

## SCALING UP LOCAL LEARNING - EXPERIENCES FROM SOUTH-SOUTH-NORTH NETWORKS OF SHARED SOFTWARE DEVELOPMENT

Jørn Braa (jbraa@ifi.uio.no)  
Dept. of Informatics, University of Oslo, Norway

Eric Monteiro (Eric.Monteiro@idi.ntnu.no)  
Dept. of Computing and Information Sciences, NTNU, Norway

Sundeep Sahay (sundeeps@ifi.uio.no)  
Dept. of Informatics, University of Oslo, Norway

Knut Staring (knutst@ifi.uio.no)  
Dept. of Informatics, University of Oslo, Norway

Ola Hodne Titlestad (olati@ifi.uio.no)  
Dept. of Informatics, University of Oslo, Norway

### **Abstract:**

There is solid empirical and analytical evidence for the importance of localising ICT in the context of use. Through hands-on experiences, the inevitable source of local processes of learning, improvisation and tinkering, an information system is grounded in situated work routines. Building on this position, but ultimately moving beyond it, we discuss the challenge of balancing two competing demands. On the one hand, allowing the required local learning as indicated above, while, on the other hand, accommodating trans-local (or "global") knowledge flows to take place. These trans-local knowledge flows are required since reiterating the full process of local learning at each and every local site is prohibitively resource-consuming; it does not scale.

Scaling of local knowledge is explored using the case of open source software development for health care within the HISP network. From a start developing information systems to support the new decentralised health structures in post-apartheid South Africa, HISP now has more than ten years experience from building South-South-North networks for shared learning in this area. Attempting to do the 'same' in each of numerous sites in a number of countries in Africa and Asia, the network needs to balance contradictory demands for local learning (thus local flexibility) with enjoying economies of scale through harvesting from investments done at other sites.

**Keywords:** Capacity building, local vs. global, standards, improvisation.

# SCALING UP LOCAL LEARNING - EXPERIENCES FROM SOUTH-SOUTH-NORTH NETWORKS OF SHARED SOFTWARE DEVELOPMENT

## 1. INTRODUCTION

The aim of this article is to explore how information systems development may be incorporated in global collaborative networks that enable sharing of both costs and knowledge, while still remaining sufficiently flexible to be developed further locally. Sharing of resources is one of the great promises of Free and Open Source Software (FOSS) approaches to software development – but it also puts demands on the local capacity in all parts of the network, ranging from software development to adapting the use context – capacity to *both* meet local needs and contribute to global development at the same time. This is a challenging combination, and there is a need to investigate the interconnections in more detail. A particular challenge is to balance the efforts going into developing the global core application (i.e. the global standard) with the efforts going into local customisation and add-ons. The more generic the global solution, the more it can be shared globally and reused, but also, the more resources will be required and the more complex the development – and vice versa. Furthermore, the more shared learning and “best practices” are incorporated in the “business logic” of the core software, the more it becomes a vehicle for shared learning – and, again, vice versa. We address these research aims through examining various processes around the design, development, and implementation of FOSS solutions in such a network.

The Health Information Systems Project (HISP) is an extensive South-South-North (SSN) network with a focus on information systems for public health care. Initiated by universities in Cape Town and Oslo, it is currently present in a number of African countries (South Africa, Tanzania (including Zanzibar), Ethiopia, Malawi, Botswana, Mozambique, Namibia, Zambia, and Nigeria), as well as in India and Vietnam. For analytical purposes, we distinguish between networks within a country (which we call “local” for the purposes of this paper), and across countries (which we call “SSN”). The local networks are comprised of various actors in the health administration (community, sub-district, district, provincial, and national), universities, NGOs, and funding providers. The global SSN network has over the last decade been engaged in developing and supporting two sets of processes. One, supporting local learning and sustainability around the application and use of computer based Health Information Systems (HIS) to address local public health challenges. Two, strengthening the overall SSN network to enable further sharing and learning across the various countries in the network. In this paper, we explain how these processes are mutually reinforcing, and that they cannot operate effectively without one another.

