The Electronic Journal of Information Systems in Developing Countries
www.ejisdc.org

COLLABORATIVE DEVELOPMENT OF GLOBAL INFORMATION SYSTEMS: TOWARD COMMUNITY BASED GENERIFICATION

Elisabeth Fruijtier
University of Oslo
Norway
fruijtie@tcd.ie

Matthieu Pinard
University of Oslo
Norway
matthiep@ifi.uio.no

ABSTRACT
This paper examines generification in open source software development of information systems for low resource environments. The challenge addressed is that of designing generification processes in which key aspects of generification – information flow, selection and prioritisation – involve a distributed community. The objective of this study is to understand how this challenge is dealt with in practice, with the aim of expanding the current analytical scope available to researchers to study generification processes as well as the guidance available to practitioners to address its practical challenges. Doing so, we build on the work of Gizaw et al (forthcoming) who broach a debate on more ‘open’ generification. We propose a reconceptualization of open generification by being more specific on its conditions and process aspects. Open, we argue, is about closed generification circuits that enable a ‘community based’ generification process.

Key words: Generification, Open generification, Open Source Community, Global Information Systems

1. INTRODUCTION
In collaborative software adaptation, the process of making IT artefacts suitable to requirements or conditions for its use is the act of (groups of) individuals who are required to correlate social and technical processes in the interest of designing functional information systems (Scheil 2008). Open source software (OSS) communities are apparent examples of such collaborative efforts given the high degree of cognitive independence among actors and effective coordination of knowledge and actions required for successful decision making within these environments (Yoo & Kanawattanachai, 2001 p.188). However, a shared interest in partaking in design does not imply unanimity. The collaborative development of complex systems is a dynamic process underpinned by actors’ different ideologies, that influence the development and scalability of the system (Constantinides & Barrett 2014).

In the negotiation of common interests, a global network of stakeholders involved in the act of collaborative IT adaptation faces two challenges inherent to (two opposing requirements of) system development. These are 1) the development of a system for use across an organisation requires the negotiation of shared interests (focussing on harmony) and 2) the development for use in particular parts of that organisation requires the negotiation of solutions that fit best to a local context (focussing on availability of alternatives). The software development and adaptation process thus involves a compromise between harmonic requirements and local alternative solutions.

The process of compromise in software development whereby a diversity of ‘local’ requirements of different user groups of a product are channelled, filtered and prioritised to inform ‘global’ design features that work across a diverse range of organizational contexts is also referred to as ‘generification’ (Pollock et al. 2007). Generification could thus be seen as the compromise reached as an outcome of negotiating between local and global requirements.

We know that in firms, the generification process of selecting and prioritising user needs tends to be shaped by management (ibid.). In comparison, open source development
design processes are perceived to be much more participatory. Nevertheless, given the aim is to cover universal aspects of local requirements (Camara & Fonseca 2007), this approach is equally bound to addressing a selective range of user needs. Here, the word ‘community’ in describing open source communities becomes problematic due to its implication of a sense of unity (Gherardi 2009) when in fact, this unity requires the constant negotiation of conflicting interests in order to develop a generic product. Especially when an open source community involves many different stakeholders, interests may differ greatly. For instance, there are also dictator-led communities in which decisions on direction, focus and contributions are made by visible leaders (Bacon, 2009 p.222).

This creates a paradox in OSS communities that are concerned with the development of a generic product, as more ‘bottom up’ generification processes need to channel the needs of a wider community. The challenge is how to deviate from issues associated with a top-down process of generification whereby information from different user groups is valued differently by managers (Pollock et al. 2007). The objective of this study is to understand how this challenge is dealt with in practice, with the aim of expanding the current analytical scope available to researchers to study generification processes as well as the guidance available to practitioners to address its practical challenges. Doing so, we build on the work of Gizaw et al (forthcoming) who broach a debate on more ‘open’ generification. In steering the research toward achieving its objective, the following research question was formulated:

What mechanisms facilitate generification processes by a globally distributed open source community?

This paper proceeds by outlining a discussion of current conceptualisations of generification in networked practices in section 2. A description of our methodological approach in section 3 is followed by a description and analyses of the empirical material gathered in relation to the collaborative adaptation of a FOSS called DHIS2 in sections 4 and 5. In section 6, we examine how a process perspective can be used to contribute to a networked understanding of generification. We conclude our paper with concrete suggestions for researchers who wish to take this debate further.

2. LITERATURE REVIEW

In the previous section it has become clear that the generification process is concerned with the trade-off between particularisation and generification that requires eliciting general requirements from the particular requirements of a few users; assessing how particular these requirements from the few users are and whether diversity should be built into the system or if functions meeting these particular few should be customised locally (Johannessen & Ellingsen 2009). As the word ‘trade-off’ implies, this is not only a logistical but also a political process.

Generification processes within collaborative development processes as part of OSS communities have been subjected to a limited amount of research. As noted by Lanzara & Morner (2003 p.1), ‘open-source software development poses a theoretical challenge to conventional ways of conceptualizing knowledge processes within and across organizations’. The authors (ibid.) have attempted to bridge aspects of this gap by developing theory on how virtual communities of open source developers use tools for for distributed collaboration. However, this analysis limits itself to an information transfer approach to knowledge sharing and does not consider the selection process of this information central to generification.

A recent study that specifically focuses on the generification process of an open source community was conducted by Gizaw et al (forthcoming) who, in an attempt to challenge the top-down nature of generification processes, introduce the concept of ‘open
generification’ as a more democratic alternative. Open generification acknowledges the need for and the feasibility of generic software and in addition proposes an alternative model for the governance of it. They define this form of generification as follows (ibid. p 1):

‘Open generification is not about managing the community of users attached to a software package by homogenization or segmentation, but aims at addressing the diverse needs of the community the software is expected to serve.’

