgap not a replacement for other capacity building strategies. Thus, when applied, staff pooling should only complement other long term human resource capacity building initiatives.

As was the case during DHIS2 implementation and reconfiguration, staff contributions to the shared pool by stakeholder organisations can vary depending on, among other factors, the existence of requisite human resource capacities within a particular stakeholder

organisation. This is summarised in the model presented in *figure 5* depicting contributions by organisation 1, organisation 2, and organisation 3 through to organisation X.

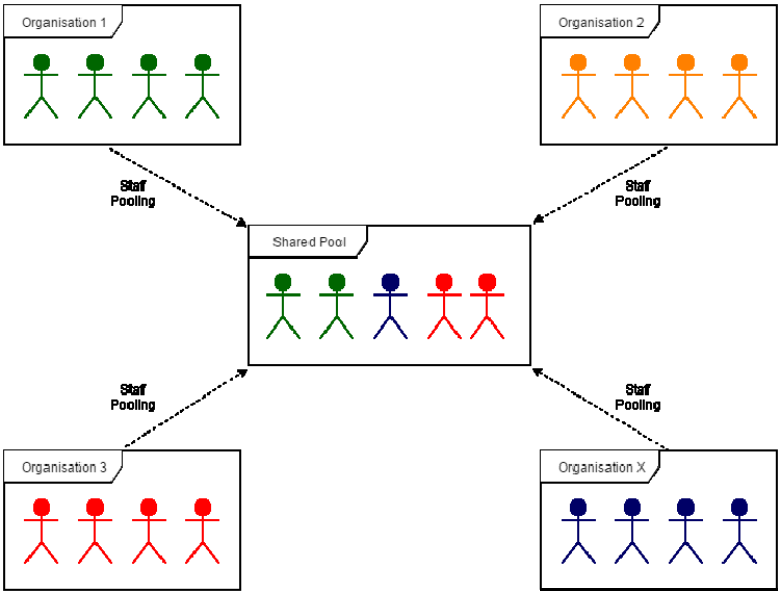


Figure 5: A Model for Pooling Human Resources across Multiple Stakeholder Organisations

5.2 Challenges with Pooling Human Resource Capacities across HIS Stakeholders

A number of challenges exist towards pooling human resource capacities needed to leverage open source health information software platforms. First, it is only possible if some of the stakeholders involved have the requisite human resource capacities. Unless stakeholders with requisite capacities can be identified, pooling human resource capacities in question would be impossible. Another challenge stems from the fact that staff pooling is not automatic; it needs to be negotiated through mutual benefits resulting from such an arrangement. Unless the mutual benefits are apparent, convincing other stakeholders to pool human resource capacities can be difficult. In Malawi, an integrated national HIS with DHIS2 at the hub is seen as mutually beneficial to stakeholders as it would provide a one-stop shop for information for decision makers.

Furthermore, there is potential for fluctuation in the availability of pooled human resources. Sometimes pooled staffs have to respond to urgent matters at their respective organisations making them unavailable for some considerable time. Pooled staffs are constantly subject to tensions between priorities common to all stakeholders and those in their respective organisations potentially leading to their unavailability. Unless there are stringent agreements, fluctuations in the availability of pooled human resources can affect the speed with which mutually beneficial tasks are accomplished.

5.3 Factors That Favour Pooling Human Resource Capacities across HIS Stakeholders

Notwithstanding the challenges, a number of factors were seen to work in favour of pooling human resource capacities across HIS stakeholders. First, information dependencies drove stakeholders that would rely on information from or exchange information with DHIS2 to render assistance towards its implementation and reconfiguration. Second, mutual interests such as an integrated national HIS and ensuring data quality created a need for stakeholders to collaborate with each other to realise such common interests. Lastly, the potential to mutually leverage one software platform made pooling resources around DHIS2 more cost effective compared to stakeholders deploying several in-house solutions. Thus, the existence of information dependencies, mutual interests and potential for mutual leverage

are some of the factors that would work in favour of pooling human resources needed to leverage open source health information software platforms.

6 Conclusions

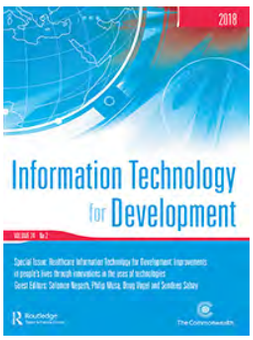
Open source health information software platforms come with unique human capacity requirements for key stakeholders in developing countries adopting them. For developing countries to fully leverage open source health information software platforms, appropriate human resource capacity must exist. HIS in developing countries are characterised by a multiplicity of stakeholders. The implementation and reconfiguration of the DHIS2 software platform in Malawi has shown that pooling human resources from various stakeholders is possible strategy to mitigate challenges with human capacities needed leverage open source health information software platforms in developing countries. However, staff pooling when applied should perceived as a stopgap not a replacement for other long-term human resource capacity building initiatives. It should also be noted that pooling human resources needed to leverage open source health information software platforms requires negotiation and is subject to constant tensions between priorities common to all stakeholders and those of individual stakeholders.

References

1. Eck A, Uebernickel F, Brenner W. The Generative Capacity of Digital Artifacts: A Mapping of the Field. 2015;
2. Ghazawneh A, Henfridsson O. Balancing platform control and external contribution in third-party development: the boundary resources model. Information Systems Journal. 2013;23:173–92.
3. Sheikh YH, Bakar AD. Open Source Software Solution for Healthcare: The Case of Health Information System in Zanzibar. In: Popescu-Zeletin R, Jonas K, Rai IA, Glitho R, Villafiorita A, editors. e-Infrastructure and e-Services for Developing Countries. Springer Berlin Heidelberg; 2012. p. 146–55.
4. Karuri J, Waiganjo P, Orwa D, Manya A. DHIS2: The Tool to Improve Health Data Demand and Use in Kenya. Journal of Health Informatics in Developing Countries. 2014;8.
5. Kandar S, Mondal S, Ray P. A review of Open Source Software and Open Source Movement in Developing Countries. International Journal of Computer Science & Informatics. 2011;1:89–93.
6. Yi Q, Hoskins RE, Hillringhouse EA, Sorensen SS, Oberle MW, Fuller SS, et al. Integrating open-source technologies to build low-cost information systems for improved access to public health data. International Journal of Health Geographics. 2008;7:29.
7. Kimaro HC. Strategies for Developing Human Resource Capacity to Support Sustainability of ICT Based Health Information Systems: A Case Study from Tanzania. The Electronic Journal of Information Systems in Developing Countries. 2006;26.
8. Kimaro HC, Nhampossa JL. Analyzing the problem of unsustainable health information systems in less-developed economies: Case studies from Tanzania and Mozambique. Information Technology for Development. 2005;11:273–98.
9. Dittrich Y. Software engineering beyond the project – Sustaining software ecosystems. Information and Software Technology. 2014;56:1436–56.
10. Staring K, Titlestad OH. Development as a Free Software: Extending Commons Based Peer Production to the South. ICIS 2008 Proceedings. 2008;50.
11. Silsand L, Ellingsen G. Generification by Translation: Designing Generic Systems in Context of the Local. Journal of the Association for Information Systems. 2014;15.
12. Tiwana A, Konsynski B, Bush AA. Research Commentary—Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics. Information Systems Research. 2010;21:675–87.
13. Boerma JT. Health Information for Primary Health Care. African Medical and Research Foundation; 1991.
14. Stevenson A. Oxford Dictionary of English. OUP Oxford; 2010.
15. Creswell JW. Research design: Qualitative, quantitative, and mixed methods approaches. Sage; 2009.

Appendix 4: Paper 3

Msiska, B. & Nielsen, P (2017*),* Innovation in the Fringes of Software Ecosystems: The Role of Socio-Technical Generativity, *Information Technology for Development, 24(2), pp 388-421*



Information Technology for Development

ISSN: 0268-1102 (Print) 1554-0170 (Online) Journal homepage: http://www.tandfonline.com/loi/titd20

Innovation in the fringes of software ecosystems: the role of socio-technical generativity

Brown Msiska & Petter Nielsen

To cite this article: Brown Msiska & Petter Nielsen (2018) Innovation in the fringes of software ecosystems: the role of socio-technical generativity, Information Technology for Development, 24:2, 398-421, DOI: 10.1080/02681102.2017.1400939

To link to this article: https://doi.org/10.1080/02681102.2017.1400939

INFORMATION TECHNOLOGY FOR DEVELOPMENT, 2018
VOL. 24, NO. 2, 398–421
https://doi.org/10.1080/02681102.2017.1400939

Innovation in the fringes of software ecosystems: the role of socio-technical generativity*

Brown Msiska and Petter Nielsen

Department of Informatics, University of Oslo, Oslo, Norway

ABSTRACT

Understanding the way information systems grow and change over time and the role of different contributors in these processes is central to current research on software development and innovation. In relation to this, there is an ongoing discourse on how the attributes of software platforms influence who can innovate on top of them and the kind of innovations possible within the larger ecosystem of technologies and people these platforms are part of. This discourse has paid limited attention to innovation unfolding in the *fringes* of the ecosystems peripheral to and disconnected from where the central software components are developed and where the resources necessary for digital innovation are scarce. Drawing upon Zittrain's characteristics of generativity and Lane's concept of generative relationships, the key contribution of this paper is a socio-technical perspective on innovation and generativity in this setting. We build this perspective of *socio-technical generativity* based on a case study of software innovation activities in Malawi on top of the health information system software platform DHIS2 developed in Norway. This case illustrates how the technical attributes of the platform played a key role in concert with human relationships in shaping innovation activities in Malawi.

KEYWORDS

Generativity; generative relationships; ecosystems; fringes; digital innovation; Malawi

1. Introduction

Information systems are no longer developed and managed as stand-alone and monolithic systems, but are parts of larger ecosystems (Hanseth & Lyytinen, 2010). Aligning with and becoming parts of ecosystems is a new and different challenge for system developers (Sommerville et al., 2012). It involves grappling with the complexity that emerges from the heterogeneity of the multiplicity of systems, functionalities and actors involved. These are the challenges associated with large-scale and complex information systems. State-of-the-art agile and user-centered methods as well as advanced programming tools and frameworks all seek to cope with complexity. However, most of them are doing so in a software-centric fashion and fail to address the broader heterogeneity of systems, users and developers involved in digital innovation (Yoo, Boland, Lyytinen, &

CONTACT Petter Nielsen pnielsen@ifi.uio.no
*Doug Vogel is the accepting Associate Editor for this paper.

Majchrzak, 2012). In this paper, our aim is to contribute to addressing this gap by bringing a stronger focus on the opportunities for innovation in the fringes of ecosystems. Fringes in our case relate to spaces where innovation activities unfold far away from and thus disconnected from the context where the central software components (typically, a software platform) in the ecosystem are developed. These are contexts where resources and human capacity for digital innovation are scarce.

The different roles of different contributors and components are a key part of current digital innovation discourses. For example, the success of digital platforms (such as AppStore) has triggered the curiosity of researchers and many discussions concerning the different roles and the relations between platform owners and app developers (see, e.g. Elaluf-Calderwood, Herzhoff, Sørensen, & Eaton, 2011; Yoo et al., 2012). With a similar focus, Tiwana, Konsynski, and Bush (2010) define a software platform as a software-based system with an extensible codebase that provides core functionality shared by modules (applications) that interoperate with it. The software platform and the modules connecting to it comprise a platform-based software ecosystem. These ecosystems represent a departure from traditional software development as they depend on leverage and rely on the expertise and initiative of a diverse developer community to generate new capabilities (innovations) on top of platforms.

While software platforms and ecosystems have attracted a general research interest toward architecture, governance and digital innovation, these topics do not have a prominent position in the ICT for development literature and debates (Nielsen, 2017). While a special issue on ICT ecosystems was published in the *Information Technology for Development* journal in 2016, the focus was on how ICTs are embedded in socio-economic contexts (see, for example, the editorial by Diga & May, 2016). The emphasis was limited to the context in which ICTs are *used*, a focus also reflected in other discussions on innovation and the role of ICT in transforming development (see, e.g. Qureshi 2013). This paper extends these discussions by studying a software platform, how it is contextualized in the fringes and in particular *how innovation is unfolding on top of it in a developing country setting* and as a part of software ecosystems composed of globally distributed participants.

Different factors influence the way in which different software platforms spur innovation and fuel entrepreneurship. Attributes of the platforms (e.g. their accessibility, transferability and adaptability) will influence who can innovate and the kind of innovations technically possible and economically viable. Zittrain (2006) has ventured deeply into these matters and based on studying the Internet, he coined its key success factor as its *generativity*. He argues that the essential flexibility of Internet as a platform is not limited to its modularity and decentralized network architecture, but also the way in which it enables and leverages innovation performed by a broad range of contributors. These mechanisms can be exemplified with the DHIS2 software platform discussed in this paper where innovation is leveraged based on a sophisticated and rich RESTful Web API that allows extensions of functionality. While discussing the way in which the capacities of different technologies and platforms are influencing innovation is fertile, Zittrain in his definition of generativity attributes innovative capacity primarily to the technology. But while some platforms may be understood to enable and facilitate more innovation than others, innovation is nevertheless a human activity. We argue, concurring with Lane (2011), that innovation is always a collective and social

activity. And in particular in a setting of complexity and uncertainty, identities and attributions of technology develop and change through collectives and based on human relationships.

The focus in this paper is innovation on top of a software platform and with a particular focus on the people located in the fringes of a software ecosystem. We argue that discussing innovation in this setting with a generativity perspective offers us a venue to understand how the social activity of innovation is technologically framed and vice versa. Our attempt is to answer the following research question: *How can our understanding of generativity be framed to provide a holistic account of both technological and social factors that constrain and enable innovation in the fringes of software ecosystems?* The main contribution of this paper is our concept of *socio-technical generativity*. We build this concept by drawing together the technology-focused work of Zittrain and the socially oriented work of Lane. We discuss this holistic perspective based on a study of innovation activities in Malawi based on the open-source health information system software platform DHIS2 developed by the University of Oslo, Norway (see www.dhis2.org). Through the discussion, we also develop our concept of software ecosystem fringes. We show that even if the technical attributes of the software platform play crucial roles in fostering and shaping innovation, it yields little if any innovation in the fringes of its ecosystem if not backed by strong human relationships.

In the next section, we introduce our concept of generativity and point to existing research in this area. In Section 3, we describe our research approach before we present our case study in Section 4. In Section 5, we analyze the case by using our concept of socio-technical generativity and we finally draw theoretical and practical contributions in Section 6.

2. Generativity

In this section, we introduce two different perspectives on generativity, one that is technology-oriented and another focusing on social relationships. Based on these perspectives, we argue for and suggest a holistic and integrated perspective on generativity: socio-technical generativity.