Given these research aims and backdrop, the rest of the paper is organized as follows. The next section discusses relevant literature, and presents key analytical concepts. Section 3 presents the research methods, followed by a description of the case itself in section 4. After presenting the case analysis in section 5, the final section summarises discussions and implications.

## 2. LITERATURE REVIEW

### Sustainability and local improvisation

There are a number of reports on full or partial failure of information systems (IS) in developing countries (Heeks et al 1999). Lack of sustainability, defined as processes that are self-sufficient

(Reynolds and Stinson 1993), is often cited (e.g. Kimaro and Nhampossa 2005). However, the requirements of modern IS often mean that developing countries cannot be expected to be fully self-sufficient in this regard. As suggested by Korpela et al (1998), sustainability would imply that the user organisation is able to manage risks that threaten the long term viability of the IS. In the context of health information systems (HIS), long term viability will require sufficient flexibility for the IS to adapt to and encompass ever-changing needs for information caused by new and emerging health challenges, such as the HIV/AIDS pandemic (WHO 2000).

Inappropriate design as related to the needs and context of use represent a major reason for IS failures in developing countries. Design and development need to be grounded in the context of use, and the capability for local improvisation in IS development play a key role in order to close this “design–reality gap” which needs to be enabled along the following dimensions (Heeks 2002):

- Technology; the IS applications need to be reality-supporting and enabling with high levels of “plasticity” and “shallow inscriptions” (Akrich 1992)
- Nature of design; local design improvisation will rely upon modularity and increments
- Local capacity; local improvisation will have to be carried out by local people understanding context , organisation and work processes, as well as the role of IS.
- Improvisation-supporting IS development approaches, such as participative approaches.

The HISP project in South Africa is widely regarded as a success, mainly because of local improvisation applied to all the dimensions above; the enabling character of the District Health Information Software (DHIS) application, the incremental and modular design and participatory approaches applied, together with the capacity of the local team covering the range of aspects of both health and software (Braa and Hedberg 2002). The aim of this article is to explore how achievements such as these may be shared across countries and developed further locally in collaborative networks.

### **Standardisation – negotiating the local-global dichotomy**

The difficult issue (analytic as well as practical) is where to draw the line between how much to keep constant (standard) across the different sites and what to open to localization. Dominant approaches to standardisation portray it as a top-down imposed ‘iron grid’ which subjects merely have to comply with (Schmidt and Werle 1998). Recent work on generic software packages have examined how to bridge the heterogeneity between different organizations and cultures, concluding that they are "brought into being through an intricately managed process, involving the broader extension of a particularized software application and, at the same time, the management of the user community attached to that solution" (Pollock et. al. 2007). We wish here to focus on the perspective of the developers and implementers. FOSS methodologies have a potential to empower local implementors to customize applications for "the very particular needs that often arise in different settings, and allows, through use, the natural evolution of information technologies and systems within unique and specific contexts" (Weber 2003).

The District Health Information System (DHIS) application software lies at the core of the HISP project. We explore the negotiations around striking a balance between the need for standards to adapt DHIS to various local contexts, while simultaneously coping with complexity by leaning towards universal solutions. Globally flexible solutions are often costly and intricate both to program and to configure as compared to “hard-coding” of bespoke

solutions closely tied to the present local situations. Local capacity, ranging from software to the context of use, needs to be developed through practical implementation efforts in each country. Sharing software raises the issue of balancing the set of standards being imposed through the software with the need for local flexibility and room for improvisation. Unless the common core is significant, the software becomes prone to fragmentation into mutually incompatible versions (a phenomenon known as "forking" in the context of FOSS), and ceases to be an effective *boundary object* allowing distributed communities of practitioners to collaborate and learn across contexts and borders (Pawlowski et al 2000). The case study explores how distributed and "shared" development of the software between different countries and contexts of use may help identifying areas of the software application that need to be open and flexible and what areas can be "fixed".