Open generification aims to challenge the limitations of what we could perhaps consider to be more ‘closed’ generification, characterised as top-down (dominated by the software vendor) and ‘somewhat manipulative’ as it treats user groups differently (i.e. not according to their needs, but according to their customer value). Their analysis complements our knowledge of generification in a valuable way, by emphasising the importance of perceiving the development of generic systems from the user’s point of view. Following a process of embedding, local developers can create local alternatives that suit local user needs. This ‘embedding’ then informs a process of ‘disembedding’, through which management decides which local alternatives need to be made generic (integrated in the ‘global’ product).

One of the important strategies in this process emphasised by Gizaw et al. (ibid.), is to decrease the level of dependency between actors by facilitating the development of local alternatives. Emphasis is on the FOSS characteristics of the product itself, which enables a way for global developers and local developers to negotiate features through a process of embedding and disembedding. The FOSS characteristics allow for innovation outside the realm of a central actor which can be fed back into the generic software and therefore transferred to other settings. We have conceptualised the (open) generification process in figure 1 as follows:

**Figure 1. Conceptualisation of the Generification Process**

Figure 1 illustrates a design cycle that starts with a (global) ‘designer who articulates a design, which is then tested in use. This in turn generates feedback for the global designer following a generifaction process. In Open Generification, local designers improve the design (creating a ‘branch’) to meet user expectations and communicate this back to the
global designer instead. What makes this process ‘open’ is the porous membrane provided by accessibility to the core code for external developers through which local ‘alternatives’ can be built and made generic. Open generification in this sense involves a shift in attention, from managing the requests of several types of users to managing (‘disembedding’) the requirements of a community of local developers instead. Since disembedding is part of conventional generification, it is unclear to what extent this process is not prone to the same limitations of being top-down in nature and motivated by the added value of user representatives.

We are inspired by this work, which we perceive to be the groundwork for our study. However, we also identify some aspects of the open generification process that require further exploration. Unlike Lanzara & Morner (2003), it remains unclear to us what mechanisms and tools are used to orchestrate the open generification process. How is ‘embedding’ and ‘disembedding’ enabled and disabled? We foresee such insights will generate a more in-depth understanding of what gives generification its ‘open’ nature, other than the characteristics of the software product or the type of community involved compared to more traditional generification processes.

In our analysis, our conceptualisation of the open generification process differs from the conceptualisation offered by Gizaw et al (forthcoming) in one important way. In the example used by Gizaw et al, the local designers access the core code to create a ‘branch’ (also referred to as forking of the core code). A fork occurs when developers take a copy of the source code of the software and start independent development on it, creating a distinct and separate piece of software. This becomes problematic when a fork created to address a local problem may not be brought back into the source code of the systems’ generic core. As discussed by Manda et al (2014), the disadvantage of developing such software ‘forks’ is that the user eventually can no longer benefit from later versions of the system and the advantage of consulting the wider community, which is especially critical with a high release pace. However, in the case discussed in the study by Manda et al. (ibid.), these disadvantages compete with the situation of the user that may require urgent adaptation, which is why generification processes that adequately channel user needs are crucial.

Accordingly, we will conceptualise the ‘open’ nature of this process not to depend on designers branching and negotiating the branch to be part of the ‘trunk’, which as seen in the case example provided by Gizaw et al is a difficult process. Instead, we depart from the assumption that local designers do not touch the trunk and instead propose suggestions to it in the form of ‘blueprints’ which can be seen as proposed pieces of code that can be pieced into the source code. This changes the conceptualisation of open generification in important ways because it reflects on the traditional ‘generification’ aspect inherent to ‘open generification’ rather than the alternatives it offers to generic design. This implies that in figure 1, the ‘design’ box on the left concerns a blueprint (or an application on the software platform that does not depend on integration) and not a ‘branch’ of the source code.

Interestingly enough, descriptions by Johnson et al. (2013) of the very same empirical setting that inspired the conceptualisation of the ‘traditional’ generification process (Pollock et al. 2007) has overlap with conceptualisations of ‘open’ generification by Gizaw et al (forthcoming). A commonality between these works is that a group of ‘power users’ (Volkoff et al 2004) or local designers in the case by Gizaw et al (forthcoming) play a central role in communicating needs of the user community. In the case of Johnson et al (2013), we furthermore see these mediators are able to increase their influence in the generification process over time. Johnson et al. (ibid.) conceptualise these mediators as ‘prosumers’; collective fora which – though providing the space for independent action and innovation by users – are managed by the company who enables (influential) relations as well as disabled them. However, Johnson et al. (ibid.) note how this nevertheless requires collaboration with
the user community. Their work highlights how consumer-company interactions are complex evolutionary duals in which users attain skills to promote their idea’s to designers, and designers attempt to remain control over their user communities. Having access to the same empirical case which inspired the ‘open generification’ approach, this research will explore the grey area that emerged between these cases in search for specific aspects of the embedding and disembedding process that make generification processes more open.

3. **Methodology**

This study is a naturalistic enquiry of participatory processes within the networked development of a free and open source software called DHIS2. A naturalistic inquiry is founded on the primary belief that phenomena should be studied in context (Frey, L., Botan, C., Kreps 1999 p 258). In line with this approach, both authors have been longitudinally involved in the activities of the HISP UIO node within the network over a period of respectively 1.5 (primary author) and 3 years (secondary author). During his involvement, the second author has experienced working closely with the development team within HISP UIO as a DHIS2 Academy and Community Coordinator. This insider perspective has been an invaluable contribution to understanding design processes and practices related to DHIS2.