2.1. Generative technologies

The concept of generativity has attracted attention in information systems research, with a particular focus on innovation with respect to information infrastructures and other forms of digital artifacts (see, e.g. Henfridsson & Bygstad, 2013). Generally, generativity is defined as an ability or capacity to generate or produce something (Avital & Te'eni, 2009). More particular, generativity is defined as the overall capacity of a technology or a system to be flexible and malleable by diverse groups of actors and in unanticipated ways (Eck, Uebernickel, & Brenner, 2015; Zittrain, 2006, 2008). The underlying idea is that the success of a technological platform hinges on the participation of independent third-party actors in the generation and production of innovations. The possibilities and incentives for these third parties to engage in innovation depend on the generative capacity of the platform. According to Zittrain (2008), there are five key characteristics of the generativity of technologies:

- *Leverage:* The extent of which the productivity of an actor using the technology is increased as compared to not using it. A technology with good leverage makes difficult tasks easier.
- *Adaptability:* The potential of the technology to be adapted for use in different contexts than the one it was designed for.
- *Ease of mastery:* How easy it is for an actor to understand a technology as well as the amount of effort required to adapt it.
- *Accessibility:* How easy it is (or barriers) to obtain access to a technology, along with the tools and information necessary to achieve its mastery. This includes the cost of acquisition, regulation and secrecy by technology producers to maintain control, which are typical barriers.
- *Transferability:* The ease of which technologies from one context can be conveyed to and re-appropriated in other contexts.

This understanding of the role of technology platforms in innovation is reflected in discussions related to the "essence" of the Internet. This "essence" is seen as important among scholars discussing the regulation of cyberspace. These scholars point to the key qualities of the Internet that are maintaining the speed and scope of innovations it has triggered – regarding both the Internet itself and its use. Lawrence Lessig (2001) has stressed the importance of the location of functions close to the application that uses the function, the so-called end-to-end architecture, originally proposed by Saltzer, Reed, and Clark (1984). This is a central principle to provide flexibility by systems design. The point this principle is making is that functionality in networks only can be appropriately implemented if based on knowledge – knowledge that only exists close to the applications standing at the endpoints of a network. And network growth is based at the endpoints and the applications, and not as a centralized activity. Both Lessig (2001) and David (2001) exemplify this argument by illustrating the Internet as a network where intelligence is in the endpoints. Because the Internet is not optimized for any application but opens for the unexpected and surprising, innovations can flourish without changes in standards or the network itself. The important role of the end-to-end architecture in the success of the Internet is also underscored by historian Janet Abbate (1994, 1999) in her analysis of the history of the Internet and this relationship between the end-to-end architecture and innovation has also been analyzed by Barbara van Schewick (2012). The discussion about the importance of the Internet's end-to-end architecture has more recently turned into a broader one also focusing on the importance of platform-based architectures and the evolution of platform-centric ecologies, typical examples being the iPhone and Android platforms and their respective ecologies (Gawer, 2011; Tiwana et al., 2010). Yochai Benkler (2006) extends the end-to-end argument by underscoring the mutual dependence of the end-to-end architecture of networks and (easily) programmable terminals. Benkler bases his argument on contrasting programmable computers and appliances. An appliance is a device with a limited and well-defined set of functions which (normally) cannot be modified after the users have bought it. Typical examples include washing machines, radios and cars. Such devices have computers inside, but their software cannot be modified by its users. Benkler is worried that several proposals for increasing security and preventing harmful use of the Internet, i.e. cyberspace regulation, will constrain the Internet users' ability to program their computers, i.e. turn

them into appliances. An example of this is found in the proposed "trusted computing" technology and how this may be implemented and ways in which it might be enforced by law.

Inspired by the Internet and lately also platforms for mobile phone apps and digital innovation in general, there is a growing technology-oriented literature on generativity. This literature attempts to discern and improve our understanding of the characteristics and capacities of technologies that turn them into platforms for innovation.

2.2. Generative relationships

Framed within organizational thinking, David A. Lane (2011) presents a different, yet complementary we argue, theory of innovation built around complexity and the cognitive processes involved in innovation. This theory consists of two main concepts: exaptive bootstrapping and generative relationships. The concept of exaptive bootstrapping describes how technologies emerge and evolve. Lane describes these processes as a five-step algorithm driven by positive feedback (Lane 2011, p. 69): (1) new artifacts are designed to achieve some particular attribution of functionality; (2) organizational transformations are constructed to proliferate the use of tokens of the new type; (3) novel patterns of human interaction emerge around these artifacts in use; (4) new attributions of functionality are generated – by participants or observers – to describe what the participants in these interactions are obtaining or might obtain from them; and (5) new artifacts are conceived to instantiate the new attributed functionality. Lane (2011, p. 71) claims that "the most important cognitive process in innovation is the generation of new attributions" and that "the most important communication process involves the aligning of attributions among agents." Accordingly, Lane understands innovation as a collective and social activity. Under situations of uncertainty and change, identities and attributions change, and agents need to track these changes carefully in their attributions of the identity of the significant agents and artifacts in their world. The process of monitoring and interpreting identities requires discourses with other agents. These discourses are channeled through the agents' informational and interpretive social networks. And from these networks, generative relationships emerge.

New attributions arise in the context of a particular kind of relationship among agents which Lane calls generative. Generative relationships may link actors from the same firm, different groups of actors from more than one organization engaged in joint projects, or agents working together under the auspices of a market system. And the generative potential of relationships among agents and their modes of interaction depend according to Lane (2011) on five characteristics:

- *Aligned directedness:* Interactions among agents are focused on achieving similar transformations.
- *Heterogeneity:* Even if having the same overall aim to support a stable system, to succeed, innovative agents have to seek out and build strong relationships with other agents that differ substantially in terms of, e.g. competence, social positions and access to resources.
- *Mutual directedness:* The group of heterogeneous actors, which have different experiences and perceive the world differently, engage with each other in a way where

they see each other's worldview and experiences as a resource rather than assuming the world views different from one's own to be wrong – which often happens.

- *Appropriate permissions:* What the individual actors are allowed to do, i.e. what Lane calls permission structures that shape appropriate permission to engage in innovation.
- *Action opportunities:* Whether the actors have the possibility to engage with one another in interactions that result in transformations not just in their own attributions, but in the structure of agent-artifact space.

2.3. Socio-technical generativity

The concepts of generative technologies and generative relationships are both focused on innovation. And they both concern how innovation is emergent and the outcome of human actors exploring new ideas and opportunities. They address the challenge of understanding how innovation happens in complex settings and how different technological and social environments in different ways enable innovation. At the same time, the generative technologies perspective primarily focuses on attributes of technologies as the enabler for innovation, where the generative relationships perspective primarily focuses on the attributes of the organizational setting of human actors involved in innovation and the relationships between these human actors. In this way, they are complementary. While different technologies have different capacities to facilitate innovation, innovation cannot be seen as independent of human activity. And the activities of humans engaged in innovation will be influenced by other humans as well as the properties of the technology they are working with. This complementarity has also been picked up by Avital and Te'eni (2009) when they argue that the extent to which innovation will take place depends on an appropriate combination of a generative technology and a generative collective of users and developers. When these elements are matched in a successful way, there is a generative fit between the two. This is socio-technical generativity.

Drawing from the concepts of generative technologies and generative relationships, we summarize the attributes of socio-technical generativity in Table 1. The attributes fall into two dimensions: the social dimension concerning how social relationships can be conducive for innovation and the technical dimension concerning the capacity of technologies

Table 1. The dimensions of socio-technical generativity.

Socio-technical generativity	
Social relationships	Technology capacities
Aligned directedness	*Leverage*
Focus on achieving the same transformation	Increasing productivity for the users
Heterogeneity	*Adaptability*
Complementarity in competence, social positions and access to resources	Potential to be used for what it is not designed for
Mutual directedness	*Ease of mastery*
Appreciation of differences and complementarities	Limited efforts to understand and adapt
Appropriate permissions	*Accessibility*
Permissions to innovate	Ease to obtain access and master
Action opportunities	*Transferability*
Opportunity to change own and influence others' attributions of technology	Ease to convey from one context to another

in supporting and promoting innovation. These two dimensions of socio-technical generativity will mutually influence each other. In this interplay, the two dimensions can positively or negatively influence as well as mutually reinforce each other. For example, a flexible permission regime around a software platform increases its adaptability and accessibility. At the same time, unregulated change resulting from a very permissive regime can result in innovations that are not compatible with a generic platform and thereby negatively affect the transferability of the resulting innovations. Ensuring transferability of innovations entails having a well-regulated permission regime to ensure compatibility of emerging innovations. With this interplay in mind, our hypothesis is that this socio-technical perspective on generativity will enable us to better analyze, understand and promote innovation on top of software platforms. In Section 5, we apply this perspective to discuss our case study of local innovation in the fringes of the DHIS2 software ecosystem in Malawi.

2.4. Socio-technical generativity and the fringes of software ecosystems

ICTs are now available and affordable for large populations also in developing countries, and Yoo, Henfridsson, and Lyytinen (2010) have argued that digital technology: " … has democratized innovation and almost anyone can now participate … " (p. 726). We define *fringes* of software ecosystems as contexts peripheral to and disconnected from the context where central software components are developed and where the resources necessary for digital innovation typically are scarce. These resources include human capacity and social relations. While also these spaces are increasingly connected in terms of high-quality and affordable Internet access, they continue to be weak in terms of human capacities, they are likely to be disconnected from innovation networks and they lack generative social relationships. Enabling innovation in such contexts will put high demands on the generative capacity of the technology and provide ways in which those located in the fringes can tap into and contribute toward generative social relationships. Software development and innovation can be seen as unfolding in a complex environment in which multiple software components exist and interact (see Manikas & Hansen, 2013; Tiwana et al., 2010). Such ecosystem perspectives are based on and understanding of symbiotic relationships between the different components and often the central position of a software platform (Gawer, 2011). Our concept of fringes turns the attention away from the role of the central platforms and the focal point becomes the periphery of software ecosystems.

3. Research approach

In the next section, we introduce a case study of the implementation and use of DHIS2 in Malawi. Developed by the University of Oslo in Norway, DHIS2 is a flexible, free and open-source-based software platform designed to support the collection, aggregation and visualization of routine health indicators in developing countries. DHIS2 is highly configurable and customizable, and has an extensive API allowing for integration with other systems and development of apps on top of it by third parties.

We have studied the introduction, implementation and later innovation initiatives on top of DHIS2 in Malawi. We analyze this process based on a perspective on innovation

Table 2. Interviewees.

Organization	No. of interviews	Designation of interviewees
Ministry of Health/CMED	6	Director, CMED Chief Technical Assistant, HMIS Chief statistician DHIS2 Technical Assistants (3)
HISP Malawi	3	Board members (3)
Baobab Health Trust (BHT)	4	Executive Director Software Development Manager DHIS2 Integration Team Members (2)
University of Malawi	1	HISP Malawi Representative
Ministry of Health/HIV–AIDS Department	2	Technical Assistant, Systems Administration (I-TECH) Technical Assistant, Statistics (I-TECH)
DTREE International	1	Project Manager
GIZ/EPOS Health Management	2	Senior Technical Advisor, HMIS Strengthening Programme DHIS2 Technical Assistant

as not only influenced by technical artifacts and their trajectories, but also occurring through a negotiation process involving a heterogeneous network of human and technological actors (Law, 1999). Our research design traces the technical components involved, the role and intentions of innovators and designers locally and globally, and how this is reflected in how the technology is used (Faraj, Kwon, & Watts, 2004), focusing on "recognising the depths of interdependence of technical networks and standards, on the one hand, and the real work of politics and knowledge production on the other" (Bowker & Star, 2000, p. 34).

The case study reported here is based on 19 informal and in-depth interviews with managers and technical personnel in 6 different organizations related to the implementation of DHIS2 and related innovations in Malawi (summarized in Table 2). The organizations included the Central Monitoring and Evaluation Division (CMED) in the Ministry of Health under whose custody DHIS2 falls; the HIV/AIDS department which hosts other health information systems for the Ministry of Health; the local organization of DHIS2 experts HISP Malawi, University of Malawi and GIZ/EPOS Health Management providing local technical support on DHIS2; and DTREE and Baobab Health Trust which are working toward integrating their electronic health solutions with DHIS2. The interviews were focused on how the different actors use and understand DHIS2 and how they contributed to its implementation and related innovations in Malawi. Where consent was granted by the respondent, the interviews were recorded and later transcribed. Otherwise, written notes were taken. While the entire set of interviews provide the background and setting of local innovation, this paper particularly draws upon interviews focusing on the implementation and innovations on top of the DHIS2 platform. These "innovation episodes" around DHIS2 in Malawi include DHIS2 integration attempts, DHIS2 reconfiguration and third-party application development exemplified by the League Table App.

In addition to interviews, document reviews were carried out to trace the implementation and "innovation episodes" related to DHIS2 in Malawi. Document reviewed included research reports, websites (e.g. www.hispmalawi.org.mw), minutes of planning and review meetings, and emails from the stakeholders mailing group. Furthermore, participatory observation was carried out whereby one of the authors is an active participant in an ongoing DHIS2 reconfiguration project in Malawi. The data collected from interviews, documents and observations combined were used to identify the "innovation episodes"

and these episodes and their context were used to focus the further analysis. Along the analysis, we built our socio-technical perspective on generativity and understanding of fringes, and its dimensions were further used to support and guide the analysis and discussion.

4. The case of DHIS2 in Malawi

4.1. The process of implementing DHIS2 in Malawi

DHIS2 is centered on supporting the collection, aggregation and visualization of routine health data, or so-called health indicators. It is developed by the Health Information Systems Programme at the University of Oslo (HISP UiO) based on the support from international donors. It has over the recent years developed into a *de facto* standard in developing countries and is widely used by Ministries of Health, health programs, donors and NGOs. In some countries, DHIS2 is implemented as the national health information management system.

DHIS2 has a sophisticated and rich RESTful Web API that allows extension of functionality through new innovations using Web technologies such as JavaScript, CSS and HTML5. These can be additional software modules that sit side by side the DHIS2 core modules or applications that run on top of it. From previously being a software system developed by the core team of developers in Oslo alone, the strategy behind DHIS2 is today one of open innovation. Extensions with apps are expected to be developed by a wide audience of developers participating and contributing to the DHIS2 software ecosystem globally. Malawi is a part of this ecosystem, but also very much located in its fringes.

DHIS2 was introduced in Malawi as part of a health management information system (HMIS) strengthening program initiated in 1999. A prime motivation behind these efforts was an identified lack of reliable information for health services planning and management due to, among other things, the existence of a number of different information systems belonging to vertical health programs. This fragmentation made access to health programs' data difficult across programs and geographies. Data remained within the different information systems of the different health programs and in separate, silo-systems. Between 1999 and 2002, an HMIS review funded by the Dutch Government was carried out. The review led to the implementation and launch of an early version of DHIS2 (DHIS) as the national Health Information System in January 2002. At the time, DHIS was a desktop-based application based on the Microsoft Access platform. Being a desktop application, it did not support shared access to information as noted by a Director at CMED: " … sharing data to geographically distant stakeholders was still a challenge." As a result, fragmentation continued, characterized by the continued existence of paper systems and multiple computerized systems. Furthermore, DHIS was not scalable due to the use of proprietary software which entailed license fees for each workstation where it was installed. A different solution was required to improve the health information system in Malawi. In 2005, HISP UiO made the first release of the open-source and web-based DHIS2.

