# PROGRESSIVE OPEN SOURCE: THE CONSTRUCTION OF A DEVELOPMENT PROJECT AT HEWLETT-PACKARD

Catharina Melian

AKADEMISK AVHANDLING

Som för avläggande av ekonomie doktorsexamen
vid Handelshögskolan i Stockholm
framläggs för offentlig granskning
25 maj – 2007 – 13:15
i sal KAW Handelshögskolan, Sveavägen 65

# PROGRESSIVE OPEN SOURCE:
# THE CONSTRUCTION OF A DEVELOPMENT PROJECT AT HEWLETT-PACKARD

Catharina Melian

STOCKHOLM SCHOOL
OF ECONOMICS
HANDELSHÖGSKOLAN I STOCKHOLM

Dissertation for the Degree of Doctor of Philosophy, Ph.D
Stockholm School of Economics 2007


Cover image courtesy of Ralph Bartsch
ISBN 978-91-7258-717-5

Keywords: collaboration, communication, open source, project management, progressive open source, software development

Printed by:
Elanders Gotab, Stockholm 2007

Distributed by:
EFI, The Economic Research Institute
Stockholm School of Economics
Box 6501, SE 113 83 Stockholm, Sweden
www.hhs.se/efi

# Preface

This report is a result of a research project carried out at the Centre for Media and Economic Psychology (P) at the Economic Research Institute at the Stockholm School of Economics.

This volume is submitted as a doctor's thesis at the Stockholm School of Economics. As usual at the Economic Research Institute, the author has been entirely free to conduct and present her research in her own ways as an expression of her own ideas.

The institute is grateful for the financial support provided by Hewlett-Packard, Ericsson, the Swedish Research Council (Vetenskapsrådet) and Stiftelsen Marcus och Amalia Wallenbergs Minnesfond. The present volume would not have been possible without the participation of employees of Hewlett-Packard. The Economic Research Institute wishes to warmly thank all involved for their generosity and openness.

Filip Wijkström

Director of the Economic Research Institute at the Stockholm School of Economics

Guje Sevón

Centre Director of the Centre for Media and Economic Psychology (P) Stockholm School of Economics

STOCKHOLM SCHOOL
OF ECONOMICS
HANDELSHÖGSKOLAN I STOCKHOLM

*EFI, The Economic Research Institute*

## EFI Mission

EFI, the Economic Research Institute at the Stockholm School of Economics, is a scientific institution that works independently of economic, political and sectional interests. It conducts theoretical and empirical research in the management and economic sciences, including selected related disciplines. The Institute encourages and assists in the publication and distribution of its research findings and is also involved in the doctoral education at the Stockholm School of Economics. At EFI, the researchers select their projects based on the need for theoretical or practical development of a research domain, on their methodological interests, and on the generality of a problem.

## Research Organization

The research activities at the Institute are organized into 21 Research Centres. Centre Directors are professors at the Stockholm School of Economics.

| EFI Research Centre | Centre Director |
|---|---|
| Management and Organisation (A) | Sven-Erik Sjöstrand |
| Centre for Entrepreneurship and Business Creation (E) | Carin Holmquist |
| Public Management (F) | Nils Brunsson |
| Information Management (I) | Mats Lundeberg |
| Centre for People and Organization (PMO) | Andreas Werr (acting) |
| Centre for Innovation and Operations Management (T) | Christer Karlsson |
| Centre for Media and Economic Psychology (P) | Guje Sevón |
| Centre for Consumer Marketing (CCM) | Magnus Söderlund |
| Centre for Information and Communication Research (CIC) | Per Andersson (acting) |
| Marketing, Distribution and Industrial Dynamics (D) | Björn Axelsson |
| Centre for Strategy and Competitiveness (CSC) | Örjan Sölvell |
| Centre for Business and Economic History (BEH) | Håkan Lindgren |
| Accounting and Managerial Finance (B) | Johnny Lind |
| Centre for Financial Analysis and Managerial Economics in Accounting (BFAC) | Kenth Skogsvik |
| Finance (FI) | Clas Bergström |
| Centre for Health Economics (CHE) | Bengt Jönsson |
| International Economics and Geography (IEG) | Mats Lundahl |
| Economics (S) | Paul Segerstrom |
| Economic Statistics (ES) | Anders Westlund |
| Law (RV) | Johnny Herre |

Chair of the Board: Professor Carin Holmquist
Director: Associate Professor Filip Wijkström

## Address

EFI, Box 6501, SE-113 83 Stockholm, Sweden • Website: www.hhs.se/efi/
Telephone: +46(0)8-736 90 00 • Fax: +46(0)8-31 62 70 • E-mail efi@hhs.se

# ACKNOWLEDGEMENTS

The work on this thesis has been undertaken at the Centre for Media and Economic Psychology at Stockholm School of Economics. I am indebted to a number of people for support and guidance. First of all I thank my thesis advisor at the Stockholm School of Economics, Professor Guje Sevón, for constructive advice, encouragement, and for sharing her extensive knowledge about processes of organizing and research in general. I would also like to acknowledge Professor Zary Segall, University of Maryland, for serving on my dissertation committee and for contributing with support, encouragement and creative ideas. I would also like to thank Professor Udo Zander at Stockholm School of Economics, who also served on my committee.

This thesis was made possible by the generous support and active participation of Hewlett-Packard and all the marvellous people that opened their worlds for me, sharing their experiences and friendships. I particularly want to thank Dr. Pankaj K. Garg for never ceasing and continuous loyal support, guidance and friendship. I had also incredible help and support from Research Manager Aad van Moorsel at HP Labs in Palo Alto, California, Cathy Ammirati Burles, Manager at HP:s Image and Printing Division in Vancouver, Washington, and Rachelle Rowland, Manager of the Collaborative Development Program in Roseville, California. I also received great help and support from Myra Scott (you 'included' me in your world and made my life so much fun!), Martin Griss, Jennie Hollis, Svend Frolund, Kaveh Eshgi, Gene Becker and Bernardo Huberman. All you guys and gals made my visit at HP interesting, exciting and fun! And most importantly, I want to thank all of you, who anonymously participated in interviews and meetings, sharing so many interesting ideas and experiences with me, which in the end provided me with the foundations for this study. Also thanks to developers from Motorola, PDG

Hightower and CollabNet who participated and supported this work. I hope to see you all again some time in the future.

During my visit at Hewlett-Packard I also had the opportunity to participate in seminars at SCANCOR, at Stanford University. I particularly want to thank Professor James March for giving me time, commenting on my work. I also want to thank Professor Pamela Hinds at Work Technology and Organization (WTO) at Stanford for giving me the opportunity to present some of my findings half-way through this project. I am also grateful to Professor Steven Weber at Berkeley for sharing experiences and to Professor Bonnie Nardi at the University of California, Irvine, for encouragement and ideas.

This study would not have been possible without the practical and financial support of Hewlett-Packard. Moreover, I am very thankful for receiving funding during my first year as a PhD student from Ericsson. Especially thanks to Sven-Åke Hellgren, Director of Ericsson Credit. I also received extensive financial support from Vetenskapsrådet (Swedish Research Council) which made it possible for me to work for three years with this project. In addition I received financial support from Stiftelsen Marcus och Amalia Wallenbergs Minnesfond.

There are also several others from the Stockholm School of Economics that have been a necessary and important part of my research. First I want to thank all of my colleagues at the Centre for Media and Economic Psychology. I particularly want to thank Magnus Mähring at the Centre for Information Management for many valuable comments on this thesis, Andreas Werr at the Centre for People and Organization for helping me to grasp the possibilities of NVivo, and Ebba Sjögren at Score for outstanding encouragement and practical support when trying to get this thesis in to some kind of shape.

Moreover, I want to thank Professors and colleagues at Stockholm University who supported me in the early phase of this PhD project. In particular I am indebted to Associate Professor Bo Green for encouragement

and support, Professor Bo Hedberg for inspiration and support, Professor Pierre Guillet de Monthoux for great enthusiasm and encouragement, and to Professor P-O Berg for encouraging me to wrap 'it' up! I especially want to thank Professor Magnus Boman at the Department of Computer and Systems Sciences, Royal Institute of Technology, Stockholm University and Swedish Institute for Computer Science for friendship and insightful comments.

This has been a long journey, and I would not have liked to walk this far without the friends I made as I went along. I particularly want to thank Camilla Dahlin-Andersson, Erik Bjurström and Sofie Roy-Ranelid who all of you, in your special ways made this journey so pleasant, joyful and interesting.

Lastly, I want to thank my family for generous support. My mother, Yvonne Milmark, helped me out in many situations and I could not have done this without you. My kids, Cezar, Magdalena and Carl-Gunnar are my inspiration. And Gunnar for love and support.

Thanks to all of you, this one is on me.

Stockholm on April 23rd 2007

Catharina Melian

# TABLE OF CONTENTS

# 1. THE TRANSLATION OF SUCCESSFUL MODES OF ORGANIZING

Many companies are fighting a continuous battle to stay in shape, to remain competitive, and find ways to improve innovativeness. Consequently they tend to imitate successful modes of organizing translating ideas to fit their particular contexts. The instance of translation conveys a process of adopting and subsequently reconstructing a non-traditional form of work organization within the boundaries of the traditional organization. One such attempt, the introduction of Open Source development will be focused in this thesis.

If a certain method or procedure is perceived as successful and innovative it soon tends to be imitated by others. Strong and thematic ideas soon rule and determine the actions of others hoping to achieve the same level of success. Through extensive research we have become aware that knowledge and technology are considered as integral and as sources of competitive advantage in the 'new economy', (Kreiner & Mouritsen 2003). Increasingly, new batteries of communication technologies enable project participants to connect to a dispersed social network (Castells 1996). And in the wake of new technologies people on a regular basis wield their personal social networks to accomplish their every day work (Nardi, Whittaker & Schwarz 2001). Following Latour (1987; 1999b), artifacts and network of actors are co-constructed in the process of innovation. Organizing for innovation is crucial for companies that are forced to quickly adapt to changing competition, markets and

technologies (Dougherty 1996; Hage 1988; Jelinek and Schoonhoven 1990; Zahra and Covin 1995).

Consequently, there is an emergent need to enable people to work together effectively through computers (Mills 1999). Research by Tankjær (1999; 2000) imply that knowledge increasingly is created collaboratively, in a more open style, i.e. utilizing an open strategy that both encourage and generate input from voluntary resources.

The aspects of enacting openness includes transformations towards a stricter focus on core competencies, while relying on third parties for supplying services as well as components facilitated and coordinated by information technology. Transformation as a phenomena presuppose the introduction of a new element. In that sense, transformation tends to influence existing roles of employees as well as creating new ones.

Openness is increasingly becoming a more important feature of organizing. We know from earlier research that information frequently tends to be shared and disseminated in an open and free fashion, altering the chain of command and channels of communication, in favor of dialogue and trust (Clegg 1990; Heckscher and Donnelon 1994; Barzelay 1992). Furthermore, the emergence of a more horizontal, decentralized approach raises fundamental issues of social control. Information technology alters and reshapes traditional ways of managing work. While monitoring is possible it seems as if a larger emphasis is put on empowerment and self-control, (Frenkel et al. 1995). Control becomes inculcated through peer pressure and through the visibility of individual performance in teams. Research suggests that the establishment of knowledge-based trust is founded on knowledge about other people and that it is usually accumulated over long periods of time through interaction, communication and courtship (Shapiro, Sheppard, and Cheraskin 1992; Lewicki and Bunker 1996).

Openness in terms of software development is facilitated by a more unified and universal approach in terms of creating common repositories of a particular kind of explicit knowledge, i.e. source code, which constitutes the building blocks of software

systems. This enables large corporations to move away from explicitly coordinating different actors towards streamlining processes internally as well as with external partners. Collaboration encompasses tearing down walls between actors and relocating work. Corporate relationships become more intermeshed, forcing new meanings in to established vocabulary. The role and meaning of suppliers, customers, and competitors are renegotiated. This transformation process encompass methodologies and social practices geared at making existing knowledge and best practices available and more open, thus facilitating and aiding organizations and its members to leverage from large quantities of information continuously created.

The implications of opening up strategic processes to actors outside the formal organizational entity are important for understanding how to organize for innovation, play and experimentation (Rushkoff 1999). Experimentation opens up for organizational exploration, thus enabling new participants to contribute with ideas and resources (March 1994).

## Collaboration and strategic intent of the organization

Business leaders worldwide believe that implementing a collaboration strategy potentially can ensure greater revenue streams, e.g. through increased communication among employees, customers and between companies, facilitated by technology (Biggs 2001). Furthermore it enables sharing of knowledge, empower people, facilitate organizational learning and bonding, and as highlighted by previous research it improves the quality of life at work (Siviter, Petre, Klein 1997).

Traditional theory on co-operative and collaborative efforts is mainly built and derived from studies of companies in a resource-based setting, i.e. subjected to the laws of diminishing returns. Over the last decades, however, companies are increasingly becoming knowledge-based, and corollary becomes subjected to the laws of increasing returns.

Computers and software programs are complex and expensive to manufacture and sell. But once invented, the incremental production is relatively inexpensive. As more products are built the costs continue to drop, and profit increases. Moreover,

knowledge does not disappear when used, but it can be used over and over again, i.e. what is considered a learning economy is characterized by a net gain in knowledge. Taking these arguments under consideration economists have argued that the pooling of resources is a viable and effective way for firms to compete, since up-front costs, marketing networks, technical knowledge, and standards may be shared (Arthur 1994).

A collaborative strategy facilitates and serves deliberate outsourcing efforts (Quinn 1992). But what does it mean to collaborate?

Etymological, to collaborate means *to labor together, especially in an intellectual endeavor.*[1] Furthermore it also means *to cooperate with or willingly assist an enemy of one's country and especially an occupying force.* In addition it may also impose cooperating with an agency or instrumentality with which one is not immediately connected. Collaboration cover the spectrum between colleagues working together in a trusting relationship, through instances of different stakeholders trying to accomplish their individual and separate goals, and even situations when adversaries or competitors are compelled to pursue acts of collaboration (Doheny-Farina 1986; Matusov 1993; McMaster, Jones & Wood-Harper 1997; Newman & Newman 1993; Self 1992; Cohen, Cash & Muller 2000).

Research by Cohen, Cash & Muller (2000:34) suggests that collaboration between stakeholders, or adversaries:

> rely on the strategic manipulation of awareness of the existence and availability of information, including documents, people, and processes. Their ability to control access to these resources by selection is one key to their success.

While people from different organizations or from different parts of an organization, evidently need to come together and collaborate, it may simultaneously become necessary to control membership and visibility while maintaining as much flexibility as possible (Clement & Wagner 1995).

[1] Merriam-Websters Online Dictionary, www.m-w.com/dictionary/collaboration

Collaboration presupposes courtship. Courting conveys not only invitation to something, but also excluding others from this. The risk we face when opening up and courting others to collaborate, is that we turn in to 'a fellow traveler or a sympathizer', in a negative sense. For instance it may connote that we sell out our own ideas to others, e.g. individual actors or organizations. It may also purport that we become less creative, through the eagerness of pleasing the other partners. We hypothetically become less prone to pursue our individual ideas. Ideas tend to be negotiated and power relations sometimes have an impact on what actually gets highlighted. From time to time ideas are concealed until political issues are settled (Newman & Newman 1993).

Other well-known negative aspects of collaboration are a general unwieldiness of cooperation and cooperative endeavors between intellectuals, e.g. instances where individuals are afraid to get their ideas taken away from them. This has significance for immature ideas that still are in an early phase, which consequently rarely can be protected by intellectual property rights, e.g. patents and copyright (Lessig 1999; 2001). Precipitate and impulsive ideas every so often tend to be withheld by individuals for social reasons, e.g. losing face or simply jeopardizing reputation. Collaboration then becomes an issue of weakening or strengthening a competitive relationship between actors in a collaborative setting.

Collaboration additionally denotes something positive in that it embraces the notion of working together, and to possibly conquer an intellectual endeavor. But it also symbolize something less positive and constructive, i.e. control, that what is not legitimate, to omit or to leave out, to be in collusion or perhaps in cahoots with the enemy. Collective work is to a large extent depending on convention, but it is also spurred through innovations (Mukerji 1998).

## Translating Open Source

In open-source software development, source code is shared and refined by any interested actor. It is a freewheeling process that previously was unheard of in the creation of proprietary software. The rationale behind Open Source development is

that the involvement of as many programmers as possible in the development of the product leads to a more useful software. It is put forward that the collaborative approach in itself leads to well-designed software.

But how can we understand the logic behind Open Source software? Plausible it becomes easier to grasp using an analogy proposed by Weber (2004). Weber uses Coca Cola as an example of a company selling bottles of soda to its customers. And we the consumers drink it. The ingredients are only generically listed on the outside of the bottle and we (the consumers) can read that it contains sugar, water and a few other ingredients combined with a secret mix of flavors that in the end creates something of great value. But, it is practically impossible to do reverse engineering into its constituent parts. Since we cannot know exactly what is put in to it – we cannot duplicate it. Open Source software is quite the other way around. In fact it inverts this logic, by ensuring that the source code must be distributed with the software, and that anyone may redistribute and modify the software for free.

The notion of Open Source really goes back a very long time into the development of a variety of libraries of FORTRAN, the computer language. During the mid 60's people began making available on university computer networks methods of plotting data, and the code was left open to the users. The emergence of the Internet, and more specifically TCP/IP, which is the Internet protocol made previously incompatible networks all compatible with each other, which has led to a very open and free-flow of information. The evolution of the browser, which in a similar way leads to the ability to view code without having a particular application also has facilitated openness and sharing. Over time things like CVS[2] and Bugzilla, which is an Open Source way of tracking bugs made it easy for almost anyone to post code making it accessible over the Internet, i.e. code that previously was hosted behind the firewalls now became available.

A recent trend in the software community is to utilize a virtual workspace on the Internet, facilitated by common toolboxes and techniques imported from the Open Source community. Actors closely related to the Open Source Movement have

established commercial entities offering infrastructure solutions to large corporation A vision of open collaboration is brought forward by Brian Behlendorf , the CTO of CollabNet:[3]

> Creating a virtual work space. Making it as easy as picking up a phone to start collaborating with (developers) and sharing ideas. That's pretty powerful. Then if I can take what's developed there and put it in some permanent archive someplace where I can refer to it later, then that's pretty cool. That plays in very well to the type of stuff we're doing here: Building software for building infrastructure that allows groups of people to collaborate on software development. This kind of real-time, shared white board chat is next stage.[4]

For many large corporations, software is believed to be the key to maintaining control while at the same time increasing productivity (Toupin 2001). Conceivably for this reason, large companies continuously seek to improve their development processes through imitation. Hewlett-Packards former CEO Ms Carly Fiorina expressed it like this when talking about Linux:

> The Linux movement is based on openness, on the idea — and the evidence — that the achievement of our collective work is greater than the sum of individual efforts. That everyone benefits when everyone else advances"... "Like the Web itself, Linux is an open source technology that spawned an entire industry — and that continues to be improved by smart minds collaborating all over the world.[5]

Nonetheless, open source practices cannot automatically be transplanted into the proprietary milieu. They have to be translated into the context of the organization. And the process of translation is influenced by different concerns of the translating organizations. Such concerns are e.g. when open-source software processes are called in questioned as if they have the necessary rigor needed to produce timely, and at the same time commercially viable products.

---

[2] Concurrent Versions System.

[3] Other commercial entities offering similar solutions are SourceForge provided by VA Linux.

[4] http://sanfrancisco.bcentral.com/sanfrancisco/stories/2001/05/21/newscolumn8.html

> Developers sacrifice individual autonomy---they perform a ballet---to get a product out early...Technical prowess isn't the top-rated feature of a programmer; predictability and consistency are.[6]

Translation processes are then, acts of invention, and is brought forward by combining and mixing various elements and constraints. According to Callon we need to perceive this version of translation in the following way:

> Translation involves creating convergences and homologies by relating things that were previously different (Callon 1980:211).

From yet an other angle, Open Source collaborative methodologies may also be viewed as a social phenomenon and as social creation in the sense that the development is performed among an array of persons, organizations and sometimes institutions with an integrated perception on how to achieve a desired outcome. The methodologies are subsequently translated (Latour 1999a), by the actors through the implementation of a common infrastructure, inserting common languages and techniques for development, and through the imagination of an idea of sharing, creating momentum through the formation of a community helping and sharing knowledge and experiences more openly. The development is being organized in projects utilizing a network of actors, consisting of employees, and third party developers and a common infrastructure for communication.

The emergence and the dependence on personal social networks in a workplace through enacting an Open Source strategy, brings forward an amalgamation of different developers, belonging to different groups, being either internal or external to the organization, suggesting a larger component of development relying on voluntary resources giving room on the expense of traditional management control, planning and structuring teams of developers.

---

[5] Fueling Innovation, Opportunity with Linux LinuxWorld Conference 2002 New York, New York — January 30, 2002

[6] Mark Evans, president of investment-performance software maker Confluence, Carnegie Mellon University Software Center roundtable in Pittsburgh, December 2001.

Open Source Software challenges existing patterns of competition in the software industry.[7] This becomes evident as large corporations are investing in Open Source Software trying to leverage the community of developers as a strategic thrust against their competition. Hewlett-Packard e.g. has for many years invested in Linux and Open Source development and they strive at continuously implement cutting-edge information from these communities into its own products and solutions (Shacklett 2007). The model for Open Source development is pretty straightforward and well documented, but what happens when corporations adopt certain features of the model, translating it into its own context? There seems to be some challenges when corporations adopt Open Source methodologies, at least at first sight given these differences:

1. Corporate software development usually has to work with fixed resource limits, e.g. time and money. Open Source development does not.
2. The motivations for participating in Open Source development are usually fun, problem solving, and fame. In a corporate project the monetary compensation is more prevalent.
3. The size of the developer community is substantially reduced in the corporate project, (half a million vs. a few thousand).

At this point in time, it is evident that the initial hype of Open Source development has faded. However, some Open Source practices, such as sharing source code with business partners is becoming more and more interesting for proprietary software producers. From this context Open Source software is distinct from the proprietary software products developed and sold by companies. Intellectual property rights (denoted by proprietary) are used in order to create incentives for innovators. Through patents, copyrights, licensing and other available means knowledge is protected to ensure that economic rents are ensured and corollary also appropriated by the innovator. The underlying rationale is that if knowledge is not protected it can

[7] Cf. The Boston Consulting Group Hacker Survey, Release 0.3, In Cooperation with OSDN, LinuxWorld Presentation 31 Jan 02, Distributed under the GNU Free Document License v1.1

be used by each and everyone leaving little or no economic incentive to innovate in the first place.

In the next section I will go in to the particular context of Hewlett-Packard.

## Translating Open Source the HP way

Software program development is an intensively personal, innovative process, sometimes compared to an artistic process or a play (La Plante and Seidner 1999). Large-scale software development, on the other hand, is by its very nature a collaborative effort (Brooks 1975). That said we recognize a noticeable tension between issues of individual versus collaborative efforts in software development.

Companies, such as Hewlett-Packard increasingly seek ways to adopt more open strategies for sharing source code[8] with business partners, subcontractors as well as customers. Tools and methods emanating from the Open Source community are developed and reconstructed in order to suit the needs of large corporations.

The underlying rationale is that a speedier and more agile approach towards innovation is more desirable than protecting intellectual property rights at all costs. Through a more open – but not entirely open collaborative relationship – the developers can receive and provide input in the development process which potentially will enhance the products.

An entire new development paradigm, the Progressive Open Source was initiated in 2001 and subsequently influenced engineering practices at HP. It was explicitly targeted towards rejuvenating software development through combining existing practices with tools and technology widely used and utilized within the community of Open Source developers.

[8] Source code is basically the running program of a system, i.e. the underlying code that constitutes the software program. Closed or "black-box" systems are typically written in binary form that makes them incomprehensible, (Neumann 2000)

Progressive Open Source sets out to establish a set of centralized techniques, tools and infrastructure enabling fast, convenient and effective communication between developers and third parties engaged in short and medium duration projects. Interoperability is sustained by standard protocols and syntaxes. Distributed collaboration is dependent upon global access, common solutions and tools, and standards. The idea of involving third parties and the customer in the process of innovation is important, since it is a way of anticipating the need, problems, and competence of the other actors. Victor and Boynton assert that co-configuration, i.e. development in conjunction with the customer, increases adaptability and learning capacity, as well as strengthening the relationship with the customer (1998).

Open Source has presented Hewlett-Packard with an opportunity to leverage a larger development community, and managers have gradually more recognized the great objectives, leveraging from a larger community of developers.

Lee Caldwell, former IPS Chief Technology Officer at Hewlett-Packard stated in 2002 that:

> HP faces the major challenge of reinventing imaging and printing in the Internet era. This means that we have to develop meaningful contributions at a faster pace and provide a way for partners to leverage our inventions. The Collaborative Development Program is a critical step facilitating breakthrough work and leveraging more quickly inside of HP and with (external) partners...

Hewlett-Packard deliberately set out to implement an approach inspired by Open Source software development. The initiative was denoted POS since it included a progressively more open approach starting from openness and sharing within the organization to include and involve a set of trusted third partners, i.e. customers as well as suppliers. Sharing, openness and collaboration were guiding stars fuelled by the belief that speed over secrecy is preferred.

The implementation of a common infrastructure for communication and development produced a development process where the feedback loop between the developers became simultaneous and intermeshed, clouding the distinction between

internal and external developers. Knowledgeable developers are a prerequisite but more and more found very cost effective, especially in countries such as India, which has turned out to be the paradise of outsourcing efforts. Hewlett-Packard is frequently using such contractors, e.g. Wipro Technologies, [9] which is a very fast growing Indian company.

Through these combined efforts Hewlett-Packard, strive to become an organization where knowledge capturing and sharing is the norm, thereby offering customers speed-to-deploy as well as innovative products and services. Consequently it was perceived as necessary to capture and leverage skills and knowledge of the members of the organization.

The POS initiative was presumably launched in order to increase the transfer of knowledge from the individual level to the organization level. In this thesis I will cover four separate programs initiated by Hewlett-Packard; Cool Town, The Collaborative Development Program (CDP), Bluestone and the Corporate Source, which are all initiated, (according to the explicit goal of HP) in order to bring together people who have requisite tacit and explicit knowledge with those who need it, inside the formal organization as well as to trusted partners.

The respective projects within POS vary as to their intended degree of openness ranging from being complete Open Source projects (Cool Town) and corollary sharing everything in the open across organizational boundaries to covering internal sharing only (the Corporate Source). Collaboration is elicited among skilled employees within the company as well as employees of other firms in established networks and alliances.

In **Table 1** I present a short genealogy highlighting some of the differences and similarities of the projects.

[9] Wipro has successfully formed alliances with companies such as Hewlett-Packard, SUN, Ericsson, Microsoft, IBM, and others providing a highly skilled workforce to a relatively low cost.

***Table 1:*** *A genealogy of the different projects that constitutes Progressive Open Source*

| **Dimensions** | **Corporate Source** | **CDP** | **Bluestone** | **Cooltown** |
|---|---|---|---|---|
| **Openness** | Internal only | Internal and selected partners | Internal aiming at complete openness | HP project/initiative Open Source code |
| **Infrastructure** | Intranet solution, HP Library | Internet based solution, | Internet solution | Internet solution |
| **Community size** | Small, grassroots driven effort emanating out of the effort of a single HP lab researcher | Large, more than 3000 cross organizational, top management supported | Fairly small 200-300 developers aiming at infinite numbers targeting complete Open Source | Small (less than 20) aiming at indefinite numbers Open Source developers |

To sum up POS is about organizing software development in a large corporation through fostering a network of communities around the development of software systems. Software systems are integral to the functions of any modern large, corporation. Such systems minimally include word processors, email, group communication systems, and often include complex systems such as Enterprise Resource Planning (ERP), Enterprise Portals, and so forth. Hence, most modern corporations of today heavily utilize and often develop software systems or their customizations.

The success of several Open Source software systems, e.g. Linux, Apache, suggests that certain collaboration practices of Open Source development methods, like open discussions for features and requirements, the ability of the user community to participate in such discussions with the developer community, is perceived as very attractive and potentially benefiting large corporate software development and customizations.

Even though HP was fairly early translators of Open Source principles[10], it was not the only corporation choosing a progressively more open approach. Sun's Community License is an example of a restricted Open Source license. IBM's

adoption of Apache in Web Sphere was similar instances where corporations adopt and contribute to existing Open Source software. Microsoft has adopted a limited openness for its Shared Source program.

We can conceive of an image where companies are dancing on a tight rope between extremes (cf. Mauss 1950/1990; Smith & Kollock 1999; Barbrook 1999; Rehn 2001).

- A commodity driven economy versus a gift economy
- Proprietary, closed software versus Open Source systems
- Market competition versus network communities
- Digital encryption versus free downloads

As companies try to adapt to changing conditions, hybrid solutions emerge. The Progressive Open Source is one such example.

In the next section I will provide a summary of distinguishing features between Open Source Development and Progressive Open Source as it relates to Hewlett-Packard.

## Juxtaposing Open Source and POS

Open Source as a phenomenon has been studied with amazing intensity over the last decade. And it has been proposed to us that we may think of it in terms of a gift system and also as mainly a social phenomenon distinct from the formal organization, (cf. Raymond 1999; Smith & Kollock 1999; Rehn 2001; Weber 2004).

The original ideas that pertain to the so called gift economy go back to studies by Malinowski in e.g. the American Northwest, Melanesia and Papua.[11] The anthropological studies of Malinowski was later theoretically developed by Marcel

[10] Starting in 2000/2001 time frame.

[11] Bronislaw Malinowski made detailed observations about tribal economics, e.g. Argonauts of the Western Pacific, 1922, Routledge and Kegan

Mauss who elaborated on a gift economy/system and when defining it he said that we must think of it in terms of an economy with one important feature which: "is clearly one that obliges a person to reciprocate the present that has been received" (1924/2000:7).

Open Source as a phenomenon seems to share some of the cultural characteristics of the gift economy, e.g. it is a system which seems to connect people, and furthermore it encourages reciprocity, i.e. individuals both give and receive in return. And the gift is the source code, which is for 'free' while it is implied that receivers also contribute something (at least bug-reports).

In **Table 2**, I have juxtaposed a set of differences between what is generically known as Open Source, with Progressive Open Source as it became known to me through this study. I have included references for dimensions where I have derived theory from studies of validity for Open Source. It does not intend to be exhaustive, since for the purpose of this study I have it in mind only as a way to delineate what Progressive Open Source seems to be.

***Table 2:*** *Open Source and Progressive Open Source – a comparison*

| **Open Source** | **Progressive Open Source** |
|---|---|
| No formal organization, i.e. rather a social phenomena | Formal organizational entity, sometimes involving 3rd party developers |
| Gift system. The institution of Potlatch, (cf. Mauss 1950)<br>Underlying notion: abundance<br>Each gift is part of a system of reciprocity. Every gift has to be returned in a perpetual cycle of exchanges | Market system<br>Underlying notion: scarcity<br>Calculated gains and losses, utility maximization |
| Very open and free flow of information | Progressively more open – however controlled flow of information between trusted partners internal and sometimes external to the formal organization |
| Voluntary teams working on projects | Designated teams, i.e. employees and/or contractors working on defined projects loyal to a formal organization |

| **Open Source, continued** | **Progressive Open Source, continued** |
|---|---|
| Meritocracy/benevolent dictatorship – i.e. Linus Torvalds rules over Linux – but grassroots are allowed to use the system for free (cf. Raymond 1999) | Traditional hierarchical organization |
| Sociological phenomena. It has the potential of bringing together software practitioners regardless of their prior record, e.g. being employed by a company, having the right educational background, age status etc. It's only the validity of the contribution that counts (cf. Smith & Kollock 1999). | Organizational phenomena. Technology in place that enables collaboration between software engineers employed and/or contracted by the organization and trusted partners. |
| No time and budget constraints. | Often strict time frames and budget restrictions. |
| Potentially limitless influx of knowledge and skills – infinite number of developers. | Controlled number of developers. Significantly smaller than Open Source community. |
| Public good:<br><br>Non-rival – i.e. one person's consumption of the good does not reduce the amount available to another.<br><br>Non-excludable, i.e. excluding others from consuming the public good is difficult or impossible.<br><br>*Challenge:*<br><br>How to get people to contribute and avoid free riding.[12]<br><br>How to coordinate a large group of individuals that want to contribute (cf. Weber 2004). | Privately owned/controlled intellectual property rights. |

[12] It is argued by Weber (2004) that free-riding may in fact be desired, since free-riders (at least a small percentage) often contribute something of valne, e.g. reporting a bug. This argument will only hold as long as an adequate nuber of developers continues to contribute to the common good.

| Open Source, continued | Progressive Open Source, continued |
|---|---|
| Motivation to contribute (cf. Smith & Kollock 1999; Weber 2004):<br>• Identity (honour, reputation, ego-boosting)<br>• Reciprocity<br>• Efficacy: individuals contribute because they have some impact on the system.<br>• Art and beauty of solving problems<br>• Work as vocation | Motivation to contribute:<br>• Monetary<br>• Patents<br>• Ranking system/reward-system |

## From a model of diffusion to a model of translation

Traditional theory of the diffusion of models, techniques and practices are often outlined as homogenous and isomorphic, e.g. Rogers (1962/1995). Rogers defines diffusion as the process by which an innovation is communicated through certain channels over time among the members of a social system. The model contains four important elements that characterize the diffusion of innovation; the innovation, the communications channels, time and finally the social system. In his classic, Diffusion of Innovations, Rogers traces formal change research to Gabriel Tarde's 1903 book, The Laws of Imitation.

As a result of the extended study of POS at HP I cultivated my ability to understand things from the point of view of the organization and its members. It is perhaps best described as an empirically driven theoretical movement and development. I began to view POS, not in the view of something static that diffuses, but rather as something moving, performing, changing as the local interpretation of the ideas emerged. Looking at the travel of ideas through the model of translation became a more elaborate analytical framework that emerged and in my view also seems to fit particularly well when studying the role played by science and technology in organizing (Callon 1986).

To explain the diffusion, dissemination, propagation or dispersion of an idea, a claim or an artifact there are at least two possibilities according to Latour (1986/1998:42). The first is to allow for an inner force, similar to the inertial force of

physics. The inertial force allows an object to travel in the same direction as long as nothing stands in its way. This possibility, the model of diffusion, does not explain how an idea, a claim or an artifact is dislodged in time and space. It merely explains increase or decrease in velocity. According to Latour (1986/1998:43) the model of diffusion defines three important elements in the propagation of an idea or an artifact through time and space:

1. The initial force that sets of the motion
2. The inertia that maintains the motion
3. The medium that facilitates the circulation

Latour (1986/1998:44) contrasts the model of diffusion with the model of translation, which as he stresses relies on the human being for propagating ideas, claims, demands, artifacts or commodities. The following aspects are highlighted by Latour (ibid.) as being important to acknowledge:

- Each and every one acts individually and may alter, modify, ignore or divert interest from the above mentioned. A correct (in the sense un-altered) transmission is rare in such a model.

- Friction is not caused by some primordial kinetic energy, since e.g. ideas and claims etc, lacks energy in them self. But, rather it is the consequence of the energy of those who work with ideas. This means that it is evident that constant movement is dependent on finding new sources of energy, since it is impossible to rely on what happened in the past.

- Each an every on who takes part in the chain of movement, the actors, will form and impact the idea according to their own respective needs. They translate and interpret the ideas accordingly.

Ideas are in constant flux as they travel from one individual to another, crossing one context entering another. Ideas are transformed. In this study it becomes relevant to acknowledge that is a very deliberate procurement of an idea, a best practice, i.e. Open Source development that serves as model of imitation for Hewlett-Packard. The model is translated and transformed through a process of reception, interpretation

and reformulation. The model is thus edited to confirm existing procedures and situation, and it becomes re-labeled. Open Source Development becomes Progressive Open Source within the realm of Hewlett-Packard. When we look at processes of imitation in this context, it is not a matter of isomorphism, but rather a process towards more variation. This study will highlight that even though certain best practices tend to be imitated by many organizations and thereby creating expectations of homogeneity and distinct conformity in the development process, the travel of ideas does not generally create identical or even similar practices.

## *Moving away from an ostensive towards a performative definition of social causality*

According to Latour (1986/1998), an ostensive definition allows in principle for the recognition of characteristics that are typical for society, and which could explain social causality. Furthermore, social actors, regardless of their relative size and importance exist in society. And in spite of the fact that they may be active (as implied in social actors), their respective activities have delimited importance, since they are regarded as part of a bigger entity – the society. The actors of society are regarded as important informants for researchers looking to discern patterns of principles that keep society together. But, the actors are merely informants, and corollary it is not possible to rely entirely on them, since they are unable to grasp the wider context.

Finally, using the proper methodology, researchers are able to arrange the views, conceptions, illusions and behavior in order to comprehend the characteristics typical of life in a society, and assemble the pieces in to a whole. Within such a system, all problems related to the origins of society are regarded as practical difficulties that potentially could be eliminated with more and better data, a more superior methodology, and a more accurate delimitation of the research problem.