The way universal solutions, predominantly from the developed countries, need, but notoriously fail, to be negotiated against the needs of developing countries illustrates the problem (Braa, Monteiro and Reinert 1995; Ryckeghem 1996; Sahay 1998; Sahay and Walsham 1997). In essence, this amounts to exploring the tensions arising from two different strands of reasoning. The former, closely aligned with readily recognizable concerns for curbing complexity, reducing risk, and maintaining control, is the argument that the only viable way to establish a global information infrastructure is to adhere to uniform, standardized solutions (Weill and Broadbent, 1998). The latter, by now well iterated and largely internalized, argues for the necessity of adapting information systems to local, situated and contextual work settings (Ciborra 1994; Kyng and Mathiassen 1997; Suchman 1987).

In particular, scholars have pointed out how this needs to be conceptualized as negotiation processes involving elements of 'work arounds' (Gasser 1986), 'drift' (Ciborra 1996a), 'improvisation' (Orlikowski 1996), and 'situated actions' (Suchman 1987). Collectively, such surprising outcomes of technologies-in-use are often lumped together under the heading of 'unintended consequences' (Barley 1986, Robey and Boudreau 1999). The extent of improvisation, drift or unintended outcomes is of course strongly linked to the notion of intended action in the first place: if all deviation from the intended trajectory is 'drift', this will necessary be a significant category.

### 3. METHODOLOGY

This article draws on experiences from a continuous software development process in the HISP network over the last decade (Braa and Hedberg 2002, Braa et al. 2004). The authors of this paper have been engaged in different ways in various processes around in software design, implementation, political brokering, and obtaining funding, in both the SSN and the different local efforts.

The HISP initiative draws upon the so called Scandinavian action research tradition in IS development where user participation, evolutionary approaches, and prototyping are emphasized (Greenbaum and Kyng 1991, Baskerville 1999). This methodology has been further developed as part of HISP into a "networks of action" approach, emphasizing the need to go beyond learning in singular locations to the sharing of experiences and knowledge between various nodes in a network (Braa et al. 2004).

While one of the authors has been actively involved to different degrees in the process right from the start in 1994, the others have been active participants in subsequent phases, including the customization and adaptation of the DHIS software in various countries. Two of the authors are currently coordinating the global development of DHIS version 2. In addition, we build on results obtained by Masters and Doctoral students who have contributed to specific project activities such as software customization or capacity development in an action research mode

(e.g. Lewis 2005, Nordal 2006, Øverland 2006).

## 4. CASE STUDY

### Centralized development and local improvisations of DHIS version 1

The first releases of the DHIS (referred to collectively as “version 1”) were developed through numerous prototyping cycles over a decade, with extensive initial trials in pilot health districts. It was subsequently scaled up to two provinces, and then to the whole of South Africa. A major reason for its portability and eventual success came down to the system’s flexibility with regard to the definition of what data items to collect, as well as a transparent and open database management system which meant that local managers and implementers could easily adapt it to the needs of any particular province *without* the need for programmers to be involved (Braa and Hedberg 2002). The ease of creating locally defined data sets combined with an emphasis on minimal national datasets to be reported upwards in the health system hierarchy. Another success factor was the strong involvement of public health professionals both in the design and use aspects. The technical team was composed of a main programmer actively engaged in the health domain together with a small private company, all working closely with public health specialists.

From 2000 onward, version 1 was adapted and applied in a number of countries in Africa and Asia. While data structures could be adapted to the context of other countries using the flexible GUI, language and special reporting requirements could not. For the first prototype in Mozambique the Portuguese translation was hard coded in the user interface. This did not work well, since it represented a “fork” of the DHIS, and the frequent new releases from South Africa could not be used in Mozambique. This sparked the development of an independent language module, where strings of text could be translated to – in principle - any language. This was the first example where specifications developed in other countries was fed back to South Africa, and contributed to south-south collaboration.