In Malawi, efforts to replace DHIS with DHIS2 commenced in 2009 with a pilot project targeting three health districts. In this process, new challenges emerged and were handled. First, the CMED of the Ministry of Health (MoH) under whose jurisdiction

DHIS2 falls lacked the requisite technical capacity and human resources to implement and rollout DHIS2. This was due to the fact that the CMED organization is dominated by economists and statisticians and without any ICT staff. MoH, like any other ministry in the Malawi Government, is allocated ICT staff by the Department of e-Government. This staff is typically dedicated to ICT and not the health information system, as noted by a CMED Director: " … staff allocated are mostly focused on keeping the ICT equipment running, process salaries and do not deal with health information system issues." In order to mitigate this challenge, a strategic relationship was formulated with HISP UiO through the University of Malawi. This brought on board three constituent colleges of the University of Malawi: College of Medicine, Malawi Polytechnic and Chancellor College. At that time, the ICT infrastructure at the College of Medicine was amongst the best in the Malawi with a stable and reliable Internet connection, while Malawi Polytechnic and Chancellor College had ICT staff which could be leveraged. In addition, a local organization called HISP Malawi was established to provide the required technical support for the implementation and management of DHIS2. To manage HISP Malawi, a five-member board was constituted, drawing its members from MOH/CMED and the three mentioned colleges at the University of Malawi. Second, as a web-based system, DHIS2 required a stable hosting space with adequate bandwidth to handle the data traffic from various users. However, the Government Wide Area Network (GWAN) through which MoH gets internet connectivity is known for being unstable; often up but also down and when up, the quality of the connection is often poor. Furthermore, "GWAN is very slow … we found its infrastructure inadequate and unstable to run DHIS2" (Chief Technical Assistant, CMED). Therefore, alternative means of hosting DHIS2 were required. There were three options on the table: (1) setting up a DHIS2 web server at CMED with its own dedicated Internet connection, (2) using a web hosting service outside Malawi and (3) hosting DHIS2 with a partner institution in Malawi. Setting up a web server for DHIS2 at CMED and acquiring dedicated Internet connection proved to be a costly endeavor. At the same time, the MoH was not in favor of hosting health data outside Malawi. Because College of Medicine had the best ICT infrastructure and good Internet connectivity, a decision was made to host DHIS2 at the College of Medicine. Third, with other higher priority health areas to finance, there is usually little left to finance health information systems initiatives. Not surprisingly, the ministry did not have funds for the DHIS2 pilot project and alternative sources of funding had to be explored. Fortunately, the University of Oslo provided the funds for a pilot project. With funds available, two fresh graduates from the University of Malawi were recruited under HISP Malawi and placed on secondment to CMED as technical assistants on the project. The pilot project ran from 2009 to 2012 and DHIS2 was rolled out countrywide in 2012 with support from various stakeholders including: Norwegian Agency for Development (Norad), Support for Service Delivery Integration (SSDI), United Nations Children's Fund (UNICEF), Centre for Disease Control and Prevention (CDC), International Training and Education Centre for Health (I-TECH), HISP Malawi and the University of Oslo (Department of Informatics).

4.2. Emerging needs for innovation

The national rollout of DHIS2 implies that data are now transmitted electronically from the district level to the national level. However, from the health facility level to the district

level, data reporting is still paper-based. This is because most health facilities do not have electricity, Internet connectivity and/or computers. Data at the health facility level are collected through a paper-based system of registers and aggregated monthly on paper forms. The forms are then sent on paper to the district level where the aggregate data are entered into DHIS2. Aggregate data are therefore entered twice: first onto the paper forms at the health facility level and then into DHIS2 at the district level. As noted by the Chief Statistician at CMED: "This is a challenge; it makes data reporting slow and tedious." This is one example of how DHIS2 has brought forward local demands for further innovations in order to support its local use. In the subsections below, we describe three different local innovation initiatives being undertaken in Malawi.

4.3. Integration

The health system in Malawi has a history of fragmentation and reporting systems that make access and sharing of data among various geographically distributed stakeholders difficult. There are human resources management, logistics and medical record information systems producing data that in aggregated form is to be entered into DHIS2. The data from these systems are manually aggregated and then entered into DHIS2. To reduce the efforts required for data reporting, there is a call for integration between DHIS2 and auxiliary systems to enable information flow electronically between the systems.

Responding to this call, in the last quarter of 2015, MoH recruited a technical assistant through HISP Malawi to work in close collaboration with other stakeholders on integrating their systems with DHIS2. One such stakeholder is Baobab Health Trust. Baobab Health Trust is a local NGO that develops and maintains a suite of touchscreen-based electronic medical record (EMR) systems used by health facilities across Malawi. The Baobab EMR suite includes applications for patient registration, in- and out-patient diagnosis, maternal care, antiretroviral therapy, pharmacy inventory control and billing as summarized in the Table 3.

The Baobab EMR suite is used in more than 60 health facilities across Malawi and there are plans to introduce it to more health facilities. Baobab Health Trust is also working together with the National Registration Bureau and the MoH on an electronic birth registration (EBR) system which will allow babies to be registered at birth. Currently, the EBR system is being piloted in four hospitals across the country. In health facilities where

Table 3. System components composing the EMR suite offered by Baobab Health Trust.

System applications	Description
Patient registration	Registers every patient and issues a unique ID in form of a barcode pasted on a patient's health passport. The barcode allows continuity of care where once scanned previously recorded patient's information is retrieved
In- and out-patient diagnosis	Linked with the patient registration module, allows clinics to record both primary and secondary diagnoses
Maternity	Registers, admits, diagnoses, discharges and refers pregnant women from antenatal ward or labor ward to post-natal ward
Baobab antiretroviral therapy (BART) system	
Pharmaceutical inventory control	For the management of pharmaceutical items
Billing	For the management of paid health services

the Baobab EMR and EBR systems are running, data are first electronically captured into the systems, and later manually aggregated for entry into DHIS2. Integrating Baobab systems with DHIS2 will potentially reduce the time and effort required to collect and aggregate data for entry into DHIS2. But getting appropriate access rights to DHIS2 from HISP Malawi has challenged Baobab's attempt to integrate: "We had one of our software developers working on integrating our EMR with DHIS2 but we were unable to secure appropriate access rights to DHIS2 from HISP Malawi to allow us test our solution" (Director, Baobab Health Trust). Even technical assistants working under CMED have experienced this: "At one time one could be granted super-user rights to DHIS2, only to realise later that those rights have been revoked without us being informed" (DHIS2 Technical Assistant, CMED). There are limits to what MoH and other stakeholders can do with respect to DHIS2. Control and ownership over DHIS2, shared between CMED/MOH and HISP Malawi, were by other actors found to be confusing. To improve this situation, a project was carried out to introduce a mirror server for DHIS2 at the MoH to run parallel with the one at College of Medicine. MoH with support from various stakeholders established a server room equipped with good Internet connectivity at the offices of its Community Health Services Unit (CHSU). The server room is hosting other information systems such as the Human Resources Management system and Logistics Management system. It is expected that once the mirroring is done, the DHIS2 server at CHSU will be the main server with the server at College of Medicine acting as a backup. This will give MoH and other stakeholders more leeway and the requisite flexibility to implement desired innovations. Furthermore, a memorandum of understanding between MoH and HISP Malawi is being drafted which when signed will formalize the relationship and add transparency to matters of control, ownership and accessibility.

4.4. Reconfiguring DHIS2

When a decision was reached to mirror the DHIS2 server at College of Medicine with a new server at CHSU, it was also observed that the current configuration of DHIS2 has some shortfalls that needed to be ironed out. For example, there are cases of data duplication whereby similar data are entered in different datasets in DHIS2 by two or more health programs. Furthermore, the way the majority of data elements are currently configured prevents certain high-level data analysis and reporting required by various health programs and stakeholders:

> Instead of just mirroring the two servers and inherit the challenges we are facing with the current configuration it is better that the new server be reconfigured in such a way that we can migrate all data from the old server but also get rid of its shortfalls. (Senior Technical Advisor, Health Information Systems Strengthening Programme, MoH)

A decision was therefore made to embark on a DHIS2 reconfiguration instead of mere mirroring of the DHIS2 servers at College of Medicine and the CHSU. Mirroring became one of the several objectives within the DHIS2 Reconfiguration project described below.

4.4.1. Aligning together stakeholders for DHIS2 reconfiguration

DHIS2 in Malawi involves several local and international stakeholders in terms of its use as well as development. Thus, the DHIS2 Reconfiguration project attracted interest from

several stakeholders working together by drawing upon their resources, experiences and expertise. This includes people from CMED/HISP Malawi, MoH ICT Department, MoH Programme Coordinators, Baobab Health Trust, GIZ/EPOS Health Management and the University of Malawi, Chancellor College. In addition, the DHIS2 Reconfiguration was supported by the University of Oslo, the Global Fund and UNICEF.

To effectively work together, there was a need for proper alignment of the stakeholders involved. First, it was important to ensure that the DHIS2 reconfiguration implementation team had proper access rights to DHIS2 at the College of Medicine. Following a series of meetings, administrative rights over the server was established and DHIS2 super-user accounts were made available to CMED. Furthermore, standard working procedures were drafted to guide the collaboration between the stakeholders by defining permission and control structures. Consultations were made with the head of the ICT department in the MoH which lead to allocation of two members of staff to work with CMED on DHIS2. A larger implementation team was further established comprising nine technical staff drawn from different stakeholders: two members from CMED/HISP, two members from MoH ICT department, two members from Baobab Health Trust, two members from University of Malawi and one member from EPOS Health Management. The implementation team was supported by a two-member administrative team: one member for CMED and one member from EPOS Health Management.

4.4.2. Capacity building and DHIS2 reconfiguration

In order to build the required capacity to reconfigure DHIS2, the implementation team took part in training activities. First, a DHIS2 Application Development workshop took place at the University of Malawi, Chancellor College from 7 March to 16 March 2016 (University of Malawi, 2016). The training introduced participants to the current DHIS2 configuration, the DHIS2 API and how to develop DHIS2 applications using them. The training was funded by the University of Oslo with support from UNICEF. It attracted participants not only from Malawi but also from Kenya, Ethiopia, Zambia and Malawi. Five of the members of the implementation team attended the training. A second training was carried from 21 March to 2 April 2016, prior to the start of the DHIS2 reconfiguration. The second training was funded by the University of Oslo with support from Global Fund. It was facilitated by a consultant and expert on DHIS2 from the University of Dar es Salaam, Tanzania. This training introduced participants to DHIS2 installation on the Linux operating system, how to configure DHIS2, and how to backup and restore DHIS2 instances. Only five members of the implementation team managed to attend both rounds of training. As a result, from 4 April to 8 April, a follow-up training was organized to bring the other four members up to speed. This was facilitated by the five members who attended the earlier training.

Effectively, the reconfiguration commenced on 11 April 2016 with the installation of a DHIS2 instance on the CMED server at CHSU. This was followed by the redefinition of data elements in order to get rid of the anomalies associated with the DHIS2 instance at College of Medicine. Programme Coordinators were consulted to provide clarification of data elements related to their programs. Where data duplication across programs was identified, respective coordinators were called to a joint meeting to agree which program would be responsible for collecting the data in question. For example, both the TB program and the HIV/AIDS program were found to be collecting TB-HIV coinfection

data in their datasets. Following consultation, it was agreed that the TB program will collect this particular data and the HIV/AIDS program will reuse this data for its own purposes. Decisions like these led to the redefinition of program datasets and subsequently their reconfiguration in the DHIS2 instance at CHSU. The implication of this is that although the DHIS2 instance at CHSU is meant to capture the same data as the one at College of Medicine, in terms of data elements and dataset configuration, the two instances are significantly different. Thus, the data in the DHIS2 instance at College of Medicine had to be migrated to be aligned with the new configuration used in the CHSU instance.

Data migration between the two DHIS2 different instances required mapping data elements from the College of Medicine instance to corresponding data elements in the CHSU instance. This involved extracting three IDs for each data element (a total of six IDs for both instances) and browsing three pages of the DHIS2 Web API (a total of six for both instances). Most datasets have hundreds of data elements. Therefore, manually mapping data elements was going to be tedious and slow, making the DHIS2 reconfiguration unattainable within the project timeframe. To speed up the process, applications to automate the data migration were required. Drawing from the DHIS API knowledge acquired during the trainings, the implementation team developed a set of applications. The first application is called *Dataset Details Lister*. This is a DHIS2 web application that lists data elements and their corresponding IDs for a selected dataset. This information is then taken to a spreadsheet to create a dataset map between two corresponding datasets of the two DHIS2 instances. The second application is called *Data Migrator*. This is a PHP application that uses the DHIS2 API to connect to both DHIS2 instances and uses the dataset map created from the output of *Dataset Details Lister* to migrate data from the instance at College of Medicine to the one at CHSU. The third and last application is called *Data Migration Validator*. This is a PHP application that uses the DHIS2 API to connect to both DHIS2 instances and checks if the data migrated across the instances are matching and flags out any mismatches. Mismatches can happen as a result of errors in mapping data elements or problems with the Internet connection. To speed up correction of such mismatches, the application was further improved to automatically fix any mismatches identified. This is only done if the mapping process is confirmed to be free of errors. Together, these three applications enabled a fast-tracked data migration process. Figure 1 illustrates the relationship between the three applications and the two DHIS2 instances.

The DHIS2 reconfiguration project employed a phased agile approach where different health programs were reconfigured and migrated in different time boxes according to priorities assigned to each program. The project was scheduled to run up to August 2017 with the first set of reconfigured and migrated programs delivered in June 2016. At this point, it was expected that those programs will make a switch from the DHIS2 instance at College of Medicine to the DHIS2 instance at CHSU. However, a challenge emerged with respect to limited Internet bandwidth at CHSU which led to the switch being delayed. The limited bandwidth is the result of the server room hosting several other web-based health information systems in addition to the new DHIS2 instance. The Internet connection is also used by members of staff of the HIV/AIDS department for various office duties. Switching the reconfigured programs to the DHIS2 instance at CHSU without upgrading the bandwidth will result in poor response time. The matter was tabled at a

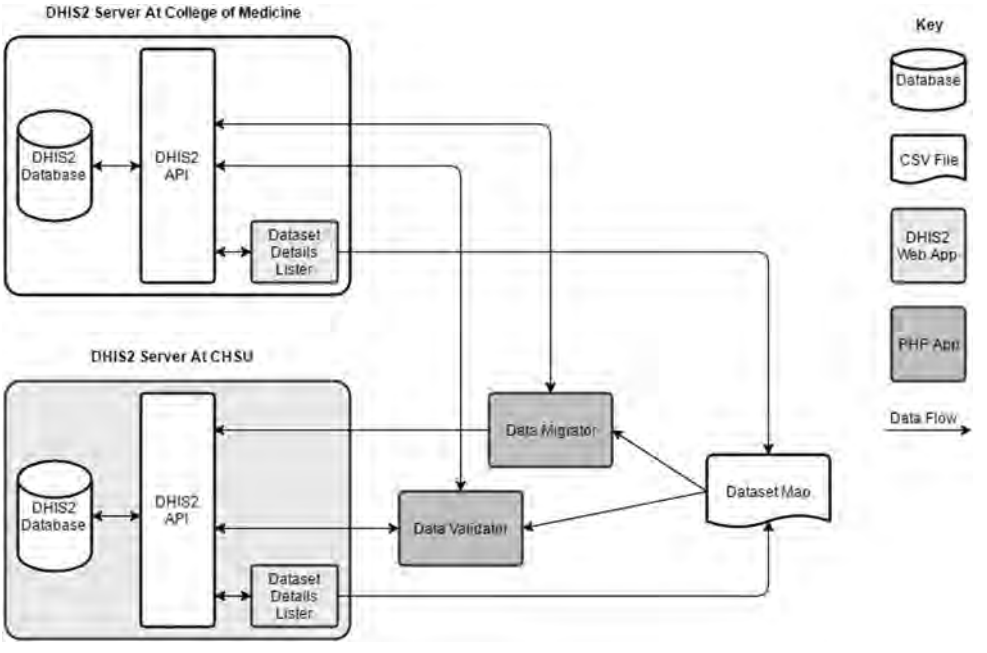


Figure 1. Applications developed during DHIS2 reconfiguration in Malawi.

stakeholders meeting and a cost-sharing agreement was made toward upgrading the available bandwidth. Although the project was initially scheduled to run up until August 2017, the automation of migration through the three applications mentioned above allowed the migration to be completed four months ahead of schedule. As a result, there was a switch to the new instance in April 2017 instead of August 2017.