Following Latour (1986/1998), the performative definition states that it is impossible in principle to define the list of characteristics that potentially could be typical for life in society, however it may be feasible in practice. The actors define, regardless of their size, what constitutes society in practice, for themselves as well as

for others. No premise is necessary regarding one actor is more knowledgeable than another. Researchers deal with the same issues as any other actor of society and strive at finding practical ways of highlighting their definitions of society.

Imitation viewed as a performative highlight an "individual capturing an idea, translating it into something that fits its own context, and materializing in into action" (Sahlin-Andersson & Sévon 2003:253). Imitation is thus "a process in which something is created and transformed by chains of translator" (2003:253). On the other hand, an ostensive definition of the diffusion of innovation implies that "what is to be imitated is seen as a given, as something objectified, something immutable, born with an impetus that propels it across a social area or space with various degrees of resistance" (Sevón 1996).

## Aim of the study

The aim of the thesis is to increase knowledge and understanding of the conditions for and the consequences of collaboration in a progressively more open context.

In this thesis I will investigate and substantiate a new way of organizing a process for innovation, i.e. research and development of new methods, products and services in a company. In an organizational practice 'innovation' not only refers to extraordinary features of a changing organization of work, but can also reside within a commonplace of improvisation taking place in everyday practices.

Specifically I try to capture how a high technology company adopts Open Source methodologies for innovation, transmitting and transforming models promoting collaborative work in a physically dispersed organization.

## Structure of the thesis

This section is a traveler's guide that intends to provide the reader with some information on what to expect when reading this thesis.

This thesis is about the process of applying Open Source ideas and methodologies in the context of a company, i.e. it describes and analyses the enactment of the idea of openness in software development practices. The research report concerns a new organizing phenomenon, which has its roots outside the traditional corporate organization.

The Open Source software movement, as such, has received much attention in the last decade, and many studies have tried to analyze and grasp its underlying rational. The movement has certain distinguishing attributes, e.g. spontaneous organization of voluntary developers residing physically dispersed whilst interacting over the Internet. This is decidedly at odds with traditional best practices in software engineering.

In this thesis, I report on a study of a particular initiative within a company in the United States, Hewlett-Packard, on their Progressive Open Source software development project. In the report I successively apply different theoretical lenses in order to understand how the project was implemented mainly based on the collected experiences of different professionals who were active in the project.

In Chapter 2 I will present the research methodology I employed in the empirical study at Hewlett-Packard. My collection of data was conducted in a field setting. During 2001 and 2002 I visited the HP labs in Palo Alto California for a period of 7 months. I also traveled to different HP locations in the United States. In this chapter is described the variety of techniques used for data collection.

Chapter 3 presents the Progressive Open Source Idea in some detail and gives a retrospect picture of on how the ideas of openness became part of a more institutionalized implementation, influencing software development practices within the organization.

Chapter 4 sets out to envisage POS as an organizing device and as a new deliberate strategy for driving innovation while upholding a from an organizational point of view relevant balance between modes of exploration and exploitation. It also touches ground with some fundamental issues related to generic HP culture. For instance, POS projects challenge the idea of the lonely inventor developing software

in 'the garage', which is a legend that goes back to the days when the company was founded by Bill Hewlett and David Packard. In the chapter particular emphasize is put on the organizational history and context out of which the POS has evolved.

Chapter 5 highlights collaboration at a distance and depicts instances where face-to-face interaction is preferred by developers. The theoretical lens applied in this chapter originates from ideas of socially shared cognition, which is a branch of psychology that leans towards viewing cognition as a social phenomenon. Taking a constructivist stance on cognition conveys making cognition integral to social processes. Open Source software development seems to challenge traditional theory on the importance of face-to-face interaction and communication, since it is a phenomenon that thrives on the Internet. We might project that POS would work just as well in a distributed setting. The results of this study indicate differently.

In Chapter 6 I take on a different theoretical lens when dealing with aspects of language in collaborative development. Through the POS, the organization sets out to implement a unified model consisting of tools, methodologies, language use and process. I particularly distinguish between language issues related to the distributed setting of the organization, i.e. encompassing cultural issues and issues related to developers speaking different native tongues. In addition I address issues related to technical languages and analyze implications of introducing a standardized toolset. Lastly, I also address issues related to non-verbal communication and particularly analyze instances where pictorial language is preferred over written language.

Chapter 7 moves in to the domain of openness and concomitant modes of management and control. I set out to do so using theory about a virtual extension of panopticon. The key control mechanism is surveillance and self-discipline facilitated by openness. Visibility suggestible imposes certain behavior and the study indicated instances of such behavior – POS thus also becomes a template of control. POS not only challenges the prevailing presuppositions for control of the developers, it also imposes changes on the role of management.

Chapter 8 comprises of summary and concluding remarks, referring back to the aim of the thesis, and the theoretical lenses applied in the study. I also discuss

openness conceptualizing and positioning it as a theoretical foundation based on the empirical study. I argue that the transition towards a more open system has consequences for the process of organizing research and development in an organizational setting.

# 2. METHODOLOGY AND RESEARCH DESIGN

This chapter comprise of three separate sections. In the first section I discuss the approach I had for the thesis, both in concrete terms but also discuss some basic assumptions, i.e. issues related to ontological, epistemological and methodological assumptions. In the second section discovering the thesis, I lay out the rational behind the thesis. In the third section, uncovering the thesis, I outline how the research has been undertaken. In the final section I discuss issues related to ethics and attitude I had towards concluding the thesis.

## Approaching the thesis

In this section I describe my approach for understanding the adoption and use of Progressive Open Source within HP. This encompasses sharing with the reader some of the basic assumptions that I had before actually designing the research project. It is perhaps best described as a process of setting the problem. Conducting research is not a simple and straightforward task. Undeniably the process of setting the problem is difficult, since the things that we attend and frame so closely are related to our own particular context of experience and knowledge. Hence what I originally intended to study was as a result of the research process itself gradually altered. Consequently the subject and object of knowledge is inescapably intertwined, which from a strict scientific standpoint tends to be problematic (Alvesson & Sköldberg 1994).

Hewlett-Packard and the POS initiative has been the focus of this study. And with risk to state the obvious, I did not enter this field of research a blank slate. That

probably would not even have been possible in the sense that without some prior understanding this experience simply would not have been intelligible to me. Only, looking back retrospectively I believe that it is not a palpable disadvantage to have fresh eyes in the process of making sense of Progressive Open Source. It has involved both instances of discovery and creation.

Throughout this research I have deliberately chosen to investigate and substantiate new ways of organizing innovation. Consequently I have set out to analyze what collaborative software in a company is all about in practice, i.e. how software developers perceive collaboration and collaborative efforts, what it is and what the results are for the organization. Hence, the approach was to explore with an ambition to capture the dynamics of POS. Because the study involved new modes of organizing it was necessary to leave the research focus broad in scope initially and try to stay as open as possible in the initial stages.

Given the initial aim of the study (which admittedly was relatively broad and open) I was convinced that ethnographic studies were most relevant. This particular set of techniques are increasingly gaining more momentum, e.g. research in the area of technology-mediated collaborative work has demonstrated that this approach can provide insights in the vicinity of issues related to social interaction, i.e. how work is coordinated and how unexpected events are handled and co-managed (Heath & Luff 1991, Goodwin & Goodwin 1996, Suchman & Trigg 1991, Bentley et al 1992). Yet other similarly related research in the area of human-computer interaction has provided insights in to the subject of collaborative work and thereby drawing important connections between human aspects and technology (Zuboff 1988; Gantt & Nardi 1992; Nardi 1995; Star 1995; Nardi & O'Day 1999). Ethnographic research has in terms of these aspects emphasized understanding regarding computer-based activities of people, especially concerning intentionality of human actors.

But why do we go on field studies in the first place? For me personally, ethnographical studies are very interesting because it enables the researcher to make a very close connection to people and how they work. It goes well with my personality since I am really curious and interested in people, and actually living in an organization for an extended period of time enables in-depth understanding of a

reality that previously was not known to me. I was plain curious to know the world of the developers, and how they *construct their worlds*. Would that be all that different from how I construct mine? Would I even be able to talk to them given my limited experience of their professional language (*software engineering)*? I was not sure if I was to be accepted. I decided to settle for the fact that they actually invited ME – I wasn't exactly intruding on THEM. Remembering what Barley said:

> The best one can probably hope for is to be viewed as a harmless idiot who brings certain advantages to this village[13] (Barley 1986:56).

Being an outsider is not such a bad thing when carrying out field studies, in fact it refrain the researcher in the field from becoming a *practitioner* in the sense assuming the role of the advisor. Czarniawska (1997) suggests that this is a role that is easy to assume as a researcher in business studies, thereby jeopardizing the research (she uses the word *meddling with realities*, which I intuitively interpret as something best left to – advisors...).

Ethnographic studies presuppose that researchers spend time on the field trying to understand ongoing changes through observing the events as they are unraveling, e.g. by listening to the testimonies of those affected (Czarniawska and Sevón 1996). This is considered important since" only what moves is visible" (Czarniawska and Sévon 1996:2).

But fieldwork does not render itself easily. In my view, ethnographic studies must include noticeably more than telling a story. To give an example, for me this approach translates in to presenting a set of ordered observations along with theory and analysis. My personal interpretation of this particular approach, and the way I set out to do the study involved planning to do interviews, participate in meetings and other work related activities, making observations, and collecting artifacts in any form that seemed to have some significance for the topic of my research topic. And then analyze it, to the best of my ability, comparing my observations with related research.

[13] Citation from Nigel Barley, (1986), A plague of caterpillars: A return to the African bush, London: Penguin, As cited in Narrating the Organization by Barbara Czarniawska 1997 p. 62

Approaching this study I consciously made a decision to try to apply and combine several research methodologies, theories and empirical material in an attempt to mitigate at least some of the inherent weaknesses and biases that often threaten single observer, single-theory studies. Triangulation then becomes an alternative to more traditional criteria such as reliability and validity. This tactic helped me to deal with some difficult questions, e.g. how to define the case I was about to study, how to determine what data to be collected, and more lately (throughout writing this report) also decide what to do with the data. Nevertheless, on the outset I never aspired much further than trying to collect 'little narratives' in an effort to gain understanding, whilst trying to capture a richer meaning of collaborative action, i.e. in the sense trying to evoke the associations with movement as proposed by Czarniawska and Sevón (1996).

It is time and again argued that detailed and cultivated understanding presupposes an inductive approach, i.e. approaching the research setting without any predetermined theory (Glaser & Strauss 1967; Patton 1990, Strauss & Corbin 1990). Such an avoidance of a predetermined theory supposedly sustains the researcher's ability to think outside the box. As suggested by Glaser & Strauss (1967) and Strauss & Corbin (1990) patterns must be allowed to spontaneously emerge. In this study however, it is assumed that such aspirations are limited in practice. As it turned out, it was because I had already some (albeit rather limited) theoretical understanding of this particular area of research that I could pose some questions of relevance for practitioners, or at least have conversations with them. I knew enough of their world to be interested in it. Without such a point of departure, I would most likely have been completely overwhelmed by the complexity of the phenomenon I intended to study, but more importantly I probably never would have gained access to the *reality* I wanted to understand more about.

The initial research questions that I posed were not completely random, but rather generated out of the academic research available in books and articles, and for that reason somewhat *purposefully* guided by theoretical assumptions as has been pointed out by Lincoln and Guba (1985). This research strategy ought not to be confused with having a completely predetermined research design. Put differently, as

my experience increased (both as a result of the extended visit at Hewlett-Packard as well as from the literature studies that took place in parallel) I continuously elaborated my assumptions and ultimately also I began looking at the phenomenon differently, something else emerged if you will. As many other doctoral students I have repeatedly returned to the question – "What is my work about?" I believe, in my case, that this has been a process of gradually becoming aware of aspects that I did not know when I first set out (Lindblom 1990).

As previously mentioned in chapter 1 and now iterated, my extended study at POS at HP and the research I studied in parallel convey what may be described as an empirically driven theoretical movement and development, closely related to that which is put forward by Lincoln & Guba (1985).

I started off this study with a set of questions derived from studies on the diffusion of innovation (Rogers 1962/1995). I hoped to know more about the characteristics of innovation as conceptualized by Rogers, e.g. relative advantage, compatibility, complexity, trialability and observability (1962/1995). Another area of interest was motivation and how collaborative environments may encourage individuals and teams, taking particular interest in rewards and recognition. These questions are derived from the work of Mauss (1924/1990), and focus on gift cultures and the way people become motivated to make contributions to a larger community. This and other closely related theories were receiving renewed interest in studies pertaining to Open Source software development (cf. Rehn 2001).

This study from the outset took its point of departure in existing knowledge that was available to me in 2000 and 2001 about Open Source development. I had just started taking an interest in Open Source development as a phenomenon and had written a paper on how Open Source projects are coordinated, which I also presented at a workshop for PhD students in Uppsala.[14]

[14] NFF 2001 in Uppsala.

Evidence of the significance of prior understanding (albeit limited) is found in the template[15] I constructed which served as a foundation for the semi-structured interviews that I later conducted with developers and managers at HP.

In addition to the above-mentioned, I was interested in collecting stories from the field in order to grasp the wider context of Progressive Open Source, i.e. where it comes from and why it is perceived as important by a large corporation to adopt methodologies and ideas from the Open Source model of developing Software.

## Discovering the thesis

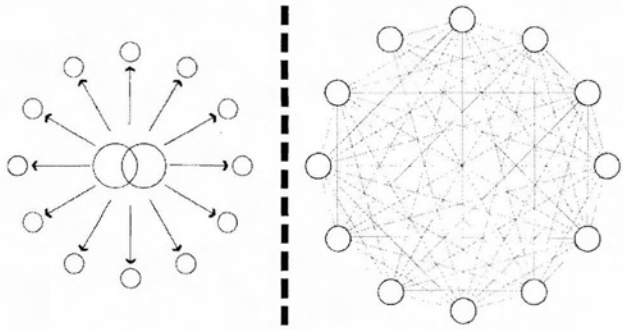
The concept of openness has played an important role for me as I began to discover the thesis. Openness tends to be praised in academia and is closely related to scientific work, connoting images of sharing and at least potentially innovativeness. But could it also somehow be related to physical proximity? Or is physical distance between collaborating actors irrelevant for innovation? Those and similarly related questions were, (and still are) important to me on the outset of this study.

A pre-study that I conducted at Ericsson[16] indicated that organizations were increasingly establishing virtual teams working in distributed settings. The organization used the word *glocal* to symbolize acting globally and locally at the same time. Two teams were merging to one, and subsequently the individuals of the community were globally distributed in order to establish close relationships with the local customers. In spite of the good intentions it seemed as if they were struggling with the collaborative effort, but rather remaining separate satellites with limited knowledge exchange and transfer. This image as presented in **Diagram 1** symbolizes their goal (to be networked) albeit the result indicated separate satellites.

[15] The complete template for the interviews is found in the appendix.

[16] I studied knowledge management activities related to risk management issues at Ericsson, Customer Finance group in 1999 and 2000.

***Diagram 1.*** *Glocal teams*



At this instant in time, i.e. in 2000, the phenomena of Open Source software development started to draw significant attention from the field of sociology and management research. What struck me, and obviously many others, was that distance and being physically dispersed while collaborating seemed to matter very little. Open Source seemed to challenge a lot of the established *truths* in existing theory of management and organization.

The Open Source community consists of individuals, (sometimes groups of individuals), who voluntary contribute to a particular Open Source product or technology, and is distinct from a formal organizational entity. Corollary the experiences from that particular setting could not be applicable to any company trying to imitate its successes – or could it?

## *Conversations with software developers in the Bay and initiating the project*

The research project that constitutes the basis for this thesis began in 2001 when I participated in an Open Source software conference San Francisco, California. Proponents of Open Source development were courting business organizations trying to convince them about the advantages and benefits of sharing software. Their main message to the participants was that Open Source entails business potential. The conference was hosted by the Open Source Initiative. The conference was chaired by

Steven Weber who is a well-reputed political scientist,[17] and who also have authored interesting books and articles on the topic of Open Source software development model, Open Source history and origins, as well as contributed to theory related to Open Source community rationale. Also present were representatives from the Open Source community itself giving speeches, e.g. Brian Behlendorf alongside with representatives from major corporations interested in understanding Open Source development and its business potential.

At this point, I had not formulated anything resembling a relevant research topic, but I had an inclination that I could at least understand some of the challenges that practitioners were struggling with, if I could only listen and watch closely what they felt were urgent issues to resolve and discuss.[18] Belonging neither to the community of engineers nor the community of business people I felt like I was witnessing something really intriguing. Why was the hacker community interested in building relationships with business and vice versa? During the conference I had the opportunity to listen and observe software engineers from different organizations. They seemed to share a common interest in the benefits of Open Source software, and in particular the process of opening up the development process in order to better capture and harness the benefits of knowledge sharing in corporate software development.

Initially I had an idea to watch and observe only and try to stay as open and close to the ground as possible. Coincidently I started a conversation with a senior scientist[19] from the Hewlett-Packard Corporation, who told me about the company and their strategy to open up the development process, and how they had plans to initiate projects showing resemblance with the Open Source model, only slightly translated into their own business context.

[17] He is a member of the Berkeley Roundtable on the International Economy and a Professor at the University of California at Berkeley.

[18] I will be forever indebted to Professor Pierre Guillet de Monthoux at Stockholm University who granted me money to attend the conference!

[19] Dr Pankaj K. Garg.

In view of the fact that HP were still very early on in their process they were also interested in documenting their efforts. And while this happened to coincide with my overarching research interest, we discussed the possibility of setting up a research project.

When I returned to Sweden, I wrote a research proposal based on what I knew about software development, Open Source and organizing. The proposal went back and forward, a couple of times between me and the senior researcher at the labs. Shortly thereafter it was accepted by management at Hewlett-Packard Labs in Palo Alto, California. The proposal resulted in an invitation to participate as a guest researcher to document the introduction of a new development paradigm within the company, *the Progressive Open Source*.

This became the beginning of the empirical study, which covers four particular programs, all labelled under the same umbrella – the POS. In the following section I will present POS, information about the project, and some considerations regarding research design.

## Uncovering the thesis, collecting and interpreting the data

### *Progressive Open Source the research project*

Progressive Open Source (POS) relates in this study to four separate initiatives or programs. Why put the four under the same umbrella? There is a point in separating the respective networks for some practical and analytical purposes. But more importantly, I choose to use POS as a common denominator, simply because it was chosen by the organization and some of its dominating actors on the outset of the translation process.

As already mentioned, the new software paradigm that HP was establishing embraced several separate programs, where the Collaborative Development Program was by far the largest. It had already in 2001 close to 4.000 developers belonging mainly to the Hewlett-Packard Corporation. A significant number of third party

developers were also becoming part of the program, and they were growing by the number as the program rapidly expanded in 2001 and 2002. This program, the CDP, was initiated by the Image and Printing Group in Vancouver, Washington.

In addition, also included in the study are three other programs that were significantly smaller than CDP. They deviated slightly in their respective goals and also in terms of desired level of openness.

The Corporate Source initiative was initiated as a completely grass-roots driven project which aimed mainly at facilitating sharing within the company. The Cooltown project and the Bluestone project aimed at complete openness using similar strategies as advocated by the Open Source community.

Hewlett-Packard agreed to support the research project and gave me access to resources and contacts. They also funded part of my travels and expenses since I wanted to travel to meet people out of different geographical locations. In addition I received a grant from the Wallenberg foundation that made it possible for me to stay in the US for a total period of 9 months.

The first visit was for approximately 7 months between October 2001 and April 2002. An additional shorter trip was made in August of 2002. And finally I made a trip in May 2003 where I met with some developers and also presented initial results at the WTO Colloquium at Stanford University.

Given the scope of the project I was about to study and its complexity, I had to contemplate on the research approach. For obvious reasons, there are a number of methodological approaches available for such a project. I could have chosen a quantitative approach, investigating and comparing structures and effects. Only as it turned out, given my initial experiences and pre-understanding it at that time seemed more appropriate to choose a qualitative approach. According to Pettigrew (1990), the research design is contingent on the problem being investigated and prior knowledge and theory. Since the adoption of Open Source methodologies in large corporations was a new phenomenon at the outset of this project, I decided that a longitudinal field research approach was appropriate.

I had initially a broad and open aim of the study, since I was interested in making available an in-depth examination of specific events and processes. I had access to a single company, HP, and had funding enough to stay with the project over a reasonable long time, which made it possible for me to explore a relatively new research territory. According to Handfield and Melnyk (1998) case studies may contribute to theory building, as well as testing and/or extension and refinement of such theory. Theory in this sense is viewed as a process rather than something static already perfected that serves as a baseline for verification and testing, (Glaser & Strauss 1967). Researchers conducting case studies (cf. Van de Ven, Angle & Poole 1989/2000) often remarks and emphasizes the iterative process of research and also admits to its sometimes untidy character. Often research as in this case, begin with a broad definition of the research problem, which as the research progress, continuously becomes sharpened. For me, the research problem evolved as I collected the data. And also as a result of becoming increasingly more familiar and up to date with research pertaining to the field I had chosen to investigate. Moreover, patterns evolved through discussions with my Professors, with fellow students and in the course of having conversations with practitioners.

## *Research method summarized*

I conducted ethnographic inspired studies at different Hewlett-Packard sites mainly between October 2001 and April 2002. The study was conducted during the initial / start-up phase in the implementation towards a more open system of collaboration, i.e. the introductory phase of POS within HP.

The approach encompass semi-structured interviews, participatory observations in meetings, both at the physical locations as well as virtual meetings, e.g. using net-meeting, video-conferences, e-mail and telephone discussion.

I was assigned a cubicle on the premises of HP Labs in Palo Alto, California, and was granted unlimited access to meetings related to the initiative, meetings notes and to the software infrastructure itself. I collaborated closely with developers and managers involved in POS. I also traveled to different HP locations in the United States; Roseville CA, San Diego CA, Vancouver WA, Boise ID, Trenton NJ. I

conducted interviews at all these locations including internal as well as external developers and managers. I also interviewed representatives for the infrastructure provider CollabNet in San Francisco CA.

This has granted me unique access to corporate data in an organization where no earlier studies in the same area have been conducted.

I spent the first months on the premises participating in meetings with developers, managers, human resource officers and POS change agents, and collecting artefacts such as written material related to POS. I also had informal meetings and conversations with different members of the organization, mostly within HP Labs. Metaphorically speaking I breathed nothing but HP for approximately two months. And it was not hard at all to spend all my time in a research setting such as HP Labs. (In fact it is hard not to be enthusiastic in such a place.) Subsequently after having completed an initial analysis I constructed a research template[20] based on the knowledge and experiences I had gained so far.

My plan was to interview developers (internal as well as external), managers and POS initiators. For practical reasons, I decided to limit my interviews to 52 interviews. They were all conducted using the same template. The template served only as a point of departure for the interviews, and the interviewees were allowed to deviate and elaborate on the topics as they felt appropriate given their respective experiences. The interviews lasted for approximately two hours each, and they are all tape-recorded and later also transcribed. The participants in the study were all granted anonymity. In addition, I made extensive field notes.

## *Coding the data*

The interviews have been coded in NVivo which is software for handling qualitative data.

[20] The complete template is found in the Appendix.

Through the research process I have continuously gone over my material, (i.e. interviews and notes) and categorized the material in different ways, in order to grasp the overall theme of my thesis. I have used NVivo as a way of categorizing my data, in large themes and then structured issues in the format of trees and sub-trees. In information systems, such as NVivo, node is the term commonly used for a place in an index. 'In vivo' is a term borrowed from grounded theory. It refers to categories that appears in the data, i.e.," well named by words people themselves use" (Richards 2005:95).

Since I had access to extensive data, it was necessary to organize it in categories in order to gain speed, reliability and efficiency. Coding passages in texts is helpful since it enables you to find and to recall ideas easy without distracting and detracting the thought process. It also helps establishing reliability and meaning in the texts. Moreover, it enhances the ability to develop many ideas and establish relationships between ideas without losing or confusing them.

I have used qualitative coding for the following purposes in order to:

- Be able to reflect on what the coded segments of the texts have told me about the category, and its meaning in the project.
- See how the category relates to other ideas that have appeared from the data, and to see if I could construct theories about how they relate.
- Be able to collect all material about a specific topic, from all the different sources I have had at hand, and to be able to distinguish between the different categories of interviewees that I have included in the study.

The coding process has involved three different types of coding. I have used descriptive coding to store information about the interviewees, e.g. what kind of job they do, how old they are, what kind of education they have, and years within the company. Thereafter I have utilized topic coding in order to establish what topics are being discussed in each passage of the interview, e.g. face-to-face interaction, medium or issues related to control. Lastly I use the term analytical coding to distinguish coding that requires interpretation from that which is more descriptive.

Analytical coding is a process that requires interpretation and reflection on meaning. I have created categories that express ideas about the data at hand.

The coding process was followed by a process of browsing through the coded texts. I did that in order to look for differences, e.g. why two people had such different views on a particular topic. I also looked in the same way for similarities. The overarching goal was during the research process to construct a broad thematic presentation of the collected data, and try to link it to theoretical and empirical findings across the existing body of knowledge

## Concluding the thesis

As is true with most qualitative research, it was hard to know from the outset where and how this project was going to end. Starting the project I had framed a general statement of the purpose of my research. As the project progressed it was necessary (more than once) to firm up the picture of what I was aiming for. The goal has been to reach beyond describing the research project but also to include an analytical dimension, I aimed at answering the "why"-questions as well. I also wanted to make some kind of theory out of the research. Not the kind of grand theory that C. Wright Mills is speaking about, but rather theories that are particular to the substance of the data I have collected, only I can also hope that they may be useful to other research. Grounded theory writing makes a distinction between formal theories which has a high level of general applicability, whereas substantive theories are more particular. In the end I wanted the following pieces to fall in to place in order to meet the goals of the project. I wanted to:

- Be able to answer my research question
- Offer analysis – not just a description of a case
- Produce at least a new local theory

In addition I wanted to make sure that:

- What I finally produce is at least a little bit more than others have already produced

- I could account for my data, i.e. I want to give the reader a sense of what is going on in the data I have collected.
- At least some parts should be useable, in the sense of allowing someone else to be able to do something with what I have produced.

## *Managing the degree of involvement*

Research is clearly a social process and it is very easy to get too involved when spending long periods within an organization. In literature, the remedy for becoming too closely involved is to alternate between fieldwork and time spent outside the organization reflecting (Pettigrew 1990). However, for practical reasons, I had to conduct the empirical part of my investigation over a period of seven months, without any significant breaks. After the initial visit, I went back to Sweden and had a chance to reflect on my experience before going back again in August 2002, where I met again with POS developers and managers to discuss some of my initial results.

A similar phenomenon is related to the not uncommon experience of researchers in the field, to be held hostage, i.e. become prey in various information games that always goes on in any social setting. Goffman (1959) describes this using a theatre metaphor. In the performance (i.e. the process) spectators and players interact, constructing a *presentation*, for an audience (in this case the researcher). The way in which the individual present him/herself and his/her activities affects the interpretation and the understanding of the researcher (Goffman 1959).

The researcher is metaphorically speaking brought on stage in an information game, and the stage is set by concealment, discovery, false revelation, and rediscovery. Performance is then defined as: "all the activity of a given participant on a given occasion which serves to influence in any way any of the other participants" (Goffman 1959:15). This suggests that fieldwork and drama are related. I guess it can be and must be. I don't immediately believe that it is wrong to be empathic as a researcher and to some extent be moved by your experience. But such instances of close involvement preferably are dealt with by making room for reflecting, even analyzing if you will.

The way I dealt with the issues of becoming too involved was to have regular meetings with my Professor, Guje Sevón, over the phone, discussing and reflecting. I also used the opportunity to frequently visit Stanford University, SCANCOR.[21] Meeting other doctoral students and Professors there, gave me a chance to put some distance between my self and the experience I had at HP.

## *Ethical considerations and reciprocity*

Reciprocity is a guiding principle of this work. When granted access to Hewlett-Packard I agreed to participate in workshops and seminars with the people involved. I have also made explicit agreements about publication, confidentiality and consent of participation for all of those who willingly shared their experiences and insights with me. All interviews were taped and transcribed. However, all the interviews remained confidential and have only been used anonymously in the thesis.

During the process of collecting empirical data I also wrote articles with members of the research team at Hewlett-Packard. Those articles were published by the company as tech-reports, and were also presented at conferences. In addition I have presented results to members of the organization in workshops at their request. I also plan to give additional feedback after the publication of this thesis.

## *Research as bricolage*

This report as a text is a bricolage. Interpretation of data involves to a certain extent a bit of tinkering. We (the researchers) borrow things from different places and try to create coherence, i.e. understanding. In this case it means making use of as much as possible of all the intimate knowledge and resources that were presented to me.

I went 'native', i.e. I went to HP and become part of their tribe for a while. I spent basically all my time at the premises, studying, eating, laughing, and listening to all the smart people. Being at the lab was a treat for me, would be for any researcher I can imagine.

[21] The Scandinavian Consortium for Organizational Research

The premises in themselves, a lot resemble factory buildings, (at least from within), e.g. the floors are neatly divided in to endless rows of small cubicles. Every researcher at the labs has his or her own little cubicle which initially seems to leave little room for privacy. Or perhaps you adjust your voice and your behaviour to fit the situation. Still that initial sensation of being watched (or overheard) at all times goes away very fast. You were only interrupted on special occasions, e.g. when the CEO (at that time Carly Fiorina) was giving talks over the loudspeakers... That happened now and then because at the time when this study was conducted the organization was suffering from a major struggle between Carly Fiorina and the family scion Walter Hewlett over how to regain some of the innovativeness and former glory of the company. As a result the CEO was frequently giving her side of the problem to 'us[22]' – the worker bees.

I remember that I could not believe how such a place could contain so many creative people... or was it in spite of it? I can't say. I just know that it was a place with professionals from a great variety of nationalities. Still nationalities seemed to matter very little. To be a foreigner was the norm. Yet they were homogeneous in other aspects. Most of the researchers were men, although there were also some women. The women in the lab were mostly administrators or human resource officers. A majority of the researchers had a PhD degree in engineering, maths, physics or similar. What seem to bring these people together were brains and a huge interest in technology. And while they were at it – they seemed to have fun.

Altogether I never felt a stranger. I was always invited, and there was always someone who was taking an interest in me and my work. I don't think I ever went to lunch by myself. But, more often we went as a group, a group of people sharing similar interests. And there was always work related discussions! But, that was only a sign of their respective interests I think. The culture itself evoked interest in science and technology, and it is difficult not to be caught up in that!

When I arrived at Hewlett-Packard I was carrying with me (metaphorically) a set of techniques to study and write down 'thick descriptions' (compare Geertz 1973).

[22] On such occasions I and them, became us.

This require - as I have mentioned before – observing, listening, and taking careful notes. But in the end, I had to trust my own instinct. But to state the obvious, as it happens in academia, tinkering is not something you do completely on your own. It's a system consisting of superiors and peers, providing feedback and criticism. I did the best with what I had, and created this *thing*.

The research process as such conveys instances where information becomes put together in new ways, i.e. as in this thesis *my* ways. The bricolage as a construction then reflects patterns of *my* thinking. I can only humbly agree with Nigel Barley (1983) that the justification of academic endeavour (not entirely but to some significant extent) not lies in the contribution to the collective, but being closely related to one's own individual development.

> Like monastic life, academic research is really all about the perfection of one's own soul. This may well serve some wider purpose but is not to be judged on those grounds alone. This view will doubtless not sit well either with conservative academics or those who see themselves as a revolutionary force. Both are afflicted by a dreadful piety, a preening self-importance that refuses to believe the world is not hanging on their every word (Barley 1983:10).

I cannot claim to speak for POS as a whole. Still some of you who participated in this study may recognize your own voices in this thesis. But what I have tried to do is to present an assemblage of the observations that I collected during the time I spent at HP. And even more, I have mixed your voices with theories and other research that I had at hand. My motivation is that such a collection will somehow reflect an image of all the ideas and experiences I had while exploring POS. Will it make the world of developers and software engineering more understandable? That, I will leave to you - the reader.

# 3. About Progressive Open Source

## and how it all started

In this chapter I attempt to present a picture of how POS was constructed as an initiative within Hewlett-Packard. Since this study was conducted during the start up phase of the POS I consequently will provide my interpretation of some parts and aspects that pertains to the initial process of translation. I will also give a retrospect on how POS was constructed through the testimonies of the early adopters of POS.

As suggested by the word translation, I (in this chapter of the thesis), am concerned with tracing the transformation of software development practices within HP and how the *hybrid* POS came into being. In this context the word hybrid suggests that POS may be perceived as a mix between traditional software practices and novel one's. Through chains of translations global ideas influence local practices, thus the innovation translation model is perceived as movement through time and space. According to Latour (1993) the world is full of hybrid entities containing both human and non-human elements.

Ideas in the hands of people take on different trajectories. As proposed by Latour (1986) people modify ideas, deflect them, betrays their original intents, add on to the ideas, make the ideas part of their own respective contexts, or even let ideas rebound. Point taken here is that in light of this context: ***hp invent***.

Following in the footsteps of the sociology of translations convey exploring how phenomenon such as POS, emerge and come into being. It also transmits exploring

how such phenomenon are constructed and maintained, and moreover how they become and are made further durable over time (Callon 1986; Law 1991; Law 1992; Latour 1996).

Latour (1996) advocate laying emphasis on how actors solicit other actors to become part of their world and how they confer visions, desires and motivations on these actors.

The essential point that I will argue in this chapter is that the introduction of innovative modes of organizing relies on a powerful enough constellation of actors to carry ideas through, and when and if it fails it perhaps reflects an inability of those involved in constructing alliances supporting its existence among the community of actors. As suggested by Latour (1996) getting innovation accepted involves having strategies in place that will create momentum, i.e. become interesting to others enough to make them follow those interests. It is a process of aligning heterogeneous approaches of local actors and teams of actors within the organization, e.g. in terms of work-routines, tools and organizational roles.

The introduction of POS was the starting point of a change process initiated as a result of an emergent need to alleviate some of the challenges HP were facing in terms of its software development processes.

## Context of perceived need of change

Research reported by Dinkelacker, Garg, Miller & Nelson (2001) indicate that software engineering continue to impose challenges for large corporations. While at the same time, several studies have evaluated and explored the successes within Open Source software systems, e.g. Apache, Bind, Emacs, and Linux (Tuomi 2001; Raymond 2001; DiBona, Ockman & Stone 1999; Wayner 2000; Moody 2001; Moon & Sproull 2002). These studies show that a large community of contributors typically develops Open Source systems in a joint effort facilitated by communication through electronic newsgroups and mailing lists on the Internet. Many Open Source systems seem to have progressed with voluntary resources, i.e. without assigning work to

specific individuals and without an explicit system-level design activity (Mockus, Fielding & Herbsleb 2000).

The Web and Linux challenges traditional software development and its success suggests that it may be more efficient and competitive to support development in an "Open Standard" way, i.e. any protocols or software interfaces are discussed in an open discussion forum with both the users and developers actively participating or monitoring the discussion. Once the "standard" has been defined in this manner, anyone can build supporting software tools to work off that standard. For example, once the HTTP protocol was defined by such open discussions, different groups of software developers could develop the clients and servers independently to complete the development of the World Wide Web.

Out of this context large companies, such as HP, are increasingly seeking to leverage from their capacity to generate knowledge while processing as well as managing information. This is a process of including a selective but yet global workforce capable of working on a planetary scale in real time which has been highlighted by Castells (2002).

Consequently the POS enterprise is networked in the sense that HP chose a strategy of shifting alliances and partnerships for different projects, thus seeking to speed up the process of innovation and overall performance. Innovation then seems to be considered a major competitive advantage for the business and the organization. This seems to pertain to many organizations today, and is supported by extensive empirical research which indicates that innovation is dependent upon the development and management of knowledge (Christensen 1997; Zeleny 1989; 1994; Sevón and Kreiner 1998).

In the following section I discuss Progressive Open Source and its implications for a large corporation.

## Progressive Open Source

Dinkelacker et al. (2001) define "Progressive Open Source" as a strategy for large corporations to adopt Open Source software development methods. In essence the concept encompasses coordinated resource sharing and problem solving in a dynamic, multi-actor virtual network. The hypothesis is that by adopting Open Source development methods within a corporation, the corporation can gain from the collaboration styles of the Open Source software methodology resulting in robust code quality, features that are well-tuned to user's requirements, strong, well-established networks of communities of practice, and so forth (Brown & Duguid 1991, Chaiklin & Lave 1993, Lave & Wenger 1991).

Progressive Open Source advocated the progressive adoption of Open Source practices by a corporation in primarily three stages. Inner Source referred to the stage in which the software source code was open only of employees of the company. Controlled Source referred to the stage in which thes source code was open to selective third parties and partners. And Open Source referred to the stage in which the source code is open to the entire Internet community.