The Mozambique activities were tightly coupled with the development group in South Africa. In contrast, the team in Ethiopia worked relatively independently with one specific DHIS release, and developed a module based on local requirements for handling data based on ICDs (International Classification of Diseases), by adding both tables code to the DHIS core. This approach was effective locally, but again created a “fork”. The code was not integrated with the core distribution from South Africa, and Ethiopia was stuck with one specific DHIS release, not able to enjoy the improved features of subsequent releases.

The use of Microsoft Office, (specifically MS Access) as the platform for development meant that the database structure was transparently available, and extensions could readily be made. The Indian team took advantage of this fact to employ local capacity to develop a range of add-on functionalities related to reporting, presentation, and use of the data collected in the system (Nhampossa and Sahay 2005). A GIS extension was also developed, and later translated into Portuguese and adapted for the Mozambique context in collaboration with local developers there (Lewis 2005). This was the first example of south-south collaboration without the involvement of the core development node (South Africa).

### Global software development and local adaptation of DHIS version 2

In October 2003, a report distributed by the Ministry of Health in Mozambique critiqued version 1 as follows: “DHIS technology is outdated by a decade”. It was recommend it to migrate to SQL server, tiers should be split, and it should be web-enabled. The content of the critique was in many ways already acknowledged by the HISP group, but the report provided

the impetus to initiate development of a Version 2 on a completely new platform. As the South African team was still hard at work with a final release on the previous platform, the development of version 2 became the responsibility of the Department of Informatics, University of Oslo. As no funding was available to hire professional developers, this was undertaken mainly as a research project involving PhD and master students, most of whom were relatively inexperienced with large scale software development. Also, no public health experts or local end users were involved.

The final major overhaul of version 1, which was still under development, was used as the functional design for version 2. A full stack of open source Java frameworks was eventually selected as the development platform, providing both flexibility and “industrial strength” capabilities and scalability. However, the combination of several such advanced frameworks proved complicated to master, and limitations in accessing the Internet in partner countries presented an additional deterrent to learning, as most of the documentation was available in dynamic forms online, as is typical of evolving FOSS frameworks.

Version 2 development was conceptualized under a model in which core development (of the database for example) would take place in Oslo along with a small number of Southern partners. In Vietnam, agreements on collaboration was signed between Oslo and a technical university, a software company, and the health authorities in two provinces in the South. However, results were not as expected, since the health services had no experience of applying IT in health, and the HISP technical team (IT students from Norway and Vietnam) knew nothing about health.

It was the development in India from late 2005 that led to the finalisation of a workable prototype of version 2. The Kerala government’s informal (but often strongly articulated) policy to support FOSS served as a catalyst for the development processes. Establishing a stable local team proved challenging, given the great demand from the private sector for skilled Java developers. Through “learning by doing”, the Indian developers learnt to deal with many local problems, requesting assistance from the SSN through the internet when stuck. Help was usually readily forthcoming (primarily from Oslo, but increasingly also from Vietnam). This ongoing development of capacity in the team has contributed to the scaling of version 2 to two more states.

Similarly, in Vietnam, the breakthrough came in June 2006, when version 2 was finally implemented in several locations in Ho Chi Minh City (HCMC). Rolling out the software on a large scale raised the need for both a customised reporting solution, a multi-language module with support for Vietnamese translations, as well as technical support to the health offices. These demands created the conditions for closer interaction between the local technical team and health staff, and with it the potential for mutual learning and increased technical capacity.

### **Flexibility and local improvisation**

The flexibility of the new global platform (where the core has been mainly produced in Oslo) is pronounced on several levels, and goes well beyond the ability for users to create new data collection items without the intervention of a programmer. The very modular web interface framework employed by the project allows for easy incorporation of various modules. For example, both the Indian and Vietnam teams have created reporting modules able to replicate the sometimes fiendishly complex paper reporting formats. These modules integrate nicely with the web interface and show up as new menu items. The modular web portal allows the local teams to assemble selected modules into a running application. For example, to promote management control over the user management and the health hierarchy, the Indian team selected to build a slimmed down version for data entry clerks, which removed certain modules.