4.5. The League Table App

Another initiative extending the DHIS2 in Malawi is the League Table App. Feedback practices such as review meetings at health districts have been taking place at all levels for a long time in Malawi (Moyo, Frøyen, Sæbø, & Kaasbøll, 2015; Moyo, Kaasbøll, Nielsen, & Sæbø, 2016). The review meetings bring together various stakeholders such as District Managers and NGOs, to examine performance data, provide feedback and develop action plans to improve the data. Feedback is also provided to health facilities during supervision by teams from the district. Districts produce and disseminate HMIS annual bulletins, as a means of providing feedback. To strengthen the HMISs, the MoH in collaboration with the University of Oslo has piloted and tested several different versions of league tables in different districts. League tables are used to compare the performance of different entities at the same level of the health systems by ranking them.

As an internal decision support tool, the league table can be developed and used by the district health management teams themselves. The league tables can also be presented in the district HMIS bulletins so that they are available for a wider group of users. The design of a DHIS2 League Table App was undertaken by two Master students at the University of Oslo as part of the course and project work (see Figure 2 for an example user interface).

Figure 2. Example League Table App user interface (not real data).

The design started in January 2015 and it culminated into the piloting of the solution in Malawi in November 2015. This combination of members with different backgrounds, interests, roles and responsibilities enriched the interactions and consultations between the developers of the app, researchers and the head of the CMED at the MoH responsible for data management and DHIS2 in Malawi (who was part of the research team).

A series of meetings were held at the University of Oslo among the researchers to plan the pilot. The identification of the indicators was based on programs with high reporting rates in DHIS2. Based on this, the League Table App used indicators, as a starting point, from the following reports: HMIS15 summary report, Family Planning, Antenatal care, Maternity and Malaria program reports. The league table was designed as a separate app using the DHIS2 Web API to access the data from DHIS2. It offers the health managers the flexibility to define their own league tables based on the indicators they find relevant and they can give the different indicators different weights in the total ranking. Currently, there is a project funded by UNICEF aimed at integrating the League Table App with another DHIS2 data visualization app, the Scorecard App. The new app will be piloted in Malawi, Uganda and Tanzania in 2017. The new app is being developed by developers from HISP Tanzania and University of Malawi.

In Table 4, the key DHIS2-related activities in Malawi in the period 1999–2016 are summarized.

5. DHIS2 and socio-technical generativity in Malawi

In this section, we link the case study to our proposed perspective on socio-technical generativity. We do so by first illustrating how the technical attributes of DHIS2 influenced

Table 4. Key DHIS2-related activities in Malawi.

Time period	Activities
1999	HMIS Strengthening Programme Starts
1999–2002	HMIS Review, establishing a picture of a fragmented HMIS and the need for consolidation
2002–2008	DHIS introduced and implemented
2009–2014	DHIS2 introduced and piloted
2015–2017	- MOU and technical working procedures to facilitate collaboration and integration - DHIS2 reconfiguration - DHIS2 trainings - League Table App piloting

innovation in Malawi, then how social attributes played a role, and finally, how these social and technical attributes were interrelated in the way they influenced innovation. By doing so, we show the strength of our holistic perspective on socio-technical generativity and how it supports the recognition of the complementarity between social and the technical factors in the way they influence innovation.

5.1. The technical attributes of DHIS2 and generativity in Malawi

In Section 3, we introduced five attributes of technology influencing generativity: capacity for leverage, adaptability, ease of mastery, accessibility and transferability. In this section, we discuss each of these attributes related to DHIS2 in Malawi. First, DHIS2 with its data management, visualization and web-based futures offered the Malawian Ministry of Health and CMED a tool they could leverage to reduce the fragmentation of the multiple existing information systems. This improved the ease of access to information across health programs and improved data quality. Second, the adaptability of DHIS2 is, for example, reflected in the three applications developed using the Web API to facilitate data migration from the old to the new DHIS2 instance and the piloting of the league table. Third, with respect to ease of mastery, DHIS2 offers a mature, well-proven and well-structured Web API and it comes with extensive online documentation.

The documentation provides users, implementers and third-party application developers the means to learn and master various aspects of the platform and the API. While the documentation was a key source of information for the implementation team, it was also observed that the documentation sometimes is vague and out of sync with the different versions of DHIS2. The amount of effort required to adapt DHIS2 can be discussed on three different levels in the case of Malawi. First, the team had the required skills to configure DHIS2 to fit the particular requirements. Second, they also had the skills needed to reconfigure DHIS2 and develop the League Table App by using the DHIS2 API. Third, even if DHIS2 is open-source software and anyone can easily inspect the code and make changes to the core, this did not happen in Malawi. There can be many reasons for this, including the lack of expertise needed to understand the core code, uncertainty about the consequences of changing it and the risks of making software that may end up incompatible with future versions of DHIS2. Fourth, DHIS2 is open source and free of charge to download and use. Without expensive licenses as an accessibility barrier, DHIS2 was highly attractive for Malawi as a developing country and with its very limited resources for investments in software. Fifth, DHIS2 is highly configurable and customizable, and has an extensive API allowing for integration with other systems. The strategy behind DHIS2 is based on supporting diverse needs from diverse use contexts

in different developing countries. Based on an approach of "open generification" (Gizaw, Bygstad, & Nielsen, 2017), DHIS2 can be transferred to, customized and implemented in most contexts and it is up to the user to configure it related to, for example, organizational hierarchies, health indicators, language and dashboards. To support the sharing of innovations made on top of DHIS2 across organizations and countries, HISP UiO has also established a DHIS2 App store. In Malawi, this enabled the implementers to configure DHIS2 to fit the context and share local innovations globally.

DHIS2 shows the attributes of a generative technology in Malawi. Compared to the earlier desktop version, it offered the required leverage (web-based), it came without the classical accessibility barriers in developing countries (no proprietary software and associated license costs) and it offered the adaptability to customize it for local needs and introduce new innovations on top of it.

5.2. Generative relationships and DHIS2 in Malawi

In Section 3, we also introduced five attributes of social relationships influencing the generativity of a technology: Aligned directedness, heterogeneity, mutual directedness, appropriate permissions and action opportunities. In this section, we discuss each of these attributes related to DHIS2 in Malawi.

First, the implementation of DHIS2 brought together people from different organizations and domains: The CMED from the Ministry of Health, researchers and technical experts from the University of Malawi and the University of Oslo, and other local and international organizations. They were all working in the same direction and toward strengthening the national HMIS in Malawi. Second, this heterogeneous group of organizations and people had different competences, social positions and access to resources. These resources were in many ways complementary. For example, the CMED possesses domain expertise but lacks the required software and hosting capacity needed to implement and use DHIS2. The College of Medicine had the infrastructure necessary to host DHIS2 and HISP Malawi and HISP UiO contributed with their capacity to implement and customize the software. Third, HISP Malawi played a key role in bringing these different actors together. HISP Malawi was established based on members from different local stakeholders. Some of the members were also graduated with PhDs from the University of Oslo. Through their education, they were exposed to and became a part of the "ideology" of HISP UiO, namely the *Networks of Actions approach* (Braa & Nielsen, 2015). At the core of this "ideology" lies the idea that action research-based interventions in developing countries must be part of larger networks to be sustainable. Sustainability and success are an outcome of collective actions by a network of engaged organizations and individuals locally and globally. This acted as a platform for HISP Malawi where differences and complementarities were appreciated, despite the diversity of the organizations involved. Fourth, the core of the network of action is the action research supporting the implementation and further development of DHIS2. Experiments, prototyping, student projects and innovation to meet user needs compose the modus of operandi of this global network. During the process of implementing DHIS2, CMED also permitted the hosting of DHIS2 at the college of medicine instead of using the GWAN infrastructure. Hurdles to innovation, like the limitation of access to integrate new solutions to DHIS2, were effectively remedied with a memorandum of understanding between MoH and HISP Malawi and the

introduction of standard working procedures. As a part of this, researchers and students from the University of Oslo got the appropriate permissions from CMED to engage with their expertise and learn from the concrete cases in Malawi. Fifth, the network of actors involved locally and globally created an environment where new attributions could emerge. This environment was conductive for innovation on several levels. First, domain experts, implementation experts and DHIS2 software experts were working together on the ground in Malawi. At the same time, this team was linked to the global network of HISP experts through personal networks and online DHIS2 forums where there is an ongoing discussion of opportunities and issues. Second, regional DHIS2 academies and workshops are arranged annually with the aim of building capacity as well as facilitating peer-to-peer sharing of experiences and ideas among different experts in the network. Third, the involved students from the University of Malawi and the University of Oslo all had a strong drive toward experimenting with and improving DHIS2 to meet emerging user needs.

In Malawi, DHIS2 is supported by generative social relationships. These relationships are composed of a strong local network of heterogeneous partners linked to a global network of DHIS2 experts. The driving force in these networks is innovation to strengthen HMISs in developing countries.

5.3. Socio-technical generativity and DHIS2 in Malawi

From the analysis above, we argue that DHIS2 and its related social relationships in our case in Malawi are generative. Through the analysis, we have also shown that it does not make sense to look at technological or social factors in separation. Generativity is socio-technical. An illustrative example of this is the DHIS2 reconfiguration and the development of the apps required (Dataset Details Lister, Data Migrator and Data Validator). While DHIS2 offered the necessary Web API, it would have been useless without the adequate administrative server rights to DHIS2 at College of Medicine, the expertise knowledge required to map the data in the different databases as well as the expertise to develop apps outside for efficient mapping, and at the same time using the DHIS2 API. And vice versa, without the required Web API, the network of experts would not have achieved much. Such a situation is well illustrated with the previous DHIS based on proprietary technology, hampering local customization and not allowing for extension of its features and thereby mandating the end users to use it as it was. The implication of this was that stakeholders had no other option but to create new systems – exactly what the project set out to avoid in the first place. A similar challenge was met by Baobab when HISP Malawi did not grant them access to DHIS2 and integrate their EMRs system with it. Based on the existing social relationships, this hurdle was quickly removed.

5.4. Socio-technical generativity in the fringes

With its platform design, DHIS2 does not only depend on the core team of developers at the University of Oslo, but also a network of local experts. Software platforms are "half products" which have to be configured, customized and extended to meet the needs of a specific context (Dittrich, 2014). And the continuous geographical expansion of the

platform has fueled its generic as well as generative nature by introducing new use contexts, use cases and application domains (Nielsen & Sæbø, 2016). As a developing country, Malawi has limited resources to invest in and human capacity to engage in software development. And as a relatively small country located geographically far away from the University of Oslo, it is in many ways at the fringes of the DHIS2 software ecosystem. From a technical point of view, the platform nature of DHIS2 makes space less a concern as long as DHIS2 offers the needed technical support for local innovation. The possibility for a stakeholder in Malawi to make the team in Oslo change the core software is at the same time limited (Gizaw et al., 2017).

While software can travel fast and freely globally and to the fringes of software ecosystem, establishing the required local capacity and social relationships is much more challenging. In the case of DHIS2 in Malawi, this included establishing access to servers, building the capacity to customize, implement and use DHIS2, and establishing and nurturing the necessary relationships to innovate on top of it. These relationships were local, but also global and linking the local initiatives with the region and the global HISP network through, for example, DHIS2 academies.

5.5. Socio-technical generativity and time

While social and technical attributes of generativity are necessary to support innovation, they develop independently and not necessarily in sync. The establishment of generative relationships does not necessarily coincide with the presence of generative technologies, and *vice versa*. For example, the local capacity in Malawi to implement, use and maintain DHIS2 has developed and matured slowly since the HMIS strengthening program was started in 1999. A key element in this was from the beginning the HISP project and the PhD program at the University of Oslo were Malawian students were enrolled and later graduated. While this capacity was building up, the previous version of DHIS and its proprietary technology basis continued to offer only limited technological capacities for innovation. This continued until the new DHIS2 was launched in Malawi in 2009. And the technical capacity of DHIS2 to support innovation did not fully come with its first release, but developed over time as DHIS2 matured as a platform with improved documentation and a maturing Web API. Turning DHIS2 into a platform was a strategic move by HISP UiO to meet a rapidly expanding demand for flexibility to meet new requirements from numerous new countries implementing the platform. To support this, HISP UiO actively worked toward establishing a global network of DHIS2 experts and software developers to develop apps. Still, the DHIS2 experts are primarily engaged in customizing and implementing the software while the software team at the University of Oslo is doing the vast majority of the software development. The relationship between social relationships, technological capacities and time is summarized in Table 5.

6. Implications for research and practice

The aim of this paper is to build and discuss a holistic account of the technological and social factors constraining and enabling innovation in the fringes of software ecosystems. The main contribution of this paper, the concept of *socio-technical generativity*, is

Table 5. Summary of time periods of DHIS2 in Malawi.

Time periods	Social relationships	Technological capacity
1999–2002	- Building a shared understanding in Malawi regarding the weaknesses, and in particular the fragmentation of the HMIS	- DHIS with limited accessibility due to license costs not matching budgets - No flexibility to change with the result of different stakeholders investing in different systems
2003–2008	- The capacity to use DHIS slowly building up in Malawi through PhD program - Malawian PhD students building links nationally, regionally and globally	Status quo
2009–2017	- Capacity to use and participate in the development of DHIS2 building slowly up in Malawi - HISP Malawi established, drawing together expertise from different domains - A global network of DHIS2 experts established with key people from HISP Malawi	- DHIS2 established as open-source and web-based software - A maturing Web API with extensive functionality and documentation - DHIS2 Academies established - DHIS2 App development workshops creating/enhancing local capacity to develop DHIS2 apps - DHIS2 App store enabling transferability of innovations within the ecosystem

developed and discussed related to activities of innovation around the software platform DHIS2 in the developing country context of Malawi.

Socio-technical generativity acknowledges the complementary roles of social relationships and the capacity of technology in the shaping of innovation processes. As demonstrated by the case in Malawi, even if a technology is generative, it makes little difference in the fringes of a software ecosystem without the required social relationships to develop and nurture novel attributions. At the same time, the existence of strong social relationships in the fringes and relationships linking the fringes to regional and global networks is not enough to enable innovation if the attributes of the technology are not supportive of innovation. We have also shown how these different attributes change, and not necessarily in sync, and relate both to space and time.

Platforms have a central role in the increasing focus on digital innovation in society in general, and information system research in particular. With the significant prospect of digital innovation as argued by the trade and popular media, this research will potentially have significant impact. As a part of information systems research already engaged in the study of software platforms and innovation are calls to focus more on strategic roles of business frameworks (Yoo et al., 2010) and innovation networks (Lyytinen, Yoo, & Boland Jr., 2016). Nielsen further argues that also ICT4D research must engage in the role of software platforms in developing countries and how developing countries can actively contribute to and not only act as users of digital innovations (Nielsen, 2017). Innovation as a human activity includes access to and the understanding of the inner workings of technologies as well as the opportunities to develop new attributions of technology and act on them.