Each one of these stages related to varying degrees of openness. The projects that I have studied have chosen slightly different approaches. It was evident that:

1. Corporate Source was initiated as an inner source project
2. Collaborative Development Program (CDP) was initiated as a controlled source project
3. Bluestone was initiated as a controlled source project which aimed at becoming a complete Open Source project
4. Cool Town was initiated as an Open Source project

In the next section I will give an historical retrospect on how and why Progressive Open Source was introduced.

## The Sirius community - a starting point of POS efforts

As early as in the timeframe of 1994/95 it was acknowledged that a new model of organizing development was needed within HP in order to cope with the challenges of the market. The demand was to develop new features, faster and to a lower cost.

In response to the acknowledged demands of the organization, The Sirius community launched a particular project, The Owen Project, which in many ways became the starting point for what was later denoted as POS development within Hewlett-Packard.

The Sirius community started experimenting with new practices adopted from Open Source methodology. And since the Owen project was an immediate success it was inscribed as a prototype model for development, i.e. inscribing in detail programs of action for the developers. It was later utilized in the larger corporate initiative which embraced the effort to open up processes for collaboration.

Openness and sharing were to become a more stabilized approach encompassing not only research but also development of new products. And in this respect the Sirius developers were early translators inscribing their visions about how to conduct development, e.g. in written reports that were presented internally within the organization as recognized best practices.

## *The Owen Project - a prototype for POS*

In this section I will provide a background on the Sirius community and a particular project – OWEN – which became a starting point of POS development.

Sirius is an architectural framework for co-operating database systems and it is also the name of a community of collaborating developers within HP that utilize the framework for firmware reuse. The Sirius architecture was originally localized within one organization, the San Diego division.

The San Diego division had made a name for themselves in the large-format plotter business. Being a very successful business unit they had practically defined

the field of competition for an extended period of time (more than 15 years). It was recognized as an important community in 1994/95 when the company made a decision to centralize the generation of print engines.

Already in 1992 the business had significantly matured and the large-format business was re-located to another business division in Europe. As a result, the Sirius team had to re-invent their business. Big changes were in store for the division which encompassed abandoning the large-format plotter business entering the fax and later on the multi-function peripheral business. The products they used to develop belonged in the high cost/profit low volume niche market whereas the new products belonged in the high volume consumer market which had considerably lower margins for costs and profit.

The goal of the organization was to make it possible for multiple divisions to reuse and apply the print engines into their respective product development, e.g. photoproducts and other multifunctional products. In order to enable reuse, it was perceived as necessary to develop a new firmware platform only the desired schedule was extremely aggressive.

The situation (as recapitulated by management of the division) was that a large firmware team, which had low overall experience in the new product category, had to invent a new platform architecture design, development and implementation, in practically no time.

In the past (before 1994 timeframe) the development process was less formalized, e.g. oral history and experience prevailed which conveyed inadequate and incomplete documentation regarding previous projects. Management urged that the development process had to be improved considerably.

The Sirius team was looking for a silver bullet which would alleviate concerns about structure and project specification as well as introducing functional verification while maintaining intellectual control over the products. Initially, they looked

internally and found an article in the Hewlett-Packard Journal by Grant Head about how to use Cleanroom Software Engineering Techniques.[23]

The San Diego team invited Grant to come and share his experiences. Only, it was recognized that the scope of Grant's project was significantly smaller than the challenges that the Sirius team were facing. They found themselves in a situation were they basically had to step back and acknowledge that a business-as-usual approach to face the critical tasks ahead was doomed for failure. What to do?

After a period of en masse meetings with extremely slow progress it was decided that a smaller group was to assume responsibility for developing the new architecture. This smaller group went on to develop a set of architectural guidelines and an architecture infrastructure that was clearly defined and documented. Once developed it was reviewed by the entire team.[24]

The review process was soon acknowledged as being critical to the team in terms of training, and especially the value of peer review (very much the same as in Open Source).

Having at this point extolled some practical guidelines and an architecture the management and the team decided to establish an interconnected core team. A structure was set up in order to manage the large amount of knowledge that had to be communicated and internalized to the entire team.

Serial communication between all the project members were not perceived as effective given the scope of the project and the speed of the development that had to occur. Instead, a core team constituting of seven of the most experienced and architecturally knowledgeable firmware engineers were selected as responsible to lead the specification of one of the seven critical components. In addition, each of the seven also was assigned as members of two other component specification teams. Ownership and co-ownership of components made the teams interconnected.

---

[23] Six-Sigma Software Using Cleanroom Software Engineering Techniques, June 1994 Hewlett-Packard Journal

[24] E.g. Gilb-type formal inspections, Software Inspection by T. Gilb and D. Graham, Addison-Wesley 1993.

Each of the core teams would meet on a regular basis to define and document their respective component's function, its interface, and its requirements of one central component. Soon the teams were showing great results.[25]

As the OWEN project was proceeding it was documented in order to capture objectives, processes and methods. The process of documentation worked as a reminder of the intention to the team and also helped facilitate communication between the engineers, especially as new members were added to the team. Some interesting observations were made regarding the core team structure and the peer review process.

The perceived positive aspect concerning the core team structure was that the chosen approach seemed to work extremely well for communication and team learning, according to the team members and management. Their experience was that three persons in each core team seemed to work particularly well, e.g. schedule coordination was minimized and the teams could run concurrently. Also, according to their experience the odd size of the team reduced the risk of splitting the team over an issue.

The perceived negative aspect were that they also reported that the core teams were less effective in terms of developing and defining processes for the larger teams, which may indicate that power position has an impact, i.e. engineers had problems enforcing process on peers.

Another interpretation is also possible – it seems as if it is important that those being affected by the process are involved in the definition of the process.

In terms of the process development it was acknowledged by the Sirius team that there must be a clearly spelled out reason in order to embark on a process of improvement. And this reason must be completely understood by the entire team.

[25] Quantitative metrics were collected prior to each review. The participants recorded e.g. preparation time spent inspecting the work-product. Also metrics were taken in order to measure length of meetings, number of items logged, number of issues etc. After the meeting they typically would measure number of verified major defects (e.g. defects that could result in customer failure), total rework time etc.

According to their experience it was perceived as crucial that a majority of participants and at least a key leader buy into the process of change.

The Sirius project was owned by management. The reason for this was (as mentioned before) that it was perceived as difficult for peers to impose changes in the process on peers, and in addition it was perceived as downright inappropriate for peers to measure each other on performance.

In order to have efficient reuse it was perceived as necessary to have a common architecture and a firmware that was flexible enough to support the unique business needs of each of the using divisions. The aim was to have one organization for this platform and leverage them into other printers. Very early, the teams were showing promising results. They had achieved effective collaboration across the IPS organization.

> Probably the best success story was firmware product development. So we had; and I was actually lab manager on one edition across the street. We had firmware modules that were developed for all-in-one and firmware modules that were developed for printers. And we basically got the team together and Kathy was instrumental in getting all of us pulled together. And we migrated to a common firmware architecture that we could use across all of our consumer ink jet printers, so the Vancouver team would be working on certain modules for driving the pen and servicing the pen, and moving the carriage axis back and forth. The all-in-one folks would be working on modules of the user interface and fax modules, and the telephony and all that kind of stuff. And the photo printer team would be working on all the firmware that were driving connectivity for photo card slots and image print things. So that's kind of the first major, from my perspective, collaboration across IPS. And the firmware was probably the biggest success story.

A centralized development approach such as pursued by the Sirius team additionally presupposed having a standardized way of managing shared sets of assets that everybody could contribute to, and also withdraw from. Consequently a designated team was in charge of coding standards.

The designated team was assuming the role as housekeepers in the community. They didn't focus on standardizing on the blocks but rather standardizing on the interfaces which made it possible to link efforts.

A core team was formed in order to manage the different efforts. They decided that the development process should be split up into more manageable pieces.

The core team also recognized that it was not desired to let the respective pieces be unconnected from the architecture and as a response they came up with the idea of an interconnected core team in charge of identifying the components of the architecture. The interconnected core team was also made responsible for assigning the components across all the developers. One team became responsible for cleaning up the architectural framework.

The initial matrix was composed of approximately thirty odd components. Members of the core team assumed multiple responsibilities for the components.

The basic idea was to avoid developers to become disconnected from the architectural intent. Developers should be free to design their components while having the architectural intent in focus. The developers had cross-accountability and responsibility between components that were interrelated. And as a result the information flow between the groups was perceived as very fast.

> And because those other two people, or two other core; were members of other core teams and also leaders of their own core team, very quickly the information flowed throughout the team, so you didn't need to; so that information flowed really quickly. You could get feedback back from the group really quickly. So you didn't have these regular all-hands meetings where you flowed information one way, from somebody coming up with this thing and everybody else taking notes. The information flowed throughout. That isn't to say that you didn't have regular meetings where something that was particularly valuable that might need some training or some architectural clarification, so we did have those things too, but the day to day learning's really flowed through the groups really quickly.

The Sirius community had given evidence that the new collaborative model of organizing development had produced:

- Faster flow of information
- Faster transfer of innovation between teams
- A coherent overarching view of the architectural intent
- More responsibility for each others contributions because developers were vested in more than just their own work
- The development model that the Sirius team was propagating constituted of a three-legged stool. More specifically it encompassed of:
    - A common process
    - Converging tools – at a minimum effective bridge tools
    - A common architecture

The Sirius community resolutely was supporting a monolithic view on collaborative work in the firmware arena and they also believed that such an approach would successfully help the collaborating teams to meet their business needs. As a result of the proven successes of the OWEN project the Sirius team became an important inspiration for POS development within HP both in terms of collaborative software development.

The experiences of the Sirius team were documented and as a result the approach was imitated and translated throughout the organization. Consequently it became both a prototype and a standard that thereafter was translated by other groups within HP. And as such also a process of inscribing patterns for future development.

## *Aligning interests persuading others to join*

The social process of introducing collaborative development involved both technology and organizational change. And as the OWEN project was evaluated The Sirius community also realized that they had to start thinking significantly more about social aspects of how to share in a multiple division environment.

How to move ideas about collaborative development outside the immediate sphere of the Sirius group?

Consequently the front players of the Sirius community as you would expect stepped up to assume leadership trying to influence the POS paradigm moving ideas on openness and collaboration forward.

The Sirius community went on to form a team with sponsors from each of the collaborating organizations. They got together on a regular basis to build the collaborative culture and also resolve issues that helped them to get the work done. The team had basically two roles:

The first was assuming the role as advocators or vanguard ardently influencing developers to collaborate and sharing information setting up an efficient communication process, defining roles and responsibilities.

The second role involved propagating the benefits of creating products that work together, starting with the print engines.

The collaboration on firmware became very successful and after some time HP made a decision to try taking it to the next level across the organization, i.e. established common software elements that worked across all the product platforms, e.g. common print engines, common firmware, common software etc.

The aim on the outset was to leverage and reuse software modules. But, it very soon evolved to encompass understanding cultural issues associated with the collaborative development process. E.g. it was perceived as an immediate need to influence developers to start trusting turning over responsibilities of the design work to each other. The goal then further expanded to involve moving people beyond the initial element of fear.

The Sirius teams advocated that the concepts that they had developed were applicable not only in the area of software firmware, but also in the mechanical and electrical domains. And as they proved significant positive results managers became

very thrilled. They were labeled "The Sirius Dream Team" by managers. And consequently their influence on future development processes was significant.

The members of the Sirius team became important actors and proponents as the initiative was progressing through sequential chains of translations, increasingly refining their concepts, articulating and documenting their efforts in internal reports. The interests of the Sirius actors were in this respect translated and inscribed into both technical documentation and also through the establishment of a preferred mode for social arrangements in terms of collaborative development.

When such process of translation and inscription is proceeding successfully, heterogeneous interests are increasingly becoming aligned moving towards a more stable actor network (cf. Callon 1991; Akrich 1992).

Their motto through this process became the organizational motto. It was translated into: *Speed over Secrecy*. This catchphrase basically encapsulated the idea, that it was perceived as better to develop new products at a faster pace than to be able to patent every step in the process. Decidedly technology was perceived as becoming obsolete at such a fast rate that it was better to be swift and agile rather than relying on patenting as a way of gaining competitive advantage.

## *The Sirius community and the introduction of POS*

Sirius development had established interdependent relationships between the different teams meaning that they had a process that supported co-development of moving changes back and forth. As it turned out the movement of the innovative ideas of the Sirius team in the hands of the other teams took on a different trajectory, it was modified, added on to and some features of the original ideas was dropped.

As has been proposed by Latour (1986) getting an innovation accepted calls for strategies aimed at the enrolment of others through making the ideas interesting to others, persuading them to follow our own interests.

In at least one key aspect the Sirius team failed to influence the process of translation, e.g. when POS development was introduced the Sirius developers were

left in receivers positions only. The interdependent relationships between collaborating actors had previously been supported by converging tools, or at least supporting effective bridge tools. The successful communication of the past started to break down. When the interdependent relationships were not upheld people started losing interest in sharing important information of bugs, which paradoxically lead to a situation where collaboration actually broke down as a result of the introduction of the larger collaborative effort, the POS. Ideas pertaining to development processes took on a different trajectory in the hands of the larger initiative. And The Sirius team was starting to lose influence.

> Yeah, again CDP tools had nothing to do with that. The process had to do with that. And the breakdown in the collaborative process. But San Diego had some responsibility for it, and Vancouver had. But that hypothesis that I was talking to you about was when we were mutually-dependent, we were really good. Really good. When we became kind of service based, we weren't very good. We started to lose some of the capability that we had. So this gets back to this core team. They worked when they had the core team people. I saw that in my work. It directly affects them, and by the way, their work directly affects what I'm doing. So we have an inter-dependency.
>
> [But here all of a sudden you lost sight of the overall picture.]
>
> So this happened, there were several things that happened. Remember San Diego had this merge, so we were in a state of leaderlessness, with respect to distance and process. What else? We kind of finished that inter-dependent model, so there was another thing that happened.

When I interviewed key players of Sirius they proposed that it was the perceived lack of interdependent relationships in conjunction with lack of support from management that turned the Sirius teams away from the successful collaborative model of the past.

Suggestible as the chains of translations were proceeding, the new hybrid POS was started to be perceived as a betrayal of the original intents of the Sirius team. And metaphorically speaking it had started become "the son of outrageous conduct", which in the ancient Greek language used to denote sheep-goat chimeras.

As POS was introduced more widely introduced as the preferred mode of collaborative software development the Sirius management team no longer liked to ride the change.

In their opinion, this was particularly related to do with the introduction of a common, standardized toolset, abandoning the original intent which had been to build effective bridges between different toolsets. As a result of the continuous chains of translations within the organization, the Sirius teams that were successful earlier on, found themselves in a situation where they perceived it as impossible to make contributions at all in the development process. They were cut off and their managers would not support efficiency losses as a result of introducing POS. This is described by one of the pioneers:

> So at this point San Diego's saying if what's best for the overall community is to be on this toolset, this is now San Diego's problem. But it's a big one. It's a huge one. One of the things that when we talk about collaborative behavior, we say that we sometimes need to make decisions that are going to be locally difficult, but are going to increase the power of the community, right? Isn't that one of those things? So the question is, is this one of those times? This is the thing I think that San Diego's struggling with. Is this the time? And the community needs to move that way, and so the individual's going to have to figure out what they're going to do about it.

The Sirius teams suggested that it was POS tools, i.e. the particular toolset chosen by POS as the universal standardized toolset that jeopardized their business not collaboration per se. And the process of standardizing tools was leaving the Sirius teams out in the open. Consequently, this was creating annoyance among these teams. And moreover they perceived this as very ironic since they actually were the original pioneers or original translators if you will of a more open collaborative development process. This is described by a pioneer:

> This is my observation; we're not listening well. We as a community aren't. We as a community are not listening well to this very big problem. How can we be the black sheep when in our own mind we are golden sheep?

Looking at POS development in the light of the Sirius community is in fact realizing that business drivers for collaboration were present long before it was translated into an idea saturating the larger organization. POS is then more of an emergent process, i.e. an evolution of what the organization was already doing in 1994/95.

Whereas the Sirius community had a very strong cultural bent it did not share common tools. They only recognized that the tools could talk to each other, i.e. they developed bridge tools. The POS paradigm on the other hand presupposes common tools, and ironically this conveyed great difficulties for many of the Sirius teams.

## *How and why POS was introduced*

There was a vision that the company needed a collaborative environment within HP. Developers as well as managers had had practical experiences of instances were they were writing and testing their own tools while there were already existing tools within the company that they could not utilize.

It was perceived as important to build an effective community and communication and collaboration was a very important part of the problem, especially in the high processing material. In the high processing maturity everybody strictly have to adhere to their respective roles. Whereas in low process maturity you have to collaborate and communicate because it feels right or that you want to contribute back to your friends.

In addition, it was perceived as the most important factor to get people to communicate with each other and to agree jointly to use certain components and also to have organizational norms to determine how to work collaboratively.

The vanguard developers were inspired by Linux Open Source, since some them were participating in Open Source development in their off time from work. The developers were also very passionate about technology and were trying to glean good ideas from some of the projects they were participating in. The idea was to share files and to put files 'out there' for others to use. This is a developer:

> And I just find it fascinating just how all these people rally around a technology, share their ideas, and strengthen this up-project, and for me it was just fun. That is why I was really interested in participating in this project (POS), because I could see the value in the Internet, how people were using that, and how people were strengthening and building these really complex tools to solve big problems.

Another developer:

> I started learning about open source, and did the typical thing most people do, read the Cathedral and the Bazaar, started reading up as much as I could about it and said its very fascinating to see how we could bring this into HP, and how it could work inside of HP in such a structured environment of Hewlett-Packard, and such an open environment of the loose cannon web developers.

Developer and later also a POS manager:

> A discussion came up about how we need to find a way we can work better with third parties, and there was some bantering about how there is Open Source, but we have security needs that we have to make sure that we don't expose our IP. Lee Caldwell (CTO) said, we're going to have an initiative that is going to solve these problems, and that man is going to lead it. And that's when he pointed to Rob Miller. Rob at just having the idea that this is just one of his jobs, and he kind of looking around and it was a combination of both Open Source and just the issue around collaboration and being able to tap into the expertise around the business and the company. So he did some investigation on what other types of projects have been successful through collaboration, and he came across Sirius. Sirius-Firmware cooperative, and I was leading it at the time, and he asked for some information. I sent him all these slides and then I asked; 'why do you want it?' and that's where he pulled me into CDP. He just shared what CDP was at first, and I said: ' well, if you're going to use Sirius as an example, then I'd like to be involved' and the reason he came to Sirius was that there are very few others, if any, across HP. This was four different divisions working remotely; one in Vancouver, two in San Diego and one in Singapore and we all shared the same firmware code for our products. It started back in -95 I think and the first year it was just two products that come out and then by the time Rob started looking at it,

> which was the summer of 2000, we were having six to ten products a year off the same code base. We had shown tremendous benefit to the business; we had shown that it takes fewer resources to not necessarily to get more done, but to get the same done.

The POS leaders were using Sirius as a business case. Initially it was called Inner source, and it was focused on finding a way to take the Open Source tools and bring it within HP. Later they added the notion of bringing in third parties. It spun off into the idea of controlled source. It is fair to say that the ideas of collaboration and open source principles came from practical experiences of the developers, only the CTO of the company mandated the project and made sure it could the proper funding.

> So controlled source had this security idea that you can have a third party linked in, but they don't see all of HP, they just see what we let them see.

Rather than continuing in two separate lines, i.e. inner source and controlled source, the HP Image and Printing division launched the collaborative development initiative in the summer of 2000. They started by trying to influence lab managers across the laser jet site and the ink jet site to join the larger initiative. The Image and Printing division were working in each of those communities separately trying to educate (persuade) them on what the opportunity was for collaborative development.

The Sirius team belonged in the ink jet division and they were very excited. On the laser site they had shared code, but they didn't do code development. And also on the laser site there was a more prevalent Silo-mentality[26] and many control issues. The control issues on the ink jet side were more amenable.

Leaders of POS had to work harder on the laser side to raise the viability in order to get funding for POS and also to attract developers. The POS lead started off with a small team of a dozen of people and started on the tools to make sure they could collaborate. They started bit by bit to break down the barriers of getting to the information and sharing the information.

[26] Silo-mentality refers to instances where internal competition, lack of synergy, shortshighted solutions and poor communication prevails over collaboration and sharing between teams and divisions across the organization.

## *Short summary of the rational behind POS and the process of implementation*

The OWEN project to all intents and purposes became inscribed in documentation and thereby providing impetus for introducing change in development practices within HP.

Through the documentation key players of the Sirius team contributed to a strong narrative by elucidating a set of conditions fundamental for the success of the OWEN project. E.g. the key players argued that the introduction of a common architecture for the development projects were an essential feature. In addition they propagated that the architecture ought to be split up into manageable pieces and that the interconnected core team ought to be responsible for delegating components across the teams. The basic idea was to try to avoid disconnection from the architectural principles. Cross accountability for the architecture and cross ownership of the process was augmented in order to ensure visibility of the big picture thus facilitating collaborative development on a larger scale. Enrolment in the network then came to be contingent on a set of strategies for motivating company wide collaboration.

The strategy for implementing POS as a larger initiative across the organization, including trusted third party developers, effectively attracted top management support.

The OWEN project was at this point recognized as a positive model and the insights and experiences of that particular project was translated to fit the larger organizational initiative, POS.

Through the translation process the innovative ideas of OWEN were modified in order to meet the ends of the larger community. The larger initiative had to resolve three immediate needs; collaboration with third parties, managing geographically dispersed teams, and supporting independent teams collaborating on the same code base.

As with any idea that travel, translation occurs entailing both changes in terms of the original idea and also as has been argued by McMaster and Vidgen et al (1997) it entails losing sovereignty. This was evident as the Sirius team was struggling to retain control over their own development process whilst realizing that what had worked for them in the past now was abandoned by the larger initiative. The Sirius team found themselves depicted as black sheep while in the past (before the introduction of POS) they were considered golden sheep.

## *The four projects - We are ONE but we're not the same*

I will end this chapter by providing an overview of the projects that fit under the POS umbrella. I depict some of the in my view more interesting features of the four projects that were part of this study. They each represent different translations of similar ideas, while still representing images of the larger initiative that more succinctly encapsulate a variety of interpretations. I will particularly devote interest to Corporate Source and Collaborative Development Program since they also attracted most of my attention throughout this study. I will summarize the Bluestone and Cooltown initiatives.

### *The Corporate Source project*

The corporate source initiative was an inner source project. It supported the use of a flat, networked organization for software development across HP. It was introduced in response to the hierarchical form of organization that had been prevalent in the past.

As was typical for many companies in this timeframe, i.e. in 2001, HP software products were organized in hierarchies, either functionally or market-driven. Hence, a product group in the printer division writes all printer software, and the operating system group in the computer systems organizations writes the operating system software. The only connection between these two groups was through the Chief Executive Officer, who often was up to ten or twenty levels higher than the engineering groups working on the product.

The source code of the software from one group was rarely available to the other group. Hence, if there was a problem in the interface between the printer driver and the spooler on the operating system, there were several layers and channels of communication and coordination that had to be crossed to address the issues.

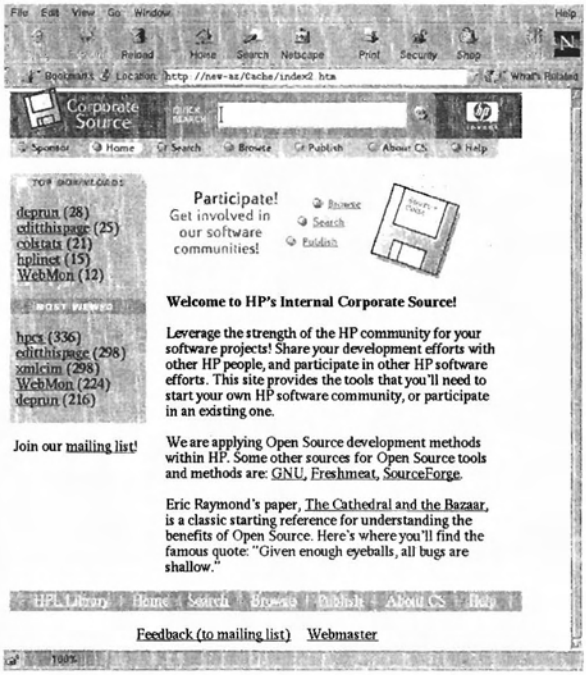
In contrast, Corporate Source advocated that the two groups (and all other groups in the corporation) should freely make available their source code among them. In this manner, the printer group should be able to make changes to the operating system spooler source code, and the operating system group should be able to modify the printer driver source code. The ownership and control of what ultimately goes in the product still rests with the original owners.

Apparently Corporate Source borrows heavily from the Open Source development paradigm (DiBona et al. 1999) and from the methods of scientific research (Kuhn 1962/1996). From the *Open Source development paradig*m, it borrows the notion of making source code available freely (openly) for all members of the community; propagating the use of open email discussions for feature addition, implementation, review and testing; and providing a persistence base for the source code and email discussions to be available long after their creation date. Following such an approach enables any new developer to quickly join a project by understanding the rationale behind some feature selection and implementation (cf. Raymond 2001). The World Wide Web (WWW) infrastructure was utilized to make Corporate Source projects freely available for all employees to browse through and participate, through a familiar employee portal.

From the *scientific method*s, Corporate Source borrows the notion of "publishing" work for peer-review and criticism, and archival storage of important experimental results for future review. Hence, the primary responsibility of facilitating the use of Corporate Source were resting with HP's research library, which was also the primarily responsible for maintaining and disseminating HP's scientific knowledge as technical reports.

The HP Research Library were hosting the Corporate Source service, along with some other Knowledge Management services, such as a database of skill set of employees, an Idea forum, technical reports, and so forth.

***Figure 1*** *Displays a typical screen of the Corporate Source service that provided a community hub for the users of Corporate Source.*




Members of the community could publish their own software, update existing software, search or browse through existing software, or comment, criticize or review existing software or discussions. Traditional hierarchical organizational boundaries were minimized by only exposing relevant information about a user's network identity and skill set.

Any given user's hierarchical position could, of course, be determined through some of the attributes of the network identity. Hence, one could not truly achieve a virtual network identity as in the case of the Open Source development, where any given user can completely hide behind a network identity. The nature of the discussions and the corresponding contributions to the software, ideas, and thoughts, therefore, most likely were different in Corporate Source than in the Open Source communities.

It was envisioned by the scientist[27] who initiated the initiative that the potential of Corporate Source encompassed realizing the potential of empowering junior members of the organization to make far-reaching and wide contributions to the corporations software, similar to what has been achieved in the Open Source communities, as e.g. when high-school graduates were able to shake up the software and media industry with the work of peer-to-peer computing of Napster[28] (cf. Clarke, Sandberg, Wiley & Hong 1999; Alderman 2001; Rose & Buchanan 2001).

### *The Collaborative Development Program*

The Collaborative Development Program (CDP) was an initiative emerging out of HP's Imaging and Printing group that develops the printers and related products for HP.

It was acknowledged that several printer products had relationships with each other and third party products, e.g. the all-in-one office jet has some features that were in common with a printer, while other features that were in common with a scanner, and so forth.

The source code for the firmware and software for these features, therefore, was expected to be shared between the groups that provide these products. Similarly, software for some of the foundational features, like networking, was common to all the groups. The goal of CDP was to foster appropriate networking, collaboration, and community spirit among the various groups that participated in such development, including third party individuals who were not employees of HP.

While the Corporate Source program was a grass-roots program from HP Labs, CDP was a group funded and organized program that had executive champions, sponsors, R&D change leaders, and information technology staff. The executive champions had been critical in establishing the credibility of the program and establishing a need for collaboration among the different groups in the organization.

---

[27] Dr Pankaj K. Garg at HP Labs.

[28] Napster is a protocol for sharing files between users. See e.g. http://opennap.sourceforge.net

The sponsors were critical in committing resources that enable short-term and long-term nurturing and success of the program. The R&D change leaders were enabling a dialogue of cultural change and training in the organizations to begin the long process of transition from a hierarchical, product-focused organization to a networked, collaboration-focused organization. The IT staff provided the critical collaboration infrastructure on a 24x7 supported basis.

The CDP infrastructure supported email discussions and bulletin boards, source code repository (searchable and browsable), and defect tracking. While the Corporate Source infrastructure was a home-grown combination of some available Open Source tools like CVS (Fogel 1999) and Mailman[29], CDP were relying on a third-party, CollabNet, Inc., to provide the tools infrastructure.

The CDP infrastructure was residing on the Internet (as opposed to the Intranet for Corporate Source). Hence, to bring in a new third party on-board the CDP infrastructure was a relatively straightforward task. Indeed, the time for establishing a new collaboration project were reduced to a matter of few days from what it used to be a few months or weeks at best. If all the project participants were from HP, then the setup took a few hours.

Each user in CDP was given a network identity, which was based on their corporate identity. The hierarchical organizational identity, however, was not that easily visible although it could be deduced quite easily. CDP organized people by projects and by default any HP employee were given read-access to any "open" project in CDP. CDP promoted the sharing of knowledge and information to build a community that would deliver on the priorities across the company. Ultimately the goal was to do away with any organizational boundaries to allow engineers to apply their expertise to provide the greatest return for the company by enabling project teams to deliver innovative solutions faster and with greater reliability.

[29] www.gnu.org/software/mailman/mailman.html

Both Corporate Source and CDP had been operational in HP for several months when I conducted my study. Corporate Source was officially launched in June 2000, and CDP was launched in April 2001, and continuous to be widely used.

Corporate Source had about 1500 registered users while CDP had 3500 users (10% of whom were non-HP). Forty-five external companies were developing projects with HP using CDP. Corporate Source had about two dozen projects, all of which were research projects that were not tied to any HP product. CDP had about 350 projects, most of which were tied to specific HP products. Corporate Source had users in forty-five countries; CDP had users from at least eighteen countries. Both community hubs had active users, although it was perceived as necessary to increase the awareness, adoption, and use of Progressive Open Source within HP at the time when I was conducting this study.

### *Bluestone*

Hewlett-Packard announced that it was to acquire Bluestone in October of 2000. Bluestone was at that time a leading provider of Internet software platforms, tools and technologies for business-to-business, business-to-consumer and mobile Internet transactions. The intention was to establish Bluestone's software as the integrating platform for HP's software offerings. Through acquiring Bluestone HP aimed at capitalizing on the growing middleware market.

However, already in 2002 HP decided to get out of the middleware[30] business, since they were never able to make significant gains. In 2002 HP software was at the bottom of the product listing on HP's web site. Software revenue dropped more 20 % as compared with the year before. The company as a result decided to cut off much of its software business.

At the time of this study, The Bluestone initiative were heralding approximately 200 core users, and close to 2000 light users. Light users were defined as those that are mainly participating in activities such as discussion boards. They typically were

[30] Middleware is computer software that connects software components or applications.

submitting remedial issues. Moreover, they did not have access to checking in and out source code for the purpose of making changes. They were mainly viewers and discussions partners with no formal impact on the development process.

Bluestone was at this time a controlled source initiative, but was ultimately targeting a complete Open Source approach. Since they were a fairly small group within HP with an independent past, they also wanted to retain control over their own development approach. Consequently they decided to invest in the same infrastructure solution as CDP, only they were fighting to retain control over their own initiative since they were very concerned with not having to compromise too much with the larger CDP initiative.

The motto of Bluestone was to make small an asset through ultimately establishing a visible presence at Apache, which was perceived as a natural outcome since they had been in long-term competition with Sun and IBM.

### *Cooltown*

Cooltown was a project initiated by HP in order to make technology freely available and Open Source under the GPL.[31] The project applied Web technology to develop systems that support the users of wireless, handheld devices interacting with their environment (Barton & Kindberg 2001).

They are (and were) a research team in a nomadic computing department which focuses on areas of web-based pervasive computing and mobility. They were at the time of this study a small community of approximately 15 specialized experts in a software centered environment. In order to get the Cooltown concept out of the labs and into practical applications, Gene Becker, director of the cooltown project, and Bruce Perens[32], HP's open-source and Linux strategist, decided to release about 150.000 lines of code under the CoolBase Open Source development project. HP also

[31] General Public License, www.gnu.org/copyleft/gpl.html

[32] Bruce Perens is the primary author of The Open Source Definition, the formative document of the open source movement. www.opensource.org/docs/definition.php

was collaborating with Carnegie Mellon and the Georgia Institute of Technology to help disseminate the Cooltown concept.

HP had been working on Cooltown for many years in the labs before deciding to approach the Open Source software community. However this was according to analyst Stacey Quandt of Giga Information Group an approach often utilized by companies for attracting interest to projects that are perceived as interesting but not really perceived as practical or useful.

> A lot of companies seem to keep throwing technology over the wall under the GPL and thinking: We'll have the community build this for us.[33]

The explicit goal of Cooltown was to create a software platform and release it Open Source, outside of HP. The goal was to establish collaborative relationships not only with people inside the company but also in other research groups in universities primarily but also potentially in other companies as well that were interested in the same sort of space of nomadic computing. But as indicated earlier, courting the community with a project which primarily is still in an early phase, i.e. basically considered a vision only was not effective in terms of getting attention from the Open Source community.

[33] Citation in CNET news, http://news.com.com/2100-1001-270341.html

# 4. PROGRESSIVE OPEN SOURCE AS A DEVICE FOR R & D

In this part of the thesis I explore and analyze changes associated with introducing large-scale information infrastructure. Companies face new barriers and challenges every day and Hewlett-Packard, as many other companies today has to be flexible and are corollary constantly changing.

Traditional theory on organizational transformation is grounded in a discourse of stability (Pettigrew 1985; Wilson 1992), whereas more recent research suggest that change is an ongoing improvisation by actors, i.e. a process not dependent on a specific time or place (Orlikowski 1996).

According to Orlikowski, technology ought to be viewed as enabling and constraining, i.e. it shapes human action and is also shaped by the action of those using the technology. The introduction of Progressive Source has imposed organizational transformation, i.e. it has substantially changed the structure of the organization as well as its practices. The transformation was enabled by technology, but was not caused by it.

Technology plays the role of mediating the changes in practices and structures. Nevertheless, the transformation is driven by actors and their visions of creating a more open approach enabled by collaborative development conducted with actors inside the organization as well as a selection of trusted external partners. More

precisely, the changes occurred through adjustments, accommodations enacted by the community of developers adopting the POS paradigm.

The introduction of collaborative software does not transform the organization into a more flexible entity, but it depends on the prevailing mental models that depict the relationship between technology and work, and the structural properties of the organization (Orlikowski 1992). In the next section I will also analyze and try to illustrate the implementation of POS technology and discuss how the use of collaborative tools changes the nature of work and the patterns of conducting work and with what intended and unintended consequences.

## Exploring and exploiting - mutually exclusive or both?

The introduction of Progressive Open Source development paradigm within Hewlett-Packard may be viewed as an effort to establish a strategy for driving creativity and innovation while maintaining a balance between exploring new possibilities while exploiting existing knowledge and skills (Holland 1975; March 1991).

Exploration embraces activities such as search for new knowledge, as well as variation, experimentation, play, flexibility, discovery and innovation (March 1991). Exploitation on the other hand captures other aspects such as refining existing knowledge, producing in more efficient ways, i.e. fine-tuning processes in order to gain a more efficient implementation.

The POS paradigm depicted aspects of exploration, e.g. it involved altering work processes introducing more openness inside the organization as well as to a selection of trusted external partners. It strived at enhancing creativity among its collaborating actors since every thing they did became visible to everybody else. Furthermore, it also strived at preventing the organization from the 're-inventing the wheel syndrome'. However, at the same time it clearly embraced modes typical of exploitation, e.g. sharp focus on creating a structured way of conducting research and development including common methods, tools and roles, in short a common cookbook. In addition, managers expressed the need to tighten the reins for the

developers. They could no longer act as lonely cowboys hashing out ideas in a creative but nevertheless an unstructured manner.

Exploitation is typically stimulated by successes in the past, whereas exploration is spurred by failures and disillusions to overcome challenges imposed (March 1999; Weick 1976, 1979, 1982).

## From garage projects to POS projects

The metaphor of the garage goes back to the days when Bill Hewlett and David Packard first started their business in 1937 in a garage in downtown Palo Alto in the heart of Silicon Valley (Packard 1995). The legendary garage in fact designated the birthplace of Silicon Valley and is now a California state historical landmark.

Garage-mentality is an attribute of an ideal way of conducting innovative work. Even today, in spite of efforts such as POS, when I interviewed researchers in the labs, they were telling me about how innovative work was, and to a large extent still is conducted.

Garage mentality takes in important aspects of invisible work, i.e. work that often isn't recognized by management. Managers may be aware that such work exists but that for obvious reasons are difficult to call attention to.

As it often turns out in engineering work, or even in a lot of work related to innovation, people are doing valuable, albeit invisible work that the organization explicitly hasn't asked for. And people for various reasons decide 'to fly under the radar', e.g. they personally are inclined to think that it may be of importance for the company, or sometimes for them individually, but publicly nobody has validated it.