3.1 **Data Collection**

Longitudinal exposure has enabled the authors to gain a real life understanding of processes within and outside the HISP UIO development node. During this time we made observations; participated in many presentations; conference calls with external experts and had informal field consultations with key informants. Both authors have furthermore participated in so called ‘DHIS2 academies’ which enabled access to various actors at different levels of the network and exposure to interactions between them. Observations during these events where documented in the form of descriptive field notes and included a one-day field visit to a local NGO. In addition, we had access to the tools used by the community for collaborative development; email lists with interactions between users, local developers and core team developers; various documentation (ie. strategic document, evaluation report) and a large volume of publications (including MSc and PhD theses) to deepen our understanding of development processes surrounding DHIS2.

The evidence presented in support of this particular case was derived primarily from 4 sources. First of all a series of recorded semi-structured interviews conducted with 10 experts, key informants and users of the software (table 1). Interviews lasted approximately 1 hour and were all recorded with exception of one interview which was documented manually. Secondly, a focus group was held with 5 HISP nodes and experts hosted by the second author in his role of DHIS2 Academy and Community Coordinator. Third, a community assignment was designed by the authors to derive input from a group of approximately 70 participants of a 6 day DHIS2 expert academy which hosted a selective group of primarily DHIS2 experts and developers from all over the world.

For the community assignment, the community present during this expert academy was divided into different user groups and assigned the task to first gather and then select methods for the community to address some of these challenges identified. Groups were formed based on their common knowledge (ie. (I)NGO’s, consultants, HISP nodes, core team, researchers etc). A shared folder in Google Drive was used as a medium for both the authors as well as the participants to access the achievements of the various groups. The different group assignments were then collated and results were themed (ie. roadmap, documentation, translations, bug action, training, apps etc). New groups were formed based on random selection and asked to come up with practical suggestions in relation to one of the themes.
Table 1. Interviews

<table>
<thead>
<tr>
<th>No.</th>
<th>Participant affiliation</th>
<th>Topics covered:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>HISP node (country A)</td>
<td>- Role and responsibilities</td>
</tr>
<tr>
<td>2.</td>
<td>HISP node (country B) / Ministry of Health</td>
<td>- Enrolment into community</td>
</tr>
<tr>
<td>3.</td>
<td>HISP node (country C)</td>
<td>- Adoption &amp; adaptation of DHIS2</td>
</tr>
<tr>
<td>4.</td>
<td>HISP node (country D) /HISP consultant</td>
<td>- What does being a HISP node entail (when relevant)</td>
</tr>
<tr>
<td>5.</td>
<td>HISP node (country D)</td>
<td>- Types of users reached and their capacity/skills</td>
</tr>
<tr>
<td>6.</td>
<td>HISP node (country E)</td>
<td>- Collaboration with other nodes and core team</td>
</tr>
<tr>
<td>7.</td>
<td>NGO (country E/ user)</td>
<td>- Attendance of- and experiences during academies</td>
</tr>
<tr>
<td>8.</td>
<td>Ministry (country E/ user)</td>
<td>(when relevant)</td>
</tr>
<tr>
<td>9.</td>
<td>Consultant (donor community)</td>
<td>- Tools and fora available for collaborative development</td>
</tr>
<tr>
<td>10.</td>
<td>Consultant (HISP UIO)</td>
<td>(when relevant)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Provision of user trainings (when relevant)</td>
</tr>
</tbody>
</table>

3.2 Data Analysis

Interviews, the focus group discussion and field notes taken during the expert academy were transcribed, coded and themed. The outcomes of the first part of the community assignment were summarised under three categories (table 3 left column), and compared with practical solutions from the second part of the assignment (table 3 right column). This categorisation was not an outcome of the exercise, but emerged from the analysis. This resulted in the overview provided in table 3. Processes are highlighted in italic when they were already ongoing during time comments were raised, processes in bold are emphasised by the community involved in the exercise as having priority.

After condensing the raw data we focused on eliciting relationships between the various stages of the development process and the types of actors involved. This stage of the analyses involved identifying and mapping out the various layers of the distributed community of actors involved in the generification process. To construct our analysis, we distinguish across two layers of the adaptation process; the different type of adaptation and the different actors involved. We follow the terminology of Grisot et al. (2014) and Star & Ruhleder (1996) to zoom into four types of adaptation: adaptation ‘of, in and on’ the system. An extra dimension, adaptation ‘to’, was added to describe those development processes which attempt to enable information systems to connect to other information systems and therefore relate to its placement in a wider ‘ecosystem’ of systems. This was illustrated in figure 2.

In addition, the various actors involved in collaborative software adaptation process may be perceived to broadly represent meta-designers; collaborators; contributors and consumers as categorised by Fischer (2009). This was illustrated in figure 3. The scope of this paper focuses but is not limited to those levels of participation which essentially describe the generification ‘agents’ in globally distributed software design. These agents are located at ‘contributor’ and ‘collaborator’ levels which mediate between the invisible users as consumers and developers as meta-designers. It is therefore at these levels (Gizaw et al (forthcoming) refer to these levels as ‘local developers’) that requests and needs can be elucidated to enable a generification process.