We have developed our concept of *socio-technical generativity* in the context of an open-source software platform and the network around it working toward strengthening the health system in developing countries. We argue that our perspective on socio-technical generativity will yield useful insights also in the analysis of other software platforms and bring understanding to how they are related to innovation. With a strategy of

supporting the implementation and use of DHIS2 as a global public good, HISP UiO is not designing an open technology platform with the aim of generating revenues. Rather, the design is geared toward strengthening national health information systems, fostering local innovation and the creation of local business in developing countries and regional collaboration. DHIS2 may have very different attributes and develop differently compared to other software platforms. For example, commercial platforms will require different business cases and typically have a much shorter time horizon to recoup investments. Thus, it remains to be seen if our perspective on generativity also makes sense related to other software platforms and other contexts.

From a practical point of view, our case study of DHIS2 in Malawi and our perspective on social-technical generativity have implications for both the design and implementation of software platforms in developing countries. First, software platforms must be designed and implemented with the necessary attributes of technology and social relationships. As a starting point, the platform must offer the required accessibility, adaptability and ease of mastery in the context it will be used. This will require developers to engage with and understand what the users need to participate in innovation, facilitate local capacity building and the development and distribution of the necessary tools, teaching materials, documentation, etc. In developing countries where resources are scarce, this is demanding. Engaging local higher education institutions to participate in and drive these processes has shown highly successful and sustainable in the case of DHIS2. Second, and as a natural part of capacity building, participants will build social relationships locally. Platform developers should nurture these relationships as well as work toward the establishment of regional and global networks of experts. These networks should not be limited to a certain group professionals, but should be heterogeneous and always include experts on technology as well as from relevant application domains. These relationships are not built overnight and careful planning is needed to ensure the resources and incentives necessary to sustain them over time. The value of creating local business opportunities and career paths for experts is a key learning from DHIS2 in this respect. Third, it is crucial to assure the openness of platforms in terms of offering appropriate permissions to a wide audience to innovate on top of it. This may simply be a matter of offering access rights easily and widely, or in other cases, long-term engagement in advocating for, making and implementing policies to establish a regime of openness.

We have in this paper developed a socio-technical perspective on generativity based on studying a software platform in a developing country. While we have illustrated the usefulness of this perspective in Malawi and related to DHIS2, it requires further research to show its value in other contexts and related to other software platforms. It would also be interesting to further explore the analytical strength of our perspective by also studying in a more focused way the role of platform "owners" (HISP UiO in our case). We have also focused on developing a holistic perspective by bringing two separate perspectives on the social and the technical together. In this, we have paid limited attention to the particular attributes of these perspectives. Further research is needed to scrutinize these attributes as a part of a holistic perspective and possibly remove, reframe or add new attributes. There is also a need to more in depth explore the relationships between the different social and technical attributes and the potential positive and negative influence between them and how they may reinforce or counteract each other. Finally, we have focused on the context of DHIS2 and how it has developed in Malawi from 1999 to 2017. But this is an ongoing

process and the way in which DHIS2 as a platform continues to develop in Malawi should be followed further.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Brown Msiska is a PhD candidate at the University of Oslo, Norway, and a faculty member of the University of Malawi. His research interests relate to open source software platforms and software ecosystems with a particular focus on open source health information systems in developing countries. In his PhD, he is looking at how to build generative capacities in developing countries to enable them to leverage and innovate on open source health information software platforms and he is using DHIS2 as the focal software platform.

Petter Nielsen is an Associate Professor at the Department of Informatics, University of Oslo, Norway. His research interest relates to large-scale and complex information systems – how they are evolving, how we can govern them, and the role of architecture. He have empirically studied and theorized large-scale and complex information systems related to digital innovation in the domain of platforms and ecosystems for mobile phone services in Northern Europe and currently health information systems in developing countries.

References

Abbate, J. (1994). The Internet challenge: Conflict and compromise in computer networking. In J. Summerton (Ed.), *Changing large technical systems* (pp. 293–210). Oxford: Westview Press.

Abbate, J. (1999). *Inventing the Internet*. Cambridge, MA: MIT Press.

Avital, M., & Te'eni, D. (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.

Benkler, Y. (2006). *The wealth of networks: How social production transforms markets and freedom*. New Haven, CT: Yale University Press.

Bowker, G. C., & Star, S. L. (2000). *Sorting things out: Classification and its consequences*. Cambridge, MA: MIT Press.

Braa, K., & Nielsen, P. (2015). *Sustainable action research: The networks of actions approach*. 13th international conference on social implications of computers in developing countries, Sri Lanka.

David, P. A. (2001). *The beginnings and prospective ending of "end-to-end": An evolutionary perspective On the internet's architecture*. Stanford, CA: SIEPR Discussion Paper No. 01-04 .

Diga, K., & May, J. (2016). The ICT ecosystem: The application, usefulness, and future of an evolving concept. *Information Technology for Development*, *22*(Suppl. 1), 1–6.

Dittrich, Y. (2014). Software Engineering Beyond the Project – Sustaining Software Ecosystems. *Information and Software Technology*, 1436–1456.

Eck, A., Uebernickel, F., & Brenner, W. (2015). *The generative capacity of digital artifacts: A mapping of the field*. PACIS 2015 proceedings.

Elaluf-Calderwood, S., Herzhoff, J., Sørensen, C., & Eaton, B. D. (2011). *Control points and tussles in flexible Mobile network innovation*. Helsinki: ECIS.

Faraj, S., Kwon, D., & Watts, S. (2004). Contested artifact: Technology sensemaking, actor networks, and the shaping of the Web browser. *Information Technology & People*, *17*(2), 186–209.

Gawer, A. (2011). *Platforms, markets and innovation*. Cheltenham: Edward Elgar Publishing.

Gizaw, A., Bygstad, B., & Nielsen, P. (2017). Open generification. *Information Systems Journal*, *27*(5), 619–642.

Hanseth, O., & Lyytinen, K. (2010). Design theory for dynamic complexity in information infrastructures: The case of building internet. *Journal of Information Technology*, *25*(1), 1–19.
Henfridsson, O., & Bygstad, B. (2013). The generative mechanisms of digital infrastructure evolution. *MIS Quarterly*, *37*(3), 896–931.
Lane, D. A. (2011). Complexity and innovation dynamics. In Cristiano Antonelli (Ed.), *Handbook on the economic complexity of technological change* (pp. 63–80). Cheltenham: Edward Elgar Publishing.
Law, J. (1999). After ANT: Complexity, naming and topology. *The Sociological Review*, *47*(S1), 1–14.
Lessig, L. (2001). *The future of ideas: The fate of the commons in a connected world* (1st ed). New York, NY: Random House.
Lyytinen, K., Yoo, Y., & BolandJr.R. J. (2016). Digital product innovation within four classes of innovation networks. *Information Systems Journal*, *26*(1), 47–75.
Manikas, K., & Hansen, K. M. (2013). Software ecosystems – A systematic literature review. *Journal of Systems and Software*, *86*(5), 1294–1306.
Moyo, C., Frøyen, M. H., Sæbø, J. I., & Kaasbøll, J. J. (2015). Using performance league tables to promote accountability and feedback in health management in Malawi. In *Proceedings of the 13th international conference on social implications of computers in developing countries* (pp. 263–276). Negombo: IFIP WG 9.4.
Moyo, C., Kaasbøll, J., Nielsen, P., & Sæbø, J. (2016). The information transparency effects of introducing league tables in the health system in Malawi. *The Electronic Journal of Information Systems in Developing Countries*, *75*, 1–16.
Nielsen, P. (2017). Digital innovation: A research agenda for information systems research in developing countries. In J. Choudrie, M. Islam, F. Wahid, J. Bass, & J. Priyatma (Eds.), *Information and communication technologies for development* (Vol. 504, pp. 269–279). Cham: Springer. IFIP Advances in Information and Communication Technology.
Nielsen, P., & Sæbø, J. I. (2016). Three strategies for functional architecting: Cases from the health systems of developing countries. *Information Technology for Development*, *22*(1), 134–151.
Qureshi, S. (2013). Networks of change, shifting power from institutions to people: How are innovations in the use of information and communication technology transforming development? *Information Technology for Development*, *19*(2), 97–99.
Saltzer, J. H., Reed, D. P., & Clark, D. D. (1984). End-to-end arguments in system design. *ACM Transactions on Computer Systems*, *2*(4), 277–288.
Sommerville, I., Cliff, D., Calinescu, R., Keen, J., Kelly, T., Kwiatkowska, M., … Paige, R. (2012). Large-scale complex IT systems. *Communications of the ACM*, *55*(7), 71–77.
Tiwana, A., Konsynski, B., & Bush, A. A. (2010). Research commentary – Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research*, *21*(4), 675–687.
University of Malawi. (2016). *Computer science department holds DHIS2 workshop – University of Malawi | Chancellor College [Academic]*. Retrieved from http://cc.ac.mw/news/computer-science-department-holds-dhis2-workshop-17-03-2016
van Schewick, B. (2012). *Internet architecture and innovation*. Cambridge, MA: MIT Press.
Yoo, Y., Boland, R. J., Lyytinen, K., & Majchrzak, A. (2012). Organizing for innovation in the digitized world. *Organization Science*, *23*(5), 1398–1408.
Yoo, Y., Henfridsson, O., & 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.
Zittrain, J. (2006). The generative internet. *Harvard Law Review*, *119*(7), 1974–2040.
Zittrain, J. (2008). *The future of the internet and how to stop it*. New Haven, CT: Yale University Press.

Appendix 5: Paper 4

Msiska, B., 2018. Cultivating Third Party Development in Platform-centric Software Ecosystems: Extended Boundary Resources Model. *The African Journal of Information Systems*, *10 (4), Article 6, 348–365*

Cultivating Third Party Development in Platform-centric Software Ecosystems: Extended Boundary Resources Model

Research Paper – Special Issue
Volume 10, Issue 4, October 2018, ISSN 1936-0282

Brown Msiska
University of Oslo, University of Malawi
bmsiska@gmail.com

(Received *January 2018, Accepted May 2018*)

ABSTRACT

Software ecosystems provide an effective way through which software solutions can be constructed by composing software components, typically applications, developed by internal and external developers on top of a software platform. Third party development increases the potential of a software ecosystem to effectively and quickly respond to context-specific software requirements. The boundary resources model gives a theoretical account for cultivation of third party development premised on the role of platform boundary resources such as application programming interfaces (API). However, from a longitudinal case study of the DHIS2 software ecosystem, this paper observes that no matter how good the boundary resources a software ecosystem provides, third party development remains a mere possibility until there exists adequate external generative capacity. Taking into consideration this observation, this paper contributes by extending the boundary resources model to foreground external generative capacity alongside boundary resources as factors that influence third party development.

Keywords

Software Platforms, Software Ecosystems, Third Party Development, Boundary Resources, Generative Capacity, Developing Countries

INTRODUCTION

Software ecosystems provide an effective way through which software solutions can be constructed by composing software components, typically applications, developed by internal and external actors on top of a software platform (Manikas and Hansen, 2013). By leveraging an existing software platform alongside applications specific to it, the time, cost and risk of implementing a software solution can be reduced. A software ecosystem consists of a software platform, a set of applications specific to the platform, a set of internal and external developers (also referred to as *third party developers*), a community of domain experts, and a community of end users whose information needs are met by the

software platform and associated applications (Bosch and Bosch-Sijtsema, 2010). Software ecosystems exist within a larger competitive environment, often competing with other rival ecosystems for end users as well as third party developers (Tiwana, 2013). Google's Android and Apple's iOS ecosystems are popular examples in this regard.

Core to a platform-centric software ecosystem is a software platform, a software system with an extensible codebase that provides core functionality shared by applications that interoperate with it, interfaces through which they interoperate, and resources with which derivative applications can be created (Eck et al., 2015; Ghazawneh and Henfridsson, 2013). The value of a software ecosystem to its end users is partly established by the range of applications specific to its platform. Applications, or apps, are add-on software subsystems that connect to the software platform and add functionality to it (Tiwana, 2013; Tiwana et al., 2010). Applications are a manifestation of "innovation on" (Grisot et al., 2014) whereby innovations are built on top of and without disrupting an underlying artefact.

Despite its benefits, the software ecosystem approach is not one without challenges. Challenges emanate from having a heterogeneous community of end users that are remote from platform developers. In this regard, one challenge faced by platform owners is lack of familiarity with the end user context which makes it difficult for them to make informed decisions on what services or applications to develop for the platform (Henfridsson and Lindgren, 2010). Another challenge is how to effectively respond to turbulent and often diverging demands for innovation from the end users (Ghazawneh and Henfridsson, 2013). Consequently, third party development which involves devolving the development of applications on top of a software platform to external developers in or close to the end user community is increasingly attractive for software platform owners (Ghazawneh and Henfridsson, 2013) as it allows them to focus on the platform core and defer satisfying specific end user requirements to third party developers. Familiarity and proximity to the end user context enables third party developers to make informed decisions on applications to develop in response to specific demands for innovation among end users.

Failure or delay responding to needs in the end user community can diminish the value of the software platform among its end users and give room to rival software ecosystems. The demise of once dominant Symbian and Blackberry ecosystems and rise of iOS and Android ecosystems is a good example in this regard. For platform owners, third party development is attractive because it increases the potential of a software ecosystem to effectively and quickly respond to specific end user needs. Third party development also works in favor of the end user community as it brings innovation close to the context of use. The proximity of third party developers to end users brings about agility in responding to innovation demands. As a result, enabling third party development is not only in the interest of platform owners but platform adopters as well.

Furthermore, third party development can accelerate innovation in a software ecosystem (Bosch, 2009) and be the basis for market leadership (Ghazawneh and Henfridsson, 2013). For example, in the year 2012 Apple's iOS ecosystem had about 200,000 external developers through which it was able to muster enough innovative third-party applications to usurp market leadership from the Blackberry ecosystem which only had 8000 external developers (Tiwana, 2013). The wealth of derivative applications needed for a software ecosystem to stay competitive is difficult to accomplish using in-house developers alone. Thus, to successfully build platform-centric software ecosystems third party development must be cultivated and this entails shifting design capability to external actors (Prügl and Schreier, 2006; von Hippel and Katz, 2002). This shift is necessary to build or enhance the generative capacity (Avital and Te'eni, 2009) of external developers.

This paper is based on a case study of the *District Health Information Software version 2 (*DHIS2) software ecosystem. DHIS2 is an open source software platform that enables governments and organizations to collect, manage and analyses data in the health domain and beyond. It is currently in use by ministries of health in more than 60 countries. The DHIS2 platform is developed under the Health Information Systems Program (HISP) at University of Oslo in Norway. DHIS2 supports third party development by providing an application programming interface (API) with which third party applications can be constructed. Despite the provision of the API and associated platform boundary resources to facilitate third party development, the quality and number of third party applications in DHIS2 app store is still limited (Polak, 2015). A large part of DHIS2's end user community is in developing countries and the shortage of ICT skills in such countries are well documented (Kimaro, 2006; Kimaro and Nhampossa, 2005; Mutula and Van Brakel, 2007). The dearth of generative capacity in developing countries leaves a majority of DHIS2 end users dependent on core developers for the implementation of desired applications. This creates an innovation bottleneck which often results in failure or delays in responding to specific end user needs.