In the past that invisible part of work was a vital part of the culture in HP Labs. And it was also something the engineers loved doing. After some time when the engineers were ready to expose their ideas, often when they believed that it could fit in really well with some ongoing business, the so called G-jobs were brought out and recognized on the public agenda of the organization. Engineers referred to it as G-jobs, and its original connotation was government job, only in practice it very seldom

had anything to do with the government. It simply meant work that you cannot tell anybody about. A G-job is in a sense a sneaky little project that you do on your own, that isn't part of your normal work, but that you as a researcher find interesting.

This is how a researcher in the labs described G-jobs:

> You couldn't tell anybody about it, you know? Okay, so it was working in the garage late at night? Exactly. And M-install arose as a way to share these things, tools that people had developed for their own use that they found useful. And it was one persons G-job to do this, because they realized that such a thing was needed.

The G-jobs were all completely unofficial and at the same time part of the HP culture. The phenomenon is also closely related to what is often labeled 'hacker-culture', and as such closely related to joy, play, entertainment, interest, curiosity, fun, passion, and so on (cf. Himanen 2001). Such a culture supports having a passionate relationship to work, and is of course not only related to software hackers but can be found in other areas, e.g. amongst scientists and many kinds of artists.

As further developed by the same researcher in the labs:

> It was all completely unofficial. Unofficial. And completely distributed. If you wanted to share something, you ran the server on your machine, and you put on your machine the things that you wanted to share.
>
> [So why is it? Has this always been like kind of part of the culture?][34]
>
> Some people around this kind of. Oh yeah, sure. It's basically a hacker culture. You've done something, other people might find it useful. It was a completely bottom-up, completely distributed, completely voluntary system which, while people were using HP-UX as the primary development system, worked pretty well. And it probably still does for the community that uses it. I've taken myself out of the community, so I'm not sure where it is now, but I would guess that you could get on a UNIX box and N-Install to HPGJDE, and you'll get the catalogue.

[34] Throughout the thesis I use brackets when I myself ask questions. Moreover, I use brackets on some occasions to hide the identity of the interviewees when I feel obliged to do so.

The rules of the garage are outlined on posters that you can find hanging on the walls inside the premises of the company. The picture on the poster shows an old garage at sunset or perhaps at dawn. Inside the garage you can see a faint light. Someone is inside working, even though it is late... or early in the morning? The text goes like this:

> Believe you can change the world.
>
> Work quickly, keep the tools unlocked, work whenever.
>
> Know when to work alone and when to work together.
>
> Share - tools, ideas. Trust your colleagues.
>
> No politics.
>
> No bureaucracy. (These are ridiculous in a garage.)
>
> The customer defines a job well done.
>
> Radical ideas are not bad ideas.
>
> Invent different ways of working.
>
> Make a contribution every day.
>
> If it doesn't contribute, it doesn't leave the garage.
>
> Believe that together we can do anything.
>
> Invent.

The rules signify issues of the culture and traditions that has permeated the company from its origin up until the present. It is part of what is known as "the HP way", i.e. it says something about the vision of the company and something about the management techniques that in different ways became the legacy of the founders of the company.

Walter Hewlett and David Packard formed organizational values that among other things encompassed the following:

> We did not want to be a 'hire and fire' - a company that would seek large, short-term contract, employ a great many people for the duration of the contract, and at its completion let those people go. This type of operation

> is often the quickest and most efficient way to get a big job accomplished. But Bill and I didn't want to operate that way. We wanted to be in business for the long haul, to have a company built around a stable and dedicated workforce (Packard 1995:129).

Other core values were reflected in the Management by Objective (MBO) which according to the founders of the company had been a vital part of the HP successes in the past. Packard is referring to a system:

> in which overall objectives are clearly stated and agreed upon, and which gives people the flexibility to work towards those goals in ways they determine best for their own areas of responsibility (Packard 1995:152).

Together with the principle of 'management by walking around' and 'the Open Door Policy' MBO were praised core values to facilitate and support a culture founded on trust, mutual understanding and open communication.

It is out of this context that the Progressive Open Source has evolved, and as such it is also a context rich of precursors and prototypes for collaborative development models utilized within the company.

At the time when I was conducting this study the company was going through a series of challenging events. The American economy was going through a deep recession. Many employees had been laid off on very short notice. This was something that the employees had not been subjected to in the past. Historically, HP had upheld a tradition of employment security even through bad times. But, a lot of that was changing when Carly Fiorina was appointed new CEO of Hewlett-Packard in 1999.[35]

In American media Carly Fiorina was described as a 'Power Babe', a blond, good-looking, and efficient female executive. Inside the company her new ways was creating some turmoil. The employees I interviewed felt like that they were perceived as interchangeable cogs in the machinery, and not as carriers of important knowledge

[35] In July of 1999 Carly Fiorina was named President and CEO of Hewlett-Packard. The following year, in September 2000 she was named chairman of the board of directors.

and skills, and moreover they openly expressed that they didn't recognize the core values of the company any more.

Many employees complained openly that competence was leaving the company by the number. Hewlett-Packard was also planning to merge with Compaq as a way of gaining competitive advantage.[36] This also created much distress among the employees.

To sum up, there were at least three important factors influencing change at this particular time:

1. A general downturn in American economy creating incentives for changing corporate strategies and goals
2. Employee layoffs in order to cope with financial difficulties of the company
3. Planned merger between Hewlett-Packard and Compaq creating distress among employees

The Hewlett-Packard organization was still hierarchically organized. Employees spent a lot of time documenting and informing their superiors. But, there were some signs indicating that they were facing a dividing line. Due to major lay offs the remaining employees had a lot on their respective plates! Working in a slim organization conveys that employees were overloaded with work! At a meeting that I attended a woman burst out:

> You know, hierarchies don't work any longer. Our plates are basically full all the time. I mean, even the bosses are basically always pre-occupied with dealing with issues of lay-offs and down-sizing. It doesn't work! So - we try to work it out anyways. Through our personal networks. And we don't announce things until everything has been taken care of. We solve what we have to do- but we keep a low profile, just to make sure it really gets done the way we have to.

[36] On September 4, 2001 HP and Compaq announce a definitive merger agreement to create global technology leader. On May 3, 2002 HP and Compaq officially merge, beginning operations as one unified company.

## *Struggling with re-inventing the organization*

When Carly Fiorina arrived at Hewlett Packard in 1999 she embarked on a journey trying to re-invent the organization. The company was struggling to regain its former glory as an innovative fast growing company with hefty profits. In spite of the global downturn in the economy Fiorina was setting high revenue growth targets for the company. Her strategy was to create cross-company initiatives in order to get all HP employees working together to help create new growth opportunities for the company. In Business week in February of 2001 she emphasized:

> What we are after is to engage our businesses in a collaborative process that leverages the unique portfolio of Hewlett-Packard in a way that differentiates it in the marketplace.[37]

In October 2000 she publicly announced that Hewlett Packard intended to back the Open Source movement, challenging the efforts of Microsoft and Sun Microsystems. In her keynote speech at the Gartner's symposium ITxpo in Florida, Fiorina gave her view on the Open Source movement:

> The open source movement is natural, inevitable and creates huge benefits. It's part of the next wave of computing, and that will involve participants and users within the industry in open source.

Fiorina also explained publicly that HP were to become a more customer-focused and innovative company focusing on business transformation and IT implementation.

In a speech at the World Congress in Information Technology in Taipei Taiwan in June 2000 Fiorina used biology as a metaphor for companies in the Information age. Biology rather than mechanics were the chosen lens of HP to look at the perceived needs for change. And Fiorina suggested:

> Biology, of course suggests permeability, openness to new influences, new ideas, new ways of looking at things, and it suggests adaptability, the ability of willingness to embrace change. And that is, of course, what invention and reinvention are all about.... If invention and creativity are

> prime virtues of this new age, then the leadership is not about controlling decision-making. Leadership isn't about setting boundaries...the role of leadership is to set people free, to empower them, to create and invent.[38]

And initially it seemed to work. The company was hitting and beating the target for the first year, but in 2001 additional cost-saving moves were initiated. Consequently those initiatives and the struggle to re-invent the organization were an ongoing discussion amongst the developers and managers that I interviewed, especially at HP labs.

On the other hand, the Bluestone teams, which recently had become part of the HP family, had an opposite view of the process of re-inventing the organization. They believed that the HP way and the organizational culture was a way for people to resist change. This is how they perceived the situation:

> I think HP's struggling with their reinvention.
>
> [You think?]
>
> Yes. For whatever reason, the culture that was in HP before Carley Fiorina was very, very deeply ingrained. In my visits to the west coast; they would frequently say the phrase, whenever somebody talks about something and were different, that that's not the HP way. The HP way was the catch phrase at HP.
>
> [So you mean that they were hiding, resisting change behind that label?]
>
> Yes. Even though that label, they couldn't necessarily define it very well, they would hide behind the label, and still do. [Is that your experience?] Yes.
>
> [I mean, I hate to ask you this, but could you give me an example of how it means, because I'm curious. Because I know that you have a reason for saying this.]
>
> I was out there; when was this; November time frame. And it was around the period of layoffs. People were very bitter. I don't know what the layoff history policy had been at HP. I had heard that there had been some work

[37] www.businessweek.com:/2001/01_08/b3720008.htm?scriptFramed

[38] www.hp.com/cgi-bin/pf-new.cgi?IN=referer

> force reductions in the past but mainly wasn't a very common thing, but it wasn't unheard of. But people were very upset, very disillusioned with the company. And from my point of view, virtually every place I've worked has had work force reductions at one time or another. There are ups and down in the companies, the economy, that's just the way it goes. So that could have been part of it, part of setting the mood at the time.
>
> [You mean that they felt too secure in their generic culture?]
>
> Yes, and very stuck to the existing policies and procedures, and not wanting to change and become more agile. (Bluestone manager - became part of HP - but used to be an independent organization. It had recently merged with HP).

## Aligning structure and strategy

### *Before implementation of POS*

As has been laid out already, HP was perceived as struggling with re-inventing the organization. The CEO of the company Carly Fiorina had publicly declared that such an undertaking required four major components that had to work together in harmony; strategy, structure and processes, rewards and metrics, and culture and behavior. And she avowed:

> How do we get our work done? Structures have to be tightly aligned to deliver on strategy. Processes are what hold organizations together. And those processes must be integrated, streamlined and working at world-class efficiency so we can deliver quickly and cost-effectively.[39]

Following the explicit intents of the organization the introduction of the POS paradigm instigated a more structured conduct for individuals and teams working with research and development. Thus collaborative development between teams and individuals working at a distance forced managers and developers to create greater uniformity in the work process.

[39] Carly Fiorina, 2000 World Congress in Information Technology, Taipei, Taiwan, June 12, 2000.

### *Cowboy engineering*

It was acknowledged that the development of complex systems and products were suffering from what they (HP managers) themselves described as 'cowboy-engineering'. As indicated by the expression it encompassed the following features:

- Little, or incomplete documentation of projects, which constituted real and severe challenges when people were rotating or leaving a project. It was also perceived as difficult and sometimes even impossible to share experiences and knowledge between members of the community of developers.
- A larger component of creativity and impulsiveness at the expense of discipline and structure.

Collaboration with external partners, i.e. partners who work at a geographical distance with no or minimal face-to-face contact, require more structure and more planning in order to work more effectively. Past experiences indicated that it was too costly to change and alter the direction of a project randomly lacking a clear method for conducting research and development. A manager at HP's division for Image and Printing described it like this:

> Because there's another element that is part of the cycle of HP, what I call cowboy engineering. They don't want to follow that, "Oh, I don't want to have to get into documentation, having to spell everything out. That wouldn't allow me to be creative and take new directions." They're so used to engineering on the fly with coming up with a creative idea, running down the hall and talking to their counter-partner and everybody goes, "Good idea, let's work on that." Or "What about this." And hashing it out in a hallway conversation where everybody goes back to their cube and gets in there and fixes and does what they want it to do. That doesn't work with a third-party supplier. You have to have some discipline. The difference between that engineer architect and Darryl on the Mac team is "he's already been down this track for a while and he is very clear on how bad that whole cowboy approach works when dealing with an outsource entity." It's expense to outsource to begin with. And when you change directions all the time, and put the supplier through a lot of trash, you simply pay for that. I would say that this tool has cost us more because of that behavior and we have had no luck. I can't influence the guy and his manager seems

> to support that direction. So, it's on of those things that it's a learning process.

## *Metamorphosis – after the implementation of POS*

According to managers and project leaders a more open approach, i.e. involving a large community of developers to participate and review the projects, required a significantly more detailed and overarching work-process. They were trying to create a system constituting of methods, rules, and clearly expressed roles guiding work conduct. It was perceived as necessary to clearly spell out (in writing) how a project was supposed to be realized in order to fit the new development paradigm.

When I interviewed early adopters of POS, it became evident that they had very consciously and deliberately worked out routines for their projects. They themselves viewed this process as a 'learning experience'.

The early adopters had utilized the skills of the Knowledge Management team of Hewlett-Packard in order to develop a process map. The starting point was an already existing outsourcing project with an external partner (WIPRO). Great emphasis was placed on key actors who were also asked to provide input.

The goal of the early adopters was to document in detail how to organize a project effectively, and what necessary steps were needed to be taken in order to introduce a new project. Their input served as a basis for documenting the work process.

As a result of the experiences of the early adopters consequently significant effort was placed on establishing a common language and a common standard for the work process. The effort to document projects resulted in a cookbook prescribing how to conduct collaborative development the POS way.

Roles were created and delineated to support the collaborating actors, guiding them. The proposed roles also provided the means to control projects, since each role was granted different layers of access to the projects. Through the creation and establishment of explicit roles it was possible to manage Intellectual Property rights,

which was an increasing concern of the organization as POS conveyed sharing sensitive information with external partners.

> So we had a technical writer work with our process map expert and she developed the cookbook. It's basically a guide for people who are engaging in an outsourcing activity. And you can go through and look at what your role is. So, if you're the technical lead, you can go in and see each of the major action items that have to occur, what the dependencies are and you can start to fill in the blanks, of "OK for my project I'm going to be working on this. OK now, I need to think about doing this next activity because it relates to something, like setting up CDP and getting people their logins. So maybe that's owned by the project lead. Otherwise that person doesn't have any way of prompting to know that they need to do something about that." They also don't know that "Oh well, before we put code out on CDP we have to have a source code CDA in place. Or we have to talk to procurement about IP protection." Those kind of things kind of fall into that document. And that's now a living a document that's being used by the Cabrio project as a guideline to help make it a little easier for them. What I'm hoping we can do is maybe go in the direction of either producing a white paper or some set of tools that we might even be able to park out on CDP somewhere that we could point people to when they are about to start a project so they have some examples of things they might want to use help make it easier for you to get up and running. That's one of the things that can cause CDP to be a little daunting for someone who's just starting to try to figure out how to use it. You go in and you take a little tutorial. Yah, you can kind of navigate around in the baseline side of it, but now what do you do with it now that you've got it? How do you engage with your partner on the outside? How do you work with other people within HP. What are the steps that make it work more smoothly. What are the things you need to think about.

The managers I interviewed expressed that the aim of the 'cookbook' effort was to collect a number of recipes on how to prepare and conduct a successful project. And managers often referred to the cookbook-examples and often expressed that it aimed at providing adopters of POS with a school-book example of how to succeed with the new POS work process.

More specifically the cookbook encompassed a more detailed specification on how to initiate and conduct a project within the general framework of POS. It also attempted to make explicit to all collaborating actors how far the relationship and commitment was supposed to reach. This was considered as increasingly important when extending the involvement to include third party developers.

The cookbook also defined the explicit relationship between internal collaborating teams. This was perceived as very important since even internal openness and sharing could jeopardize customer relationships. A frequently used example that managers were referring to was that HP often sell systems and products to competing customers. And Sharing may not be appreciated by customers.

The cookbook also spelled out specific requirements and demands of the developers, e.g. the requirement to explicitly delineate the particular issues that needed to be addressed in order to make the working process smother.

Moreover, developers were reminded to ensure that critical security measurements were taken in order to protect Intellectual property rights.

Berger and Luckmann discuss recipe knowledge and define it as: "knowledge limited to pragmatic competence in routine performances" (1966/1991:56). In essence Schutz (1944), who perhaps were the main influencer and inspiration for Berger and Luckmann, suggested that that knowledge is derived from practical experience of the world. And recipe knowledge is a way of providing a purpose or meaning-context for individuals and also for explaining those. He then proposed three ideal types:

- The ordinary man on the street
- The citizen who strives at being well-informed
- The expert

Schutz argued that knowledge tends to be socially distributed according to the ideal types.

For the ordinary man on the street it is suffice with 'recipe-knowledge'. This means, that the ordinary man does not need to understand (not relevant to the immediate purpose) the more advanced knowledge behind preparing a meal (e.g. when baking I don't need to understand the chemistry of yeast or any other scientific aspect of bread baking). According to Schutz (1944):

> ...the knowledge of the man who acts and thinks within the world of his daily life is not homogeneous; it is (1) incoherent, (2) only partially clear, and (3) not at all free from contradictions.[40]

What kind of relevance do the ideas of Schutz have on POS? Why even bring the ideas into this context?

Evidently it would not be completely accurate to use the 'cook-book' effort as an example of how POS potentially threatens to delimit the knowledge of developers using the system. Cleary, the ideal types were theoretically intended, and as such they are to be utilized for the purpose of explicating the social distribution of knowledge. But still, it may help us to start analyzing if, (as the cookbook example seems to indicate), relying on recipes for conducting collaborative development in fact threatens to delimit the interest and/or responsibility of the individual to experiment and to explore. What will in fact be the impact of a more structured and controlled development process, what happens with innovative and explorative modes of research and development within the POS initiative?

Thus the theory of Schutz (1944) conveys additional issues that seem to have a potential impact on developers in the light of the more structured approach offered by the 'cookbook'. We may begin to ask whether if (as the cookbook example seems to indicate) developers rely increasingly on recipe knowledge, i.e. incomplete, only partially clear, and even contradictory knowledge, what it will do to their capacity to judge the validity of a proposed recipe. Will they even care to question if the recipe is accurate, efficient, and so forth?

[40]Also in Schutz, A., The stranger: an essay in social psychology. In: Collected papers. Vol. II. Studies in social theory, The Hague: Martinu Nijhoff, 1964, 93

What are the boundaries, in terms of information seeking behavior, between the 'ordinary' albeit the 'well-informed' developer, and the expert? Will it have an effect on innovativeness and creativity of the organization in the future?

I don't intend to answer these questions in detail, perhaps not even on an overarching level. For the purposes of this study it is suffice to raise such and similar concerns.

The introduction of POS was considered by managers that I interviewed as a way of bringing more structure in to the work process. This in turn was perceived as necessary when introducing new developers, especially third party developers with limited previous experiences of working with HP. Accordingly POS was considered by managers as a crucial and important step towards establishing an effective knowledge transfer between engineers, internal as well as external. New co-workers were to be provided with the 'cook-book' and they were also to be given access to already existing projects in order to quickly grasp how to do research and development the HP-way.

> Then we're going to have a set of training tools we're going to use so when a new engineer comes in, the first thing they're going to do is get pointed at CDP and say "Here go and learn all this stuff. That's your job for a month. Go focus on this and learn everything you can. And Oh, here's a few defects to start to introduce you to the code." And that becomes the process whereby we will be bringing our own engineering staff up to speed more quickly. If you can cut the time that it takes to train an engineer to be productive and effective by two thirds, CDP, has more than paid for itself. So, that's a big plus too.
>
> [They must think this is a very interesting opportunity for them to come in and apply their knowledge.]
>
> It's a very interest tool. They way I got introduced to it was I was talking with [Person X] who was saying, "Have you heard about CDP?" The minute he started showing this to me, I looked at him and thought, "This is the answer to my prayers." I was looking at, we were just starting to do software and firmware outsourcing. I was looking at all these potential issues.

> Every engagement we learn from and we get better at it. When I think about where we stand now as opposed to a year ago, it's really an amazing progress. I envision that 2-3 years down the road the struggles we have here are going to seem so distant and it's going to seem like such an easy thing to do. We'll have our processes so nailed down that people who are coming new to the outsourcing process will slide right in and have no problems with it. So much of what makes it work well is having the infrastructure and the process in place and community internally within the Vancouver labs that understand how to work with an external set of partners. That's definitely not second nature now but it will be. It's just a matter of having a significant number of engagements that people have the exposure to the process. We work the kinks out of it and get better at it over time. So we're driving all our suppliers to embrace CDP, to learn as fast as possible and to drive toward an independent development model so they don't have to rely so much on us to solve their day to day problems that they become very independent. Take the problem that we pose to them, go out and figure out the solution, and check in with us on technical issues. If we need to redirect we do, but it's mainly that they're driving, they're pushing their momentum forward to deliver for their target dates that we set for them. That's very different than the hand-holding model that we started out with and we've evolved that over the course of three projects. It's literally night and day. The necessity is mother invention. We really can't handle these folks. We need to push them to be more independent and we just keep telling them. We're moving away from them. "No don't call us. Go fix it. Find a solution and you tell us what your options are and we'll tell you which one we think makes the most sense given the context." It's been real interesting.

## Knowledge and management, relevance and perspectives

POS, I posit, is a device for sharing knowledge, or at least this is what seems to justify the introduction of the project. In my view, it is also a way of organizing innovative work.

As has been highlighted in previous sections in this chapter, adopting a more open approach may also be viewed as an attempt to re-invent the organization,

making it more flexible and more prone towards outsourcing and distributiveness of the workforce.

It is impossible to ignore that the notion of knowledge embraces many philosophical aspects and its character is in many ways both elusive and abstract. Nevertheless, knowledge has become perhaps the most desired object of management in many organizations today. This clearly constitutes a challenge when transforming desired outcomes of knowledge and know-how into action-oriented attempts in an organization.

As I have suggested already the act of transforming knowledge and information in to a form that actually can be managed is a significant part of the POS project. Indeed, POS, while attempting to make knowledge and information more available and open, it at the same time facilitates attempts to ascertain control and ownership of that knowledge. Throughout all my interviews with managers issues of Intellectual Property (IP) rights seemed to remain important for HP. Perhaps those issues were at least slightly less of a concern, considering the organizational motto 'speed over secrecy' which was equally often referred to. In my view it was the means of protecting IP rights that were taking new forms. Moreover I recognized that:

- Rules were utilized for the purpose of developing a robust process for managing outsourcing efforts.
- Managers were keen to define IP-rights early in the development process.
- Managers often instructed and reminded developers to think before speaking. It was suggested by managers that IP-rights were relying on the best the best efforts of the people involved in the projects.
- Projects were typically structured in a hierarchical way in order to control access to the projects. Defining different roles for the developers were part of what I denote as part of structure and hierarchical control. The roles typically specified who could do and see what in the projects. Through such structures HP obtained layers of security.
- Members of HP were constantly reminded to avoid cross-pollinating, i.e. avoid sharing information that is company proprietary.

- The organization had to maintain a necessary balance between openness, i.e. sharing while at the same time upholding security. Security concerns often stretched out to include the external collaborators attitudes, and measurements to uphold security.

As has been indicated by a vast and extensive literature on knowledge management, knowledge, in order to be stored and later retrieved, needs to be codified (cf. Nonaka & Takeuchi 1995). Similarly the POS project seems to focus on making software and its running code available to a community of collaborating developers. Efforts were devoted towards allocating, transferring and sharing the software and the ideas that evolve around the development. Solutions and ideas were to be shared in a communicable format, i.e. the software were presented in form of running code and the discussions were articulated, mainly in writing. The ultimate goal was to ensure that knowledge resources were available to the teams 24/7.

Productivity was according to managers I interviewed supposed to be achieved through applying knowledge to knowledge. The knowledge of others in the collaborating network was to be used in new situations and also act as a foundation for similar products being developed in parallel. The objects, i.e. software, obviously contain knowledge, albeit the knowledge itself is hidden in the product.

Kreiner and Tryggestad (2002) uses the metaphor 'packaging for knowledge',[41] which has a strong rhetorical connotation, implying that the customers who finally utilize the product do not need to understand everything that goes in to the product. It is suffice that the customer understand what it is and that it suits the customer needs. When utilizing the product the knowledge contained becomes unleashed for their respective purposes.

However, even though the POS project ultimately aimed at producing products that were compatible with each other to the benefit of the customer, the focus for this study is the process whereby knowledge is created, i.e. produced rather than ultimately consumed.

Interestingly enough when looking at Open Source projects the distinction between users and producers becomes blurred. Producers and users of the software may in some cases be one and the same. On the contrary and more traditionally, looking at POS projects, the distinction becomes clearer. Not everyone was invited to participate in the development, but rather a selection of trusted partners. And Intellectual Property Rights were still considered a very important issue, ultimately shutting competitors and also potentially the customers out.

Stewart (1997:108-109) suggests that the goal of the management effort is to make sure that knowledge that "belongs to the organization as a whole can be reproduced and shared". And looking at POS structures, roles and boundaries were in place and acted as sanctions against complete openness and scrutiny of the black-box, i.e. the proprietary object.

A large extent of the literature on knowledge management focuses on intellectual capital and knowledge assets. According to Kreiner (2002) the attempts to transform knowledge into manageable assets often involves making knowledge explicit. Only, this on many occasions tends to be a difficult task, especially since knowledge often tends to be hard to define and even harder to grasp. If knowledge, (as in relation to new product development processes[42]), is difficult to manage from a perspective of ownership and control. How and if, can it be made manageable? Kreiner (2002) highlights two separate managerial efforts:

1. The first effort rests on the notion that knowledge is a resource in itself and consequently the managerial goals are focusing on the acquisition, codification and the transfer of knowledge. This effort relies heavily on providing explicit knowledge or rather the focus is on information.

2. The second managerial effort is focusing on mobilizing knowledge through managing contexts and facilitating interaction. The role of knowledge

[41] Adopted from Peter Drucker 1993.

[42] I define new product development as a process by which knowledge is built into the products produced within the POS paradigms (e.g. printers).

management accordingly becomes the act of attracting and adding together knowledgeable people.

The two alternatives indicate that it is not suffice to rely only on a strategy for making information and knowledge available. Given this insight as suggested by Kreiner (2002) what are the possible implications for POS?

It is perhaps relevant to question to what an extent the POS is just another type 1 (above) example of knowledge management efforts, i.e. just any another effort to acquire, codify and disseminate information and readymade cookbook solutions , i.e.,exploiting mode, rather than stimulating a constant quest for answering new problems and issues , i.e. explorative mode?

The constructivist theories of knowledge development are in a distinct contrast to the more objectivist perspective. The biggest gap in agreement is whether knowledge is depending on the knowing subject, e.g. a person, a team, or a firm, or independent on it (cf. Kuhn 1962/1996; Feyerabend 1975; Berger & Luckmann 1966/1991; Habermas 1973; Popper 1972; Popper & Eccles 1984).

The constructivist approach assumes that knowledge within a group, an organization or an individual is depending on the knowing subject transmitting knowledge through social or cognitive processes (Von Krogh & Roos 1995). For knowledge to evolve at the social level, individuals must share subjective knowledge with others. Schutz and Luckmann (1973) use the term objectivation which is "the process by which the externalized products of human activity attain the character of objectivity" (Berger & Luckmann 1966/1991:78). It is clear that such a process, i.e. objectivation, covers more than just communication.

As pointed out by Von Krogh and Roos (1995) organizations generally makes use of at least three channels, i.e. language and signs, tools, and marks.

Individuals convey their subjective knowledge by talking and writing, i.e. using language. However, as have been pointed out by Polanyi (1962, 1967) a significant part of the individuals stock of language is tacit. When trying to share knowledge, individuals need to complement linguistic expressions (oral or written) by using signs

such as e.g. gestures, evoking facial expressions, but also through drawing. In this study, developers (engineers) have indicated that they find it hard to use POS simply because it favors written language. They are not used to, i.e. do not feel comfortable sharing ideas, solutions or conversations articulated in writing.

Individuals also convey their subjective knowledge to others through creating and applying tools to solve tasks (Von Krogh & Roos 1995).

Marks are "the result of acts established by the one acting in order to hold onto a definite element of knowledge and to mind one of this" (Schutz & Luckmann 1973/1985:274).

Individuals continuously participate in the creation of social knowledge of an organization through communicating with others, e.g. speaking, writing, drawing, using tools and marks.

However, and this makes sense when talking about POS, and the cook-book example. The objectivation of an individual's knowledge may be affected by the process of legitimation. Berger and Luckmann describe legitimation:

> ...as a process is best described as a 'second-order' objectivation of meaning. Legitimation produces new meanings that serve to integrate the meanings already attached to disparate institutional processes. The function of legitimation is to make objectively available and subjectively plausible the 'first-order' objectivations that have been institutionalized (Berger & Luckmann 1966/1991:110).

Legitimation consists of: 'what everybody knows' about a particular social world and also explains and justifies through cognitive and normative elements (Berger & Luckmann 1966/1991:111). Furthermore, Berger & Luckmann (1966/1991:112-113) suggest that subjective knowledge must be made legitimate on at least four levels in order to contribute to the creation of new social knowledge.

Firstly, subjective knowledge must be conveyed by using language and signs, marks and tools that are accepted by the community of developers (as in POS).

Secondly, subjective knowledge is made legitimate by referring to or evoking organizational stories or myths.

Thirdly, organizations have a set of standard operating procedures (e.g. quality control measurements, accounting principles). Individual knowledge may be made legitimate by making references to these espoused theories or made concrete by using them or supporting them.

Fourthly, organizations tend to have continuity of paradigms which give meaning to activities and experiences (Pfeffer & Salancik 1978; Parsons, 1956; Parsons 1960). Why is that? According to Berger & Luckmann such paradigms put "everything in its right place" (1966/1991: 116). There are industry specific recipes that delimit what is considered acceptable behavior of the organization (Spender 1990).

The process of legitimation serves to prevent an individuals stock of knowledge from disturbing continuity and regularity of the organization.

The introduction of a cookbook way of doing software research and development within the POS paradigm may be viewed as a process of legitimation. The legitimate knowledge should be captured according to the 'cookbook'. Experimental behavior, such as depicted by 'cowboy-engineering' was perceived as counterproductive for the goals of the organization.

## Learning while doing

This study seems to indicate that structures are developed and improved as HP go along adopting the POS way of conducting research and development. There was no time left to reflect upon whether they were doing it the right way – but rather the development of new processes had to take place while attending to business as usual.

## *The metaphor of the running car*

A manager that I interviewed particularly highlighted POS as a learning experience, something that was introduced while trying to attend to business at usual. She was using the metaphor of the running car, which to me indicated the immediate concern of the organization to improve processes in order to enhance its ability to compete in an increasingly fierce competitive market situation. When POS is introduced it is not yet made stable, but it is still evolving. This is her story:

> When I think about the fact that we didn't know about CDP really very much at all in February when I agreed to take on software work. [Person X] and I took a trip in April to try and identify some firmware outsourcing entities for him to work with, by July we were launching our first project and we launched it on CDP with all its quirks and its problems. It's just been of those things where we just keep accelerating the pace and the learning processes are exponential. The metaphor that I used is that we are driving the car down the road and we are improving it as we go. We're in there tinkering under the hood, somebody's hanging off the side fixing the door lock, and that's what we have to do because the flow of business, the need to hit delivery dates for the products that have to go to market is so overwhelming that we don't have the luxury of sitting down and saying, "Well let's just take some time to go off here on the sidelines and figure out how to do this right." We literally must do it at the same time as we're trying to deliver a product to market. [Person X's] got a lot more gray hair than he had last year and so do I.

The managers also described in essence the type of environment that POS was implemented in. It was an environment where people knew each other inside an out. She admitted that she would not have liked embarking on such an event without a well-functioning team. It was a very delicate assignment and a lot was at stake. One small mistake when implementing POS may potentially jeopardize their result and ultimately their existence. She used another metaphor to capture how much was at stake for them when implementing POS:

> I'm thinking about the winter Olympics. Somebody coming down that ski slope and dodging in amongst the flags. They make one little move to the right when they should have gone 10 degrees to the left, and they don't fall

> down, they still go to the finish line, but they lose the race. It's those little tiny course corrections that have to happen, critical path, real time, as fast as possible that CDP can fill those gaps. It can help us stay on the gyroscope. They help us stay on track and help us go on in the right direction.

Through my interviews with developers and managers it was clear that they were struggling with adopting POS. Some developers tended to adopt the approach more easily than others whereas others were still trying to learn how to make use of POS. They were exploring how to relate to rules, roles and tools provided by POS. On some instances, 'super-users', i.e. early adopters of POS actively and seemingly voluntary assumed the role of helping others, guiding them if you will. And those super users of POS actively supported POS and facilitated spreading the positive aspects of it to others.

## Leveraging best practices

The goal of POS is ultimately to collect best practices through enabling evaluations and comparisons between projects. The developers can at least theoretically stand on each others shoulders in the sense avoiding repeating mistakes, and copying already existing solutions and so forth. However, an ongoing concern of some of the developers that I interviewed related to issues of how to structure information in such a way that the best practices actually are made easily available to the rest of the community.

This study was conducted during the start-up phase of POS. And it was at that time not very stable and in many aspects still an evolving project. The community was growing very fast initially, and it was utilized for many different reasons. Put together, the popularity and lack of coherence in terms of structuring information, it was by many viewed as 'just a bunch of data'. Developers were requesting a function – a match-making function that would help them navigating through the projects seeking meaning. This particular developer expressed that he was actually longing for a group of people that could help him looking for similarities between projects. This is how he describes the challenge of finding relevant projects:

> So if HP had somebody that was always perusing, or a group of people that were always perusing all this open source, looking for patterns, then they could send notices to these two different projects. They might not know about each other because they're geographically dispersed or organizationally dispersed, and then they would say hey you two, you should meet each other. You're doing similar things. And try to put those links together between the projects.
>
> [Like match makers.]
>
> Yeah, really. That's what it might take because you're right. There is this problem where CDP was laid out, a lot of people saw the advantage. Everybody jumped into it, they're using it for all sorts of capabilities from investigations to shipping products, documentation only to actual source code. And now there's such a volume of data, now what do you do with it? How do you actually? Is it going to grind to a halt because it's just too popular? When you talk to people it's always about how they have overload with information. But they don't really have a technique for actually leveraging information and actually putting it, reusing and helping them. No real knowledge management strategy.

Developers were struggling with establishing structure and processes in terms of setting up projects. There were no established templates from the outset of POS which often made it difficult for developers to browse through projects and really understand what the projects encompassed. Often the projects were given code names which meant little or nothing to those not immediately participating in the projects. Developers established their own best practices and this is how they went about:

> But to really understand what each project is I would have to go in and view the documentation that they might have online. Because a lot of times a project is given a code name, so for example Yatra. What does Yatra do? To understand it I would have to look in documentation. To look at documentation they would have to publish it on CDP, and they would also have to publish it in a way that would make sense, for example they would need an introduction to explain what is Yatra. A lot of times, and to me this is another area where structure would be good, a lot of times the project is just a collection of miscellaneous notes that maybe two engineers on a team have been keeping up to date, and other engineers don't. So

> maintaining the documentation, the structure of the documentation, that's not there in CDP right now.
>
> [So there are still some more processes that have to come in place in order; perhaps in templates for how you document, or how you use the tools of CDP?]
>
> That's right. And then you would need to make sure that people use those templates.
>
> [But do you in your team, have you established such standards for your own working within your team and your projects? Do you work with templates? Or if you start up a project and maybe [Person X] has another project, do they look the same, or do you work within the same line?]
>
> Yes, even amongst the three projects I have, I'm involved with, my two projects look very similar. My device plug-in project and also my localization project. And that's because I'm the author and I used the same format for both of them. But even between my project and the project that I'm participating on which is the Web Jet project, there is a different mentality between the two. For example, CDP allows you to check your documentation in the CVS browser from CDP, but it also allows you to just check documents into CDP and then get a version control software. Well, I'm checking all my documentation into CDP, that way I can go back and some versions very easily. I'm sorry, checking in to CVS; that way I can use CVS's versioning capabilities to go back and look at older versions and in all documents and see what changed and when, and these type of things. But not everybody does that.

And the teams were struggling initially. The information was available through POS, but the developers perceived that the processes and structures were still lacking, or were at least not very consistent which made knowledge transfer difficult. This was a frequent comment:

> I don't really know what that project is. I would have to wade through all these different documents just to try to understand it. I'm just thinking; collaboration, sort of, between people, and in order not to reinvent and to be able to elaborate and reuse and so forth, that would take some work. The information is there right now, but to be able to use the information properly and then to transfer knowledge then it takes some kind of structuring that is not still there.

## POS as a communication device

As a result of studying POS it in a wider sense became applicable to me to think of POS in terms of a communication device, i.e. as a tool for sharing information in the form of source code and written texts.