In the case of Ethiopia, there was a strong requirement to collect more fine grained information about each disease. Although not going all the way to a full patient data system, the health authorities wanted to add International Classification of Diseases (ICD) codes, and to break down the statistics collected by sex and age groups. Again, the modularity of the system allowed for increased detail at the data entry level in a separate input module, which could easily run alongside the global system, though the data were partially stored separately.

Yet another example of local variations in application of the general software stems from India, where the use of 16 official languages and almost as many different alphabets mean a strong emphasis on multi-language support. In general, Java is strong in this field, and this is also true for the Indic scripts. Thus internationalization of the user interface became. However, data clerks in clinics will often prefer all text to be entered and displayed in the local language, whereas managers at various levels often prefer to operate in English. Also, data will have to be reported in English to the federal level. This means that not only the user interface, but also all the names of districts and diseases, which constitute the data in the system, need to be available in multiple languages. Also, when expanding to several new states, the project soon faced the fact that IT and public health experts from one state often meet a language barrier when working in another state. Setting up the database and working with minimal datasets soon becomes unwieldy when working with foreign languages. Lacking a full blown internationalization of all aspects of the database, the Indian team made use of “shortnames” for English equivalents of the local Hindi or Gujarati script versions of diseases and clinics to enable the use of two languages in parallel. While the eventually successful generic solution providing full support for all elements in the database was being worked on in Vietnam and Oslo dragged on because of the intricacies involved, the local team found workable local solutions which fulfilled the needs of the users and decisionmakers.

### **The hundred flowers of reporting**

The process of developing a reporting solution for DHIS 2 illustrates some of the local-global tensions in the SSN, and also consequences of design-reality gaps (Heeks et al 1999). Routine and ad-hoc reporting is one of the most context-specific parts of the software, with a wide range of local requirements within the SSN.

In January 2006, a technical team from Norway and Vietnam worked in HCMC on developing a global report solution that would serve both the apparently quite simple report demands of HCMC and the more general requirements of the global SSN. At the same time, there was immense pressure on the HISP India team to roll out DHIS 2 in Kerala state, and the most crucial missing part was the report module. The requirements from Kerala were too complex for the Norwegian/Vietnamese team to incorporate it into the global solution, and the Indian team lacked the necessary “DHIS 2 skills” to develop a generic and integrated solution. The combination of these constraints forced the India team to develop a local “quick fix”, which in many ways was inconsistent with the core architecture and technology choices of DHIS 2. Even though the solution solved a critical problem in Kerala, the Norwegian core developers objected strongly to seeing that their global “best practices” standards for software development had been thoroughly disregarded. At a time when the core developers, lacking understanding of the use context, struggled to develop a generic solution for reports that could be shared across countries, the Indians developed a less general tool that lacked the necessary flexibility to be shared globally in the SSN.

The local solution to the Vietnamese reporting needs were somewhat hampered by the emphasis on also meeting the more complex and generalized global requirements defined by the coordinators in Norway. While the Vietnamese requirements at first seemed quite simple, the implementation process and the subsequent interaction with end users sparked a series of

new requirements and improvements to this module, a “discovery” of requirements quite similar to the complex ones initially elicited in Kerala. Given the increased technical skills of the developers in Vietnam, and more importantly their much improved knowledge gained through extensive user interaction, the local developers were able to handle these additional requirements. However, the coordinators and developers in Norway were frustrated to see that these improvements, of which many could benefit the whole SSN, were placed in a separate local Vietnamese module as a fork to the global solution. After strong pressure from Norway, these local improvements were finally incorporated as part of a global report module.

Based on the continuous Vietnamese improvements to the global report module, the Indian team has also started to use some of the functionality, such as flexible and rapid viewing of simple reports, a feature the more static “quick fix” did not support. This then led to an unfortunate complex situation for the users, where two different report modules were in use at the same time. However, as the Vietnamese module becomes “globalised”, the “quick fix”, can be phased out and fully replaced – a year after being absolutely crucial to getting the system up and running in Kerala and the rest of India. This rather chaotic repository of report solutions also caused a lot of confusion and delay in making reports in Ethiopia, which, though an early pioneer in the use of Version 1, was less well integrated in the SSN when it came to developing Version 2, with the result being yet more local fixes to be able to meet the tough deadlines of the local health authorities.