Coming from this background, this paper uses the DHIS2 software ecosystem as a case to address the question: how can third party development in a platform-centric software ecosystem be cultivated? The boundary resources model (Ghazawneh and Henfridsson, 2013) provides a theoretical account for cultivating third party development premised on the role played by boundary resources in enabling third party development. However, as observed in the DHIS2 software ecosystem, providing platform boundary resources alone is not enough as a catalyst for third party development. Besides platform boundary resources, third party development also depends on the generative capacity of external developers. With this in consideration, this paper extends the boundary resources model to foreground external generative capacity alongside boundary resources as factors that have influence on third party development.

BOUNDARY RESOURCES MODEL

Platform boundary resources are software tools and regulations that serve as the interface between platform owners and third-party developers facilitating the development of third party applications (Ghazawneh and Henfridsson, 2013). Such boundary resources typically consist of software development kits (SDKs) and application programming interfaces (APIs). Platform boundary resources have the power to stimulate or constrain generativity in a software ecosystem (*ibid.*). The boundary resources model, illustrated in Figure 1, is a set of related constructs that provides an intellectual structure with which the role of platform boundary resources in stimulating third party development can be understood.

In the model, *boundary resources design* is a process involving the platform owner developing new or modified boundary resources to enhance external contributions and address control concerns. The design of boundary resources can be reactive or proactive. The design of boundary resources to enhance opportunities for external contribution increases the scope and diversity of the software platform and is thus denoted *resourcing* the software platform. On the other hand, control concerns may emerge when third party developers launch applications that represent potential threats to the platform. As a result, boundary resources may be (re)designed to address the platform owner's control concerns, a process referred to as *securing*. Finally, *boundary resources use* is a process involving third party developers developing end user applications by leveraging the boundary resources offered by the platform owner.

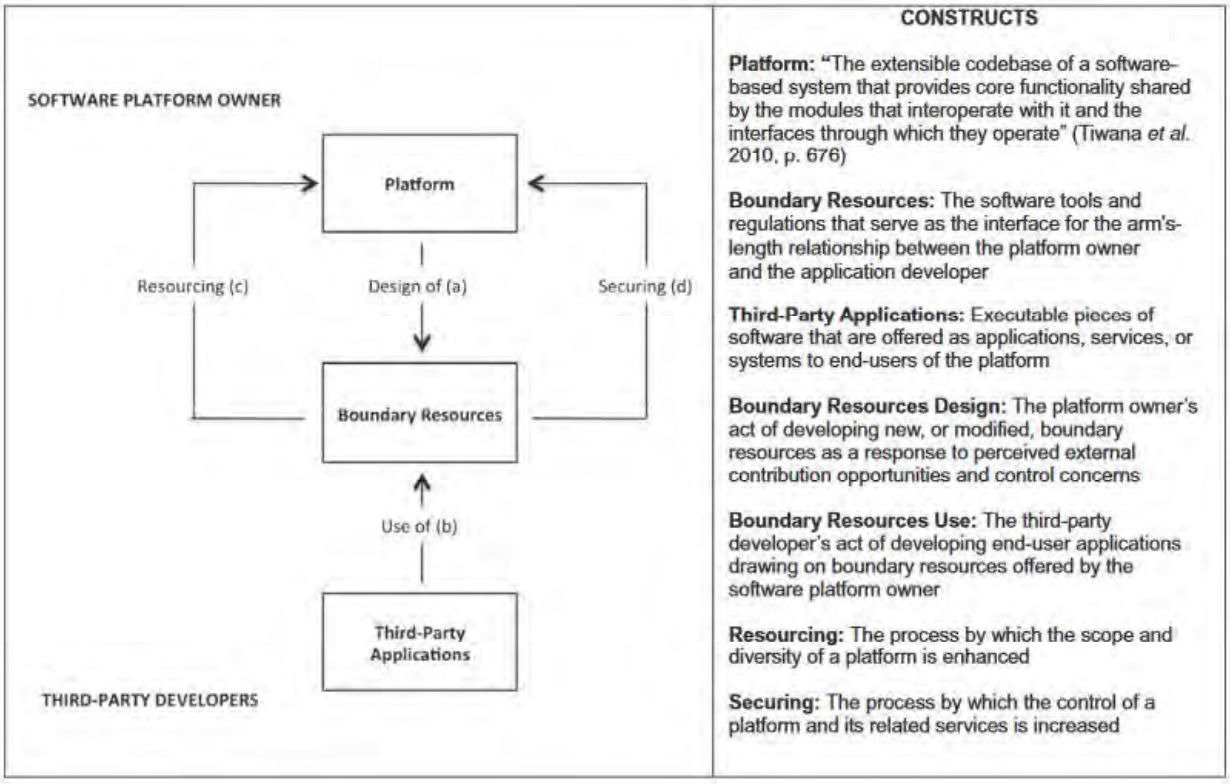


Figure 1: The Boundary Resources Model (Ghazawneh and Henfridsson, 2013)

Generativity and the Boundary Resources Model

The term generativity has been used to describe the ability of a technology artefact, for example a software platform, to attract or evoke innovations from external actors. In general terms, generativity refers to the ability or capacity to generate or produce something (Avital and Te'eni, 2009). With respect to technology artefacts, generativity has been defined as the overall capacity of a technology or a system to be malleable or changed by diverse groups of actors in ways unanticipated by its creators (Eck et al., 2015; Zittrain, 2008, 2006). This means that the extent of third party development in each software ecosystem depends on the generativity of the corresponding software platform.

Notwithstanding the importance of software platform generativity, third party development is an activity carried out by human beings and is subject to their competences. Therefore, the generative capacity of external developers is equally important in cultivating third party development. In this case, generative capacity is an attribute of a person that refers to one's ability to produce something in a particular context (Avital and Te'eni, 2009). Therefore, third party development in a software ecosystem depends on the generative capacity of its third-party developers as much as it depends on the generativity of its software platform. The boundary resources model, by focusing on platform boundary resources, emphasizes the importance of software platform generativity and backgrounds the generative capacity of third party developers.

Community Boundaries and the Boundary Resources Model

A software ecosystem has several communities of practice (CoPs). A community of practice is a group of people pursuing a common objective (a joint enterprise), sharing a repertoire of tools and ways to solve recurring problems (a practice) and having a shared domain of interest (an identity) that makes them different from other people (Wenger, 2011). With respect to software ecosystems, each of the communities of internal developers, external developers, domain experts and end users constitute a community of practice. CoPs are important social units through which competences and experiences can be exchanged and acquired.

The existence of a community of practices implicitly suggests the existence of boundaries. Boundaries are a separation between two groups of people arising from differences in enterprise, repertoires and capabilities (Wenger, 2000). Boundaries are channels through which competences, experiences and resources can be exchanged, allowing co-learning across communities and enriching capacities on each side of the boundary. The exchange across boundaries is facilitated by three bridges: boundary objects, boundary interactions and brokers (Wenger, 2000). Boundary objects are artefacts, for example tools and documents, linking or shared by communities of practice across a boundary. Boundary interactions are events or encounters (for example visits and meetings) that provide direct exposure to members of an external CoP. Lastly, brokers are human actors that operate between communities of practice and engaged in 'import-export' of competences, knowledge and resources.

The boundary resource model is based on the boundary object (Bergman et al., 2007; Carlile, 2002; Star and Griesemer, 1989; Wenger, 2000) construct. The model focuses on the role played by software artefacts, such as APIs and SDKs, in cultivating third party development. The model is, however, silent on the potential role-played human agents operating between internal developers and third-party developers in shaping third party development. Similarly, the role of boundary interactions such as events aimed at enhancing the capacity of third party developers is left in the background. Our understanding of how to cultivate third party development can, therefore, be further strengthened by drawing insights from boundary interactions and brokers in addition to those from boundary objects.

METHODOLOGY

As stated in the introduction, the aim of this paper is to address the question how can third development in a platform-centric software ecosystem be cultivated? To answer this question a longitudinal case study (Creswell, 2014; Yin, 2013; Zainal, 2007) involving the DHIS2 software platform was carried out. The transitions DHIS2 has undergone to allow and foster third party innovations make it an ideal case for addressing the research question above. DHIS2 traces its origin to DHIS 1.0 which was a desktop application based on proprietary Microsoft technologies such as Microsoft Access. Challenges with proprietary technology led to the release of DHIS2, an open source web based application. Growing worldwide usage of DHIS2 made it difficult for University of Oslo to effectively and timely address divergent user requirements. To relieve the burden on University of Oslo there was a need to support third party innovations as a result DHIS2 embraced a software platform approach. The ongoing quest to encourage the development of third party applications on top of it makes the DHIS2 software platform an interesting case in trying to understand how to cultivate third party development.

Different data collection strategies were employed during the case study running from 2014 to 2017. These included: interviews, field experiments (pilots), participant observation and document reviews. As part of observations, in 2014 the researcher attended four weeks of the open source software development masters course at the University of Oslo. During the course master students are taught the

fundamentals of DHIS2 application development. Going further with observations, between 2014 and 2017 the researcher attended three DHIS2 expert academies in Oslo (Norway) and 1 DHIS2 application development academy in Dar es Salaam (Tanzania). These observations were complemented by field experiments (pilots) in Malawi in form of two application development workshops carried out in Malawi. The first workshop took place in March 2016 and focused on development of web applications on top of DHIS2. The second workshop took place in October 2017 and focused on development of Android applications on top of DHIS2.

Over the period between 2014 and 2017, several interviews were carried involving organizers, facilitators, students and participants of DHIS2 related courses, academies and workshops. Participants in the academies and workshops were people interested in developing or already developing applications for the DHIS2 platform. These interviews centered around the various boundary resources, boundary interactions and human agents and how they are shaping third party software development in the DHIS2 ecosystem. Respondents in the interviews were given an informed consent form ensuring their privacy and confidentiality among other things. Altogether, 33 interviews have been conducted between 2014 and 2017. This is summarized in table below.

Table 1. Data Collection Summary

Period	Data Collection Event	Interviews	Other Techniques
August 2014	Open Source Software Development course, MSc in Information Systems Program, University of Oslo, Norway	0	Participant observation
August 2015	DHIS2 Experts Academy, University of Oslo, Norway	2	Participant observation
March 2016	DHIS2 Web Apps Development Workshop, Zomba, Malawi	5	Field experiment, Participant observation
August 2016	DHIS2 Experts Academy, University of Oslo, Norway	3	Participant observation, Meetings
January 2017	DHIS2 Web Apps Development Workshop, Kampala, Uganda	7	Field experiment, Participant observation
August 2017	DHIS2 Experts Academy, University of Oslo, Norway	3	Participant observation, meetings
October 2017	DHIS2 Android Apps Development Workshop, Zomba, Malawi	2	Field experiment, Participant observation
November 2017	DHIS2 Web Apps Development Academy, Dar es Salaam, Tanzania	11	Participant observation

Document reviews were carried by going through documentation, reports, research accounts and other written material on DHIS2. Further data was collected through participation in projects that aimed at either building capacities of third party developers' capacity to build DHIS2 applications or actually developing a DHIS2 application. This included participating in the content development, facilitation and organization for the DHIS2 application workshops in Malawi. In addition, data was collected by participating in the mHealth4Afrika project which is an ongoing joint project involving University of Oslo (Norway), University of Gondar (Ethiopia), University of Malawi (Malawi), Strathmore University (Kenya), Nelson Mandela Metropolitan University (South Africa) and International Information Management Corporation (Ireland) building a maternal and child health records system on top of DHIS2.

The data collected using the various data collection techniques outlined above was recorded in various forms: field notes, google forms and sheets, audio recordings and photos. This data included among other things existing boundary resources in DHIS2 ecosystem and their generative attributes, proposed boundary resources, capacity building initiatives and respective factors for the success or failure of such initiatives. Where audio recordings were captured they were later transcribed to provide a corresponding textual account. Using the boundary resources model, generativity concept - generative capacity, and community of practice concepts - brokers, boundary interactions and boundary objects as analytical lens the data collected was thematically analyzed. The results of this analysis are presented in the discussion and concluding remarks sections below.

THE CASE OF DHIS2

DHIS2 (figure 2) is an open source software platform developed by the Health Information Systems Program (HISP) at the University of Oslo (HISP UiO). It is used primarily to collect, manage, aggregate, analyze and visualize routine health data by ministries of health and other stakeholders in developing countries. Although it is primarily built for use in the health domain, the generic nature of DHIS2 has seen its use in other domains grow over the years. Over the period of its existence, DHIS2 has evolved from a desktop application built around proprietary technologies to an open source and web based application, and now, to an open source software platform with a RESTful Web API that supports creation of third party innovations using ubiquitous web technologies such as JavaScript, CSS and HTML5.

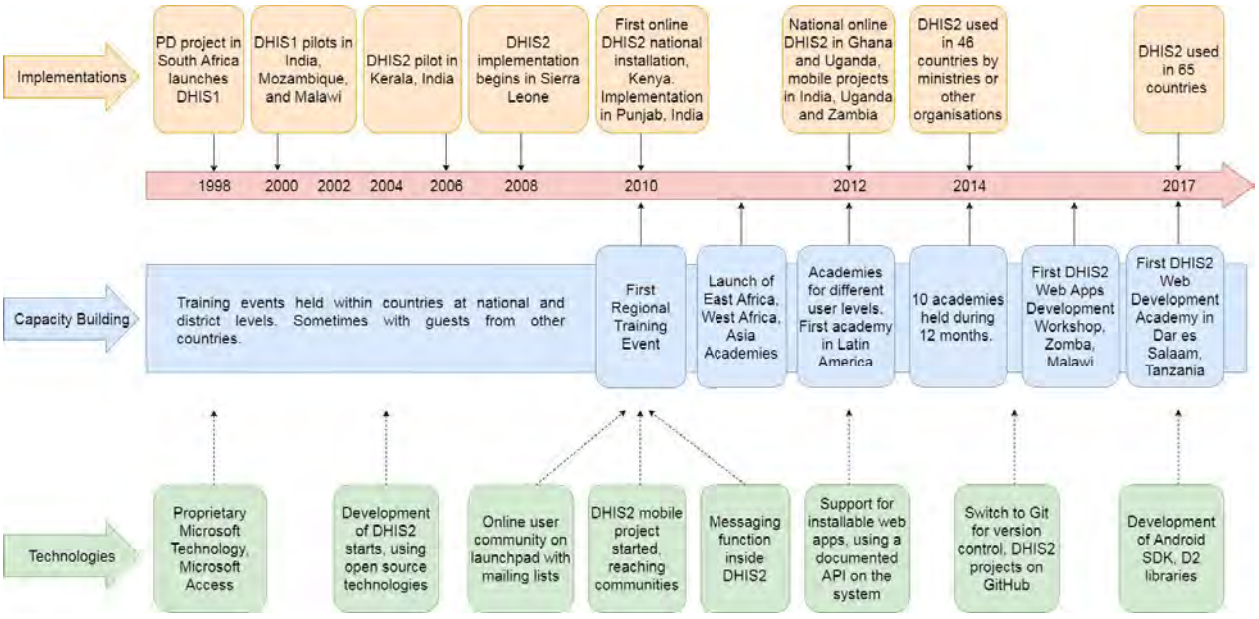


Figure 2: DHIS2 Timeline (adapted from a slide in the DHIS2 Online Academy)

DHIS2 traces its origin to the Reconstruction and Development program in post-apartheid South Africa under which HISP was initiated in 1995. One of the aims of the program was creating a unified HIS across the country. To fulfil this aim, a pilot project to develop a district-based HIS in the Western Cape Province was proposed. Between 1996 and 1998, the pilot project received financial backing from the

Norwegian Agency for Development Cooperation (NORAD) and resulted in a prototype of what was called District Health Information Software (DHIS) released in 1998. Successful implementations in the Western Cape led to the adoption of DHIS by the Eastern Cape Province in the same year and by the year 2000 several other provinces in South Africa had adopted DHIS.