Teams of developers were utilizing POS for communicating – even though they were co-located. They did that in order to create a collective history of the projects they were developing. And since the teams were constantly changing members such a collective history of discussions regarding the projects were facilitating new team members as they are brought in. This is a developer discussing how they perceive POS in terms communication and knowledge transfer:

> CDP is all about knowledge transfer, whether its knowledge transfer between parties during the engagement of the project, you can use it to get another party up to speed so they can function well. You can use it to do a lot of things to keep things moving ahead because communication is really the fundamental issue when you're working with someone who's not on site. It's a fundamental issue when they are on site, but at least you can walk down the hall and talk to them. In fact, [Person X's] team is now using CDP for communication intra-team. So these guys all site on the same row and they're all shooting stuff back and forth across forums. I said, "Why are you guys doing that?" And he said, "First it's good discipline. We talked about it and decided it would be good discipline and second it's so then we have a history. So if somebody leave and somebody else comes in and replaces them you can actually go in there and see what our team's thought processes are, how we work together, who's done what on which project, it becomes and efficacious way to transfer information to someone who's coming in new. He's got so many people coming in to his team. So as these new folks come on, I want to have that information all ready to hand, so they don't have to spend weeks trying to pick it out of somebody's head. You just go look and see the history of a whole discussion. It's a very powerful communication tool.

The metaphor of POS as a communication device also consists of issues related to embracing a new process. In order to make learning and knowledge transfer more

explicit between developers they needed to have a mechanism for it (process). Some developers remained reticent in terms of adopting POS.

The reasons for reticence varied, e.g. some people were reluctant to share ideas in the first place due to the company policy which involves rewarding engineers for filing patents. Other developers were just unfamiliar with how development was set up within the POS infrastructure and process. At the point of this study POS was still immature but evolving. This developer suggests that the sharing of best practices within the community is a way to counteract reticence:

> The next hill to climb is going to be making the mechanisms for that learning or that exchange more explicit, making it easier for people to adopt CDP for their next project. I think where there's reticence, it's partially around, "I don't want to share my intellectual property. I don't want to share my inventions with anybody." But the other level of reticence is "I don't even know where to start with this thing. How would I run a project. I know you've got forums and you have this feature and this aspect to it, and there's this tool, but where do I begin." They say that the hardest thing for a writer to do is to stare at a blank piece of paper. Because where should your first mark be? What's the first word in that story. And everybody goes through that when they have to write something. Whereas if you just draw a line on the page, it's no longer pristine. This is now scratch paper because I drew a line across it. So, I can just start to write something, so if it's not good, it's OK. It's like it doesn't have to be perfect because I'm just practicing. Giving people that jumpstart is going to be the next hill for CDP. It's not a hill that necessarily the folks at CollabNet should try and climb or anybody else outside should climb. I think it's really something where the user community really needs to start sharing best practices. It's funny, after talking, really what we need is a best practices Web site on CDP where within IPS, people are sharing templates that have worked so well for their teams. It becomes a tool for communication and expanding people's knowledge within HP who are engaging in the outsourcing practice, just making it easier and better for us. It would be really useful.

Reticence towards POS was a frequent topic during interviews and other conversations I had with developers. And even though many of them, (some were

eagerly joining others were reluctantly joining) were actually adopting POS it was often a mandated decision to join. Developers often expressed that they perceived it as important to have active management support and encouragement. It was perceived as important because HP was at this time regarded as a hierarchical organization, and developers were constantly evaluated on their performance and at the same time showing respect for authority of their superiors. And given the time frame of HP, which I laid out initially in this chapter, people were getting laid off and the company was suffering from the global recession in 1999-2001. The developers were asking for some personal evangelism, i.e. managers that actively express and reward participation, in order to actually spur developers to participate.

> [Have you seen anybody so far being acknowledged for using open source?] No. Informally yes, formally no. [Would it help?]
>
> Absolutely.
>
> [Who would be that person? Would it be your lab manager?]
>
> Well, if I could pick anyone, I would start with the department managers, because I think that we, having worked in Silicon Valley for as long as I have, I still find HP kind of really strange, and that we are so concerned about hierarchy and authority, where if you went to a number of other companies around here, and if you had a lab manager say something compared to one of the co-workers, nobody would care. But here that means something special because we have this notion that hierarchy can convey some value.

## From a paradigm of distributed responsibility to a paradigm of a big family

The teams were forming their own respective paths while relying on that the knowledge and experience of the other teams gets contributed back to the system.

> We look to draft where it makes sense, and where there's a piece that doesn't fit our business, yes, we'll do something different. But then look to contribute that back, because you don't want to be different for the sake of being different, you want to be different because you have a different problem domain that you're trying to solve. And then look to share your

> solution with anybody who may also stumble across that same problem domain. We're pretty big believers in the big family.

It was acknowledged that the future success of Progressive Open Source depended on whether the company (HP) was going to ensure that there were sufficient interdependencies between the teams. Managers and developers respectively agreed that HP had a history of having a fierce distributed responsibility for their respective end results. A manager expressed it like this:

> HP divisions don't like to be that tightly coupled with other HP divisions.

## Conclusions

Discussions around knowledge management ultimately boil down to intricate and philosophically difficult issues. It is painfully clear, that a large bulk of knowledge management literature focuses on *managing* knowledge, while avoiding the concept of knowledge itself. Knowledge is celebrated (together with technology) as a main source for gaining competitive advantage in modern economies (Kreiner & Mouritsen 2003).

In this study, I limit the definition of knowledge to encompass knowledge as resources, which when properly used becomes productive and resourceful to the organization that I have studied. However, and this is perhaps key, knowledge does not render itself to be defined as easily as other important resources utilized by an organization, e.g. land, buildings, capital and so on. The real challenge becomes how to manage and make manageable something that in itself is so lucid and difficult to grasp.

Regardless of whether knowledge itself can be rendered by knowledge management efforts, and even though they at best may be viewed as efforts of sharing explicit knowledge (and even being reduced to mere information sharing) they are the focal interests of organizations and individuals that for different reasons need to work in a distributed setting. In fact, collaboration presupposes that technologies which facilitate virtual communication (in different forms) are in place.

It is commonly acknowledged that information is no longer a scarce resource, but it is rather the proper tools that help people navigating that are lacking. POS aimed at facilitating both sharing knowledge and creating new knowledge.

Before we can anticipate sharing and creation, knowledge must be collected, certified, and distributed (Kreiner & Mouritsen 2003). It is a process of turning knowledge into something that can be effectively communicable. Clearly the aim of knowledge management is to turn expertise into even more valuable assets. The back side of the coin however, that more information necessarily doesn't move knowledge further.

POS, especially the CDP version of POS, gained a lot of momentum initially. Many of the developers saw the advantages. However, since they had little structure initially as to what to store within the system, they were using it for all sorts of capabilities, e.g. everything including investigations to shipping products, documentation and source code. The result was a large volume of data. The problem arose as what to do with it. The biggest threat was that the project was going to grind to a halt because it was just too popular. When talking to the developers they all confessed to have overload with information. However, they didn't really have a technique for actually leveraging information in order to effectively reuse and help them to create new things.

Initially there was no distinct strategy in place to avoid problems concerning the familiar phrase: 'garbage in – garbage out'. Also, projects were not listed and defined consistently by the developers which added on to the difficulty to get a good overview of existing projects. The recognized, albeit still lacking, remedy was to construct a template as how to present and define projects in order to achieve greater consistency. Other reported difficulties was how to evaluate the legitimacy of information stored within POS and how to make sure that the information was updated and not obsolete.

Knowledge is already the object of managerial decisions and efforts. Still knowledge to a large extent is hidden in the individual and remains subjective and tacit. Its medium, i.e. the individual developers of POS were in many respects

subjected to the knowledge management efforts. Regardless of whether the knowledge management solutions worked well or not – efforts were continuously superseding each other in order to make it possible for people to work together and share experiences and know how. Technology corollary in this study is the technological solution, i.e. the system and the project that I denote POS. Technology in some respects may also be viewed as the body of knowledge that is reflected in the application of methods, techniques and skills employed by the POS project.

Finally, was POS explorative or exploitative in its character? The truth is probably both. It was an example of two different development attempts. The way work was organized was explorative in its character. Whereas the production of code i.e, in the form ofsoftware and products, tended to be more exploiting, e.g. more products were developed in parallel, ideas and solutions were supposed to be effectively re-used as to prevent re-inventing the wheel.

# 5. COLLABORATION AT A DISTANCE AND FACE-TO-FACE INTERACTION

**Collaboration developer's definition:**

There are many aspects to it. At one level, collaboration is just different individuals that might not typically work together, coming together to meet a common goal. That might be one level. I guess another perspective would be that two individuals or groups working on different projects sharing pieces of those projects together toward a common goal, or toward separate goals. So instead of the open source community has people all over the world all working on Apache, for example. And it could also be collaboration where two different teams are working on similar projects that might have similar capabilities, but they're not identical. But they still share pieces amongst themselves, so that the final product the customer says, well this looks like a product ABC, and the other product is XYZ, and they don't see that there are any similarities. But actually, if you go under the covers, development is collaborative to meet that, because they're using common components, for example. Does that make sense?

**Collaboration manager's definition:**

I would say, for instance, a definition would be to achieve over geographical and organizational distance the same level of close cooperative development that you get from a small team. Because the industry's been ever-shifting, and we're somewhere in the middle right now, but it used to be that development, the good, cooperative development was done locally. Everything else was done by very cumbersome meetings, trips, document exchange, and usually either not electronic or over some cumbersome electronic form. But never was it very interactive. But if you can close the geographic distance to make someone in a different organization or in a different location as though they were local, it greatly shortens the development time and improves the quality of the product.

POS development had the ability to connect people and information assets, and it enabled cross-functional teams, working in different geographic areas, across different time zones. It corollary promised to save time, money and other resources. POS supported group processes provided by communication technologies. The developers typically interacted with POS artifacts, and their action was corollary mediated. The group process typically consisted of idea generation, problem-solving activities, information exchange, clarifying efforts, handling conflicting interests, negotiations, decision making and managerial efforts.

The goal of POS was to ensure that all members of the team had access to the same information and instructions, i.e. that the teams were able to form, interact and ultimately perform their tasks *with none or minimal meeting face-to-face interaction.* And whilst POS was a technology-based information system, we must move beyond considering technological issues and also include concerns of organization structures and processes, including human aspects. The purpose of this chapter is to delineate situations when face-to-face interaction was preferred by developers.

Prior findings indicate that the human face is a very powerful human referent – if not the most powerful (Sproull, Subramani, Kiesler, Walker & Waters 1996). When observing someone's face, we can 'read' emotion states and perhaps also personality attributes such as e.g. friendliness or optimism (Ekman 1982; Warner & Sugarman 1986). Other research indicates that facial appearance has an impact on expectations for interaction, (Hilton & Darley 1991; Snyder 1984). Facial appearance and expressions tends to be interpreted by others and corollary indicates what can be expected of a situation. For now, it is suffice to remark, that many of these signals can be misinterpreted, leading to a vide array of difficulties related to personality and emotion, which in turn have significant impact on social behavior.

In **Table 3** I compare virtual interaction facilitated by POS with face-to-face interaction. The dimensions are partially adapted from research by Bavelas & Chovil (2000). However the content is concluded from this study and relate only to the context of POS.

***Table 3:*** *Virtual interaction compared with face-to-face interaction in the context of POS*

| **Virtual interaction facilitated by POS** | **Face-to-face interaction** |
|---|---|
| Dialogue was mainly written and posted in forums. Corollary it was static in character. It could be read and reread by members of the community, leaving permanent records. And although you may get a swift response on an issue – you could not count on it. | Is dynamic and ephemeral in its character, developers respond to each other immediately without being 'reviewed' |
| Consisted of written text and code. Developers had to be able to articulate ideas in written text since POS did not favor drawings. It was context free and words become very explicit. | It is an ongoing continuous flow of words and acts merging. Developers can complement words with drawings on e.g. a whiteboard. Gestures and facial expressions complement conversations. Interaction includes tacit elements. |
| The target group of the forums were the community of developers, not the individual | Dialogue is a social interplay between two or more developers being physically in the same setting |
| The writer and the reader resided in different places. Text in the forums was addressed to a general audience. Since everything became visible in the forums – you had to mind what was articulated. | The developers interact in the same setting and can benefit from contextual clues and information from the ongoing discussion. |
| Conversations had to be followed by certain norms and rules of conduct, i.e. information were limited in order to safe-guard IP's. | Norms are freer, although developers have to be reminded to 'think before they speak'. |
| Environment of structured communication | Free flow of communication |

As indicated, POS was introduced in order to open up the boundaries for collaboration with third parties more vigorously than HP had done in the past, and to facilitate developers to find ongoing projects and other developers with common interests.

The first step was to create a more organic model of community building, a virtual electronic community as opposed to the physically assigned workgroups that they had used traditionally.

The second step was to invent an environment of structured communication, i.e. communication should be structured in the sense that it had to be captured, catalogued, and inventoried.

The third step was to ensure an effective collaborative management space, in which commitments could be tracked and progressed towards those commitments and moreover be monitored in some sort of overlay fashion.

POS developers collaborated on projects, indulged in discussions and decision-making, sharing documents and code. Collaboration was relying on an organization that potentially could facilitate coordinated activity using resources outside the organizational boundary, i.e. it may include third parties to conduct development in order to obtain a team with specialized expertise to complete the development of the product.

Often developers resided in multiple and physically wide geographic areas, leaning on a common infrastructure provided by POS. POS was in this sense facilitating virtual organizing, since work was conducted mainly relying on electronic technology.

The virtual aspects of POS encompassed the following characteristics:

- *Transcendence-* the POS teams transcended time, distance and organizational size.
- *Infinite* – The POS teams could at least theoretically have infinite developers enrolled, (even though the largest project studied, CDP, encompassed approximately 3000 developers). Collaboration was facilitated by Internet technology that enabled developers to share code and other information (Exception Corporate Source that resided inside HP only).
- *Openness* – Participation and contribution was open, only slightly safe-guarded by rules, and roles. POS developers were not anonymous in contrast to Open Source developers, which altered the conditions for communication, leading in my interpretation to a self-regulatory control instilled by the openness of the system.

Technology provided by POS was introduced in order to support learning and to spur innovation, e.g. projects were often used as tutorials to facilitate training. It was also

used for explorative purposes, i.e. to share new experiences whilst avoiding 're-inventing the wheel'.

POS supported interaction between collaborating actors internal as well as external to the organization. In light of this, communication technology provided by POS may be regarded as one of the many extensions on human modes of communication, and as such it enabled communication and collaboration across geographical distance and different time zones.

Prior research recognizes that most distributed work requires mediated communication (Nardi & Whittaker 2002). Additional research indicates that face-to-face conversation is fundamental for language use and as a result has a very profound impact on human interaction and reciprocal understanding (Clark & Brennan 1991; Kiesler, Siegel & McGuire 1984; Rutter 1987; Short, Williams & Christie 1976).

It is suggested by Filmore that the use of language in face-to-face interaction

> is the basic and primary use of language, all others being best described in terms of their manner of deviation from that base (Filmore 1981:152).

Berger and Luckmann contend that:

> The most important experience of others takes place in the face-to-face situation, which is the prototypical case of social interaction. All other cases are derivatives of it (Berger & Luckmann 1966/1991:43).

The face-to-face situation as such encompasses sharing the present with the other person and the 'here and now' of each individual becomes impinged on each other. Berger and Luckmann continue:

> Every expression of mine is oriented towards him, and vice versa, and this continuous reciprocity of expressive acts is simultaneously available to both of us. This means that, in the face-to-face situation, the others subjectivity is available to me through a maximum of symptoms (Berger & Luckmann 1966/1991:43).

In an attempt to summarize important insights from research on issues related to collaboration and communication at a distance I wish to stress the following in regard to when and if face-to-face interaction is perceived as important, particularly bearing relevance on POS development:

Face-to-face communication and collaboration seems to be important in problem solving situations since it facilitates the necessary presuppositions for mutual understanding (Clark & Brennan 1991; Kiesler, Sigel & McGuire 1984; Rutter 1987; Short, Williams & Christie 1976). Others even assert that it is irreplaceable (Nohria & Eccles 1992; Handy 1995; Hallowell 1999; Olson & Olson 2000; Olson, Teasley, Covi & Olson 2002).

Face-to-face communication and collaboration may under certain conditions be regarded as costly and disruptive, leading to the conclusion that communication technology such as: email, chat, or electronic forums is to be preferred (Hollan & Stornetta; Sproull & Kiesler 1992; DeSanctis & Gallupe 1987; Jarvenpaa & Leidner 1999; Morley & Stephenson 1969; Nardi, Whittaker & Bradner 2000; Walther 1994).

The positive aspects of collaborating face-to-face seem to diminish when co-workers are not in immediate close physical proximity of each other. Previous research indicated that co-workers residing more than 30 meters apart, are as effective as those collaborating across separate continents (Allen1977; Kraut, Egido & Galegher 1990).

Regardless of the impact on the effectiveness of collaboration it is generally acknowledged that face-to-face communication is the most information rich medium (Doherty-Sneddon et al 1997; O'Conaill, Whittaker & Wilbur 1993; Short, Williams & Christie 1976; Daft & Lengel 1984; Clark & Brennan 1991; Clark 1996). Close physical proximity seems to encourage and enable collaboration. The spatial dimension is important, even though its importance seems to vary.

## Collaboration in POS

It became evident to me through this study that POS development presupposed a collaborative mode resting on a threesome model:

1. It was a community of internal as well as external developers. And it was implied that including more developers would facilitate better and more efficient knowledge sharing as to avoid re-inventing the wheel.
2. POS was a way to bring in more structure around decision making and resources, e.g. by introducing a cook-book example of best practices together with formalized rules and roles relating to specific and desired conduct of the developers.
3. The information system of POS consisted of a set of standardized tools for development and communication.

During interviews it became pertinent that he POS system also needed to facilitate means to enable developers to reach common understanding in terms of the proposed work on the projects they were collaborating on. Many developers referred to the importance of face-to-face interaction on particular occasions when resolving specific aspects of development. The developers frequently returned to this issue, even though many of them had no face-to-face time at all with some of their team-members. The most common explanation among the developers I interviewed were that HP had an extensive and overarching travel freeze that was both related to the 9/11 incident, and also related to the cost saving moves that I have already touched upon in Chapter 4.

The Open Source phenomenon has proven to us that the importance of face-to-face interaction is significantly reduced in such a context, possibly to have little or no relevance for the successes of the projects developed.

When studying the Linux kernel, Moon and Sproull (2002) presented a set of lessons for organizations to learn in terms of improving distributed work. Moon and Sproull (2002:397) particularly denoted the qualities found in the Linux developers

and they reportedly exhibited certain qualities, e.g. taking initiative, being persistent and pursuing activism.

Moon and Sproull (ibid) also suggested that formal and informal reward systems need to be in place supporting sharing and discouraging hoarding. Moreover, they suggested that work products ought to be transparently accessible to anyone in the system. This is related to issues of Intellectual Property rights which are traditionally utilized by organizations as compared to Copy left[43] which is often utilized in Open Source.

Moon and Sproull (2002:398) did not out rule the importance of strong leadership as provided by the founder Linus Torvalds himself, but suggested that alternative models probably would work just as well.

But why where the POS developers that I interviewed not completely satisfied with collaborating in a virtual setting? And was there perhaps a pattern to be discovered? It was really something that might strike us as odd, particularly when considering that many of the developers were actually participating in Open Source on their free time and moreover some of them had even been keen proponents of bringing Open Source in to the realm of software development practices at HP. We might at this point start to ask why – but I will consciously avoid answering at this point.

## *Striving to achieve common ground*

Whenever people need to accomplish collective action, i.e. to cooperate they need to reach common understanding of what to accomplish. Clark and Brennan suggest that:

> It takes two people working together to play a duet... To succeed, the two of them have to coordinate both the content and process of what they are doing (Clark & Brennan 1991:127).

[43] According to Stallman: "Copyleft is a general method for making a program free software and requiring all modified and extended versions of the program to be free software as well". In Free Software Free Society, selected essays of Richard M. Stallman, 2002, p. 89

Common understanding or rather the process of accomplishing *common ground* is according to Clark and Brennan (ibid) dependent on what the individuals want to accomplish together, and also the medium of communication.

POS developers were as already reported struggling with establishing common ground with no or minimal face-to-face interaction. This was a frequent comment amongst developers and described by one of them:

> We can write and we can talk and we can do a lot of things to establish common ground. This is just my experience that it is only just two or three of the whole team actually runs the show. The challenge is how do you without seeing the person, how do you know that the entire group understands it. You can usually tell by their reaction or their expression. You can't get that all the time over the phone. Getting closer to having things like videoconferencing would help that. Teams need to budget time, to spend money, and fly to the other locations. There is nothing that beats being face to face occasionally.

Common ground constitutes of mutual knowledge, beliefs and assumptions (Clark & Carlson 1982; Clark & Marshall 1981; Lewis 1969; Schelling 1960). According to Clark & Brennan all collective actions are built on common ground and how it accumulates (1991). In communication, common ground gets updated through a process of grounding (Clark & Schaefer 1987, 1989; Clark & Wilkes-Gibbs 1986; Isaacs & Clark 1987). Grounding has different shapes depending on how the communication takes place, e.g. through face-to-face interaction, or mediated by computers. The process of grounding is also shaped by the purpose of the communication.

Clark (1996) suggests that it is preferable to strive at grounding with those techniques available in a medium that lead to the least collaborative effort.

Different mediums (face-to-face/ telephone/ fax/ teleconference/ computer mediated communication) have different constraints, i.e. they require different efforts and corollary different costs are involved to achieve effective grounding. In some instances, face-to-face communication can be too costly for the organization, and mediated communication may be preferable (Hollan & Stornetta 1992; Sproull &

Kiesler 1992; DeSanctis and Gallupe 1987; Jarvenpaa & Leidner 1999; Morley & Stephenson 1969; Nardi, Whittaker & Bradner 2000; Walther 1994).

POS collaboration was mainly taking place outside the face-to-face realm. When comparing characteristics in the Progressive Open Source (POS) of this study to the characteristics of face-to-face conversations brought forward by Clark (1996) and Clark and Brennan (1991) it becomes evident that the modes of communication differ in certain aspects.

## *Characteristics of face-to-face collaboration compared with POS collaboration*

In **Table 4** I present a set of characteristics that pertain to face-to-face collaboration and to characteristics that are valid to POS collaboration. The dimensions, i.e. characteristics of collaboration in social space are adapted from Clark & Brennan 1991 and Clark 1996.

***Table 4:*** *Characteristics of face-to-face collaboration compared with characteristics of POS collaboration*

| **Characteristics of collaboration in social space** | **Characteristics of face-to-face collaboration** | **Characteristics of POS collaboration** |
|---|---|---|
| Co-presence | The participants share the same physical environment | No/Minimal co-presence. Internal and external participants can work from different geographical sites |
| Visibility | The participants can see each other | No/minimal visibility. (Limited use of videoconferences as a complement) |
| Audibility | The participants can hear each other | No/minimal audibility (exception e.g. telephone conferences among project leaders) |
| Instantaneity | The participants perceive each others actions at no perceptible delay | Fairly high, but not complete instantaneity (the participants can view each others contributions, although they are geographically dispersed often working in different time zones) |

| | | |
|---|---|---|
| Evanescence | The medium is evanescent – it fades quickly | Permanent records |

| **Characteristics of collaboration in social space, cont.** | **Characteristics of face-to-face collaboration, , contined** | **Characteristics of POS collaboration, , continued** |
|---|---|---|
| Recordlessness | The participants actions leave no record or artifact | Records – (participants actions are open for scrutiny and it often results in a permanent artifact – the source code, and written documentation in forums) |
| Simultaneity | The participants can produce and receive at once and simultaneously | Fairly high simultaneity – although limited capacity to participate in oral communication. |
| Extemporaneity | The participants formulate and execute their actions extemporaneously, in real time | Deliberate actions. |
| Self-determination | The participants determine for themselves what actions to take when | Limited self-determination (limited use of language due to the written medium, text-based communication lack body language and other non-verbal cues.) |
| Self-expression | The participants take actions as themselves | Self-expression exists |

Participants in face-to-face interaction routinely use a signaling system whose function is to enable the interacting parties to coordinate with respect to meaning (Kraut & Lewis 1984, Kraut, Lewis & Swezey 1982; Duncan & Fiske 1977). These aspects of conversational exchanges allow interactants to construct their shared communicative environments on a moment-to-moment basis. As a result, the meanings of utterances are more appropriately thought of as a joint product arrived at collaboratively by the participants than as a property of messages encoded by the speaker and decoded by the listener (Clark & Wilkes-Gibbs 1986, Krauss 1987).

Different types of communication arrangements (e.g. messages transmitted face-to-face vs. those transmitted over the telephone; spoken vs. written messages) can affect the process of constructing shared communicative environments in several ways (Clark & Brennan 1991, Krauss & Fussel 1990, Rutter 1987).

First, different modes of communication may limit or alter the type of information one has about one's addressee and, consequently, the sorts of prior suppositions one can make.

Second, the amount and quality of feedback and the ease with which it can be obtained may vary substantially among different modes of communication.

In Progressive Open Source, developers were to a large extent depending on communication arrangements such as e.g. electronic forums, secure email, code repositories, project documentation and telephone. Since most of the communication were presented in written form and also openly available to the rest of the community of developers, communicators had to be cautious on how they expressed themselves as to avoid misunderstandings. They had also to carefully consider what kind of information to reveal, since 3rd party developers were part of the community. IP rights had to be defended while leaving non-crucial information free and open.

POS development encompassed large and sometimes very complicated development projects, including a vast array of collaborative relationships between developers. POS development was constructed to support distributed software development, i.e. members – (internal as well as external) – formed project development teams regardless of where they were physically located.

In practice, the POS teams were formed and conducted development without ever meeting each other face-to-face. Some teams however were still relying on face-to-face time at specific occasions, e.g. when it was practically feasible and justified given the task at hand.

My interpretation is that that there was a trade-off between functionality and economic criteria, i.e. there was a desire to enroll as many actors as possible regardless of where they resided geographically whilst coping with the constraints of having no or minimal face-to-face interaction.

Previous research indicates that many innovative and advanced applications (such as POS) have failed. Not because of inadequacy of the technology, but rather as a consequence of the way the systems are designed, i.e. not taking into consideration

the way in which individuals interact and collaborate in the traditional physical work environment (Grudin 1988; Markus & Conolly 1990; Galegher & Kraut 1990; Moran & Anderson 1990).

I am not suggesting that the idea of POS in itself was at fault, only that certain tasks or situations were perceived by developers as more easily coordinated and handled in real time face-to-face interaction, since some activities were more intertwined and perhaps more inseparable from concerted interaction with other developers. We can perhaps label this a preferred participation framework, i.e. face-to-face interaction was considered the preferred participation framework in particular situations (cf. Goffman 1981).

In the following section I will give an outline of when it was perceived as important for the developers to actually meet face-to-face. I have categorized the occasions in terms of being related to:

- The Work Environment
- Technology Development, i.e. task related

## Work Environment

In this section I will report on issues directly related to the work environment, i.e. occasions that were related to existing in a social environment provided by POS and to which its members somehow had to adapt.

According to Berger & Luckmann (1966/1991), social life is made possible because individuals succeed in creating common frameworks. Interaction and understanding are keys for collective action, and corollary trust building efforts and resolving conflicts are important.

Participants of POS were interacting to construct shared conceptions of what to accomplish in their respective projects. In this sense, POS can be perceived as a technical system, since it consists of a "specific combination of machines and methods employed to produce a desired outcome" (Sproull & Goodman 1990:255).

Prior research has indicated that people act differently in the presence of other people than they do when they are alone (Sproull, Subramani, Kiesler, Walker & Waters 1996).

## *Instances when face-to-face collaboration was preferred*

In this section I will present frequently reported instances and situations were developers and sometimes managers express that face-to-face collaboration is to be preferred. This is their experiences. It is worth noting that most of the developers and managers that I interviewed already were collaborating with no or minimal face-to-face interaction.

Developers frequently returned to experiencing that there were certain dynamics that would not come across in a virtual setting. Face-to-face was often perceived as crucial for understanding and responding to the needs of the collaborating partners. This was highlighted by a developer:

> For instance with one of our suppliers, we'll be traveling down to do what we call a retrospective with them. Over the phone or even with a videoconference, you can't see facial expressions very well. But to be around and see a pause or see an expression positions you better to know how to respond or to know what the real situation is.

Throughout interviewing and discussing issues related to virtual collaboration it became obvious that it constituted a challenge for many of the developers. This is a summary of the most significant reasons they often referred to:

- In a virtual setting it was perceived to be more difficult to check for alignment, i.e. were the developers 'on the same page' – or not?
- In a virtual setting it was perceived to be more difficult to encourage dialogue between developers.
- Virtual settings were perceived as less personal, e.g. it did not allow the collaborators to read each others body language.

- In virtual settings personalities was perceived as not showing off as well as in face-to-face situations.

Developers also referred to other situations were they either tried to ensure face-to-face interaction, or were experiencing difficulties when face-to-face interaction was not practically feasible. I will present a few of the most common experiences developers had.

This example refers to a situation related to bringing in third parties in to the development process. Reportedly before allowing third parties to become collaborating partners HP managers made sure that they got a chance to meet face-to-face in a formal meeting. They were also eager to meet the key developers – even before they were allowed to bid on the project. This is how a manager describes the situation:

> But all the vendors that I'm now working with, first I had a formal meeting with them where they came here. I've toured their facilities, and I've met their key players, all before we got into the bidding process. It was part of the qualification of who do we even want to bid on the project.

The following example relate to another issue frequently reported by developers working in virtual settings. It was also something that I had also noticed when participating as an observer/listener in meetings regarding CDP.

The meetings that I refer to were often managed (chaired) by one person. Often the meetings were held over an electronic web conferencing system.[44] This technology encompasses having all the participants securely connected over the telephone while also allowing the participants to follow the meeting agenda on their respective data screens.

Minute notes were taken simultaneously – often by the person who was in charge of the meeting (often a CDP manager). My experience was that very few of those that were actually participating were actively taking part in the discussions. It

[44] Web conferencing from WebEx.

was often only a one-man/woman show in the sense that the participants (almost without any exceptions) only replied if directly asked to do so. And mostly the discussions were related to the already established agenda which to a large extent had action items (things that people were explicitly asked to do or had been asked to do on previous meetings) and their responses were often mostly related to presenting a result or a solution to something decided in previous meetings. Frequently people would log in to meetings – but then go on to do other things. Sometimes they were 'caught in the act', i.e. the person who chaired the meeting would actually pose a question – and that person would not respond back.

Through my interviews it became pertinent that negative feedback reportedly were withheld in POS. Discussions on the forums were in a way self-censoring, i.e. members did not express their day-to-day difficulties instead discussions were held at a general, conceptual level. This phenomenon is well captured by this developer:

> I probably don't express this so much because I'm just one small division out of lots of divisions, and they tend to not want to get too much into specifics of one small business. I mean it tends to be higher level or general, so it's really not a forum for some of the things I have. I feel like I'm always the one that's bringing up all these problems, and I really hate to be the one that's all negative, but it's really not a forum where they want to hear specific problems. It's really they want to talk general concepts, so it's a very broad group. I mean this vision team includes people like for training and people from HR, and they're really not interested in what goes on day to day in software engineering at HP.

Yet other developers commonly returned to instances of trust, or perhaps rather lack of trust when interviewed. Their experience was that trust tended to develop slowly in teams with little or no face-to-face interaction. Developers reported that they perceived it as difficult to get to know each other and become confident in how to address and present ideas and issues. This developer and project-leader expresses a similar experience:

> I don't know if you've heard about the organizational liaison community. They're trying to build that community and have representatives. And I did assign somebody in my team to be the organizational liaison. So he is now supposed to gather that input on specifics, but he hasn't really; I don't

> think they meet very often. I think it's once a month for an hour. And again it's one of these communities like division from where you've never met face to face, and you don't know really who's out there. I've never seen these people that call in, and you don't really; same with the organizational liaison. It's going to take some time to really build a working community because they don't know each other and it's not a comfortable thing yet. I think he feels like he need to develop a mechanism to gather input here, and he needs to get comfortable with how to present it and who to present it to, and whether they can take action. So I would say that's kind of an immature process as well.

It also was a frequent topic during interviews issues related to authenticity, i.e. the instance that developers perceived it as more difficult to be authentic during meetings in a virtual setting, especially in big groups were people don't know each other from the outset. They often described that they preferred meeting in smaller groups, face-to-face when resolving conflicts. This is illustrated by a developer:

> And it might be a place to talk about how you would do it, but even so it's a pretty big group, which doesn't mean face to face, so people aren't necessarily being authentic in these conversations. So it's kind of a dysfunctional way to make these types of changes, but I think the problem is huge, and when it does get solved, it'll probably be solved in a smaller group, maybe just development managers figuring out how to do this better.

It was also interesting that many of the developers were so conscious about being permanently recorded, either in text or during video-conferences. A manager described for me how he was eager to capture the flow of ideas during meetings, but that developers were reluctant to allow it. The manager describes it like this:

> I think people are very conscious about anything that's permanently recorded, of voicing it. Many people; I suggested this at one time also. A very, in the best technical meetings there is a flow that develops, and an awful lot of technical details come out in a very short amount of time. And I've found that frequently after the meeting we would lose some of those details, that people would forget. People would solve this or didn't really dismiss this idea...
>
> [So you generated a lot of things and they just get lost.]

> Right. Somebody suggested once let's videotape the meetings solely for the purpose of remembering afterwards what was said. And maybe we only need the tapes for a week or something. Absolutely no one wanted that. Nobody wanted to be recorded, people were afraid of being on record as saying something and then being help accountable for it later. And that's not at all what it was for. It was just for the purpose of capturing the ideas so that we could slowly unwind it afterwards and remember everything that was said, because some of it happened so quickly.

## *Virtual collaboration is perceived to work better*

In this section I will present frequently reported instances and situations were developers and sometimes managers express that virtual collaboration seems to work to their satisfaction. It is worth noting that the developers and managers that I interviewed already were collaborating with no or minimal face-to-face interaction.

The interviews typically conveyed that initial face-to-face interaction facilitates establishing a collaborative relationship. According to the experiences of the developers virtual collaboration seemed to work suitably well once a relationship was already established. This is depicted by a developer:

> The other thing is by doing that, you get to know a little bit more about who that person is and how you might have to work with them, so there's a little bit of personality development that you learn. Those are things you can't get through electronic means. There's certain, I can't even describe it. It's relationship, you have to have a certain amount of physical knowledgeable relationship with a person. I work with a company in Italy now, and just heading over once and they've come over here several times, and they have a local team that just having interfaces with them regularly. When I get on the phone I can have a more efficient effective conversation because I understand who that person is and how they feel about things.

During interviews it became obvious that developers were developing strategies for upholding distributed relationships with collaborating actors. E.g. communication between the distributed team members would often be kept on target by allowing communication to go through designated team-members, i.e. communication was funneled through developers responsible for particular issues. These designated

communicators collected all the relevant issues and questions in to one focused message a day. The designated communicators acted as intermediaries or filters for communication. It was highlighted by developers that the strategy of delimiting communication through one or a few collaborators were facilitating communication and enabling virtual collaboration. This strategy is described by a developer:

> One thing that helps is that all the communication from Wipro[45] has been through one person, at least all of the technical questions and there's another person with more of business questions. They usually bring up all the questions in one message for the day or something. But that helps a little bit because you don't get like ten messages in one day you get one, maybe two.

Many developers explained that POS solutions typically seemed to be more beneficial for teams that have to work in a distributed setting. And the advantages seemed increase with growing distance between members. This instance was elucidated by a developer:

> I think it works better if we couldn't just walk downstairs and talk to the person we need to talk to. If it were all just strictly within a small company, I don't think it would have an advantage. If there were two different buildings located across town, then the advantage would be greater. And if there were multiple buildings, like in HP we have people on the same project in two different HP buildings, and we have us way down here in San Diego, then the advantage is a lot greater.

The developers I interviewed often were collaborating with third party developers from India. Since India is quite far from the US, project teams consisting of developers in the US and in India rarely or never got face-to-face time. They relied solely on the forums for communication regarding the projects, which they thought was quite efficient given the collaborative tasks they had at hand. This is the experience of a developer collaborating with third party developers from Wipro:

> We would never meet with the people in India; it's just too far to go. But the travel, that's also a long flight and a huge time difference. I should say

[45] Third party developers residing in India.

> that the forums are like newsgroups, and there's enough there for us to communicate back and fourth to keep them going.

It was suggested to me during interviews that virtual collaboration seemed to work better when the collaborating actors had high technical competence, i.e. when the collaborating actors were perceived to have the ability to solve problems independently. This and similar comments were reported by developers:

> I had one experience when I was working with Microsoft I outsourced a print driver development to a team in Microsoft Japan and we never even talked to them on the phone we did just email with them and that worked out, I think a lot of it has to do with people on the other side. Their technical competency was very high and didn't require a lot of hand holding, they were up to speed very quickly and they were able to understand our source code and our tools very fast and didn't require a lot of assistance.

Developers and managers often referred to privacy issues and discussed them in different ways. It is clear that privacy can take on multifarious meanings, and it clearly not conveys one coherent concept. My interpretation is that privacy in the case of POS had something to do with protecting a private sphere, i.e. when developers were not meeting face-to-face they expected to be more private, they were not keen on revealing all sorts of information, but liked to keep it somewhat more on a general 'private' level.