## 5. ANALYSIS

In the initial development phases in South Africa, local learning came about through a merger between the technology and health knowledge domains. High levels of participation, involvement of public health specialists, and technology applied by people experiencing real problems served as important mechanisms to aid the learning process and avoid large design-reality gaps.

The role of the SSN was initially to create institutional structures of collaboration, which provided the framework for the merger of knowledge. As the system was ported to other settings, experiences from Mozambique, India and Ethiopia gave practical evidence of the limits of the existing architecture, and raised the need to de-concentrate the development from South Africa. However, the health-technology gaps in the other HISP countries remained an issue of concern.

The start of version 2 entailed a conscious effort to create learning within the SSN. Impediments to this learning process came from the high threshold level of knowledge required to master the full stack of Java frameworks, the lack of public health knowledge, and the geographical and cultural distance between Oslo and user sites. However, as we have seen from the development of a report solution in Vietnam, an incremental and participatory implementation process sparked local learning and a bridging of the initially wide design-reality gap. In the HISP India team, with many years of experience implementing DHIS version 1, the initial lack of technical skills to support the new platform and take part the global software development has been reduced as a result of the practical implementation process. This ongoing development of capacity in the team has contributed to the scaling of version 2 to two more states. Increased field level implementation creates a demand-push dynamic, helping to attract monetary resources, which can help to hire more skilled developers and also public health specialists.

Arguably, the most effective learning mechanism was through practical implementation of the DHIS application in the context of the health services, and processes to fit the technology to local needs. The participatory implementation process can be understood as a mutual learning

process bridging the design-reality gap (Heeks et al 1999). In South Africa, the basic functionalities of DHIS were developed over time in close collaboration with users. Similarly, the DHIS 2 development demonstrates that practical implementation involves social dynamics of reciprocal commitments and “push and demand”. Only when the developers in Vietnam were “pushed” out in the field to implement the DHIS 2 did their contribution to the global software development take off - due to the increasing need for reciprocal commitments. Similarly, in India, the commitment to implement in Kerala sparked the finalization of the first DHIS 2 release, which helped create capacity in the HISP team to approach other states. This shows that learning cannot take place through simple diffusion, but requires a process of translation where the final essential piece of knowledge must be developed – innovated – close to the context of use.

### Balancing the support of the global and the local

The DHIS addresses the critical need for computerised health information systems to be flexible in order to adapt to and encompass the ever changing needs for information. The core idea of the DHIS architecture is to provide this flexibility to the various local contexts in the SSN in a standardised way through a complex meta-design. Core elements of a HIS such as data collection sets and organisational structures can be set up generically in each context, and basic functionality to capture and analyse data is provided through standardised tools. The more advanced peripheral functionality such as tailored reporting, custom data capturing, and special analysis tools are kept as external modules that can be loosely-coupled to the standardised core to provide the user with an integrated user interface.

As we have seen from the case of developing a generic report solution, it is difficult to draw a line on where to standardise and where to allow for local improvisation. While this distinction might seem relatively clear from an architectural viewpoint, this case was influenced by many other factors such as lack of human resources with the right skills and extreme time-pressure in Kerala. While in Vietnam and India the need to meet local user requirements had top priority, the coordinators in Norway needed to balance the importance of local problem-solving and improvisation such as the “quick fix” in Kerala, with the more long term goal to build a SSN where software and best practices regarding implementation and use are shared among the nodes. Though the resource-constrained SSN would benefit from such a sharing of software and expertise on one standardised and generic reporting solution, this has proven very difficult to develop. Currently there are four different report solutions in use around the SSN, of which some are not even possible to share across countries. Table 1 serves to summarize the tensions inherent in the local-global dimension.