We then adopted a process perspective in order to gain a deeper understanding of the dynamics between these actors, their involvement and the stages of the development process and the mechanisms at their disposal which enabled this involvement. This led to a second series of more detailed, holistic displays (figure 4) that captured the phenomena that was observed. Finally, an abstraction of these more detailed displays led to the creation of a conceptual framework (figure 5) that enabled us to enter a dialogue with the literature.
4. CASE DESCRIPTION

DHIS2 is an open and free source software tool (FOSS) to collect and analyse health data which is accessed at local, regional (district) and national level for monitoring and evaluation purposes. The development of the DHIS2 is the result of a collaborative effort between the University of Oslo and a wider network of partners united under the Health Information Systems Program (HISP) to improve health systems in low resource settings. DHIS2 is primarily used by Ministries of Health (MoH) and large international NGO’s. Today, the HISP initiative is comprised of a global network of actors from over 60 countries.

In the development of DHIS2, a FOSS approach is complemented with a platform strategy (Manda et al. 2014; Braa & Nielsen 2015). The FOSS approach encourages a wider community of users to contribute to the IT artefact which serves as a generic solution for all. Flexibility of the software product enables the user to highly customise the software’s interface. A platform approach enables those users who require customised requirements that cannot be integrated to the core to extend the software to develop their own applications with the opportunity to share them with the wider user community (Braa & Nielsen 2015).

4.1 Adaptation ‘of, in, on and to’ DHIS2

The various elements of the platform approach, which will be discussed in more detail in this section, are meant to address one of the major challenges of implementing a FOSS approach in the given global context which have previously been discussed in relation to ‘forking’. Enabling others to contribute to the features of the platform or extend it with their own applications is meant to solve part of this problem.

Adaptation ‘of’ the system refers to the development and improvement of DHIS2. This process is primarily driven and controlled by HISP UIO with new versions of the system being released quarterly, fuelled by input from the wider community of local developers and partners. The other forms of adaptation (adaptation on, in and to the system) are to a high extend dependent on adaptation processes of the core. Over the years, a network of HISP nodes has gradually emerged and is increasingly being strategically formed for the purpose of decentralising these aspects of the development process (Braa & Nielsen 2015).

Customisation of the interface of the system, discussed in this paper as adaptation in the system, involves making the system fit for local use. This requires all kinds of actions related to setting up the system such as creating organisation units; indicators; data elements, dashboards; authorisation of user-groups; pivot tables; designing tracker programs and forms and defining access rights and authorisations. Setting up and maintaining the system may be done by local or regional HISP nodes, or may be organised by the users themselves.

DHIS2 academies have become central efforts in engaging the DHIS2 community in strengthening the capacity of national and regional users to successfully set up, design and maintain DHIS2 instances (‘in’ adaptation). These intense training programs on use and adaptation of DHIS2 target users both at beginners as well as more experienced levels. A training database (‘trainingland’), which is based on a fictional country, enables participants of academies to explore the functionalities of the software. ‘Regional’ academies are to be distinguished from ‘expert’ academies which are hosted once a year. Expert academies have a more conference type of set up in comparison to the other academies, with less emphasis on training and more on collaboration and ingenuity.

The ability to extend DHIS2 instances with applications enables local developers to solve challenges they face in the adaptation process through the development of extensions and applications on the system. This is different from apps that belong to the system’s core platform which is formed by a collection of ‘core’ apps. Apps that are externally developed are predominantly designed locally for individual use (customisation). Nevertheless, a small
selection of apps that is applicable for more ‘global’ use is made available in a DHIS2 ‘app store’ where they can be shared with other users. Adaptation ‘to’ refers to the coordination of interoperability of DHIS2 in relation to various other systems as part of a larger information system infrastructure (‘architecture’). Such efforts are necessary to make data available to all relevant users across systems composing the broader architecture. This will require not only technical gateways but also harmonization of definitions and dimensions in order to combine the data in meaningful ways. Of, in, on and to dimensions of the adaptation process are illustrated in figure 2.

![Figure 2. Dimensions of System Adaptation](image)

**4.2 A Community of Generification Agents**

In the absence of formal procedures, the concept of a ‘node’ in the HISP network is subject to different interpretations. Braa and Nielsen (2015) discuss how the expansion of the nodes in the HISP network has become more diverse. Development nodes (outside the HISP UIO there are currently 3) focus on the development of the core system. Together with the core team members at UIO, they would categorise as ‘meta designers’ in figure 3. Implementation nodes comprise of (teams of) developers primarily concerned with designing for local or regional requirements and can be seen as a focal point for DHIS2 support in a certain country or region. Some nodes comprise of both (core) developers and implementers. Nodes may also provide (non-technical) public health or health management related support.

Within figure 3, these nodes would often have the role of ‘collaborators’. The term ‘contributors’ is used to capture those users with knowledge ‘in house’ (or outsourced) to customise the software or with the capacity to hire their own developers. These users are therefore capable to actively interact with the design and negotiate their needs with either the core team or local HISP nodes. This is in contrast to users who lack this interactive capability and would categorise as ‘consumers’. The capacity of contributors and collaborators may vary. Within the developing country context in which DHIS2 is implemented, international non-governmental organisations (INGOs) automatically categorise as power users (Volkoff et al. 2004) given their vast capacity to hire their own developers to customise the system. Other contributors may be local NGO’s or Ministries (sometimes with the support of (I)NGO’s).

In the following sections we will look more closely at how these generification agents (are enabled to) represent user groups. Doing so, we will adopt a more critical lens in attempting to understand how this networked generification approach, though arising from a ‘bottom up’ tradition, also has its issues that require attention, especially as interventions in these environments (and this one in particular) mature.
5 FINDINGS
The previous section provides an overview of the types of actors involved in various dimensions of the collaborative development process of DHIS2 and their level of involvement. We will build on these insights to identify the mechanisms at the disposal of these actors at different development stages which enabled this involvement, in order to reconstruct the infrastructure through which generification is enabled.