Through the interviews it became pertinent that virtual collaboration facilitated by technology such as e.g. video phones, were not gaining momentum, mainly because of privacy issues.

In this case, when discussing the potential use of a videophone, I assume that it is the information that is traveling in both directions, i.e. between the two talking on the phone that is at stake. Privacy undoubtedly has something to do with the subjective experience of having power to control information about oneself. It relates closely to the definition found in the Merriam-Webster dictionary, i.e. "*the quality of*

*being apart from company or observation* or *freedom from unauthorized intrusion"*[46]. But, in my interpretation it also had something to do with privacy which is considered a core value by most Americans. This is succinctly expressed by a manager:

> This is a group of innovators - this is not the standard population. And I had done an informal poll of some people at the time. I said assuming video phones were available at a reasonable cost, that they weren't outrageously expensive, would you buy one and use one? And I didn't get a single yes, not a single yes from this group of innovators, which totally shocked me. And their answers were mostly ones of privacy. They said, well what if I don't want to see, if I don't want the other person to see me? I said well, you turn the camera off. Well, I don't want the other person to get offended that I'm not letting them see me.

The last example that I bring up in this section relates to teams that never meet face-to-face. They often described that even though they didn't prefer to working and communicating in a distributed setting, they still believed that the communication mediated by the POS solutions worked sufficiently enough. At least good enough to be convincing to people that the benefits were larger than the drawbacks. They often referred to motivation, and how important it was for them to understand the value of working physically dispersed. Moreover, it was important for them to be convinced that the tools of POS were really working sufficiently enough. Typically the developers were under significant time-pressure which added on to these concerns. This is well captured by a developer:

> I'm working in a really complex environment. I can see the value of learning this tool, they will get it, especially if they're working with somebody that's using the tool and says yeah, I've been using it. It really has helped me. So I think that all of us that are working on CDP see the value of having to work across these geographies. It's really hard to work with a team that's far away. At least we're on the same time zone, we can call each other. But we don't travel very much. As a matter of fact, these teams have never met face to face.

[46] Merriam-Webster's Online Dictionary, http://www.m-w.com/dictionary/privacy

[And it still works?]

It still works, yeah.

In **Table 5** in the left column I have summarized those situations and instances when virtual collaboration was perceived to work better for the developers. In the right column I present those situations when developers reportedly preferred face-to-face collaboration. It is worth noting that many of the developers were collaborating with no or minimal face-to-face interaction with some, or a set of their team members.

***Table 5:*** *Situations when virtual collaboration was perceived to work better and situations when face-to-face collaboration was preferred*

| **Virtual collaboration was perceived to work better** | **Face-to face collaboration preferred when:** |
|---|---|
| Once a relationship was established, preferably f-2-f- | Trying to understand and respond to needs of collaborating partners. Getting to know each other on a team |
| If communication was funneled through designated team members | • Checking for alignment, i.e. are developers 'on the same page – or not'<br>• Dialogue between developers needs to be encouraged or enhanced<br>• Body language was perceived as important<br>• Trying to grasp the personality of a collaborator |
| When collaborating partners were geographically dispersed, in different time zones, ruling out f-2-f | Initiating third party collaborators |
| If the collaborating actors had high technical competence in terms of the project requirements. | Resolving conflicts and being authentic |
| If privacy issues were resolved | Building trust |
| If developers recognized the value | Seeking effective feedback. Negative feedback tended to be withheld in POS |

## Technology development and learning

In this section I will discuss issues that relate to technology development and learning.

The goal of POS was to stimulate learning and problem-solving through making projects available for others to study and imitate. Newcomers were granted access to the community's expertise, tools, and also the system of rules and norms that guide development within the POS paradigm.

Drawing on the community of practice literature, we may depict such an approach as emergent, self-reproducing, and also evolving entities and actors that are distinct from, and that often also extent beyond formal organizational structures, using their own organizing structures, norms of behavior, communication channels and history (Brown & Duguid 1991; Lave & Wenger 1991; Barab & Duffy 2000; Schlager et al 2002).

With the introduction of POS the strategic intent took a new path moving towards a more learner-centered and community based model, rather than relying on a teacher centered model of instructing. Whereas developing an online forum is not very difficult, attracting developers who will form a community is quite another thing. And in order to design online technology that supports professional development, it is necessary to understand the needs of the developers, and the work processes. Since communities of practices cannot be forced, but rather is an emergent phenomenon as suggested by literature, designers of POS needed to have awareness of the characteristics of the existing community in order to nurture it.

In terms of problem-solving activities, it is often assumed that the best way to improve the performance of a group is to improve communication between the members of the group. However, this is not always true. Under certain circumstances increased richness of communication may result in undesirable properties at the group level (cf. Hutchins 1991).

Groups can be better at generating diversity of interpretations of e.g. problems and solutions, but having generated a useful diversity, they then face the problem of resolving which solution is the best.

The POS community faced cognitive tasks that were beyond the capabilities of any individual community member. The performance of these cognitive tasks was

therefore always shaped by the social organization. However, the social organization may or may not have been appropriate to the task.

Face-to-face interaction and communication was perceived as crucial in the early phases of project development, i.e. the phase of the project where the architecture is laid out. In that particular phase engineers felt that it was urgent to capture human aspects of communication, such as facial expressions and body language in order to evaluate the content and the quality of the information shared.

In addition, the white board was frequently used to share ideas and to encourage brainstorming. The early idea phase when the architecture is being established and decided upon engineers expressed that communication mediated by the system was not sufficient.

I was frequently reminded during interviews that developers often perceived it as far more difficult for them to transfer information and knowledge in a distributed setting. And they typically referred to the early phases of project development (architectural work design phase) were they reportedly like to get together brainstorming, using whiteboards, flip charts and peg boards. This is well described by a developer:

> I don't think you could have seven electronic white boards going on seven different PCs around the globe and get the same feedback and energy as you're going to get if you have seven people sitting in a room with flip charts all around, and ink boards all around. That's when you're highly productive. But those are instances in time that build fundamental decisions that then can't be executed in isolation, so the tool steps in after the fact.

In an attempt to summarize the interviews, I conclude that it generally was perceived among developers[47] that finding and establishing common ground in the early phases of project development was easier when and if it was possible to see people's facial expressions and their body language. Moreover, finding and establishing common

[47] It is worth iterating that many of them actually collaborated with no or minimal face-to-face interaction.

ground was easier when they were in a face-to-face situation, since they often preferred sharing immature ideas in a closed environment. This is well captured by a developer I interviewed:

> What it won't help with at all in any way is the early phases of project development, which is where you need lots of flip harts and peg boards, and you need people to be brainstorming on well what if we did it this way? What does that mean over here? And quickly go over there and jot down a note related to that component. Walk back over to this component and say oh it works, now we can this behavior here. And what does that behavior mean? Well it's now put this storage over here. And that's the type of stuff that I don't think you're going to get dynamically through a system. It is a face to face meeting of multiple times, of multiple hours, of coming to grips with the fundamental decisions about building something.

Developers also referred to instances of issue resolution, i.e. a situation where the problem is already defined and the design already constructed. And they often told me that issue resolution (as defined above) seemed to work satisfactory for their purposes. This was a situation where the problem and the design was already worked out, which conveyed formulating researchable questions that could be posted within POS, and sometimes even externally available to Open Source communities. This line of reasoning was fairly frequent and pinpointed like this by a developer:

> Because now once you have this architecture or this component model, or whatever it is you're building, whether it's an install over a system or a set of documentation, you've got to brain storm. Everyone's on the same page. Now you can chop it up and say you do this, you do this, you do this, you do this. That's when this team will step in. I don't think you can step in too much before that.

On the other hand, developers depicted that when new components or functionalities were being introduced, or when a fundamental flaw was found, the developers preferred face-to-face brainstorming sessions with the right people. It is underlined in this quote by a developer:

> It's known as joint application development. You need the right people in the right room, to talk at the right level. And you need that to be face to face. It doesn't have to be; not all rules of having a meeting. It's got to be

> kept on target, it's got to be kept focused, it's got to be inside conversation, it's got to be parking lotted, issues have got to be captured and worried about later. It's got to be productive. And you get meetings that are face to face and are energetic, and that are brain storming, you want to let them flow. When they're not going to flow we're trying to go through some electronic kind of thing.

The developers often explained during interviews that the learning curve was considerably longer on POS. This was particularly emphasized developers that were working remotely from an HP site, (often 3rd party developers). A lot of the problems that 3rd parties encountered were related to not being experienced about the 'HP way' of conducting software development. Learning took longer because they were lacking face-to-face time with HP collaborators. They were often left on their own figuring out how to contribute to the projects. External collaborators often experienced that it was difficult to understand the big picture of projects and corollary it was perceived as harder to contribute. This is a third party developer:

> A long learning curve - longer than anticipated - simply because of the unknown. It took longer to get everything together than we thought it would. Just to getting the environment set up, as well as getting the project set up ...Getting people on the team who were not familiar with a lot of the ways; you give someone an existing code base that is huge, that's this is how the printer works. Now we need you to add something to it. So there's also a learning curve of having to figure out how all this stuff works so that you know the right places to add your stuff. Which isn't necessarily directly related to CDP, but it is one of the variables that cause a longer learning curve and getting up to speed on the whole project. I think most of the learning curve time is related to the fact that we work remotely from an HP site. That was difficult. It is related to the ability of doing software development outside of the HP environment.

Developers and managers often told me about issues related to telepresence. This encompasses issues related to the introduction of technologies that allows a person to feel as if he/she was present or to be presented, at another location, regardless of their true location. Such technology aims at providing the feeling of being in that other location.

The teams that I interviewed mostly utilized NetMeeting. NetMeeting basically was at the time of this study a tool that will enable the collaborators to see each others computer screens while sharing a conversation over the telephone. The goal of NetMeeting is to make it possible to display information to each other facilitating communication. Generally the developers and managers that I interviewed confessed that troubleshooting across NetMeeting was perceived as slow. Moreover, it was perceived as being significantly disadvantages over face-to-face interaction. By and large developers and managers were very conscious about being permanently recorded, regardless of media. This is exhibited by a developer:

> [What happens when you communicate in a virtual setting?]
>
> You don't get any feedback, so people are afraid to take risks. That's the thing that's just awful. So what you're saying now is presenting some ideas at a really rough stage, that might be fine in a face to face brainstorming. To write it down and post it someplace could really come back and haunt them. I totally understand why people wouldn't be amenable to that. Whereas you're implementing something; it's not risky, it's not controversial, it's something that everybody's bought off on, and I think that's part of HP's cultural problem right now, is so many groups are working remotely, over NetMeeting, and you sit in these meetings on the phone, and somebody's presenting something that nobody buys in to, and nobody will say it over the phone. You know? They won't say you know what, that's ridiculous!
>
> [How can you work around these things?]
>
> You don't. It just becomes very dysfunctional, because what happens is I think, at least in our case, unless it's really something you can't live with, you just let it go. You say well, okay, that's not very efficient, but it's not going to kill us if they go and do that. We'll just let it go. And then the times when you do speak up and try to get something on track that you think is not right, it's very difficult to work through, because you're not face to face. Can't see the people, and it's just really bad. So mostly it's the best way to work is to make sure the people have clear ownership of something, and they have the freedom to deliver that thing on their own. So it's just the interfaces where the controversy is. You know, oh, they're passionate about this; and it just takes a long time over the phone to get to know what people feel strongly about or don't care about, or one of those

> hot buttons in people. So we would almost always have a kick off meeting somehow, if we were going to have a group work together on something controversial, as opposed to relying on the tool to shut it down. I know. This travel freeze is really hurting.[48]

It was stressed by several developers in different teams that collaboration facilitated by POS seemed to work better in situations of mutual interdependence between teams. Mutual interdependence in their opinion ought to encompass a development process constructed as to proactively deal with bugs that needed to be fixed. This and similarly related problems were reported by teams. This is a developer:

> The consequences are severe, in that we were shipping and people were shipping a defect that they didn't need to have shipped, for quite a long time. And San Diego was not shipping because they had fixed it.
>
> [So what were the ultimate consequence of all this?]
>
> Well, they had to do a rule; so the good news is we told them we had a fix for it, and they were able to adopt the fix. The bad news is we took way too long to close the loop on that, in fact I think Vancouver said that they had a fix and they shipped down the fix for it and we said we already have this fix. Yeah, again CDP tools had nothing to do with that. The process had to do with that. And the breakdown in the collaborative process. But San Diego had some responsibility for it, and Vancouver had. But that hypothesis that I was talking to you about was when we were mutually-dependent, we were really good. Really good. When we became kind of service based, we weren't very good. We started to lose some of the capability that we had. So this gets back to this core team. They worked when they had the core team people. I saw that in my work. It directly affects them, and by the way, their work directly affects what I'm doing. So we have an interdependency. But here all of a sudden you lost sight of the overall picture. But some of the things that we used to do really well we're not doing anymore.
>
> [Like for example?]

[48] The travel freeze was related both to the 9/11 incident and also related to the cost saving moves of HP previously reported in Chapter 4.

> Well, that change, sharing changes. Another thing interesting happened to do with tools. We've lost; in our quest for a common tool set, we've lost a little bit of focus on what the effect; I don't know if it's what the requirements are, or what the whole; I guess it's the whole system,

In an attempt to summarize interviews it was suggested to me by developers and managers that the lack of interdependent relationships between teams was creating situations where:

- The collaborating groups were dismissing overarching goals to pursue respective group goals, (this was commonly denoted as Silo-mentality by developers and managers during interviews).
- Existing collaborations were breaking down due to shift of tools imposed by the collective POS initiative. Previously they had been relying more on bridge tools to maintain effective knowledge transfer and exchange between divisions.

Still, it became pertinent to me during the interviews that most POS teams expected to be able to minimize face-to-face interaction. The main bulk of the projects, i.e. general status, news and threaded discussions were already available to the community of developers which further alleviated the need for face-to-face meetings. This is expressed by a manger:

> [But do you still feel that maybe you would need less space to place interaction after you have implemented POS?]
>
> We believe so, yes. We believe that; or it will be more focused. Whatever interaction does occur, it will be more focused on a specific topic, because the general status and news should be already dealt with.

Yet, other developers challenged the view of perceived need of face-to-face interaction. And it was often stressed that face-to-face was preferred on instances relating to the customer, e.g. when fine tuning solutions. This is expressed by a developer:

> [So what kind of information or knowledge do you think could not take place in this kind of a virtual setting?]

> You would lose the fine-tuning of the solution to the problem. You would lose the fact that I work directly with the people who have the problem, to understand their business, to figure out which parts of it are more important, which ones are less important. So how do you make a custom solution for their problem?

In **Table 6** I present a summary of the most frequent comments developers and managers expressed during interviews. In the left column I present those collaborative work activities that developers perceived to work positively well in a virtual setting. In the right column I have put forward those collaborative work activities that developers preferred to solve face-to-face. It is worth re-stating that many developers and managers were already working with no, or minimal face-to-face interaction.

***Table 6:*** *Work activities preferred in a virtual setting compared with situations when collaboration face-to-face is preferred by developers.*

| **Virtual collaboration worked positively well for:** | **Face-to face collaboration was preferred when:** |
|---|---|
| Issue resolution | Establishing architectural design work |
| Reuse of existing solutions | Problem solving in early stages of project development |
| Established interdependent relationships between collaborators | Introducing new components. |
| Keeping track of the general status of the project, news and threaded discussions | Resolving fundamental flaws of the project |
| | Enhancing problem solving efficiencies |
| | Interacting on specific/focused topics of critical concern to the project. |
| | Fine tuning solutions with the customer |

## Rationale behind choosing computer mediated communication in research and development.

The interviews that I conducted with developers and managers indicated that lack of face-to-face communication constituted a perceived challenge in many situations. It was corollary interesting to look closer on how high ranking managers at HP were motivating the Progressive Open Source initiative, since it so heavily relied on computer mediated communication. When interviewed Lee Caldwell, former IPS Chief Technology Officer HP (2002), proposed:

> HP faces the major challenge of reinventing imaging and printing in the Internet era. This means that we have to develop meaningful contributions at a faster pace and provide a way for partners to leverage our inventions. The Collaborative Development Program work is a critical step facilitating breakthrough work and leveraging more quickly inside of HP and with (external) partners...

As Clark (1996) has pointed out different mediums have different constraints, i.e. they require different efforts and as a result different costs are involved when

individuals are striving at mutual understanding when collaborating. I could simply presume that large corporations, such as HP, have concluded that face-to-face communication to a large extent was too costly and did not make sense given a number of reasons. In addition, the size of the organization is one such reason, and moreover that experts were and probably will be increasingly more geographically dispersed, and lastly because the perceived challenges a head required a more effective sharing of expertise preventing re-inventing the wheel syndrome.

Looking back, product development at Hewlett-Packard historically used to be conducted by small teams working with one product at a time. But, as the company grew and more products were added on, it was acknowledged that more products needed to be developed in parallel. This required more individuals and teams to be involved in the process of research and development. Experts were increasingly globally distributed, and corollary the need for effective knowledge sharing had become at focal point when managing the organization. Product development was also conducted differently than in the past, e.g. the development of a printer required the active involvement of many teams, and it was the combined effort of both hardware and software development. There was a perceived need to stay in sync with what the other teams were accomplishing in order to fit their respective pieces of code in to a whole product.

It is worth depicting that software development differentiates itself a bit from traditional hardware development. Software tends to more subjective, perhaps a bit more on the artistic side. As one engineer expressed the difference:

> I think software engineers have a much bigger component of working in an artistic space, having space to think for themselves, having people who understand I have to own the decision. You know hardware, I get a set of specs coming to me. If I start delivering something that clocks a little faster than I was told, that's a problem. Whereas in software I deviate a bit, make something that's a little different or better or more enhanced, that's actually rewarded. It's actually looked upon about WOW, you know. To some degree... So it's subtle, but it is different, and it'd be interesting to see how it all evolves.

It was recognized by top management that it was needed to combine the efforts of many different experts, software as well as hardware specialists. And to conclude this section of this chapter, we may assume that it was perceived as a challenge to enable knowledge sharing and transfer, and that it possibly had an influence on the decision to open up the processes of research and development.

### *A short summary on why HP decided to introduce POS*

Before the introduction of POS the organization of activities related to research and development was perceived as fragmented by high ranking officers. Knowledge was perceived as residing in many different physical locations, across time zones and geographical sites. The explicit goal of POS was to form a community of developers working in a collaborative mode, sharing expertise and know-how.

It was also recognized that most products were to be developed by geographically distributed experts. HP had increasingly chosen to outsource significant and sometimes even critical parts of development. The aim was to distribute tasks that were perceived as non-strategic to highly qualified organizations, albeit in low-cost countries, e.g. Wipro in India. The level of outsourcing had increased significantly over time. HP had consciously strived at decreasing overhead costs and development costs by outsourcing the non-strategic parts of the development process. Outsourcing had become more common.

The process of opening up research and development was closely intertwined with the outsourcing strategy that the organization had chosen. The different POS efforts supported outsourcing, i.e. it made it easier for external actors to contribute. The goal was to facilitate geographically distributed collaboration.

There was a general demand due to a competitive market to speed up the process of research and development. By distributing work over different time zones the aim was to achieve 24/7 development.

A common approach for research and development (common methods, languages and tools) was presumed as a necessary condition for effective

collaboration. The standardized toolset was perceived as constituting a real challenge for some teams, and some of them reportedly were having serious troubles adopting the POS toolbox. (I will go in to that in more detail in Chapter 6).

Still, outsourcing was sometimes perceived as ineffective by HP developers. The initial cost of bringing external actors up to speed, i.e. educating them, was considered very high. HP was making a large investment when adopting POS, and hoped that it would pay off in the longer perspective.

Initially HP picked a strategy that invited many external partners to participate and also compete. The HP internal developers invested a lot of time in building relationships and creating a common understanding about HP research and development. Fairly soon, the strategy was reformulated and it was decided that it was necessary to choose one or a few strategically interesting partners. HP developers then experienced that they had wasted a lot of effort when it later on turned out that many external contractors never even where invited to bid on the contracts.

Developers were indicating that in their view HP had hired a group of inexperienced engineers to be responsible for the outsourcing. The HP developers also were inclined to believe that those engineers had a very limited network, which made them having difficulties navigating the community of developers, i.e. they had little or no insight in 'who knows what'.

The outsourcing model in use distinguished itself from the standard model of outsourcing which often encompass designing and designating a particular task to be completely executed by an external contractor. HP had chosen to strive to always keep a part of the development process internally. This was justified by the idea to control strategically interesting and valuable competence in house. In essence this means that if you view research and development in a value chain, dependent on which external partner you are collaborating with, the collaboration was to take place further down in the value chain. But, if the collaborators proved themselves reliable and competent, HP may decide to bring them earlier in the value chain. In the past HP did not usually outsource a complete project, but rather try to control and define their projects and their own internal competencies, before integrating the external

resources. Progressive Open Source was perceived as a way of incorporating external resources without losing control over the projects.

### *The establishment of social commitment*

It was perceived as necessary and important for HP when including external partners to teach them what it meant to do development in the HP context. As a result of opening up the processes of development for external partners HP was becoming increasingly aware that they did not have an efficient process in place at the outset. The process of opening up the development process forced them to formalize and make their processes more explicit. HP managers suggested that interaction with third party developers were also fostering the behavior of the HP developers.

The external actors had to meet the requirements of HP in terms of formal qualifications and standards. They also had to have an infrastructure (bandwidth) that would allow for large amounts of information (code) to be transferred between the collaborating actors. It was also perceived by HP as important that the network had a built in redundancy, i.e. if a channel of information goes down, other parts of the system needs to take over. Those external partners that could not meet the standards were filtered off. The introduction of POS conveyed collaborating with a common infrastructure and that in turned also had an impact on third parties, since they too had to make similar adaptations and also invest in higher technological capacity.

### *Technical solutions*

The organization had chosen a strategy which encompassed developing a rich array of products and they should all be compatible with each other. The organization wanted to invest in a software infrastructure that would allow external partners to collaborate over the Internet. Management decided to embrace an Open Source strategy.

The management team behind POS had chosen Linux development as a prototype (or rather the Apache web server) and translated that model in to their own context.

Hewlett-Packard strived at combining hardware and software development. Traditionally collaboration has mostly focused on software development. The goal was to create a mix of the two including third parties in the development. This raised the demand to develop common tools and languages. And it was recognized that there was a need on an ongoing basis to reach outside the boundaries to third parties in their supply chain more than what they had done in the past.

An Open Source strategy was perceived to provide the organization with several advantages. By choosing an open system (not Microsoft NT) the organization could avoid expensive licensing fees. By choosing Open Source software infrastructure and tools the organization could avoid becoming dependent on Microsoft. The Open Source platform also provided more freedom to include external partners. External partners were required to adopt the same platform. They were also persuaded to use the same tools and languages. HP offered the external partners an outline of the architecture of the product to develop.

## Conclusions

Within an organizational environment, such as POS, it is possible to discern how certain tasks and situations related to the social work environment are embedded in practices that seem to favor face-to-face interaction over virtual collaboration. Developers rely on clues provided in this setting in order to make sense of and coordinate certain actions and activities. POS development, regardless of the efficiency of the technology at hand, seems still to be dependent on a realm of tacit interaction used by individuals in real-world situations.

A major challenge for dispersed organization conducting research and development is to organize work so that the participants can effectively use one another's expertise and know-how without frequent face-to-face interaction. In this case study, the participants in the project experience situations that require close physical proximity.

It is often claimed that multidisciplinary collaboration promotes innovation. Innovation is in this case study defined as successful implementation of creative

ideas, tasks, and/or procedures (Amabile 1988). In engineering, innovation is closely linked to technical discoveries or insights, new ways to use existing technologies, or radical approaches to solve problems (Hargadon 1998; Henderson & Clark 1990; O´Connor & Rice 2001; Utterback 1994).

There is a tension between the benefits of working distributed across disciplinary and organizational boundaries versus the costs of organizing work and relationships facilitated by computers mediated communication. Multidisciplinary projects may require new approaches to organization, fluctuating between different modes of communication, i.e. face-to-face versus computer mediated.

In this chapter I have used face-to-face dialogue as a prototype to evaluate communication mediated by technology. I used it as a screen in order to fully grasp the challenges that people need to cope with when working physically dispersed while still depending on efficient collaboration.

There are at least three very important features of face-to-face interaction that influence communication. According to Bavelas et al (1997) face-to-face dialogue encompass unrestricted verbal expression, meaningful non-verbal acts such as gestures and facial displays, and instantaneous collaboration between speaker and listener. Clark & Brennan (1991) also suggested that face-to-face conversations are basic and that they exhibit features that have an impact on communication taking place subjected to other conditions. It is assumed that face-to-face conversation is the principal setting and that no special skills are required. Corollary, communication outside the face-to-face realm requires special skills, some that takes years of schooling, and many people suffer severely never quite mastering the skills needed to excel.

Features such as co-presence, visibility, audibility and instantaneity reflect the immediacy of face-to-face conversation. In conversations taking place over the phone, e-mail, or facilitated by computers when working remotely albeit collaboratively, people try to accommodate to the limitation offered by the communication system in various ways. Some people seem to adapt quickly and are very efficient in coping. Others seem to find it harder to function effectively when communicating outside the

face-to-face situation. Clark and Brennan (ibid) also suggest that the medium itself has far reaching effects on the course of language use. Non-verbal communication such as speech and body gestures is evanescent, whereas writing is not. This implies that those systems that depend on or that favors written communication has impact on the communicators and the way they are coping with the limitations of the system. In addition, Clark & Brennan (ibid) also contest that face-to-face conversations allow participants to be in full control, whereas communication in other settings restrict participants in what they can say and when.

When communication is taking place outside the face-to-face realm it becomes obvious that the non-verbal aspects of communication, i.e. anything other than words themselves that communicates or affects the messages embraced by the words are left out. Meta-communication is commonly used to describe the nonverbal process.

Important features of the nonverbal process are e.g. paralanguage. It is an inflection or an emphasis applied vocally which is applied to words that can have an effect on the impact of the message. It has the capacity to completely change the meaning. Silence also in an important tool in communication. But, outside of the face-to-face situation, it may mean nothing, i.e. an important decoder to reveal feelings and attitudes are left out. Cultural implications are also important in communication since it in many ways define how people think, act, live and communicate. With cultural aspects of communication I am including not only regional differences, but also differences that depict numerous cultures of the world. It is essential to know more about people and their background in order to understand their way of communicating. Another important type of nonverbal communication is body language, also known as kinesics. Body language provides instant feedback in a face-to-face communication which improves its effectiveness.

In this case study, people are to a large extent depending on communication other than face-to-face. This has implications on their capacity to collaborate. They also use different strategies coping with the limitations that the system imposes on them. Some people are naturally visual thinkers, which also makes it harder for them to function in a system favoring written communication.

But why (as I asked earlier in this chapter) where the POS developers that I interviewed not completely satisfied with collaborating in a virtual setting? And I know iterate that they were already copying with a minimum of face-to-face interaction, establishing strategies and so on.

Moon and Sproull (2002) suggested that there are lessons to be learned from Open Source development. And I will recapitulate the most relevant lessons they suggested and compare with POS development.

Firstly, Moon and Sproull (ibid) suggest that Open Source developers exhibit certain characteristic, e.g. they take initiative, are persistent and are generally labeled as active. This seems to indicate that organizations ought to consider this when recruiting. Only, POS developers are also frequently Open Source developers. So that does not seem to contribute to a complete understanding of the perceived problem.

Secondly, Moon and Sproull (ibid) argue that a reward system ought to be in place that support sharing and discourage hoarding. This seems to be somewhat of interest to POS, since developers within POS were still highly motivated by filing patents. In fact, filing patents was more rewarded officially than collaborative behavior. Even, though some managers reported putting larger emphasis on such behavior in the yearly evaluations of the employees.

Thirdly, in response to the lessons proposed by Moon and Sproull (ibid) similar to Open Source projects, POS projects are transparently accessible, albeit to selective actors complemented with rules and roles that somewhat were restricting sharing, especially pertaining to collaboration with third parties.

And fourthly, POS did not lack strong leadership, in fact quite the opposite. The initiative received a lot of attention and support from high ranking officers within the organization. This was true of POS as a whole, except perhaps corporate source, which did not receive sufficient recognition.

Perhaps we can conclude that the most important factor is hidden in the obvious, i.e. the fact that POS projects are not Open Source projects. They are subjected to other economic realities. The projects typically have to be developed

within very sharp time limits, and at a cost that is competitive in a market situation. And all developers were painfully reminded of that when HP were facing financial difficulties.

The perceived need of face-to-face interaction may ultimately be understood by contemplating all the complexities that large software development projects consists of. And since, as traditional theory on communication indicates, face-to-face is the most information rich medium, corollary it becomes more critical in an organizational context to rely only on mediated communication techniques.

# 6. THE TOWER OF BABEL – A QUEST FOR UNIVERSALITY IN LANGUAGE

> All translation, I suppose may be reduced to these three heads. First, that of metaphrase, or turning an author word by word, and line by line, from one language into another... The second way is that of paraphrase, or translation with latitude, where the author is kept in view by the translator, so as never to be lost, but his words are not so strictly followed as his sense ... The third way is that of imitation, where the translator (if he has not now lost that name) assumes the liberty not only to vary the words and sense, but to forsake them both as he sees occasion; and taking only some general hints from the original ... work as he pleases (John Dryden, *Preface to Ovid's Epistles*, 1680).
>
> Poetry is what is lost in translation (Robert Frost, Quoted in *Robert Frost: a Backward Look* by Louis Untermeyer 1964).
>
> Language is by its very nature a communal thing; that is, it expresses never the exact thing but a compromise ... that which is common to you, me, and everybody (T. E. Hulme, *Speculations*, 1924).

This chapter deals with aspects of language in a collaborative environment. More specifically, it deals with issues of language as experienced by engineers working in different geographical sites, with different native languages, utilizing common tools for developing software in a corporate context. In the POS project at HP, a unified model of tools, methodologies, language use and process, was

implemented as to prevent developers to fight with each other over how to develop the projects, and also producing products that are amenable for reuse and also are compatible with each other.

This 'harmonization of language use' seems like the story of the Tower of Babel, backwards. In the beginning of time all people were speaking the same language given to them by God.[49] When man tried to build a tower that would reach the heavens God punished man: *"If as one people speaking the same language they have begun to do this, then nothing they plan to do will be impossible for them".*

The biblical story offers an explanation on why we have so many different languages in the world. From the perspective of the Old Testament, the tower of Babel becomes a metaphor for the division of the human race into groups unable to communicate with one another. The biblical story of the tower of Babel may also be viewed as a metaphor for the introduction of technology in to the life of man. By being curious man pushed towards tasting the fruit of knowledge. When thrown out of Paradise man faced precarious tasks, living life in uncertainty and constant endeavor. New tools and methods for survival were needed in a world of increasing challenges. The urge to move further (higher) lead to additional risk taking. The mutual language of the past (when living in Paradise) was abandoned in favor of developing new technology and corollary new languages. The common understanding was forever lost.

According to the Genesis, the tower of Babel is the second major engineering effort, after Noah's ark. And it is also considered a fiasco. Why? When trying to make a closer assessment of the Babel project the first thing that strikes us that it is that it seems to have all the prerequisites for success, e.g. a clear mission of what to accomplish, plenty of people working on it, an abundance of material, they had plenty of time (no time constraint), and adequate technology. So why did the project fail before it reached the limits of

[49] Genesis (11:1-9)

technology? A plausible interpretation is possibly that it failed because people were unable to speak with each other (break-down in communication) and hence they were unable to coordinate their efforts (collaboration was inhibited). Lack of communication lead to disputes and enmity, and ultimately isolation was preferred over internal strife.

The introduction of a common system for developing software is an effort to unite different areas of expertise. The idea is to promote common tools, rules and language use when developing new products in order to gain more efficient reuse, products and services that are compatible with each other, and ultimately more cost effective. Is it paradise regained or will new problems arise as a result?

It may be asserted that alignment of technology, i.e. alignment of technology and language use, in itself causes loss of efficiency for some groups of experts. It may also be claimed that problem solving becomes myopic, i.e. all problems must be solved with the same tools. Is it favorable to use many languages or is it supreme to use a 'universal' language? These questions are rhetoric, and cannot be answered within this scope of research. However, in the chapter I will exhibit some excerpts of statements concerning difficulties of interpretation and understanding that has been experienced by developers in their encounter with people, cultures and languages other then their own.

POS is an organizational enterprise and in that respect it shows resemblance with the biblical story of the Tower of Babel. As Kaghan & Phillips (1998) has suggested, focusing on the construction of the tower helps us to view the Tower not only as an object but also as a locus of collective activity. This interpretation concerns language and its relation to the activities of the community involved in erecting the tower, and corollary the role of language as a mediator between differentiated but mutually interdependent communities. Accordingly, this interpretation highlights the role of communication in large collective endeavors. When applying this perspective of the interpretation to the

context of POS, it appears as if it's the tower of Babel backwards, i.e. it's a process of building a global network enabled by a standardization of language.

Developers from different subcultures, though within the same discipline of work, have to apply a common language and common tools to achieve collaboration within the POS world of software development. Unless an individual learns the language of the given field by becoming familiar with its rules, tools, and other modes of communication, he or she will remain unable to communicate with others in that field. It is a process of involving a large set of groups that has to act together and communicate in a network of interdependent systems.

The field of software development is a discipline that requires production, reproduction and expansion of formalized knowledge (mainly but not only in the form of code). Furthermore, knowledge possessed by the engineers are also embedded in the technical methods and specialized languages and vocabularies used by the community. In addition, tacit elements, such as community practices are also perhaps as important.

The results of teams are to be shared within the POS framework, in order to achieve more effective development producing products that are compatible with each other. However, and this has e.g. been pointed out by Latour (1987), results are not always incorporated in a predictable way, i.e. results are sometimes ignored and sometimes used, intentionally or sometimes unintentionally. It becomes evident that the inherent quality of the knowledge produced by POS is dependent on the understandings of the community of practice that produce the knowledge (the community of developers) and the various teams that are to make use of the knowledge produced by the system (and they may be internal as well as external to the organization).

The spelled out vision of POS is to move from a local to a global approach of development, with a clear concept of the production and maintenance of a unified approach controlled by management. Open Source development, and

corollary also POS development strives towards greater openness and sharing. However, openness and sharing between units and teams conveys giving up utilizing local languages and tools adopting the modes of communication supported by the POS system. Is this merely a dream of Paradise regained? A quest for universality in language, i.e. a language that can be spoken by everybody, has obvious advantages albeit its character of being imperfect.

Considering the use of many languages, there are several hurdles to overcome in order to achieve effective collaboration within the POS framework. I will treat all of them as issues of translation, or interpretation, and address them in the following order, issues related to:

1. The global distributiveness of the developers and corollary the fact that they have different native tongues, i.e. having to adopt U.S. English as the standard language of communication. In this category we will find also issues related to different cultures.
2. The different technical languages and tools in use, i.e. the battle of tools.
3. Communication using written language rather than pictorial language.[50]

## Language, cultures and global distributiveness

Translation has to do with the interpretation of meaning. A distinction is usually made between translation (which applies to ideas expressed in writing) and interpreting (which applies to ideas expressed orally or by body language). Etymologically, translation connotes meanings of carrying across, or bringing across. According to Brown (2002:5) translation seems to be related "to the process of making connections, of forging a passage between two domains, or simply as establishing communication". And Brown (ibid) continues: "Translation is...an act of invention brought about through combining and mixing varied elements".

[50] Pictorial language was favored by many developers that I interviewed.

> Considered from a very general point of view, this notion [translation] postulates the existence of a single field of significations, concerns and interests, the expression of a shared desire to arrive at the same result (Callon 1980:211).

Callon suggests emphasizing the way translation occurs on a common site and where significations, concerns and interests are mixed together, and as in the case of POS, this becomes the way translation takes place within Hewlett-Packard (the common site) and how the concerns and interests by various groups are intertwined, mutually influencing the local translation, and possibly also the outcome.

Translating interest is the key for understanding the process, according to Latour:

> We need others to help us transform a claim into a matter of fact. The First and easiest way to find people who will immediately believe the statement, invest in the project, or buy the prototype is to tailor the object in such a way that it caters to these people's **explicit interests.** As the name 'inter-esse' indicates, 'interests' are what lie *in between* actors and their goals, thus creating a tension that will make actors select only what, in their own eyes, helps them reach these goals amongst many possibilities (Latour 1987:108-109).

According to Callon, translation processes are enacted, i.e. it involves the process of establishing something entirely new.

Important concern for this study is related to the act of communication and its three dimensions: The developers, the ongoing projects, and the community of POS developers.

Translation theory purport that the more we know about the developer and the message produced by that developer, and the community of developers, the better acquainted with the particular acts of communication we will become. Having insights and knowledge about the circumstances that relate to the developers, improves the understanding of the messages being produced.