| Modes of software development in the SSN: | Benefits  | Challenges  |
|---|---|---|
| <b>Local improvisation</b>                | <ul style="list-style-type: none"> <li>System tailored to local context</li> <li>Local flexibility enables learning</li> <li>Rapid response to local demands</li> </ul>   | <ul style="list-style-type: none"> <li>Harder to integrate with the global modules especially in future releases</li> <li>Often more static as it is targeted as solving one specific problem (that might change over time)</li> <li>Limited support from the network</li> </ul>                      |
| <b>Global standardisation</b>             | <ul style="list-style-type: none"> <li>Generic solutions more adaptable to change</li> <li>Easy integration also for future releases</li> <li>Global support and documentation available</li> <li>Likely to be maintained and improved</li> </ul> | <ul style="list-style-type: none"> <li>Solving many problems in one solution is often complex and resource-demanding</li> <li>Standard solutions to context-specific requirements are often difficult to maintain over time as the requirements keep changing and become even more diverse</li> </ul> |

Table 1. Benefits/Challenges with the local and global software development in the SSN.

## 6. CONCLUSION

Developing information systems for public organisations in developing countries is in many ways a thankless task – a bewilderingly complex tangle of local and national requirements combined with often dismal infrastructure, stifling bureaucratic inertia, limited local IS capacity, and severely scarce resources. One approach to these challenges is to collaborate on sharing development resources and experiences in mutually supportive “networks of action”. HISP is an attempt at doing exactly this, in a way that involves countries both in the North and the South, in order to ensure broad contributions on both technology and actual implementation issues, aiming to make the contexts of use and development inextricably interlinked. However, such an effort will not succeed unless there is significant overlap between the solutions deployed in the various nodes in the network, i.e. a global standard. The extreme complexity and cost of supporting a large variety of local solutions would soon deprive the network of a sufficiently significant boundary object around which to collaborate and the various solutions would drift apart. On the other hand, for the global solution to be used locally, it must be fairly generic and flexible *in certain aspects* which are crucial in local contexts. The process of discovering what these aspects are unavoidable entails trying to come closer to the South and develop the application in close contact with the context of actual use.

Implementing more and more generic and flexible solutions in the global software will however require considerable resources and take time, that might be better used on local add-on solutions. In order not to lose momentum locally, or globally, it is important to continuously negotiate an “appropriate” balance between the local and global efforts. If momentum is lost locally, one risks that local learning processes come to an halt, and eventually that less local learning is shared in the global network and incorporated in the software. On the contrary, if one loses momentum in the global network, one risks that the global efforts are disintegrating into independent local projects with limited shared learning.

## REFERENCES

- Akrich, M. (1992), *Shaping Technology/Building Society*, MIT Press
- Barley, S. (1986), Technology as an occasion for structuring, *Adm. Science Quarterly*, vol. 31, pp. 78-108
- Baskerville, Richard L. (1999), “Investigating Information Systems With Action Research”: *Communications of the Association for Information Systems*, Vol. 2, Article 19
- Braa, J., and Hedberg, C. (2002), “The Struggle for District-Based Health Information Systems in South Africa,” *The Information Society* (18:2), pp. 113-127.
- Braa, J., Monteiro, E., and Reinert, E. (1995), Technology transfer vs. technological learning: IT-infrastructure and health care in developing countries. *IT for Development*, 6(1):15 - 23
- Braa, J., Monteiro, E., and Sahay, S. (2004), “Networks of Action: Sustainable Health Information Systems Across Developing Countries”: *MIS Quarterly*, Vol. 28 No. 3, pp. 337-362
- Ciborra, C. (1994), The grassroots of IT and strategy. In *Strategic information systems—A European perspective*, eds. C. Ciborra and T. Jelassi, pp. 3– 24. Chichester: Wiley.
- Ciborra, C. (ed.) (1996), *Groupware and teamwork*, John Wiley