Figure 4 maps out the various routes a request can travel once the original generification cycle between designer and user is broken and extended. We can broadly identify 3 ‘adaption’ circuits, where different types of adaptation blend with various actors in the role of generification agents through a variety of mechanisms. A potential direct link whereby design features come to the attention of designers through use was illustrated with a dashed line since this relationship did occur in the data but was not further explored in line with the study’s scope. In this section we will elaborate in more detail on the present mechanisms that enable generification as part of the adaptation process.
5.1 Mediators

The term ‘generification agents’ used earlier is illustrative of the role that nodes and consultants, as well as individuals that would categorise as contributors, play in mediating between local design and generic design that happens at core level. As seen in the following example, these actors become focal points and representatives for users who experience certain needs while using the software in their daily practice.

‘For us [local node], we have been in charge of overseeing all messages and feedback communication within DHIS2 [referring to messaging functionality](…)That user feedback is not automatically exported to Oslo [HISP UIO], but we can present the same thing as an experience or as a requirement in a different forum [where it gets communicated to HISPUIO].’

HISP node representative 1 country D

This example furthermore illustrates how these agents become the first step in a generification process through which these needs are channelled onwards. In their mediating role, they are required to negotiate between what is required in practice and what is technically possible or desirable, as seen in the following excerpt:

‘We usually call it ‘asking for juice’ like… Being in a meeting or in a training and someone goes: ok, so this can be done, why doesn’t it do this? So usually we call it like someone wants a juice in a cup and you want the DHIS2 to pour you juice.. Because some functionalities can be very complex.. (…) You sort of mediate, if you think this is something that is nice to have and it can be done as a developer and then you can present it to the higher level and maybe help writing a blueprint for it’

HISP node representative 1 country D
The availability and accessibility of mediators to end users (consumers) varies, as is their mediating capacity. For instance, (local) NGO’s or Ministries may have different experiences in the ability to 1) identify end user needs (for instance during local trainings) and then 2) channelling them up to either regional nodes; directly to the development team, or -where possible- cater for them themselves (through for example app development or customisation of the configurable layer). In addition, the level of support provided from a HISP node to these users may vary depending on the capacity of these nodes and the capacity of users to attract support. To better understand this mediating capacity, we need to have a look at the mechanisms in place for actors to act as generification agents.

5.1.1 Collaborative Platforms
Most of the process of suggesting or implementing adaptations to the generic core takes place on two collaborative platforms, ‘Launchpad’ or ‘Github’. These platforms are primarily used by collaborators and to some extend by contributors to communicate with meta-designers and other developers to report bugs and suggest and monitor the implementation of new features. One collaborator describes:

‘If I have a requirement then I won’t send an email [referring to the user lists] but I can create a formal blueprint within the DHIS2 Launchpad account (…) If you write a blueprint it gives some sort of discussion forum for people to discuss (…) set priorities and those kind of things.(…) If it gets prioritised] then I can see: Ok, this is scheduled for [version] 2.9 or for the coming two releases..’

HISP node representative country A

From this excerpt it becomes clear that being able to write a blueprint increases the ability of developers to convey their requests. It also is illustrative for the way local developers act as mediators in channelling user needs. This is important given the way most users without a technical background will not understand the ‘language’ spoken on Launchpad, as explained by the following developer:

Launchpad is mostly a developers thing (…) it is quite easy to use. Most people know you can use it for registrating bugs, but this collaborative development; writing blueprints, suggesting for additional features.. These kind of things - most people [other than the nodes/ consultants] either they don’t know they apply or they don’t know how to use it..’

HISP node representative country A

However, the platform is not used as a ticketing system and decision-making processes are not always transparent, as noted by the following participant of an expert academy:

It is impossible to understand how bug reports are addressed [by the core team], some take eight years and some are addressed right away (…) The same with blueprints: there has to be a rough assessment’

HISP node representative country C

In addition, an analysis of the activities on Launchpad shows that bug reports or new blueprints are not always allocated to a specific developer in the network and as a result a certain amount of ‘ghost’ blueprints and bugs ‘hang’ in the system somewhat indefinitely.

5.1.2 User-list
There are several email lists, of which the ‘user list’ and ‘developers list’ function as communication channels between contributors, collaborators and meta-designers. These lists
enable directly seeking assistance from other community members or developers in addressing challenges. Developers lists are mostly used by HISP developers (n= 462) whereas the user lists is used within and to some extend across community layers by both nodes and a diversity of technical users (n= 970). These lists are places where meta-designers articulate new designs. The other way around, these lists are media for local developers (be it contributors or collaborators) to comment on design choices. This is illustrated in the following excerpt from the user list posted by a meta-designer:

DHIS version [2.xx] is out with a lot of great new features and improvements... [28 improvements listed with screenshots and link to demo plus further information on documentation]

To which one of the collaborators replies:

WELL DONE - some highly useful new features, which hopefully also will resolve long-standing bugs/issues (e.g. like problems we've had with approvals)

Occasionally, the user list will prove an effective medium to request specific feedback from the community on future plans to make changes to features ‘of’ the system (in relation to the system’s ‘roadmap’). In some other cases, such plans surface in discussions about certain features. For instance, an email thread on the user list that starts as a ‘common’ type of question requesting assistance regarding the creation of an ‘aggregation query builder’, received the following response from a meta-designer:

In fact, aggregation query builder is already scheduled to be removed in version 2.23 and will be replaced by program indicators. Admittedly there are a few cases where persisted aggregate event data could be useful, however we think that generating the aggregate values "on the fly" will provide more flexibility and be less of a load on the server by avoiding large, nightly batch jobs.