From South Africa DHIS was later introduced to other developing countries including Mozambique, Malawi and India. At the time, DHIS was a desktop application built using Microsoft Access. This presented two challenges. First, as a desktop application it lacked support for sharing data between geographically distant stakeholders. Second, the use of proprietary Microsoft technology made scaling the solution costly as rolling out entailed paying license fees for each workstation where DHIS was installed. To address these two challenges, HISP UiO, with support from various international donors, embarked on a project to develop an open source and web based version of the software. This resulted in what is now known as DHIS2.

Platformisation of DHIS2

With DHIS2 now a free, open source and web-based application the problems of shared access and cost of scaling were alleviated. This has since led to increased adoption of DHIS2 by ministries of health and non-governmental organizations in developing countries. It is currently in use in more than 60 countries. This includes, among others, countries in the Eastern and Southern Africa region such as Uganda, Kenya, Rwanda, Tanzania, Malawi, Zambia, Zimbabwe, South Africa and Mozambique.

The increased adoption of DHIS2 came with its own challenges. First, because of increased adoption there was also an increase in user requirements that the developers at HISP UiO had to address to satisfy the end user community. At the same time, the different context under which the end users operate meant that the user requirements were divergent and often competing. Effectively and timely addressing such divergent end user requirements became a challenge. This resulted in complaints and queries from the end-user community regarding how user requirements are prioritized and addressed.

With the switch to being open source, it had been envisaged that the innovation burden on DHIS2 will be shared between HISP UiO developers and external developers in the end user community. However, the core of DHIS2 was developed using Java and requisite skills were not as ubiquitous among external developers in the end user community. As a result, a large part of the end user community remained dependent on HISP UiO for the implementation of requirements specific to their context. This created an innovation bottleneck against developers at HISP UiO resulting in delays and sometimes failure to respond to some end user requirements.

The core of DHIS2 is developed using Java and a stack of related software development frameworks. For most external developers, developing for DHIS2 meant learning a new programming language as well as a stack of associated technologies used and therefore not as attractive. To effectively and timely address end user requirements it became necessary to devise a way to allow external developers develop solutions on top of DHIS2 without requiring them to learn Java and a stack of software frameworks used by HISP UiO. This has seen DHIS2 evolving from a mere open source software to now an open source software platform.

DHIS2 Boundary Resources

As a software platform DHIS2 (figure 3) provides a range of boundary resources to enable third party development. First, within its core DHIS2 supports the creation of custom modules that can co-exist with core modules. Such modules must be written in Java alongside frameworks such as Spring and Hibernate. Second, DHIS2 now has a RESTful Web API with which external developers can develop third party applications on top of DHIS2. The API continues to evolve with each new version of DHIS2 with the aim of offering better support to third party developers.

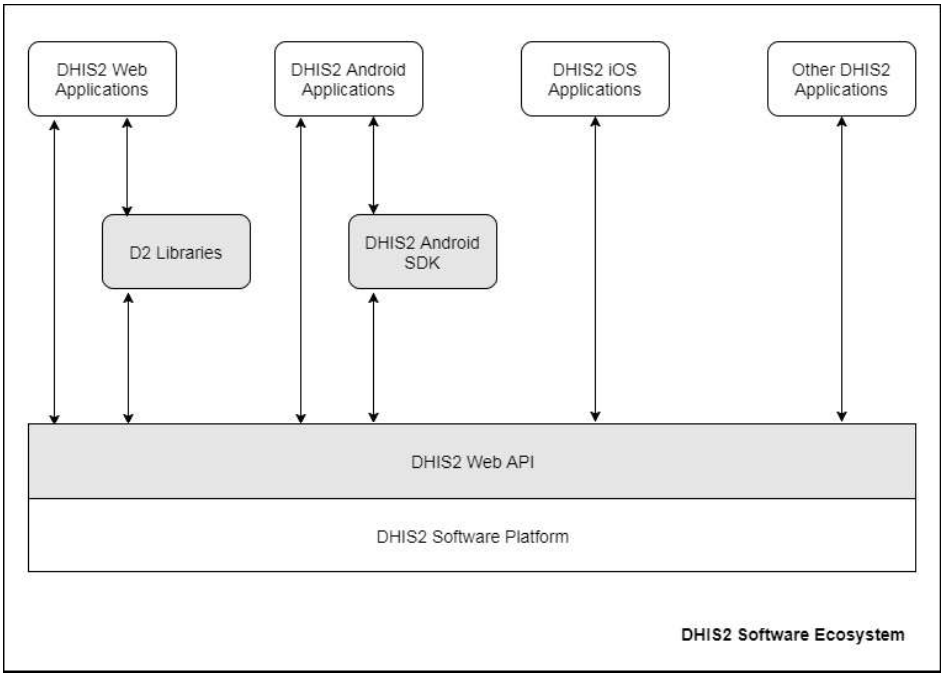


Figure 3: DHIS2 Software Ecosystem

Using the Web API, external developers do not necessarily need to learn a new programming language just to develop applications for DHIS2. As long as the programming language they are familiar with has features to handle web-based data they can use it to develop applications for DHIS2. Typically, third party DHIS2 applications are web applications written using JavaScript, CSS and HTML5. However, the flexibility of the Web API allows for creation of different kinds of applications as well. For example, the popularity of Android as a computing platform in developing countries has attracted interest in DHIS2 Android applications.

In addition to the Web API, HISP UiO has developed other auxiliary boundary resources to support specific developer needs. One of such resources is the d2 library, a JavaScript library which provides a level of abstraction above the Web API allowing third party developers to develop web applications without requiring in depth knowledge of the API. Another of such auxiliary boundary resources is the DHIS2 Android SDK, currently in beta, which abstracts the Web API when building Android applications on top of DHIS2. Besides boundary resources created by HISP UiO a number of community-driven boundary resources. For example, some third-party developers have created "seed applications" which are dummy applications containing boiler plate code that allows new developers to quickly get started building DHIS2 applications.

To facilitate distribution of third party applications, a dedicated online app store for DHIS2 web applications (figure 4) was created and can currently be accessed on https://play.dhis2.org/appstore/. There is also a dedicated app store for DHIS2 Android applications which can currently be accessed on https://www.dhis2.org/appstore-android. The number of available applications in both app stores is however limited. At the time of writing, there were 5 applications in the DHIS2 Android app store and 13 applications in the DHIS2 Web Applications app store. Most of these apps, particularly on the DHIS2 Android app store, are developed by HISP UiO.

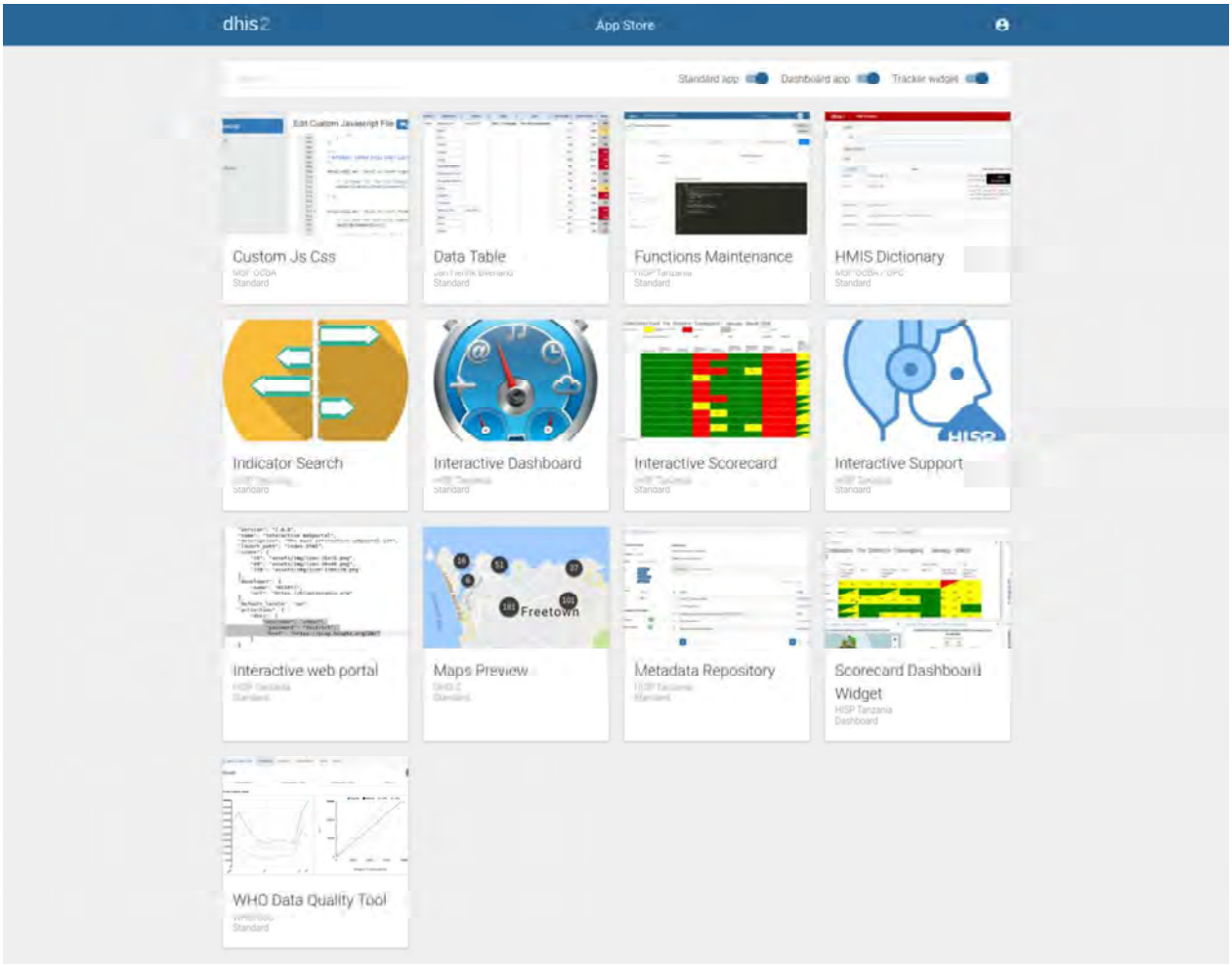


Figure 4: DHIS2 App Store

Building Third Party Development Capacity

Despite the existence of a range of boundary resources to support development of third party applications on top of the DHIS2 platform, the range of applications available in its app stores is however limited. Furthermore, several applications in the app stores have been developed by HISP UiO itself. Providing the boundary resources has not necessarily resulted in distributing the innovation

burden as much as it was envisaged in the platformisation of DHIS2. This is partly attributed to lack of awareness and requisite development capacity among potential third-party developers.

To address the capacity and awareness gap, HISP UiO, in collaboration with its global partners, has several initiatives aimed at enhancing the generative capacity of prospective third-party developers. Since the inception of DHIS2, University of Oslo has been running a master course on Open Source Software Development centered on the software. The course has evolved over the years to embrace the platform approach taken on DHIS2. In the course, students work on different projects developing third party applications on top of DHIS2. In Malawi, the University of Oslo is working in collaboration with University of Malawi where a similar course was introduced as part of a Master of Science in Informatics program.

Within the DHIS2 community, DHIS2 academies have been a popular vehicle for building different kinds of capacity among end users and technical personnel (e.g. as shown in figure 5). Topics at such academies have ranged from information use through to customization and implementation of DHIS2. The DHIS2 community is now exploiting the same vehicle to cultivate generative capacities of prospective third-party developers. With funding from UNICEF, in March 2016, HISP UiO in collaboration with the University of Malawi held a DHIS2 Application Development Workshop in Zomba, Malawi as a pilot for a potential DHIS2 Application Development academy. The workshop drew participants from Malawi, Kenya, Ethiopia and Zambia. A similar pilot workshop was carried out in January 2017 in Kampala, Uganda and attracted participants from Rwanda, Kenya, Uganda, Tanzania, Malawi, and South Africa. Following feedback from the pilots a full scale DHIS2 Application Development Academy took place in November 2017 in Dar es Salaam, Tanzania.

Figure 5: DHIS2 Academies Website

In addition to academies, other avenues for cultivating the generative capacity of third party developers are being explored. For example, recently the University of Oslo entered an agreement with University of Malawi and the Eduardo Mondlane University in Mozambique for a staff and student exchange program which will among other things allow HISP UiO to host technical personnel from Malawi and

Mozambique as a form of internship. In Tanzania, HISP Tanzania has an internal developer mentorship program where experienced DHIS2 developers work with new recruits to build their capacity. This has led to HISP Tanzania amassing adequate local capacity which has enabled them to develop several third-party applications on top of DHIS2. All these efforts are being done in complement to existing efforts on the boundary resources described in the earlier section.

DISCUSSION

The platformisation of DHIS2 and the provision of various boundary resources has made it possible for external developers to develop third party applications on top of DHIS2. The aim has been to usher in an era of distributed innovation whereby the innovation burden on DHIS2 is distributed between internal developers at HISP UiO and third-party developers within the ecosystem. HISP UiO sees the provision of boundary resources as critical to the achievement of this aim. This agrees with the intellectual account provided by the boundary resources model (Ghazawneh and Henfridsson, 2013). Without boundary resources, third party developers cannot extend DHIS2 functionality without the risk of forking the software. With appropriate boundary resources, useful third-party applications can be developed while at the same time circumventing forkability.

However, empirical data from the DHIS2 ecosystem indicate that provision of boundary resources is not in itself an endgame in cultivating third party development. Besides providing boundary resources, it was observed that third party development requires building generative capacities of third party developers. This agrees with the argument that third-party development demands shifting of software design and development capabilities from internal to external developers (Prügl and Schreier, 2006; von Hippel and Katz, 2002). This calls for the extension of the boundary resources model to foreground generative capacity alongside boundary resources as factors having impact on third party development. Sections 5.1 and 5.2 present two dimensions that can be used to extend the boundary resources model. Section 5.3 concludes the discussion by providing an extended boundary resources model and is followed by concluding remarks.

Software Development and Capacity Building Boundary Resources

The boundary resources model is based on the boundary object concept (Ghazawneh and Henfridsson, 2013, 2010). Drawing from this concept the model defines boundary resources as software tools and regulations that serve as an arm's length interface between platform owners and third-party developers facilitating the development of third party applications (Ghazawneh and Henfridsson, 2013). With this definition, our understanding of boundary resources is limited to technical artefacts, such as SDKS, APIs and libraries that are provided alongside a platform to facilitate third party development. However, the boundary between platform owners and external developers is not characterized by boundary objects alone. Besides boundary objects, boundary interactions and human agents as brokers are deployed to facilitate third party development. For example, the DHIS2 ecosystem has application development academies and workshops as boundary interactions aimed at cultivating third party development. Such boundary interactions rely on facilitators and mentors entrusted with the task of instituting a shift in software design capabilities from internal developers to third party developers. Just like the boundary objects, these boundary interactions and brokers can be regarded as boundary resources facilitating third party development. This calls for extending our understanding of boundary resources to include brokers and boundary interactions in addition to boundary objects.