- Gasser, L. (1986), The integration of computing and routine work, *ACM Trans. on office information systems*, vol. 4, pp. 205-225
- Greenbaum, J., and Kyng, M. (Eds.) 1991. *Design at Work*, Lawrence Erlbaum, Mahwah, NJ
- Heeks R, Mundy D, Salazar A. (1999), Why healthcare information systems succeed or fail. Accessed 2006-11-19 from [http://www.sed.manchester.ac.uk/idpm/publications/wp/igov/igov\\_wp09.pdf](http://www.sed.manchester.ac.uk/idpm/publications/wp/igov/igov_wp09.pdf)
- Heeks, R. (2002), "Information Systems and Developing Countries. Failure, success and local improvisations." *Information Society*, vol. 18, p. 101-112
- Kimaro, H. and Nhampossa, J. (2005), "Analysing the problem of unsustainable health information systems in low income countries: case studies from Tanzania and Mozambique." *Journal of Information Technology for Development*
- Korpela, M., Soriyan, H. A., Olufokunbi, K. C., and Mursu, A. (1998), *Blueprint for an African Systems Development Methodology: An Action Research Project in the Health Sector*". In: C. Avgerou (ed.), *Implementation and evaluation of information systems in developing countries*, IFIP WG 9.4:Vienna. pp. 173-285
- Kyng, M. and Mathiassen, L. (eds.) (1997), *Computers and design in context*, MA:MIT Press
- Lewis, J. (2005), *The Design and Development of Geographical Information Systems for the health sector in developing countries: Case studies from Indian and Mozambique*, Unpublished Master thesis, University of Oslo, Norway
- Nhampossa, J.L. and Sahay, S. (2005), "Social Construction of Software Customization: the case of health information systems from Mozambique and India". IFIP WG. 9.4 conference in Abuja ; Nigeria
- Nordal, K. (2006), *The Challenge of Being Open - Building an Open Source Development Network*, Unpublished Master Thesis, Department of Informatics, University of Oslo
- Orlikowski, W. (1996), "Improvising Organizational Transformation Over Time: A Situated Change Perspective", *Information Systems Research* (7:1), pp. 63-92.
- Øverland, L. H. (2006), *Global Software Development and Local Capacity Building: A means for improving Sustainability in Information Systems Implementations*, Unpublished Master Thesis, Department of Informatics, University of Oslo
- Pawlowski, S., et al (2000), Supporting shared information systems: boundary objects, communities, and brokering. *Proceedings of the 21th International Conference on Information Systems*. Brisbane, Australia, pp. 329-338.
- Pollock, N., Williams, R., D'Adderio, L. (2007), *Global Software and its Provenance: Generification Work in the Production of Organizational Software Packages* *Social Studies of Science* 37: 254-280
- Reynolds J. and Stinson W. (1993), *Sustainability Analysis, Primary Health Care Advancement Program*, Aga Kahn Foundation, Bangkok, Thailand.
- Robey, D. and Boudreau, M.-C. (1999), Accounting for the contradictory organizational consequences of information technology: theoretical directions and methodological implications, *Information Systems Research*, 10(2):167-185

- Ryckeghem, D. V. (1996), Computers and culture: Cases from Kenya. In Information technology, development and policy, eds. E. M. Roche, and M. J. Blaine, pp. 153–170. Aldershot: Avebury.
- Sahay, S. (1998), Implementing GIS technology in India: Some issues of time and space. *Accounting, Management, and Information Technology* 8:147–188.
- Sahay, S., and Walsham, G. (1997), Social structures and managerial agency in India. *Organization Studies* 18(3):415–444.
- Schmidt, S. K., and Werle, R. (1998), *Coordinating Technology. Studies in the International Standardization of Telecommunications*, Cambridge MA: The MIT Press.
- Suchman, L. A. (1987), *Plans and situated actions—The problem of human machine communication*. Cambridge: Cambridge University Press.
- Weil, P. and Broadbent, M. (1998), *Leveraging the new infrastructure - how market leaders capitalize on information technology*, Boston: Harvard Business School Press
- WHO (2000), *The World Health Report 2000. Health Systems: Improving performance*, Geneva: World Health Organisation