The response of this meta-designer came as a 10th reaction in this thread and led to among others the following response by a collaborator:

If not further work on the Aggregation query builder feature, please don't remove it. We actually find it very useful and is critical to our workflow. For example, we need to get Total patients seen in a facility. Now with queries the user can see total figures, plus when needed segregate them by age, gender, visit type (new, follow up etc.) and many more categories in Pivot tables which provides an easy interface to suffice all this.

In these examples, the email lists serve as a forum for designers and collaborators/contributors to articulate design choices and requirements and provide context to speak to the imagination of the other actor. In this case, the proposed change to the system would disturb work processes. However, these dialogues occur spontaneously in relation to specific requests for support, and are therefore unlikely to reach the awareness of most users of the list given that many support issues are raised on the list each day and searching the archive is not a straightforward process.

5.1.3 Regional and Expert Academies
The DHIS2 academies have an important role in enabling communication between members of the core development team and collaborators and contributors as well as among them. Dialogue is enabled by the ability to discuss and become familiar with various use-cases.

---

1 Data retrieved on 25.08.2015
through which different users can identify shared practices. In addition, it is enabled in the form of feedback sessions and opportunities during an academy. During a regional academy, this session was guided by a Word-document that was put up on the screen while participants were asked to provide their feedback to the developers in the form of requirements for changes or new features. When participants proposed a requirement, the facilitator would take notes in the displayed document visibly to all participants. This furthermore enabled participants to make corrections as well as made them aware their feedback was being documented and therefore taken into consideration. However, during an expert academy, the use of this document created confusion and to some extent had an opposite effect, when this method made visible how during a feedback session the presenter only started taking notes half-way into the session, after a number of issues had already been raised without being given the same treatment. Consequently, participants started joking about how, when a requirement does not end up on the list, it will not be followed up on. Also, the following comment illustrates how a sense of being heard does not result from merely taking notice, as one participant commented:

‘When we do have a chance to give feedback we need to know if what you write down on your list will be implemented.’

Participant expert academy (Donor community representative)

A similar observation was made during a regional academy, where a developer from a neighboring country expressed his frustration during the feedback session held because issues he flagged now he had already raised at a previous academy. This can create tensions between users and mediators, who may at times feel quite powerless in their position as the ‘messenger’ when they are both unable to address especially pressing issues with a local solution or see their requirement integrated in the core.

Compared to a regional academy, the expert academy hosts many more members of the core team which makes the process of giving feedback more efficient. Not only does this make them more accessible for feedback as a result of their physical presence, it also enables better communication (articulation) about the implementation of features, plans and prioritisation in general - which to some extent also provided input for discussions with the audience. These expert academies specifically target a selective group of collaborators.

DHIS2 academies serve an important value as physical meeting spaces for enabling participants to not only enter a dialogue with designers but also to formulate or voice collective concerns. During an expert academy, a session which walked members of the community through the latest developments surrounding DHIS2 (also referred to as the roadmap) illustrated how the interests of the present community may vary greatly. One announcement that sparked a diversity of reactions concerned efforts to develop GIS features in DHIS2 in such a way that data could be used to predict outbreaks of a certain disease and pro-actively inform decision-making. This pro-activity was a significant change compared to the way design had thus far always focussed on enabling re-active use of (health) data. For some members of the community, this raised questions about the stretching of resources in relation to the unlikely functionality of such a feature within the work environment they experienced, where ‘re-active’ use of (health) data is already problematic and broadband issues do not allow for ‘heavy’ features of this kind. During this discussion, another member of the present community however expressed a different take on these developments, by saying:

‘Most of us are professionals, we have a lot of ideas and most of the work you do is the simple stuff. It’s acceptable to do some fun and interesting ‘fancy’ stuff, if only for the developers who do it to recharge their batteries.’
In this example a negotiation of interests takes place. Interests such as the allocation of resources in relation to functionality of the system are compared to a different interest, namely the allocation of time and resources which is considered to be a reasonable trade-off for an increase in work motivation.

Despite the advantages of these academies for enabling generification (as a by-product of the training itself), they are limited in size and occurrence and therefore limitedly accessible. Currently, there can be approximately 12 regional academies per year, whereas the expert academy and the NGO academy only happen once a year. Plus, although the participation fee is as low as possible, they furthermore pose a barrier and often mean certain users at lower levels (such as districts) are excluded from participation.

### 5.1.4 Informal Channels

As seen in previous sections, the user list is not a primary mechanism for collaborators or contributors to articulate their needs. In addition, the use of tools such as Launchpad offers limited transparency about the development process or prioritisation of requests. Nevertheless, as illustrated by the following quote, an important aspect of the generification process happens informally:

> `Whoever [meta-designer] you are exposed to gives you an access so that you can even... exchange ideas before requests. (…) I might have a problem – let’s say a client says he wants this (...) if I think I don’t have the complete solution I can personally ask let’s say [name lead developer], what is your suggestion? What can you... what is your input? He can give me his input and then I can work.’

**HISP node representative 2 country D**

This participant furthermore points out a need for capacity in order to be able to benefit from the platform approach whereby adaptation ‘on’ and ‘to’ may solve limitations experienced with the system (‘of’):

> `You need to be open - that is you have a problem, it can be implemented or if not: what can you do at the local level because sometimes the problems could be more specific to your environment and not ‘global’ so... you also need to have that ehm... intellectual [capacity] to see how you can resolve that problem while you are waiting on ‘the global’ to resolve it…’

**HISP node representative 2 country D**

Here, personal relations from a collaborator with a meta-designer give access to negotiate requests or support a mediator’s capacity in solving problems through other forms of adaptation.