Extending our understanding of boundary resources to include brokers and boundary interactions allows for a distinction between software development and capacity building boundary resources. Software development boundary resources are software artefacts that are used to develop third party applications. Such boundary resources include SDKs, API, seed applications and libraries. On the other hand, capacity building boundary resources are boundary objects, boundary interactions and brokers that are used to build the third-party development capacity of external developers. Within the DHIS2 ecosystem, such boundary resources include capacity building events such as application development academies and workshops, boundary objects such as developer documentation, and brokers such as academy and workshop facilitators and experienced developers acting as mentors for new third-party developers. To cultivate third party development, both software development and capacity building boundary resources are required.

While this paper uses the distinction between software development and capacity building boundary resources to extend the boundary resources, further distinction can be made between endo-platform boundary resources and exo-platform boundary resources, and between platform-owner driven boundary resources and community driven boundary resources. Endo-platform boundary resources are boundary resources that come bundled as part of the software platform. These include software artefacts such as APIs, SDKs and libraries. On the other hand, Exo-platform boundary resources are boundary resources that exist outside the software platform. These include software artefacts such as documentation, boundary interactions such as academies and brokers such as facilitators and mentors. Going further with this discussion, platform-owner driven boundary resources are those that are put in place by the platform owner to facilitate third party development. On the other hand, community-driven boundary resources are those that are derived through efforts by third party developers or other stakeholders other than the platform owner.

Internal and External Generative Capacity

The case of the DHIS2 reveals that third-party development requires more than just providing software development resources such as SDKs and APIs. There is a need to propagate software design and development capabilities to external developers to enable them to develop third party applications. Therefore, provision of software development boundary resources must be complemented by mechanisms aimed at building the generative capacities of external developers. In the DHIS2 ecosystem such mechanisms include application development academies and workshops. In literature on software platform generativity, the term generative capacity has been used to describe a collection of competences that enable an individual to produce something innovative in a particular context. With respect to third party development, generative capacity relates to a developer's ability to develop third party applications on top of a given software platform.

In this regard, distinction can be made between internal generative capacity and external generative capacity. Internal generative capacity refers to platform owners' software development and related competencies that enable them design, develop and maintain the software platform and platform owner driven boundary resources. On the other hand, external generative capacity refers to third party developers' software development and related competences that enable them to design, develop and maintain third party applications and community-driven boundary resources. No matter how good the software development boundary resources a software ecosystem has, third party development remains a mere possibility until there exists adequate external generative capacity.

Extended Boundary resources model

Taking into consideration the foregoing discussion the paper presents, an extended boundary resources model in figure 6 is shown below. Below the figure descriptions of each of the construct used in the model are provided. In the extended boundary resources model the paper introduces the constructs *internal generative capacity*, *external generative capacity*, *generative capacity use* and extends the *boundary resources use* construct to include the use of capacity building boundary resources by platform owners and other stakeholders in the ecosystem to build external generative capacity. In addition, constructs *software development boundary resources* and *capacity building boundary resources* replace the boundary resources construct. Adding these constructs helps foreground other factors critical for third party development besides provision of software development boundary resources.

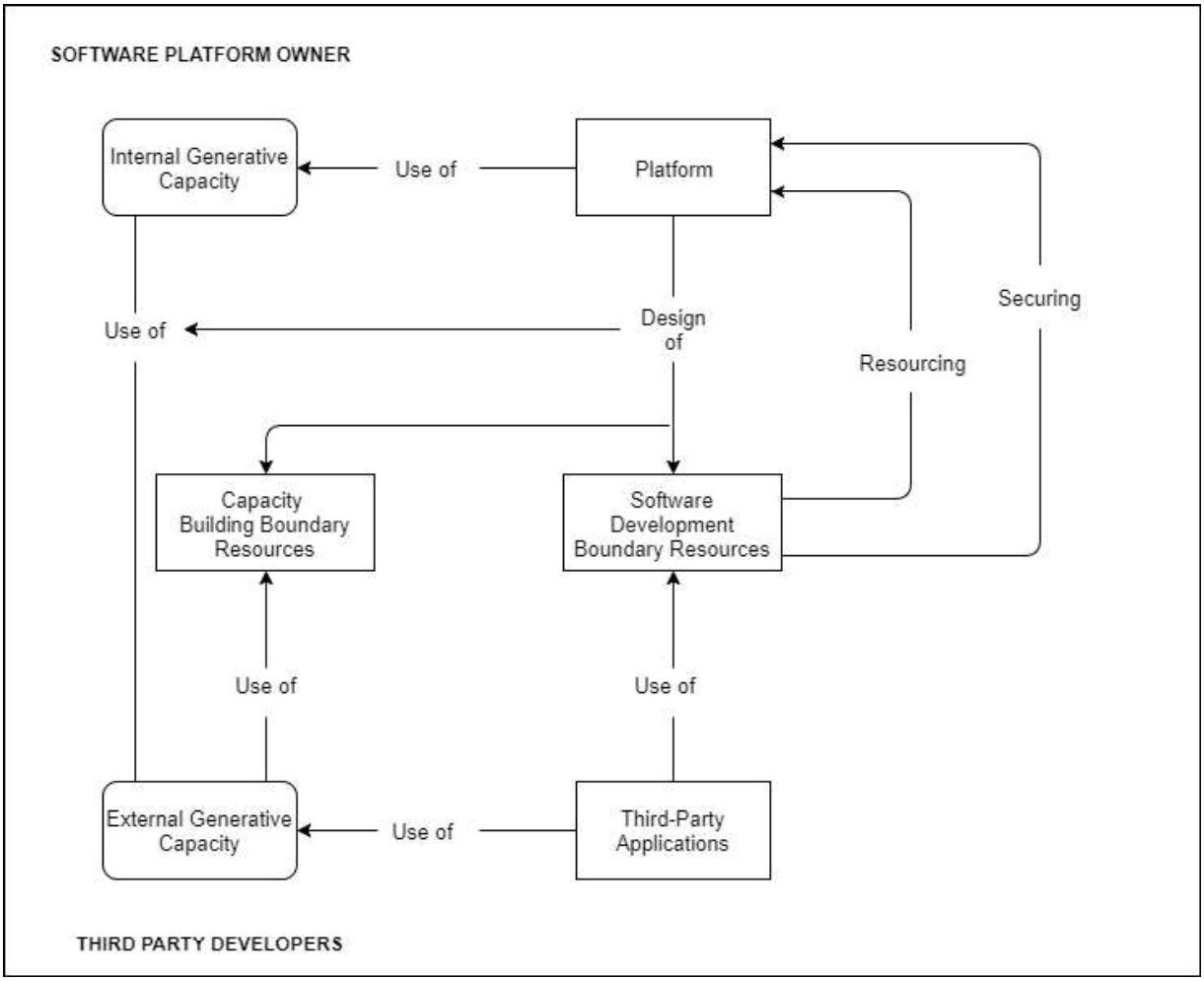


Figure 6: Extended Boundary Resources Model

Constructs

Platform: "The extensible codebase of a software based system that provides core functionality shared by modules that interoperate with it and the interfaces through which they interoperate" (Tiwana et al. 2010, p. 676)

Software Development Boundary Resources: boundary objects in form of software artefacts that are used to develop third party applications. Such boundary resources include SDKs, API, seed applications and libraries.

Capacity Building Boundary Resources: boundary objects, boundary interactions and brokers that are used to build the third-party development capacity of external developers. Such boundary resources include, amongst other things, capacity building events such as application development academies and workshops.

Third Party Applications: Executable pieces of software that are offered as applications, services, or systems to end users of the platform.

Internal Generative Capacity: platform owners' software development and related competencies that enable them design, develop and maintain the software platform and platform owner driven boundary resources

External Generative Capacity: third party developers' software development and related competences that enable them to design, develop and maintain third party applications and community-driven boundary resources

Generative Capacity Use: the use of internal generative capacity by platform owners to design, develop and maintain the software platform and platform owner driven boundary resources, and the use of external generative capacity by third party developers to develop third party applications and community-driven resources

Boundary Resources Use: the use of software development boundary resources by third party developers to build third party applications and the use of capacity building boundary resources by platform owners and other stakeholders to build external generative capacity.

Resourcing: The process by which the scope and diversity of a platform is enhanced.

Securing: The process by which the control of a platform and its related services is increased.

CONCLUSION

The boundary resources model aims to provide an intellectual account for cultivating third party development in platform-centric software ecosystems. Empirical data from renowned platform centric software ecosystems such as Google's Android and Apple's iOS renders credence to the argument advanced by the model that boundary resources are critical to third party development. In this study, empirical data from the DHIS2 software ecosystem also supports this argument. However, further observations reveal that boundary resources are not in themselves an endgame in cultivating third party development. Analyzing the data with respect to constructs from generativity literature reveals that external generative capacity has a role to play in cultivating third party development.

In addition, analyzing the data with respect to constructs from community boundaries literature reveals that boundary resources are more than technical artefacts – they are socio-technical artefacts comprising of not only boundary objects but also boundary interactions and brokers. Extending our understanding of boundary resources to include boundary interactions and brokers, allow a distinction between software development and capacity building boundary resources both of which are required to cultivate third party development. The paper makes further distinction between exo-platform boundary resources and endo-platform boundary resources, and between platform-owner driven boundary resources and community driven boundary resources.

Based on these analytical results, the paper has proposed an extended boundary resources model presented in the discussion above. In the extended boundary resources model the paper introduces the constructs internal generative capacity, external generative capacity, generative capacity use and extends the boundary resources use construct to include the use of capacity building boundary resources by platform owners and other stakeholders in the ecosystem to build external generative capacity. In addition, constructs software development boundary resources and capacity building boundary resources replace the boundary resources construct. Adding these constructs helps foreground other factors critical for third party development besides provision of software development boundary resources.

In doing this, it is worth noting that every research has its limitations. Further knowledge can be obtained by augmenting this research with further studies. In the future, for example, research could look at the interplay between factors that influence third party development as put across in the extended boundary resources model and developer motivation factors. There are a lot of accounts on developer motivation factors in the open source literature which can provide a good starting point(Aknouche and Shoan, 2013; Fogel, 2005; Hars and Ou, 2002). Notwithstanding the importance of such interplay, for the sake of brevity and simplicity, such an analytical account is beyond the scope of this paper.

REFERENCES

Aknouche, L., Shoan, G., 2013. Motivations for Open Source Project Entrance and Continued Participation.

Avital, M., Te'eni, D., 2009. From generative fit to generative capacity: exploring an emerging dimension of information systems design and task performance. Information Systems Journal 19, 345–367.

Bergman, M., Lyytinen, K., Mark, G., 2007. Boundary objects in design: An ecological view of design artifacts. Journal of the Association for Information Systems 8, 546.

Bosch, J., 2009. From software product lines to software ecosystems, in: Proceedings of the 13th International Software Product Line Conference. Carnegie Mellon University, pp. 111–119.

Bosch, J., Bosch-Sijtsema, P., 2010. From integration to composition: On the impact of software product lines, global development and ecosystems. Journal of Systems and Software, SI: Top Scholars 83, 67–76. https://doi.org/10.1016/j.jss.2009.06.051

Carlile, P.R., 2002. A Pragmatic View of Knowledge and Boundaries: Boundary Objects in New Product Development. Organization Science 13, 442–455.

Creswell, J.W., 2014. Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. SAGE.

Eck, A., Uebernickel, F., Brenner, W., 2015. The Generative Capacity of Digital Artifacts: A Mapping of the Field.

Fogel, K., 2005. Producing Open Source Software, 1st ed. O'Reilly.

Ghazawneh, A., Henfridsson, O., 2013. Balancing platform control and external contribution in third-party development: the boundary resources model. Information Systems Journal 23, 173–192. https://doi.org/10.1111/j.1365-2575.2012.00406.x

Ghazawneh, A., Henfridsson, O., 2010. Governing third-party development through platform boundary resources, in: The International Conference on Information Systems (ICIS). AIS Electronic Library (AISeL), pp. 1–18.

Grisot, M., Hanseth, O., Thorseng, A., 2014. Innovation Of, In, On Infrastructures: Articulating the Role of Architecture in Information Infrastructure Evolution. Journal of the Association for Information Systems 15.

Hars, A., Ou, S., 2002. Working for Free? Motivations for Participating in Open-Source Projects. International Journal of Electronic Commerce 6, 25–39.

Henfridsson, O., Lindgren, R., 2010. User involvement in developing mobile and temporarily interconnected systems. Information Systems Journal 20, 119–135.

Kimaro, H.C., 2006. Strategies for Developing Human Resource Capacity to Support Sustainability of ICT Based Health Information Systems: A Case Study from Tanzania. The Electronic Journal of Information Systems in Developing Countries 26.

Kimaro, H.C., Nhampossa, J.L., 2005. Analyzing the problem of unsustainable health information systems in less-developed economies: Case studies from Tanzania and Mozambique. Information Technology for Development 11, 273–298. https://doi.org/10.1002/itdj.20016

Manikas, K., Hansen, K.M., 2013. Software ecosystems – A systematic literature review. Journal of Systems and Software 86, 1294–1306.

Mutula, S.M., Van Brakel, P., 2007. ICT skills readiness for the emerging global digital economy among small businesses in developing countries: Case study of Botswana. Library Hi Tech 25, 231–245. https://doi.org/10.1108/07378830710754992

Polak, M., 2015. Platformisation of an Open Source Software Product: Growing up to be a generative software platform. University of Oslo, Oslo, Norway.

Prügl, R., Schreier, M., 2006. Learning from leading-edge customers at The Sims: opening up the innovation process using toolkits. R&D Management 36, 237–250.

Star, S.L., Griesemer, J.R., 1989. Institutional Ecology, "Translations" and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. Social Studies of Science 19, 387–420. https://doi.org/10.2307/285080

Tiwana, A., 2013. Platform Ecosystems: Aligning Architecture, Governance, and Strategy, 1 edition. ed. Morgan Kaufmann, Amsterdam; Waltham, MA.

Tiwana, A., Konsynski, B., Bush, A.A., 2010. Research Commentary—Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics. Information Systems Research 21, 675–687.

von Hippel, E., Katz, R., 2002. Shifting Innovation to Users via Toolkits. Management Science 48, 821–833. https://doi.org/10.1287/mnsc.48.7.821.2817

Wenger, E., 2011. Communities of practice: A brief introduction.

Wenger, E., 2000. Communities of practice and social learning systems. Organization 7, 225–246.

Yin, R.K., 2013. Case Study Research: Design and Methods, 5 edition. ed. SAGE Publications, Inc, Los Angeles.

Zainal, Z., 2007. Case study as a research method. Jurnal Kemanusiaan 5.

Zittrain, J., 2008. The future of the Internet and how to stop it. Yale University Press, New Haven, [Conn.].

Zittrain, J., 2006. The Generative Internet. Harvard Law Review 119, 1974–2040.