According to my apprehension of the POS project, there was something inhibiting/blocking/standing in the way which rendered it impossible for straightforward exchange of information (sometimes in the form of written text, and sometimes in the form of plain conversation, and sometimes related to tools or the system itself) Serres refer to this as 'noise' (1982) Only, without such 'noise' there would be no communication according to Brown (2002:7). And Serres (1982:79) continues:

> Systems work because they do not work.... There are channels, and thus there must be noise. No canal without noise. The real is not rational. The best relation would be no relation. By definition it does not exist; if it exists, it is not observable.

Accordingly, noise is related to the relationship between sender and receiver. It induces difference which leads to transformation of communication. And also, noise is what ultimately propels the process of innovation.

Communication between humans is both a dynamic and a complex process. It involves linguistic and non-linguistic information and it is sometimes enhanced by communication technologies. Such technologies may enhance the ability of individuals to interact over time and space constraints. However, they have to sacrifice non-verbal and contextual information and corollary the richness of human communication is reduced (Dertouzos 2001; Gundling 1999). In addition, such mediated communications may also disclose psychological and social differences inherited of different cultures and/or effected by a use of English as a foreign language.

One such disclosure was expressed by an engineer in charge of the support function of POS.:

> From the support side, strictly from the support side, the biggest challenge that we face is the multiple cultural differences that we see. There's language barriers that we all run into, and when you're trying to explain to someone, here in the United States we have a specific

> language set that we use frequently. When you're dealing with a user in India, he may not understand the same terminology or slang words that we use frequently, so the people that we have working on the support desk must very carefully structure how they speak, and that's been the most, the biggest change that I've had to deal with, with the support staff that I have.

Another example concerns the difficulty to communicate, given different native languges:

> [So if we look at the spectrum on challenges, you mention culture differences as the biggest one. So you sort of leave the tools and the technology aside?]
>
> In some cases we do. When a user calls and we're having a difficult time understanding what their actual need is, are they having a hard time getting access? Are they trying to do work with the CVS[51] side of it? Are they trying to actually upload source code? You need to try to find that out, and track that information down first. They're not always clear in explaining it because they've not used the tool before. [Okay. So it's hard for them to detect where the source of the challenge is.] Right. So the technician on our end has to be able to ask the questions to get a concise answer from the user, and not offend them at the same time. It's very easy to insult someone when you use a wrong word that in their everyday language may not be the same as what you use it as. So it's... Want an example?
>
> [Yes, give me some examples.]
>
> Well, when you work with users from Israel, they take great offense to anything that you may say that implies they may not be understanding what you're saying. And they are quick to hang up on you, and they are quick to go over your head to management. And we've had that happen, we've had that happen multiple times, and so now we know when a user calls; and we all have caller ID and we

[51] CVS is the acronym used for Concurrent Versions Systems. CVS is a version control system, an important component of Source Configuration Management (SCM). Using it, you can record the history of sources files, and documents. It fills a similar role to the free software RCS, PRCS and Aegis packages. CVS is a

> know what country the phone call's coming in most of the time. When we know that a call's coming from Israel, we know you very carefully think your phrases out before you say them. I mean, that's just the way it is. When I'm dealing with [Person X] I can speak quickly because we're pretty much on the same language level. When you deal with a user in India you have to speak very slowly and very clearly, because their English skills are not quite up to what we're used to. But it happens. You will insult people once in a while, and take your punishment with it, and off you go.

The examples tell us about well-known problems in intercultural relations; of acceptance and understanding of other people. These problems have been focused in a great number of studies, and are well understood by now. For our purpose, we need not here go into this domain. However, it is obvious that the POS project experienced problems of translation, both between different native languages and between different culture codes.

## *Architecture problems and scalability issues*

A 'global' workforce also means a large scale, i.e. adding more and more humans and non-humans. In order to achieve greater speed and reliability of the system POS were facing the challenge of scaling up to meet the demands of the rapidly growing community. The growing pains that POS were facing forced a version upgrade, i.e. in organizational terms a translation from a small to a larger scale. And the translation required taking into account a subset of constraints, e.g. the implications of adding new features (components) to the system given the size of the community. Such a translation was perceived by HP developers and support staff as an obstacle, thus creating a tension, or a dependence on 'others', i.e. the external vendor that are the providers of the POS architecture.

production quality system in wide use around the world, including many free software projects (www.nongnu.org/cvs/).

The concerns and interests by the developers, the support team and the external vendor became intertwined, thus influencing the translation. Ramping up then became the process of establishing something new, something precarious, and risky when so many are relying on the same system. The tower of Babel was shaking in its very foundation and the cracks emerged ad hoc threatening the ongoing development. The dependence on a single system of communication exacerbated the problems. Performance is a general issue with almost any large system, and also if the system supports a large number of users, scalability becomes yet another important and difficult issue.

This developer was describing such an event of scaling up the system, and the implications which also provides some insights on the potential vulnerability of large community relying on a common system.

> See that's where we, I think, we have better insight than they do, because when we look at our databases and we do things, we always say how do we scale this sucker? How do we scale this thing, that's the most important thing, because we, no matter what feature we add, it has to scale to thousands and thousands of people. And a lot of times they (the external vendor) forget about that, you know? [How big the community is.]
>
> Yeah, exactly. One developer will test it and it will work very fast on his machine. Well that's great for one machine, but when you have 500 machines hitting it as one time, what's the effect on it? And this is a new experience for them. They've not ever developed an application this large, so there are growing pains. And we've dealt with this in the past, and we'll continue to deal with it in other relationships.
>
> [But what did you do? If everything was down, what did you do? Did you just go back to the...]
>
> We gave them time to make changes to the code, to make it faster. [So is it solved now?]
>
> We're at about 90%. It's taken 2½ days of long conference calls and late night programming, but we're to the 90% effective stage. And I would imagine by the end of this week we'll be back to where we

were. But part of that, the biggest part of that is a new product, a new company, new size market, your test plans have to change to that. And so we're helping them now to develop test plans, where they had done it previously. So now we know, well, I've stepped in dog do. Now I have to clean it off and write a better plan to avoid it next time. And that's what we'll do in the future.

[So was this too many changes at the same time that they hadn't really; they had no simulations?]

Well they did, but they weren't trying to test the right areas of the product. A lot of the changes they made, we didn't know some of the architectural changes until it was too late, otherwise we would have warned against some of those changes.

[And so they went ahead with something that you weren't really aware of.]

I really like this. I mean, when the site goes down, that's when my support team does its best work, because we can handle the calls, we can get the information in, we get the information out, we get it to Collab, we coordinate with [Person X] and [Person Y] I and [Person Z] and all the team to say how do we go forward with this. And then we test it.

[So how would it happen, how many calls does he get?]

We had 57 calls in two days, and normally we get about 45 calls in a 5-day period. So we basically tripled our normal call rate in two days.

[So what happened with the other one, this big community? They must have known something.]

Most people are more passive/aggressive. The people that call are the ones who...

[Who don't care or they're waiting for something to happen?]

Well, like for me, for example. I don't call the support desk. I never typically do, I just sort of okay, I understand this now. I'll go do something else and they'll come back and try it again. You know? Most people; this isn't the first time we've had down time. So they're I would say earlier on, if this would have been the first time, we

> would have been out the roof, but this is sort of okay, another roll-out, another problem. We've sort of got a track record on it.

## The battle of tools

This section highlight issues related to an ongoing *battle of tools*. Although I use the term allegorically it denotes the important role tools play both from the developer perspective as well as the organizational perspective. As have been iterated many times in the thesis, innovation played a key role for Hewlett-Packard when introducing POS. In other words innovation was not regarded as something that happens randomly or through pure serendipity (even though that probably occurs also).

The realities of producing products that are special purpose systems, e.g. a printer, is that the computer and corollary the software is completely dedicated to the device or system it controls. Many of the teams that utilized POS were producing such products, i.e. they had to comply with the realities of embedded systems. Consequently, the development was subjected to particular management efforts.

Technology as provided by POS, allegedly facilitated the establishment of standardized processes, rules, roles and tools across distributed teams and locations. The standardized approach was implemented for purposes of instilling greater compliance and coherence, but also for improving development processes through enabling developers to be more productive.

Market consideration were crucially important since the embedded systems often are mass-produced, corollary benefiting from economies of scale. Hewlett-Packard, were (and are) competing on a global market and thus issues such as cost efficiency and productivity improvements were important whilst taking into account security and intellectual property protection. That said and emphasized we can conclude that it was an environment of high complexity

both in terms interaction between systems and between the software within the embedded components produced.

A standardized approach supposedly (at least hypothetically) enable software code re-use and re-engineering, facilitates rapid innovation and release cycles that intimately responds to shifting market demands. In the light of what I have mentioned standards and standardization play important roles as coordination mechanisms and they become *regulatory instruments* much in the same way as directives, markets, formal organizations, and normative communities (Brunsson & Jacobsson 2000).

POS was in a broader notion similar to the meaning of the term standard, since it includes rules, roles and most importantly – a common toolbox, which clearly defined how software development was to be conducted, it became a regulatory instrument which delimited the use of locally adopted and adapted best practices and tools. The toolbox consisted of tools commonly used in Open Source software development, with minor exceptions (some tools were *translated* and *transformed* in order to better serve the particular needs of the organization, e.g. in terms of protecting IP rights). Ultimately, the goal of the toolbox was perceived as the necessary lubricant facilitating global collaboration, producing products that work together, i.e. one solution, common architecture, and complete visibility among collaborating actors.

> We wanted it standardized so that everyone is using the same tool. Thus everything is in the same place. Second we wanted to have a tool that was based on CVS, which is more mainstream than open source community. Reason being some of our products we had the opportunity to move them to an open source model and maybe contribute them to the open source community. In order to understand how that community operates we needed to be running that. The third reason was, a littler bit better performance from the remote locations. People in Mount Laurel could get to the information rather quickly. Some of their remote labs had trouble getting to all of the different systems that we had. By going to CollabNet where it is hosted in an environment that is built for remote access we're going

> to stop that problem. There are the three main driving goals. We also wanted to take costs into account. For this solution everyone was running their own collaboration environment each with a major lab. That is a lot of duplication of effort. It's just that we have to spare at tools. It's not one place to get everything. It's a bunch of different tools. I think everyone will embrace going to a single source for everything. Sure there will be some hick ups in trying learn it. If there is anything significant then we haven't implemented it right. It needs to be easy, intuitive and that is the driving force.

The move to a standardized integrated toolset meant giving up local efficiency in order to enable the larger community to collaborate according to developers. This was a frequent comment from developers:

> It's given me one set of integrated tools that are good enough. I wouldn't say they're the best tools, but they were good enough to get my job done.

The managers I interviewed often made comments related to the growing community of developers with different needs. As POS expanded it had to include a vast array of groups who previously used to work with specialized tools.

> I guess as the community grows, I guess you have different needs. That is happening. A good example of that is San Diego utilizes something known as change sets that allows batch up changes for particular feature or defect together and more easily then exclude that for one product and include it for another rather than saying that you have a common code base and everyone draws equally from it. That's something that's unique to Sirius and CDP has grown well beyond Sirius, CDP needs to meet the needs of all its users rather than just the Sirius users. That means that we are moving from CVS to MKS[52], but we're not moving the change sets. That means that we have implementers in San Diego who are not at all enthusiastic about going from their change sets into CDP. Sometimes to be perfectly

> honest, we have built tools to assist us but that isn't the only way of doing it. It's just the way that everyone is used to doing it and change sometimes is challenging. For myself change in challenging. But for some people change is very challenging. So these problems could grow as the community grows also.

HP has a history of being a decentralized company in the sense that the local business units were allowed to have independent decision making and that was perceived by managers as an asset of the company. When introducing a standardized process for development (including a common toolbox) that independence was a source of tension, a battle of tools if you will. The process of standardization was mandatory, i.e. a decision was made above the local management of each business unit. And the independent decision making of the past was delimited. The rationale behind implementing the standardized toolset was according to their own statement and assertions business reasons, e.g. cost-saving and issues related to providing better and coherent service to the customer. This is how a manger perceived the challenge:

> It can and one of the greater challenges that HP faces is that a strength of our company has been the independent decision making of each entity. In that independence, you have also created many problems. The prime example that comes to mind is you have a product based on another product within our division. Product A chose to do things using this particular tool set. Product B which is based on Product A decides to use yet a different tool set because they have they right to make those choices. My team then has to support both Product A and Product B in the maintenance phase. That means my team now has to learn two sets of tools. That isn't efficient for my team. But then it takes us work to move everything that was done in Product A to a common tool set between those two products. That's within a division. But when you start talking about interdivisional, you have many, many different tool sets, each one chosen for a perceived slight advantage. But you look at the whole of

[52] Mortice Kern Systems Inc. (MKS)

> it, you say we'll do those slight advantages outweigh the cost of maintaining each one of those different tool set and the training necessary to convert an engineer from one tool set to another. That's independence, but that also fights at your collaborative efforts. I am working on this tool set. You're working on that tool set. If we want to work together, I want you to come use my tools, and you want me to come use your tools, rather than having a common set between us that we're both using so it's easier for us to share what we're doing. So how do you think you should resolve this conflict? The way that it's being resolved is painful to many engineers, similar to the vendors. They're being told what that solution is. There's a lot of kicking and screaming about "Wait a second. You took away my right to make this decision." And the answer is "Yes, we did and there's business reasons behind it." HP is trying to centralize their support. The support of all those different tool sets is very expensive. When you can centralize into one support organization with one contract with one company, versus well, as an example, I was working on an effort with a third party that provided a real-time operating system. There were 34 HP contract with that company, which meant that it was close to the situation that when you sit down an airline, the person sitting next to you paid a different price and you're both going to the same place. And why is it different? Well, because they bought their ticket at a different time from a different agent. Similarly, the contract that we had was different than all the other 34, which meant that some were better, some were worse. And none of them were the same, which meant you had 34 purchase agents that had to maintain that relationship. That was real common. From what I heard from working with tool vendors, they said working with HP was like working with a different company with every division. Trying to consolidate us into that common look is painful because people can't choose their own tools, but it's very advantageous.

Managers often discussed the advantages of using a standardized toolset, e.g. reducing costs. A standardized toolset also made it possible to centralize the support function (again cost saving reasons). Other frequently advantages mentioned by managers were that they anticipated that a common standardized process would inhibit reinventing, facilitate reuse of existing solutions, which in

turn would also lead to faster learning within the community. The biggest issue (which was a recurrent theme during the interviews both amongst managers and developers) was the transition to a common source code management system (SCM). Again, (according to their own respective testimonies), for many teams this was synonym to give up (efficiency) in order to enable overall performance. And in some cases, interdependent collaboration which previously had worked well between teams, threatened to break down because the standardized toolset (SCM) was not sufficient for their respective purposes, and also in other cases the business units were not able to make a business case (prove its utility) out of applying the common toolset. Corollary the 'independence' of the business units was difficult to manage. This is a representative comment of how managers in charge of implementing POS perceived the situation:

> Then there are the benefits of being on the same toolset, there's the benefit of reduced cost, everything is centralized, you don't have to have a support team here. The benefit of some common processes, so people don't have to keep reinventing, the learning curve is less. What are the biggest challenges ahead, do you think? The obvious one is getting everyone on a single SCM[53], and that has lots of dimensions to it, the biggest one will be pulling people off of an established system that they know and rely on very much, so that's going to be a challenge.

Some of the local business units were very reluctant to join POS. The main reason behind this was that they perceived that the tools simply were not good enough. Less effective tools, i.e. the standardized toolset offered to them by POS meant giving up too much in order to enable overall performance.

Some teams wanted to wait for an improved version of the toolset. The local management was not ready to sacrifice their own results, and my interpretation is that these teams were at the point when I conducted my study

[53] Source code management (SCM)

(which was in an early POS-phase) basically doing so well that the reasons for joining POS simply were not compelling enough. And they still maintained an independent stance towards setting goals and delivering results. A local manager of one of the teams that were resisting the standardized toolset (change):

> Yeah, they're changing at least the source code control system, so we'll see how that goes.
>
> [Do you think that's the reason because it didn't have all the features that people required?]
>
> Yeah, CDP, for me, I see works really well for small development teams, with its current toolset.
>
> [How many, what do you mean my small?]
>
> Maybe five to ten people working on a code set at once. I see issues with the current source control system if you were to have a larger development group, especially if they were diverse location wise where they can't easily communicate the same, "I want to get my changes in. So a lot of my input comes from people that live with it day to day. I would say that the innovative pieces that we don't have to do anything, we don't have to do a lot of work to get up and running on the tool. You know, it's accessible, and our group has three sites that collaborate. One in Corvallis, one in Cupertino, and one here in San Diego, and they're pretty small teams. So it gives us a platform where we can all access each other's code and documentation and test records, without having to learn or install or maintain a whole big platform between three groups, because we don't have very much infrastructure that we can use. Now from the developer's point of view, I think that they think it's pretty far behind; the actual source code management technology is pretty limited.
>
> [Really. In what respect, do you think?]
>
> Well, many of them came from a Clear Case environment where it just had a lot of different tools from branching and comparing and putting notes in. So they feel that it's pretty limited as far as sophistication of the particular tool. And then even like the defect tracking tool that we use, our quality department won't even use it.

> They refuse to use it because they say it doesn't have enough information in it. Again, it's like when you try to do something that's collaborative, sometimes it distills down to the common set of things that everybody needs, and then people are moving off of other tools that maybe were more specific, and configured for their specific needs, and they miss those capabilities. So we're getting by with the defect tracking tool, but it's pretty bad.
>
> [So, if not, what are the obvious impediments or downsides, from your perspective?]
>
> So basically what I hear is that the tools just don't have some of the features that we're used to having. I know that other people; at least this group is not so hung up on change sets and things like that, but when you have a lot of sophisticated parallel development going on, you really need a lot of help from the tool. Being able to track changes, and back changes out, and track different versions, and share code, and watch what's being changed by other people. I'm really not able to tell you exactly what is going on, but it's based on CVS[54] they tell me, and it's just pretty ancient technology, very file-based, you know? It just doesn't really do a whole lot, and it may be just in the user interface, where it doesn't do a good job of displaying what was changed and different version numbers. It's just not the sophistication that the people I was talking to came from a Clear Case environment, which you can have your own user profiles versus project profiles, and you can do a whole lot of really complex stuff.

The POS community was growing very fast in the start-up phase. More and more developers were joining the collaborative effort which brought the POS lead to perceive the original toolbox as insufficient. While a standardized toolset was desired it was not static, in the sense that the increasing demands of the community mutually forced the POS lead to instigate a process of implementing a better and improved version of source code management.

[54] Concurrent Versions System (CVS)

The process of translating the needs of the community was an ongoing process, but also, it had mutual repercussions. Each transformation incurred certain amount of pain to the community, since they were impelled to change. And to mitigate the growing pains the POS lead tried to facilitate transformation though providing contextual support, in order to ease the process of migration. A POS manager described the process like this:

> It's interesting we have just chosen SCM in the last month. We're starting the pilots next week. And so a lot of the folks, CVS is working just great for them right now, they just didn't know what else was coming down the road. Until now we've been hesitant to say, " Oh, we have this next great tool." Because we weren't sure which one we were selecting, and now we're going to start the pilots, so we're still cautious to say that we're moving to MKS at this date because we want the pilots to prove. We haven't quite got to that yet, but I do know that there are teams that are saying, "I don't want to switch. It's going to cause me pain, I have a good process that works." And so a couple things were trying to do is, one: we're going to keep CVS around for at least a year after MKS has deployed, so people will have a consistent system. Another thing is providing support for all the migration of code from CVS to MKS, and do that for them so they don't have to do it. Along these lines as well, they aren't going to stop babying CollabNet and our support team or our framework team, they're not going to stop making improvements on CVS in the meantime because MKS is going to ramp up slowly. Also with MKS we're hoping that having a single source code system and bringing everyone onto that would be a benefit enough in itself that people would be willing to switch again if there's this support that will help you migrate and there's also that we're putting processes in place for MKS, so people don't just have the SCM tool, they actually have some tools to help them build their code with it as well.

Each migration incurred pain to developers since it involved learning new tools and processes. Learning to use the new tools was often left to the individual users, even though team leaders often take on the role as 'super-users' leading and facilitating learning and change. The effort involved in learning new tools

depended on their respective experiences of working with similar tools. This is the experience of a project leader:

> [So it took you a while to get started, to get all of these practical issues done. They had to do the training themselves?]
>
> Yes, and I had to try to write up some, and I'm still trying, documentation to get our team up to speed on CDP and CVS. At the same time I sent their team leader down there, the documentation I put together, then he modified it for their team and tried to get their team up to speed on using it. How well different people adapt to it varies. We have some developers within our team that has used CVS on their own, so they were pretty comfortable with moving to the CVS model, then other developers have used the HMS[55] system and so there's a paradigm shift of how you use the old system and how you use CVS. There's a little bit of pain associated with getting shifted over to the new system, because it's not just learning new commands, it's how you interact with the new database is different with CVS versus our old system.

The standardized toolset was a living entity. It evolved over time as the presuppositions for development changed. The developers were forced on a regular basis to switch tools and learn over again. This process was ongoing and inevitable. A developer describes the process like this:

> We went into a meeting, into a staff meeting, a couple of weeks ago and one individual started talking about CVS and one because most of the group had just at that point had learned within weeks they were at that point within just weeks of starting to use CVS. And one individual who is the least in some what of a position to change the tools that we use said that CVS or CDP are old hat and it is now the latest is MKS. And at that point there was a collective groan from the entire group because the last thing we want to do is learn a brand new tool. And then six months later have to learn a new one all over again. It seems to me that what ever you do in the future, we need to be a little bit aware that just changing tools for changing tools sake is

> not a great idea unless this new one has some compelling reason over CVS and CDP, or CDP people ought to be aware of what is happening out there and change the mood. [You mean to have a more long-term perspective?] Yeah. Changing the tools is hard on everybody.

And since the process of transformation was endless, i.e. the toolset was on a regular basis subjected to major changes and alterations, some teams decided to wait to join POS in order to avoid pain, or at least minimize the incurred pain. This developer (and project leader) described this process of transforming the standard toolset as something that POS leaders not publicly wanted to talk about. The developers perceived that POS leaders were being tactical and not telling the truth about how the toolset is constantly subjected to change – as not to scare teams to join the POS effort.

> Well, it is, but they can purchase it and license it to use at all these sites. So it's at least a major company where you expect it to be there. But what we don't want is something that will end up taking a huge amount of maintenance and support, and I think they believe they can have between the X company, which I think is Y systems and CollabNet. They believe the support can happen between those two companies, and that HP will not pick up a big support burden. And they don't want HP to have to develop tools that require a lot of maintenance and support. But they've also been a little reluctant, there's been some discussion about developing anther tool beyond MKS, of providing another tool beyond MKS, but they have been reluctant in CDP to publicly explore that, because they are afraid that if they say MKS is not the final tool, there will be a third phase. They're afraid then people will use that as an excuse not to go on MKS. Because if they said okay, here's MKS. This is the second phase, and we're working on a third phase to meet more demanding requirements, then they'll have all kinds of people saying well we'll just wait for the third phase, because we don't want to switch twice. And so they're reluctant to say that there's anything besides MKS.

[55] Source code management system used before introducing POS

> They're kind of taking the position with MKS that's it, that's the end, that's all you get.

## *Technical offerings of POS must meet the business needs of the community*

Transformation of the system was an ongoing process. And the POS lead played an important role in terms of updating the community on planned changes. They were responsible for taking the best interest of the whole community into consideration. Technical improvements had to resonate with business needs. A POS leader and manager describe the transformation process like this:

> [When you decide on an improvement, is that posted on CDP too, so that people can follow...]
>
> Actually yeah, it's not well enough. A couple of things in term to the community that I'm leading, so we have the sponsors, we're working with them to make sure they're aware of what CDP is providing and get their help sometimes if we need to raise awareness and push people into migrating. Also we need to make sure our technical offerings meets their business mead. That's what the lab manager leveled from the ground up were talking with, we have the organizational liaison community. So it's through them that we have our monthly meeting and that was one of the first agenda topics this last week when we had our monthly meeting.

## *Replacing many tools for one*

The process of replacing a large set of specialized tools for one standardized toolset involved great risk, especially if the community is fairly heterogeneous, as in the case of POS. Some groups of actors within the community tended to be more dominating than others, influencing the process of translation and transformation more successfully than others. It was a political process. Attending only to the needs of the dominant actors ultimately jeopardized the collaborative idea, which embraced the notion of giving up (at least a little) in

order to contribute to 'the big family'. The process of transformation was evolutionary, in the sense that major actors tended to, at least tried to block out the interests of others. Metaphorically speaking the battle of tools was threatening the erection of the Tower of Babel. This is a POS support person in charge of keeping the standardized toolbox up to date:

> Another issue, I think, from a broader perspective is the R & D community. And if they disengage from us, then we have credibility problem, we have no direction, and that's a risk, I think, we're always worried about. Like John Lee and the DTS teams, Michelle Watson and the SCM teams, they really have gathered people that have a big stake in those areas from the R & D community to represent their needs. And we can't guess what their needs are. Right now we do have them a part of the CDP community, but in the future, I don't know what that's going to look like. If we meet all their needs do they go away, and we're just happy, or are we always in this evolution, that's something I'm not clear about.
>
> [Okay, but you did raise some interesting issues here, but could you be a little bit more specific about those two communities, and why you are concerned that they may not...]
>
> Well right now they have specific deliverables. The deliverable is to bring on this new tool called Scarab.
>
> [Are they concerned about that issue, that particular issue?]
>
> Yes.
>
> [Okay, and why is that. Is it because they feel that it is not suitable for them?]
>
> No, they feel it's suitable, but scalability is always the biggest concern. You can have a flashy tool that works great for 5, 10 people, but when you try to scale it, and after that's where everybody's worried, because this tool is meant to replace hundreds of other defect tracking tools. And it's been sort of positioned as the next generation tool, and it hasn't been tried or true, so everybody's nervous about that. It's a newly developed product, it has no track records, it's hard to know whether it's going to be good.

The POS support team in charge of providing the standardized toolset, was in a very difficult position trying to make the community as happy as possible. The standardized toolset was perceived as an important cost saver for the organization. But, on the other hand, the local business units were still relying on the independence they had been granted in the past. The independence to say yes and no to implementing changes was still very much an option, at least for large and influential groups within the community. Taking all this into account replacing many tools for one became like walking on mines, i.e. at least difficult. And the support person continued to describe it like this:

> It's dealing with the unknown. You are trying to develop a tool that you want everybody to use, and like [Person X] mentioned, there are many tools that are currently in use. And each individual pocket is quite happy with their tools. To bring them over in to one tool, you know that you're going to have some resistance, so you're trying to write the best tool that you can up front that's going to handle everybody's needs. And that's one of the concerns, is have we touched everybody to make sure their needs are being met? And that little bit of fear, of not knowing whether you've actually gotten a hold of that last engineer, to have him give you his idea. If you miss one person is that one person going to cause you enough, raise enough of an issue that it's going to stop the development or stop the implementation of your product.?
>
> [So, could that actually happen?]
>
> Oh sure, sure.
>
> [Meaning what? If they say that this Scarab is not going to meet their demand...]
>
> They would refuse to; they would not use it until a certain; they'd hold out. Yeah, we agree with that you guys are doing, but we'll have to wait until this feature is implemented.
>
> [They're afraid that it's going to be disruptive to their...]

> Yes, and one of the things that they're trying to do is to not have to support multiple platforms. Use one, use Scarab[56] as an example. If you do that then you don't have to support the older applications. That's cost savings for HP, because you don't have to keep the one person trained, and you don't have to upgrade that system. That money then can go towards the new Scarab product, so the longer these small applications stay out, the more cost this has for HP. The sooner this can be developed, the more cost savings. And we all the know that cost is a big determining factor, and whether you can use something or not. Will it save me money, will it make my team faster, those type of issues.

Standardized tools that worked well in Open Source software development was translated to the corporate context, only the conditions for doing so were significantly different in the HP business context. We may perceive development in the organizational context as subjected to limitations in terms of resources (time and money constraints) which make the process of standardization much more complicated.

Standardization inevitably leads to instances where teams could cope with those constraints. And the support person continues like this:

> And that's the fear when you're running it, trying to do that for a company that has 80,000 people. It's great to do it for one R & D lab that has 100 people, because you can touch all 100 people, but to try to touch that many usually is difficult. I agree. I'm going to jump back and go on a tangent here, but it's like the open source community is a very different place. Oh, that looks fun, I'll go there, you know? Here, do it. This is money on the line, this is business. Get to work. It's not that bad, but it's not as free and loose as the open source community. Oh, on my spare time I'll do that, you know? When I get a chance. Here we have milestones, deadlines, and in the open source community they have loose goals. It's not as...[It's okay

[56] Scarab is a Content Bug Sharing System (CBS), in which you can track issues for all types of projects: technical and non-technical, and where you can insert new issues. It is provided by CollabNet the external vendor in charge of the providing the common tools.

to grow organically.] Exactly. And that's not the case here. We have time to market is everything, so we have to have goals and milestones, and if you don't then it's hard to get things out the door.

[But is it okay to have different paths for a while, just to keep the community happy? I mean, if they're afraid to move to a new potentially disruptive technology, is it okay, or how do you deal with things like that? I'm not saying that I'm sitting here with some information, but this is just hypothetical. What if you have a community that is very dependent on a specific tool, and it's crucial for them to actually be in business? So what do you do? Do you take the risk of putting them on something, or what do you do? Or do you still keep supporting?]

Yes. Oh yes, that makes perfect sense. We have a tool like that. We have a tool that the lab uses. And to try to show them that the new tool will be just as effective and efficient as the tool they have currently has been one person's full-time job, basically. They go meet with the lab to find out what their specific needs are, bring that back, meet with the defect tracking team, meet with the source code management team and say this is the specific need for the lab. Can we incorporate that? And then it goes back to the lab and say we can meet 90% of your needs now, the other 10% will be in the future. Can you live with that? And the labs with then come back and say we need 95% of those features now. The other 5%, why don't we go back and talk to the defect tracking team, and we need those 5% additional features. And their push back is every day we're late on our product is a million dollars. Are you willing to put that on the line? And they're not. They've got things working. They have a known environment. They know how it works, they've evolved it over several years, and it's a difficult situation to ask somebody to lose a million dollars a day just to transition to a new product.

The POS support team tried to facilitate the transition for the teams when the timing was right. Migration from one tool to another was instigated in between projects, when time and money constraints were less crucial. This process was perceived as significantly easier for small teams than for large teams with specialized needs.

Collaboration facilitated by a standardized toolset has many advantages in terms of saving costs for supporting several platforms and tools. But the decision to choose standardization over specialization was not a straightforward solution for all the teams, since it ultimately was a business decision subjected to the laws of scarcity of resources. The support person also highlights how translation and transition goes hand in hand and that it is an ongoing process of mitigating the needs of a large community. He continues:

> But there are certain lulls in a project, or down time after they release a product. So there's certain opportunistic areas that we can jump in and work with them to transition them into this environment. A lot of the things they do to make their product work so well is a process. They use a lot of processes. The tools are just sort of secondary, and a lot of times these tools; they could either meet the 5% by more process or not use it at all. A lot of times they don't even need those features. They just are so used to it. They're so used to seeing it on the screen. I have that button, I have never clicked it, but it's always been there. It's that typical answer, because it's always been done that way. And so what the open source environment has done is come in and said we're not going to do it that way anymore. We're going to develop these features for you, and we're going to develop them quickly and efficiently. Can you see that? And part of the lab team has been very forthcoming. Boy, this is a great tool, and I'm so glad that I'm using it. But then you have other people that have come back and said the tool is not what we want. We don't want to take the down; but the lull time that [Person X] was talking about after a product release and before something new is coming is how we've keyed on some of these other small projects. We've gotten them off of their small system and brought them into CDP. We've used that lull time to convert everything over, and it's worked very smoothly. We'll have a challenge when we do some of the bigger instances, like the R & D lab, and things like that. So it's really been a lot of marketing. As a support person we do a lot of marketing. [Person X] was talking about the consulting. We go and actually meet with the people we want to use the product.

[But can you also provide how to migrate from one tool to another? Or are they expected to take care of a lot of those costs themselves?]

Well a lot of the R & D labs already have their own administrators, and so personally we haven't taken on a lot of that responsibility. We've said you guys know your processes. Here's out tool. We can talk technologies, how does this map to this, but we've left it up to their administrators of their current tool to help migrate that to this environment. But in the future we think that's an area that we can add a lot of value. So yes, that's something we're looking at. One other thing I was going to say is the goal for a lot of these people, a lot of these developers, they're subject matter experts in this particular area. And they're really focused on I need the best in class source code management tool. That's all I care about. And they forget about the bigger picture. And the bigger picture is collaboration, and as a result, there's things you have to trade off in order to have that environment. You can't have the best in class source code management tool in order to have collaboration. Eventually it could be that way, but that's just not the case today. You can have the best tool, but if the users can't get to it and can't use it, there's no point in having that tool. And that's the one thing that the CDP project has done, is we have made it so anybody with the right permissions can access these items. And that's the big value-add that we have. We're not working off of a server under somebody's desk. We're working in the enterprise, inside and outside the HP networking structure. And we're the only application that does that. And it's really hard, because our community is developers, right? We're delivering a tool to developers. This is their area. What are you doing giving us tools? This is our area, you know? So that's always a challenge. We should be writing this tool, not just using it. Yeah, I should be giving you guys tools.

[I guess you have to take everyone into consideration and come out with something that will potentially work for everybody.]

Right. There is a risk to that. Well we go through prioritizations with the R & D area. They're the ones who are helping set the priority and help setting the schedule. Yeah, there is a risk there, but we try to get more feedback from the R & D community on those areas that are

> most important to them. And try to minimize the risk by getting as much information up front as we can. A lot of the tools that they currently use are similar, but it's not an exact. And the feature set that [Person X] was talking about, that I mentioned earlier, having the button that you've never clicked on, but it's always been there, the feature set that they often; they'll look at is and say well, I need this one specific feature. And we'll go back and talk to them and oh no, I've never used it, but I want it to be there. I need to have total administrative access to your server. Well you don't have it currently, so why do you need it now? Well, I just need it because. So you have to answer that question.

## Language use preferred by developers

In this section I will discuss language use preferred by developers. It focuses on how they prefer to communicate in certain situations and also instances related to constraints of the medium, i.e. the POS infrastructure for communication. The insights are closely related to the interviews I had with developers (engineers) about how they prefer to communicate and how well the medium (POS) suit their respective needs.

A typical POS project consisted of developers from many different countries, often working remotely distributed both in terms of time and space. And still, in spite of ethnical and languages differences it seemed as if they were able to communicate reasonably unconstrained and effortlessly in collaborative projects.

In social sciences it has been conventional to find and explore differences between cultures, and in doing that research has primarily regarded differences between individuals as belonging to subgroups in terms of e.g. culture, gender, and also sometimes differences are explained as being pure exceptions, abnormal, deviant, or even as being random fluctuations (Maruyama 2005a).

Contrasting this view, Maruyama proposes that "in any social, cultural or gender group, even if the group is 'ethnically pure', there is heterogeneity of

individual perceptual/cognitive/cogitative/action types[57]" (2005a:2). And even further his research seems to indicate that: "these individual types are trans-social, transcultural and pan-genderical" (Maruyma 2005a:2).

Through Maruyama's research transcultural individual types were being depicted and categorized in four different PCCA-types.

The relevance for this study is that engineers as a group across cultural boundaries seemed to have much in common. They shared the same professional language, but perhaps more interestingly they tended to be non-verbal thinkers and nonverbal communicators, which meant that they prefer communicating with pictures, drawings and body language such as e.g. hand movements.

Maruyama pointed out that the effectiveness of their communication often become significantly reduced or even nullified in organizational settings depending on verbal documents. Verbal thinkers on the other hand are abounding among administrators and they are often incapable of understanding the importance of nonverbal thinking and nonverbal communication, "it is as impossible and hopeless as to explain color to congenitally blind persons, or music to congenitally deaf persons" (Maruyama 2005b:2). For the purpose of this study it is suffice to consider the following:

- Pictorial messages contain more information than written messages given the same space restrictions
- Pictorial information is faster to comprehend than verbal information, since information is simultaneous whereas verbal information is sequential
- Pictorial information may express relations which normally cannot be expressed verbally (e.g. describing a person's face to someone over the phone)

[57] Hereafter abbreviated PCCA.

Maruyama assert that engineers across cultural boundaries share similar epistemological traits, i.e. exhibit similar approaches towards problem solving activities, and similar pattern- recognition (2005b.) Pattern-recognition (cf. Margolis 1987) similarities in turn are explained by sharing a common language (professional language), similar formal education and corollary they have access to similar methods in situations that require problem-solving.

Taken together this may at least to some extent explain the success of Open Source software projects (cf. Moon & Sproull 2002). Developers collaborate, often without ever meeting face-to-face and it works regardless of cultural and/or native language difference.