Some participants noted that the focus of the development roadmap seemed to correlate with big projects valued highly by the core team. When there was a lot of focus from the core team on the implementation in a certain country or a certain project, a lot of requirements for this project would be fed into the roadmap. One participant shared the perception that how to get an issue onto the roadmap has to do with the ability of collaborators and contributors to ‘flag attention’, stating ‘obviously, you get attention with money’.

### 5.2 Community Assignment

The previous findings are complemented with outcomes of a community assignment, in which the community present at an expert academy voiced where and how they would see
room for improvements in the open generification process. The left column summarises items raised whereas the column on the right zooms in on specific process aspects.

<table>
<thead>
<tr>
<th><strong>Table 3. Community Assignment</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Assignment 1:</strong></td>
</tr>
<tr>
<td><strong>Summary of community needs raised</strong></td>
</tr>
<tr>
<td><strong>1. Informing community about community procedures:</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
| | | **1.1.4 split expert academies into technical and non-technical**
| | | **1.1.5 establish clearing house (app store) for developed apps**
| | | **1.1.6 send out a list of academy attendees to all the participants**
| | **1.2 Where to find information** | **1.2.1 create structured list of profiles of community experts** |
| | | **1.2.2 develop Wikipedia type of dictionary for terms used in DHIS2**
| | | **1.2.3 archiving requests better and create most frequently asked questions from user lists**
| | | **1.2.4 videos for specific use cases or scenarios**
| | | **1.2.5 establish QA/QC for developed apps**
| **2. Enabling community engagement** | **2.1 Development of the roadmap/apps** | **2.1.1 distribute custodianship of components of the roadmap**
| | | **2.1.2 develop collaborative development and user needs forum to develop apps (online, mailing list and face to face)**
| | | **2.1.3 develop collaborative mechanisms (such as Launchpad, Github or Google groups) to synergize development and prioritize user needs**
| | | **2.1.4 slow down release schedule (...) Need for implementers to plan. Feature-driven releases (as opposed to timed)**
| | **2.2 Transparency/prioritisation (roadmap)** | **2.2.1 introduce voting process on blueprints and provide comments to the blueprints**
| | | **2.2.2 need for more intentional collaboration with survey based tools for aggregating survey data.**
| | | **2.2.4 communication of the features (roadmap) with community (open for discussion)**
| | **2.3 Introduce steering committee/member development** | **2.3.1 identify coordinator for managing community experts list**
| | | **2.3.2 create a location for organisations to upload and submit documentation for review by community members**
| | | **2.3.3 create special interest groups that take responsibility for translations in particular languages and develop review process**
| | | **2.3.4 designate a bug manager – rotate responsibility between HISP nodes**
| | | **2.3.5 designate release manager**
| | **2.4 Inclusion of developers from within and outside the ‘HISP’ community to contribute to the development of DHIS 2 and related processes** | **2.4.1 hire experts locally for translations**
| | | **2.4.2 build development capacity in nodes to fix bugs**
| | | **2.4.3 accreditation of trainers and organisations**
| | | **2.4.4 nodes to do level 1 academies**
| | | **2.4.5 create a reward mechanism for experts creating training videos for specific tasks or scenarios**
| **3. Interaction between community members** | **3.1 Sharing best practices** | **3.1.1 create community profiles which include documentation that highlights organisations, use cases and contact information**
| | | **3.1.2 encourage community to share the problems solved and experiences in stack exchange or user list**
| | | **3.1.3 Wikipedia for DHIS2 use-cases**
| | | **3.1.4 create a sharing space for training resources**
| | | **3.1.5 communities need to have something in common (...) need to create or make visible that the community is a community of communities.**

The Electronic Journal of Information Systems in Developing Countries

www.ejisdc.org
In table 3, 11 out of 33 suggestions (highlighted in grey) concern improvements in the generification process at the level where requests are channeled from contributors and collaborators back to the meta-designer. These suggestions have to do with making the process to file requests (bug reports and feature requirements) easier as well as more transparent. Some suggestions indicate the community wants more influence in negotiations regarding their integration (‘of’). Also, the suggestion of collaborative mechanisms to prioritize user needs (2.1.3) indicates that current mechanisms are inadequate (or inadequately used) for such purpose.

6. DISCUSSION

Generification is a process of an evolutionary and cyclic nature, which emerges from the bouncing back and forth between designer idea’s interests and practical implementations with those of local stakeholders. Central to this process is how users or their representatives are engaged and valued in the process. Accordingly, this requires mechanisms for engagement as well as transparency. Based on the study’s findings, we can understand this process as a dialogue between gain and loss that occurs at each information-processing step taking place across three interrelated circuits. Open generification, in our perspective, attempts democratic forms of engagement and validation of knowledge regarding what needs to be included in the core of design (‘of’ and ‘in’), while enabling those aspects that fail to be included to be enabled at its periphery (‘on’ and ‘to’). Through the development of a conceptual framework of the generification circuit (presented in figure 5) this paper contributes to an understanding of how these distinctions are made.

Moving away from perceiving generification processes to be either ‘open’ or ‘closed’, this paper proposes to look at them as circular. This way, one could envision them to be both open and closed at the same time as part of a balancing act that may furthermore vary as time passes. What such an understanding does not reveal is how different agents are valued, which especially in a developing country context is an important concern. What it can do however, is increase transparency where it is lacking in order to improve the mechanisms in place for a more equal and/or transparent validation process. Mapping out generification circuits is thus about making the commonly ‘invisible’ practices and mechanisms of processes of embedding and disembedding in generification, visible, and therefore may be used to combat power inequalities in line with an open generification ideology. In the DHIS2 case, such an understanding reveals that at the level of these software founders (meta-designers), mechanisms in place are lacking in offering such transparency which would be able to shed light on the democratic nature of the generification process. We argue this is necessary for determining how ‘open’ they truly are.

Accordingly, this paper expands on the conceptualization of ‘open generification’ as a cyclic and multi-dimensional approach. Open, we argue first of all, is about closed loops (no matter how short or long they are). This implies that, should these circuits be interrupted at any point, requests cease to be conveyed and the generification process is ended. In conceptualising this process, we are inspired by the conceptualisation of ‘circulating references’ used by Latour (1999) to designate among others the quality of understanding chains through which information is transferred and transformed in their entirety. We follow his description of such chains in our understanding of generification circuits in the sense that ‘truth value circulates here like electricity through a wire, as long as the circuit is not interrupted’ (Latour 1999 p. 69). This was illustrated in figure 5. Similar to the concept of circulating references, generification circuits can help reveal what mechanisms enable an organisation to determine what is lost and what is gained as information is transferred as part of the generification process.
As seen in figure 5, adaptation ‘of’ the product is a mandatory passage for ‘on’ and ‘to’ forms of adaptation to become available for customisation ‘in’ the product. This also makes visible how the negotiation process this requires core to traditional generification processes remains present in so-called open generification (Gizaw et al, forthcoming).

In addition, our data suggests closed loops to be both about mediators and mechanisms. We have referred to these mediators as generification agents; contributors and collaborators, described elsewhere in the literature as ‘power users’ (Volkoff et al. 2004) or ‘prosumers’ (Johnson et al. 2013) who, based on their capacity, represent user needs and speak the language needed for development in transferring these needs to meta-designers (Johannessen & Ellingsen 2009). As seen in the paper by Gizaw et al (forthcoming), the opening of generification processes requires a transition between (passively ‘organised’) user groups into (active and empowered) ‘prosumers’, ‘local developers’ or what we call generification agents. Data indicates making this transition calls for mechanisms that further enable actors to climb the ‘generification ladder’.

In FOSS communities, participation in the design process tends to be earned with expertise, time, passion and commitment. Open generification in FOSS is no exception from this practice (Gizaw et al forthcoming). However at the same time, Gizaw (ibid.) note that openness is constrained especially in developing countries where skills, infrastructure, funding and political commitment may be scarce. As noted by Gizaw et al. (ibid.), the generification process between collaborators and the meta-designers is prone to power discrepancies in favour of meta-designers and power users that have access to stable funding resources. However, it needs to be taken in to account that the capacity of meta-designers is limited that disembedding processes can be practically challenging (ibid.).

Accordingly, once mediators have acquired a position in the generification process their influence can vary (Pollock et al. 2007; Johnson et al. 2013). In comparison to the definition of ‘prosumers’ by Johnson et al (2013) as part of a top-down generification process, the DHIS2 case tells us that in open generification, local developers are also part of a collective fora; provided with the space for independent action and innovation by in this collaborators; in a collaborative relationship with ‘managers’; nevertheless ‘managed’ by a
coordinating entity (meta-designers) enabling (influential) relations as well as disables them for various reasons. What matters then, is to understand (and discuss) the influence of various generification agents on the generification process. For this, we need to draw the attention to the mechanisms made available to them. Specifically, the capacities of mechanisms in place to 1) draw in generification agents and 2) close the generification circuit between them and meta-designers.

7. RECOMMENDATIONS AND FUTURE RESEARCH

According to a free and open source approach, software development is opened up to a wider community in an attempt to employ its rich knowledge base. In using this approach to make a generic software product, how input from that community is retrieved defines how ‘open’ this ‘sourcing’ process is. In networked FOSS development, this requires putting in place mechanisms that benefit transparency, accountability and equality in design processes of a wider community of mediators and users. Such applications of mechanisms would need to fit into an overall ‘generification’ strategy. Tools and mechanisms supporting an open generification strategy need to be initiated by meta-designers who have the power, knowledge and legitimacy to shape the generic product.

In realizing such a strategy, future research could benefit from investigating the ‘epistemic stance’ of organisations or networks, which has been defined by Fayard et al (forthcoming p 9.) as follows:

‘a collectively enacted attitude toward the pursuit of knowledge, which reflects beliefs about reality and the “true” sources of novelty, and is expressed through certain modes of pursuing and evaluating knowledge. This concept enables us to unpack how an organization, for which the pursuit and creation of knowledge is crucial, responds to and enacts possibilities offered by new IT-enabled practices.’

Understanding the epistemic stance of an organisation could enable it to critically reflect on the IT tools and processes in place for enabling generification. This could help explain how knowledge flows of different agents are valued and why bottlenecks in generification circuits occur. However, in order to give organisations analytical tools to do so, the concept of epistemic stance requires further development.

In addition, and to some extend in relation to this, it is important to note that the collaborative development efforts by open source communities encompass two processes of a very different nature: collaboration and adaptation. These processes in turn place very different requirements on the generification process in terms of the desired level of convergence and divergence. In the DHIS2 case, an unequal representation of these mechanisms that focus more on technical aspects of development potentially limits the capacity of the core team and the wider network to absorb and process information provided by/or hidden in the community.

In this light, outcomes of the community assignment as well as its process furthermore indicate that the open generification process itself (and its mechanisms), not merely its generic products, are a fertile topic for negotiation by the community involved. The community assignment discussed in section 5.2 serves as an example of a generification exercise initiated by the authors, in which different stakeholders were grouped and asked to select and prioritise aspects of the generification cycle they would like to see improved. Outcomes of the community assignment furthermore suggest an interesting alternative research angle would be to look at mechanisms at other stages of the innovation circuit, for instance mechanisms surrounding the communication of usage and capacity building of mediators.
8. REFERENCES