## *Engineers prefer drawing*

Many developers expressed that they felt more at ease communicating face-to-face in problem solving situations. This initially seemed odd to me given the successes of Open Source development, which is a phenomenon that takes place in a virtual setting on the Internet. Only, as I discussed it further it became evident that it wasn't the medium that was the main obstacle but rather the language use. When solving problems developers expressed that they preferred drawing solutions over articulating solutions in text. This suggested to me that it was drawing pictures that facilitated their capacity to solve problems. This is how a developer describes it:

> Of I don't know if you know that word this... Of having a black board or on a white board group of people... Use our pictures we can't do it and that has always been the fun part of being with.. I will I would certainly miss that but you know maybe eventually the Web tools will be good enough so that the virtual black board will draw pictures everybody will share them any where in the world eventually that could happen. That's that probably is the only limitation I can think of one you couldn't use virtually.

While I was interviewing the developer was using his hands to gesture, and as I listened to the tapes and comparing with the notes I had taken during the interview I realized that his body language was compensating for the words that were lacking. He had trouble articulating in words. What he was really missing was to draw solutions, since that was his way of solving problems. Only, with POS, he had to articulate his ideas in text, and that was perceived as much more difficult for him. And then he went on saying:

> Just to reiterate that as an engineer I value drawings I even in high school I would take the napkin in the coffee shop and draw pictures and through out my entire engineering career I've drawn on pieces of paper, black board and shared my ideas through pictures or schematic. It's hard to do it if you in a controlled environment ... Not impossible but it's harder. [Would you say that it's easy for someone who is more comfortable writing or formulating rather than drawing or talking or verbal?] Yeah for a person who that's writing I think it's much better for them. But engineers are notoriously visual and pictorial.

## *The fishbowl effect*

Through my interviews a phenomenon that I will hereafter refer to as "the fishbowl effect", was frequently occurring and referred to by the developers. It relates to openness and visibility of communication facilitated by POS. It conveys a feeling that everything that you do and say becomes immediately obvious to everyone else in the community. POS technology facilitated openness but consequently it also besieged privacy.

Do we have to conceive this as a problem? I will consciously avoid answering this particular question, and only relate to the fishbowl effect as a source of privacy disutility and only from the point of view of the developers.

This perceived disutility often is the result of the disclosure of information that an individual does not want to be publicly exposed. What immediately jumps to mind is that perhaps this is really a good thing! And that it will

improve compliance to the system and its rules and processes! But I on the other hand often heard that developers frequently avoided exposing their ideas, especially verbally as in articulated text on the forums provided by POS. This was peculiar to me especially when considering that they also reported that very seldom was people publicly criticized, i.e.,' flaming'[58]was practically never heard of. And the reasons that developers often were expressing pertain to the presentation of self, which was perceived as important in the community of developers. What you say and how you do it in other words had connotation to who you were as an engineer. It displayed aspects of their qualities as being knowledgeable and qualified persons, even though in practice these particular aspects (in this case writing skills) have little or even no significance for the developer's ability to perform.

This developer talks about inability to articulate (in writing) ideas, which most likely says very little about his ability to develop software:

> I don't want to give you the impression that I don't like CDP or that it's not a good tool because, like I say, 90-95% of it is great and very useful and the other 5% stands out because it gets in the way of what you're trying to do. But, as far as that goes, I think we've covered the important stuff, like the fishbowl aspect of feeling like you're being watched with every little thing you do. To a certain extent I don't know how valid a criticism it is because I haven't talked to anybody who said, " I saw a really dumb post of yours upon the forum," or anything like that. [But you still feel that way, than it is important. It still concerns you.] It does affect the way that I post to the forum. Right. Another interesting thing is that, within a lab like this, just kind of word of mouth about an engineer's quality of work or something like that. I guess, what I'm trying to say is that it's difficult to judge another engineer until you've worked with him. You hear different things about different engineers and then when you go to work with them it may be a personality issue, or something else like that, and you may get along great with the engineer and think the

[58] Flaming is the act of sending or posting messages that are deliberately hostile and insulting, and which

> quality of their work is great. Whereas, under something like CDP, I could see where if someone who is a bad speller or has bad grammar or not a good writer or can't get his ideas across in writing as being looked down upon. It is a very much a written medium that you're dealing with and that engineers have to deal with and engineers are not known for their grammar or spelling, and so to a certain extent, I can see that being a real issue.

In the start-up phase of POS discomfort among developers stemmed from being *the* first person actually putting informal conversations and minute meetings out for others to view and review, and again "the fishbowl effect". This developer urged that it would have been preferable to keep the forums small initially, i.e. not to include the whole community at once, in order to establish a culture around openness and sharing. It was desirable to protect the presentation of self from unwanted attention. This developer talks about how the POS was set up in order to enable openness and sharing within the community and also how initially developers were reluctant to start using the forums:

> Well they were setting up a lot like that, so that people could go find all these reports, and they could go back and look at the data sets and so forth. Or they could peruse over them, respond to them, like, though I don't understand what you were trying to do by adding this level to that, or adding this catalyst to that. And the people can reply, try to clarify their reports, and so forth. So it had been fairly successful for this one community. When they came over to engineering, I was really excited about the idea of putting something in. I fought to actually be in charge of it, and I wasn't, unfortunately. And the person who was, I could not talk them into setting up lots of small, little Notes areas. So instead, they set up one big R and D-wide area, and the fishbowl effect stopped it. So nobody wanted to be the "the" person to put their meeting minutes out there. Nobody wanted to be "the" person to put informal conversation out there. There was no culture in place to use it, and nobody wanted to be the first person to walk out on the dance floor. So you just couldn't get it

happens occasionally in open source development.

> going. And I kept saying you need to make this small. Let's have one just for the eight firmware people on this project, so that they can pass information back and forth about it. You know, the build broke again last night; whoever did this, at least be aware it's broken, and if you did something yesterday maybe take a look and see if you were the person who broke it.

The "fishbowl effect" seemed to have an impact on how freely developers expressed themselves. They became a little bit more restrained and aware of how they express themselves. As illustrated by this example:

> Now there was a little bit of a fish bowl effect on Kona.
>
> [Okay.]
>
> So there were like four active people, and if you went in and looked at the list of people who had subscribed or whatever, there was like 200 people on the project. It's like, okay. So that hindered us a little bit. We were a little careful not to just say whatever, because we knew that there were a lot of people watching and listening. Maybe not too attentively, but just curiosity, people who had asked for access so they could go take a look at it.
>
> [But could you actually block off certain areas if you wanted to?]
>
> No. You might be able to with the tool, but as group we didn't get in to try and manage it that proactively. We would have had to have worked to do that.
>
> [So you really had to think twice before...]
>
> You're right. Probably the only thing that it caused us to be careful about is, how can I say it? In this particular there was a lot of porting of code from an origining project. We were creating a variant of it, so when they updated the original project, the parent project, we needed to take their changes and merge them with ours. And you just have to be a little more professional about how you reference the fact that the parent project might have broken everything. Or they completely rewrote something, and so you have to keep in mind that a lot of people are seeing this, whereas if it was over the phone, you know, I might have said, well what the f... are they're doing. You

> know, in an email I'd just go it looks as if there's been a lot of change here. So it didn't really hinder anything too much, but there was a little bit of a fish bowl sensation. It caught my attention only because back in Digital Equipment we would literally parade something, and there would only be five of us in it. And we'd yell at each other. But it was the sort of yelling between friends where you knew how to take it. So it's the same things we would say in accord or to each other, what were you thinking, you know? Or you'd make fun of somebody. You can make fun of somebody you know really well and there's no hard feelings. Everybody just laughs, including the person you're teasing because what were they thinking? We would do that in the Notes files because they were kept real private. And here it was a little more professional, which is probably not a bad thing anyway.

## Conclusion

POS was project that strived at constructing a scientific language, perfect in its chosen range of competence, i.e. within the field of software development within Hewlett-Packard. More specifically, it was a search for a translocal[59] globalized language that would support the distributed organization consisting of experts from different geographical sites. Eco (1995:73) has pointed out the distinction between the perfect language and the universal language:

- The perfect language – that which truly mirrors the nature of the world.
- The universal language, which might be imperfect, but can be spoken by everybody.

It was obvious that POS was a quest for universality of language, albeit striving for perfection it will never really truly mirror the needs and meanings of a dispersed community.

[59] Cf. Czarniawska 1999, 2005:110.

It's impossible to discard the political aspect of standardizing communication and language use within an organization. Searching for universality and perfection in language use has been equaled with fundamentalism, in the sense clinging to and glorifying particular sets of values (Berger and Luckmann 1995:25). Translated in to the context of POS, this means adopting standards, albeit from the Open Source, still meticulously translated into the context of HP, which means open and accessible only to those trusted.

Opening up development at the same time presupposes limiting the variety of language and tools. Groups of developers had to learn new languages in order to communicate and collaborate at all. The POS project was controlled, sanctioned, and supported by management. Paradoxically, in order to become efficient, the software community had to continue their relentless search for the universal language. Only the more successful they became the greater

> ...the chance that the repeated confrontations with plurality waiting outside the organization's door will have strongly traumatic, not to say existentially threatening influence on their members (Czarniawska 1999/2005:111).

If a common language leads to a totalitarian approach, i.e. less flexibility and room for creativity, then perhaps the search for perfection through a universal language potentially may become a totalitarian trap.

If the 'bazaar' it the metaphor commonly used for Open Source development[60], then for POS we must use the metaphor of the 'exclusive club', with membership rules controlling access, as well as conduct and modes of communication. The POS paradigm involved a belief that it was possible to identify the unique, necessary and invariant aspects of knowledge related to software development, only in practice it seemed as if there was a large portion

[60] Cp. The Cathedral & the Bazaar, Eric S. Raymond, 2001.

of complexity and heterogeneity that perhaps was better maintained and protected by greater diversity in terms of practices, languages, and tools.

If knowledge is to be understood as something that people do together, interacting not only with humans but also with non-humans, e.g. computers, then knowledge is to be viewed as a process embedded in the discourses and practices of the POS community. A certain element of heterogeneity always exists, even within a fairly distinct paradigm, which in this case study became evident when recognizing that concepts, techniques, languages and tools were used and translated into alternative forms by the different communities. Different teams inherited languages, practices, tools and even different rationales for their respective work that lead to difficulties or even no interest in acting as one. Therefore, I believe that we have to place larger emphasis on local systems of meaning and action and how it relates to the larger network of discourse and practice within the POS, making explicit efforts to understand 'language-in-use' and 'culture-in-action'.

Since managing the system becomes a delicate assignment, maintaining control while avoiding fundamentalism, I will focus on those issues in the following chapter.

# 7. MANAGING AND ORGANIZING – THE POS WAY

In this chapter, I will consider openness and concomitant surveillance as facilitated by POS technology which possibly implies self-disciplining behavior leading to outcomes of normalization. After that I will address how POS management distinguish itself from more traditional management and organization, discussing concepts pertaining to Open Source Software Development. Finally, I will touch upon issues relating to the role of management in POS.

## Visibility and openness

In order to understand how the introduction of POS influenced issues related to management and organizing, I analyze in some detail the concepts of visibility and openness, and also analyze norms that can structure thought and action in organizations. In particular, I discuss how visibility and openness as related to POS seems to influence developers and managers.

Openness and visibility somehow alters communication. In interviews developers often brought up that they had become increasingly more conscious about how they expressed themselves, e.g. when posting ideas on POS forums. It was evident both in terms of what information could be shared, how ideas and

suggestions were laid out, but also in terms of making sure that the text in itself was 'correct' in the sense correctly spelled, correct grammar and so forth. My interpretation is that the openness of POS in this sense led to instances of self-disciplining behavior. This is a developer:

> You are going to a broader audience; you definitely have to be... if I post something to a broader audience I always review my messages to make sure that there are fewer grammar problems. I definitely feel comfortable, but I'll be more careful and polish it before I send it out.

It was also evident that managers expected that the open environment of POS would yield a higher excellence in the performance. This is a HP manager commenting on what he expects might come out of the openness provided by POS:

> Working in an open environment creates a higher level of excellence in what people do, and a higher level of accountability for what you do. So it's not, again it's all exposed. It's all up front. And there's some vulnerability that comes with that initially, but as people get more comfortable with that I think they're also feeling pretty good about this is my work. I am going to contribute to the broader effort.

Self-disciplining behavior is closely related to what I will define as techniques of normalization, or norms which structure thought and action into categories such as correct and incorrect, desirable and undesirable behavior. A central concern of such acts of normalization is the notion of visibility as a prime factor for discipline and management. Along the same line of reasoning it is relevant to discuss the role of information and technology as means of orchestrating control within an organization. Cooper and Burell (1988:105) complement these ideas by viewing organizations as social machines that also produce elaborate discourses of information and knowledge in which human subjects are an indispensable part, of the material flow on which the discourses are inscribed.

I now introduce Michel Foucault's notion of panopticism, not because it is entirely undemanding to grasp, but because it is still a straightforward notion

which already has a familiar application within organization studies.[61] Moreover it fits particularly well for reflection on the introduction of new technologies in organizations and how they pertain to issues of management and control.

Panopticism is closely linked to the concept of openness. In essence the concept of openness is both the focal concern and enabler of POS. POS encompass introducing a system which is progressively more open internally as well as externally.

One of the many contributions of Foucault, in Discipline and Punishment, is that he acknowledges that control no longer requires physical domination over the body, but rather can be achieved through isolation and continuous observation. And for these reasons, I am going to argue that physically distributed collaboration between actors facilitated by technology that facilitates observation and monitoring through the establishment of open records of communication and work processes is a form of organizing that seems to alter the behavior of developers, and therefore also most likely the presuppositions for management.

Openness is a concept widely embraced by contemporary organizations as well as phenomenon of collective action, such as Open Source software development. Stewart Clegg (1998) suggests that perhaps we cannot assume openness as superior to its opposite, i.e. concealment. We can only rest with the assumption that when confronting in detail what is at stake, we understand that the situation is far more complicated (cf. Schwarts, Leyden, Hyatt 1999; Weber 2004) And that perhaps we can only assume other disciplinary practices. Clegg suggests:

> One should not assume an analytical endorsement of 'openness' in favour of 'concealment'. Whereas concealment has been the basis for the practice of modernist organization in the past, and such practice has become increasingly subject to criticism, one should not assume

[61] The principle of the Panopticon has been used e.g. when analyzing surveillance in research areas such as

> that technologies of openness will deliver a liberal ideal of an organization world of free and equal individuals. To practice openness, as much as concealment, also requires disciplinary practices of power - this much, at least, one should know from Foucault (Clegg 1998:45).

But, let us think more closely about what the concept encompasses. The panopticon is a concept originating from an architectural innovation consisting of a twelve-sided polygon with a central tower from which the superintendent (manager) could observe the behavior of institutional inmates.

The panopticon, as envisioned by Jeremy Bentham in the late eighteenth century, enabled the guards of the prisons to gaze at the inmates, only the inmates were not able to see the guards due to a carefully contrived system of lighting and the use of wooden blinds. It was an all-seeing place proposed with the intention of designing a safer and more humane prison.

Moreover, the Panopticon was conceived as a solution to control problems and even though initially applied to the context of the prison, its applicability also encompasses organizations. According to Foucault (1977:83) they often seem to resemble prisons.

The key principle of Bentham's Panopticon was inspection by an invisible God (the manager). However, the electronic surveillance of POS leaves slightly less room for such asymmetric surveillance, since the observer may also be observed, corollary the panoptic control of POS deviates from the original on certain aspect, e.g. developers are supposed to collaborate rather than remain in solitary confinement. Seclusion is replaced by inclusion.

As I have already touched upon in Chapter 5 and Chapter 6, POS facilitated the introduction of roles. The introduction of roles was a way for managers of the organization to delimit the available information, and also

human resource management and accouting (cf. Townley 1993; Hoskin 1998).

delimit the possibility to make contributions to projects. Users were defined in terms of being light-users or core-users, internal and external and so on. The POS was in that respect constructed to yield the appropriate information to anyone in the 'right' position. This feature was not utilized by all teams, but it was something that was desired by developers and managers, albeit for different reasons. They typically addressed this issue as a need for 'layered information'.

Foucault summarizes the major expected effects of the Panopticon:

> To induce in the inmate a state of conscious and permanent visibility that assures the automatic functioning of power. So to arrange things that the surveillance is permanent in its effect, even if it is discontinuous in its action; that the perfection of power should tend to render its actual exercise unnecessary; that this architectural apparatus should be a machine for creating and sustaining a power relation independent of the person who exercises it; in short, that the inmates should be caught up in a power situation of which they themselves are the bearers (Foucault 1977:201).

Power is in Foucault's analyses relational, i.e. it becomes apparent when exercised. For this reason, power must not be associated with specific institutions, but rather has something to do with practices, techniques, and procedures. Power in that sense is employed at all levels and through a wide array of dimensions. As Foucault suggests we must look beyond the concept of power as merely a commodity, it is no longer relevant to pose questions such as 'who has power' but it is more relevant to look at the 'how' of power, i.e. scrutinize the practices, techniques and procedures that alternates power relationships.

It is compelling to use the panopticon as a metaphor for understanding surveillance and enclosed techniques for management in the context of POS. Giddens (1987) also has contributed to ideas pertaining to issues of openness and surveillance. He suggests that: "surveillance in the capitalist enterprise is the key to management" (Giddens 1987:175). Moreover he suggests that we need

to make a distinction between two different types of surveillance (Giddens 1985:14-15):

- Surveillance as the accumulation of coded information
- Surveillance which is linked to the direct monitoring of subordinates in the organization

When thinking of the Panopticon and its original intent, i.e. to control inmates residing in prisons, it may at first sight look outrageous to even try to bring it in to the organizational context. We know that developers are at least somewhat free to leave the organization, and for sure they don't spend all of their time in the sphere of the organization. This may suggest that the disciplinary power of the POS system would be considerably diluted. Additionally, the developers are presumably the experts, and the organization relies on their knowledge for future success, i.e. the organization is equally depending altering the power relationship.

## *Visibility a presupposition for POS development*

In POS development the idea of visibility was important, since rendering someone visible tends to have an effect on the behavior of individuals, i.e. it drives certain behaviors out, i.e. people self-regulate. In Foucault's words:

> ...power is exercised by virtue of things being known and people being seen in a sort of immediate, collective and anonymous gaze (Foucault 1980:154).

In the POS community everything was visible (or at least significantly more than before). Visibility seemed to impose certain behavior and expectations from managers, e.g. developers had to be more mindful about conversations as not to reveal that which was not intended to be shared outside the organization. Developers were continuously reminded as not to cross-pollinate, but in way that was suitable, i.e. with a proper etiquette. This is a manager of a team of developers:

Everything is visible. Your private conversations are not private anymore. You always have to be mindful of your etiquette. You can't chew somebody out because everyone sees that. So you need to be aware. Was I justified in calling attention to this problem because our forums are seen by every member of the community. When you're in an email situation, you sometimes will have very private discussions. This doesn't facilitate itself to that. But you get that sense because you are communicating to them and they are responding directly to you. You forget that there are all these other observers watching what's going on. I haven't seen people be reluctant to share information because they were going to lose credit for their invention or that someone else was going to be able to take that further. I think that speaks somewhat to the cooperative spirit with most HP engineers. In fact, one of the challenges that I face is reminding HP engineers that they're talking to someone who isn't an HP engineer and that they don't need to know everything about all our other products. All they need to know about is that one area. HP engineers want to share and they often view the vendor engineers as within their same engineering community. You have to remind that that's their company. This is our company. We share things with other HP engineers that we can't share with these other engineers.

[So there is a double notion of collaboration. You have to really think about what you do?]

Particularly no my team where we are working simultaneously with different vendors on similar things. We have different code names for the same project. You have to change your lexicon to map to the vendor that you're dealing with and you have remember it's OK to talk to them about this, but this other vendor, you can't talk about that same thing, even though they're working on a similar project. We have information flowing from both vendors to us, but we're not cross-pollinating from one company to another. It's coming into HP, but not out, that it's securely got to their facility and not walked out the door. That's one area that we look at: their security and how they maintain intellectual property. Vendors that we work with, work with our competitors. So, we're very interested in how they isolate, within their own facility, clients.

## *POS encompass controlled openness*

Even though POS presupposed openness, it set out to be a controlled openness and sharing. POS developers had to be reminded on what could be shared and discussed and with whom. There were instances when developers forgot, and started sharing information with third parties that were not supposed to receive this information. The role of the managers then became to step in and stop conversations, while trying not to make too much noise about it. This is a manager:

> Occasionally, we've had lapses where we've mentioned the wrong company and caught ourselves very quickly. We try not to make a big deal because that calls more attention to your mistake. We just move on quickly from it. That is a challenge, not only to have the engineers learn what they can and can't share, but also realizing what they can share with one vendor they can't necessarily share with another.

## *Balance between openness and security*

The members of the POS community had digital badges. The badges were used in order to maintain a sufficient level of security. But it was pertinent that maintaining security could not rely only on the security measures provided by the POS technology. The POS system as such also rested on trust in terms of trusting third party developers to be cautious with sensitive information regarding the collaborative projects. Moreover the POS system also presupposed self-discipline in terms of creating awareness of what was acceptable and not in terms of sharing information.

Through the interviews it became apparent that it was perceived as a tricky act of balancing the tight rope between openness and security. This is a manager's view:

> CDP is built around digital badges. So we have to get digital badges issued to our vendors. One of the challenges is identifying those and getting them added into the system so as they add somebody new

> that CDP access doesn't become the limiter to their ability to bring somebody new online. They have to have digital badges and then onsite they have to have the physical ID badges as far as security. The other area that is extremely built on trust is we know that information is securely exchanged between us and the vendor, but we don't know what happens to that data once it's on their side, on their local system. You have to trust that it's not walking out the door.

## *The perceived downside of openness*

Openness was also perceived as a threat to developers. This was particularly a hot issue when developers were laid off, as discussed earlier in chapter 4. Developers often expressed that they experienced that they were training third parties that later on received outsourcing contracts. Managers on the other hand conveyed a slightly different concern, since from their perspective HP developers and HP teams were significantly understaffed, and consequently they often articulated that HP developers should be alleviated from parts of the development that were not considered critical, but rather focus the efforts on being innovative. This is the view of a manager:

> The downside is there's a lot of fear about that. "he's going to steal my work" or "this isn't protecting HP's intellectual property, if we let a third party have access to it." There is a lot of concern about that. In fact, I'm having a meeting next week with somebody in the lab. An engineering manager and her team who were trying to understand better, what is the right way to proceed with this? They're very concerned about sharing outside.

Managers and developers both depicted that collaboration and sharing with external partners involves potential risks, since sharing typically involved exposing ideas that previously had been protected more urgently for securing Intellectual Property rights. This is another manager:

> I think that's the next frontier that we have to conquer where people are more comfortable in collaborating with third parties and they will

> see it in the spirit of who we can improve HP. One of the fears that we encounter from the engineers is two fold: one is that you that will enable other engineering groups to do the work that is now done by engineering groups within HP. [Like a fear that they're going to lose their jobs?] Yes. That's looking at it from the level of they're saying, "The work that I'm doing is going to someone else, it's still going to be an HP product, but my job may be at risk." The other is "Well, you will share information with the company and then they'll go work with one of our competitors and the information from our products will flow into our competitor's products and HP will have fewer sales, then I'll loose my job that way." Those fears exist within our community in terms of collaborating, and it's one of the areas that we're struggling, because we have to overcome that to allow people to feel better about sharing and collaborating with others.

## *POS and corporate memory*

POS developers typically expressed that they liked the idea of being part of the corporate memory, i.e. their work being centrally stored, however, they often returned to expressing concerns regarding complete openness, and in my interpretation 'losing complete control' over their own work and their personal integrity. This concern is further expanded by a developer in the Lab:

> And one of the things that is very important about this whole notion of centralized back up is to look at you do need the trust, you do need the security, you do need the access control. I'd love to be able to have it so that my work is in the corporate memory, but that doesn't mean that anybody else should necessarily; that anybody else should be able to modify it, or that just anybody should be able to see it. [So you would like to determine who could actually see, or you as an individual should in control of these things?] Yeah, I would think so. For some things you have to, because when you start getting into things like personnel documents, you have the notion that just anybody could get in and see performance evaluations... [Yeah, that's true.] It's a little bit frightening. And the fact that anybody could get in and see email, obviously the company has an

> interest in being able to if it needs to, but that doesn't mean that it should be possible to casually look at things.

But it also became apparent that for some meetings, e.g. technical meetings which are typically held for purposes of solving problems, developers were reluctant to allow permanently recordings. A manager wanted to videotape these meetings in order to capture the flow of ideas. But developers would not allow it.

## *Openness and hiring practices*

I often discussed with developers and managers issues related to openness and visibility in the development process. Managers typically perceived openness as something that would potentially enhance and refine both the structure and the quality in terms of the development process. Developers often agreed, and in particular they felt that documentation concerning the projects was expected to improve, which often was considered as good engineering practice anyway.

But, openness also seemed to make developers more concerned about how and when to express ideas, especially ideas pertaining to the early idea phase of a project. They also were concerned about how to articulate themselves 'properly', i.e. correct grammar and spelling and so forth.

Managers on the other hand often discussed that typically they were looking for people who would feel confident in expressing themselves in an open and more visible setting, such as POS. Some managers expressed that hiring practices ought to reflect this view. According to a developer that belonged to the Bluestone group this was already reflected in their official hiring policy. A Bluestone developer and project leader:

> No, because everything has to be very visible in collaborative development. Everything is recorded, everything that you post is recorded and read, and stuff like that. Now I think this room, and I think physically the roof of this building in general has a higher tolerance for that than other places I've seen and other places I've

worked. People here are generally more willing to expose their ideas, accept open criticism, understand that almost every development is iterative and you're going to make mistakes the first couple of times around, and that's why you do it. You do it to constantly improve. And so yes, you can always look back to the previous project and point at things you've done wrong. It doesn't mean you're a bad person, it doesn't mean you're a bad developer. It's just the stage of knowledge that everybody was at collectively. And you move on. So hopefully we can get better acceptance of that here, and maybe even in the technical development population at large. [But what is the explanation do you think for this behavior here? For this culture here? What is it about this culture that is so allowing, for making people being... It is something from the Blue Stone time, or what?] I believe it is. Is it something I can put my finger on? That's difficult. Part of it, frankly, is hiring practice. I think we look for a certain kind of person.

[What would that be? Outspoken people?] Yes. The kind of person who would succeed at collaborative development, working very well with others and in both give and take. Not just a leadership position, but also in the position where he may ask a stupid question. [So you mean, do you actually have the policy before?] The official hiring policy at Blue Stone in development was, considering you remember, Al's five things, it was to hire people who were smart, who; What were the five things? He's going to kill me. [I'm not going to hold you accountable.] People who get things done. That one was added at the end. But there were three more. Respect for individuals. And the important point was that there are some very smart people here, and smart people sometimes have a tendency to look down on people who are not as smart as them. And so the point was that that's not tolerated here. Everybody contributes in their own way, and, well, that's it. So respect for individuals, smart; I think one of them was a thirst for knowledge. So that was, maybe that one there might be the most key of all, is that everybody who works here always wants to know more. They want to read, they want to look something up, they want to figure out how something works. They're not just so focused on the exact task that they have to do. And that might even be, that thirst for knowledge might be the whole curiosity aspect, and the well

> what is this new technology? How can I use it to help me? And that might be the key, the most important thing about people here.

## Traditional vs. POS management and organizing

The traditional conception of organization encompasses the idea of a well-defined unit with clear boundaries. Open Source development by contrast cannot be depicted as organizations albeit it is a very organized way of conducting software development.

However, POS development and the organization around this particular form of software development emanated from an organization, HP. It was an extended net of different activities involving actors from HP and from a selection of trusted partners, which crossed each others respective trajectories striving for innovativeness and product development at a faster pace. The collaboration was made possible by a system for information and knowledge sharing. The collaboration between different units (external or internal) was perceived as necessary, not only as a result of a deliberate organizational plan, but because the products (e.g. printers) were the result of a large set of connected efforts and actions. In fact, it was the separate efforts and actions that were connected, rather than the actors themselves.

As I have already started to pinpoint, organizing and managing work according to the principles of POS distinguished itself on several of the defined features and dimensions offered by e.g. Weber (1978). In brief, Weber explored rational-legal relationships, with the distributions of resources based on status positions (1947; 1978). In such a system individuals relate to each other through the roles they are ascribed. Examples of these relations are found in corporate structures organized as hierarchies and where authority tends to be delegated from a central position to those in subordinated positions. And even though bureaucratic relations based on a rationalized division of labor and authority enforced through general principles and rules, may be (at least theoretically) anticipated in organizations with flat structures, there still must be a

considerable division of power and labor assigned to positions rather than individuals.

And as we begin to unravel some characteristics of traditional management and organizing and contrast it with POS we can initially discern that the boundaries of the POS enterprise were neither clear nor definite. Albeit it did signify a social unit, its members were constantly changing and it was not closed for outsiders. In addition, its identity were shared by internal as well as external developers, and corollary there was not a very strong differentiation between the personnel and resources that belonged to the HP organization, in fact it diminished over time and got more difficult to discern as the projects proceeded.

The POS way of organizing presupposed a central coordination function provided by the system itself. But, it is not obvious that the system provided a locus of final authority and power to enforce binding decisions on the community of developers.

Whereas the traditional view on organization suggested that leaders were able to control the concerted efforts turning it in to a unitary, hierarchical actor, POS organization relied to a large extent on a self-regulatory discipline enabled by the visibility offered by the system. When everything became visible action could potentially be monitored by every one else, including management.

In **Table 7** I have attempted to characterize traditional management and organizing, using metaphors that pertain some meaning to each of the dimensions that are analyzed. I have attempted the same approach for POS management and organizing, thus enabling some overall comparisons. Similarly I use metaphors that convey meaning for POS management and organizing.

***Table 7:*** *Traditional Management and Organizing compared with POS Management and Organizing*

| **Traditional management and organizing** | **POS management and organizing** |
|---|---|
| Conflicting interests internal actors, often | Blend of interests founded on equality, |

| | |
|---|---|
| adversarial external relationships, Metaphor: Silo mentality | established interdependent relationships, collaboration and co-learning. Metaphor: The Big Family |
| Restricted access to knowledge/information even internally. Metaphor: Garage mentality, The independent inventor | Progressively more open access to knowledge/information, i.e. layered information. Roles and rules as management principles. Metaphor: Soccer-team metaphor |
| To-down imperative control Metaphor: Cathedral (cf. Raymond 1999) | Creating learning communities: Metaphor: Bazaar (cf. Raymond 1999) |
| Fixed structures and procedures, Metaphor: Machine (cf. Morgan 1996) | Flexible structures, emergently (progressively) more flexible structures, Metaphor: Brain (cf. Morgan 1996) |
| Detailed and task-related assignments, managers: *manage, select, train, and monitor* performance. Behavior and action based on command and discipline Metaphor: bureaucratic organizing (cf. Morgan 1996) | Self-managing/regulating groups, creativity and innovativeness, empowered work, drafting (biking metaphor), electronic surveillance. Managers *energize communities of interests.* Behavior and action based on common understanding. Metaphor: Panopticon (cf. Foucault 1977) |

## *Concepts and ideas from the world of Open Source influenced POS management*

It has been argued that technology sometimes can replace formal organizational rules and structures, when coordinating and governing complex activity systems (Lanzara & Morner 2005). When studying coordination in large-scale Open Source software projects they conclude that artifacts and tools become critical elements in the process of creating meaning and understanding among globally dispersed teams addressing collective tasks.

Lanzara & Morner (2005:68) examine how artifacts are inscribed with technical, organizational as well as institutional knowledge and they argue that components of organized human agency and knowledge are inscribed into and delegated to technology. The locus of their research interest is Open Source software projects.

Open Source projects are similar to POS projects albeit not sharing entirely identical conditions. Open Source projects depend on voluntary participation of members, whereas the POS projects relied on traditional organizational structures and resources.

The POS projects were initiated by an organization, Hewlett-Packard. The deliberate strategy of the organization encompassed the active pursuit of more openness and sharing inside the organization itself in order to spur innovation and research & development, sometimes including external organizations and individuals. The POS projects were in this respect certainly open for a larger group of developers, i.e. those that were accepted by the system, but the projects were still managed and kept in control by management. And this, I will argue, is also the case to some extent in pure Open Source projects.

Successful Open Source projects are often ruled by an 'informed elite' sometimes labeled as benevolent dictators getting their mandate on their professional merits (cf. meritocracy). The Open Source projects are free to view and to utilize, but the project development (where the project is going) is determined by those in charge of the project.

### *Social rules governing behavior in Open Source*

Raymond (2001:73) exemplifies some of the social rules governing behavior in Open Source development:

- There are strong social pressures against forking[62] projects. It rarely happens unless it is absolutely necessary and it always involves public self-justification and the projects have to be renamed

[62] Fork: The most important characteristic of a fork is that it spawns competing projects that cannot later exchange code, splitting the potential developer community. The open source licenses do nothing to restrain forking, however in practice forking almost never happens according to Raymond. In fact, ( and in contradiction to the anyone-can-hack anything consensus theory) the open-source culture has an elaborate but largely unadmitted set of ownership customs. These customs regulate who can modify software, the circumstances under which it can be modified, and (especially) who has the right to redistribute modified versions back to the community.

- It is not accepted by the community to distribute changes to a project without the cooperation and acceptance of the founders of the projects, except in special cases like e.g. trivial porting fixes
- It is not accepted to remove a persons name from a project history, credits, or maintainers list without the person's explicit consent

The successful Open Source projects, (e.g. Linux and Apache) are characterized by having a large group of well-informed users who have an interest in sharing the improvements of the system with each other. The projects are open in the sense that all the participants can monitor the progress of the projects, but how the project progress is to a large extent decided by the initiators (moderators) of the project.

### *Electronic artifacts and their roles for managing and coordinating activities*

Lanzara and Morner (2005) contend that electronic artifacts play an important role supporting the design process of Open Source development. I agree, at least to an extent. Electronic artifacts are very efficient for dealing with e.g. e-mail lists and communication between participants in a community (developers and users). People inscribe knowledge as well as agency in the artifacts. The artifacts become possessors of dynamic means of expression for human activity and agency, thereby replacing or substituting for people in many activities and descriptions in complex networks of human and non-human actors (Akrich & Latour 1992).

When trying to understand Open Source and POS as phenomena it becomes relevant as a concept to introduce the idea of inscriptions. According to Morner and Lanzara (2005) technology may be viewed as inscriptions in terms of being:

- A tight net of multi-various software objects
- An electronic medium for programming and communication.

Morner and Lanzara suggest that we may view Open Source as a complex system, a large-scale interactive system with mechanisms for variation, selection and stability in order to develop. Large Open Source projects that reach a critical mass, becomes less the result of deliberate management, than an evolutionary result of a complex interrelationship between processes and activities. Variation is combined with a selective retention for the purpose of development.

How can Open Source handle so much variation and still maintain its ability to develop? The answer is simple, because it relies on standards, rules, patterns of behavior, structures and meanings.

In the case of POS we have to consider the rules, (sometimes not even formulated in writing) that were strictly governing where the project was going and by whom.

But, it is fair to view the source code as a playground for variation and openness towards implementing new knowledge within the realm of the POS organization? The incoming variation in the POS system was not (at least theoretically) as high as in Open Source system. However, incoming variation also depended on the attractiveness of project.

It certainly adds to the analysis if we insert technology in the analysis and choose to view the POS as an activity system consisting of human as well as non-human actors. Artifacts have a stabilizing effect on the processes and relationships in the network. It is not possible to ignore their importance. Artifacts help regulate the system and they focus the interest of the developers, help to manage and coordinate action and finally convey the communication that is so critical for POS development.

We know by now that Open Source software projects sometimes attract thousands of skilled programmers collectively using and producing code via the Internet. However, the vast majority of projects never attract enough interest to gain any momentum. Open Source development seems to be a decentralized,

interactive and fairly unmanaged process, but that is mainly on the surface. The successful projects are often governed by a meritocracy. The benevolent dictatorship ruling the world of Open Source is in fact often headed by an undemocratic and authoritarian leader who exercises power for the benefit of the community, rather then for his or her own self-interest. This is true, at least in theory. The classical benevolent dictator supposedly often focuses on matters of public importance.

To summarize, the Open Source world of development is ruled by the enlightened elite of developers – a meritocracy. The leaders often claim to act as benevolent dictators, and they are very often the founders of the projects. The founders, sometimes together with early adopters, set the ethos of the projects. Consensus is the chosen decision-making tool – even though that in practice most of the time may be consensus among the founders and not the entire community. Theoretically the right to fork a project is upheld – even though that in practice rarely happens, especially strong social pressure in the form of norms and rules work against forking.

## *Managing the POS system; technology, rules and roles*

In POS development technology take control over important aspects of the process of coordinating work, e.g. Open Source software a long with some proprietary was utilized within the POS framework. But, moreover to apply technology in order to manage and control the development process, certain rules of conduct were elaborated in conjunction with designing organizational roles tightly knitted to the projects.

The roles were designed as to regulate who could do what and how and worked in conjunction with the technology provided by the POS system. This is to some extent also true in Open Source development. Activities are coordinated (mainly bug-reporting) efficiently by the tools and while major changes to the system tends to be decided by the key actors, i.e. the initiators of the project.

The management of the POS system was depending on technology for coordinating the important inflow of information to the system (discussions, bug-reporting etc), however decisions on design and major implementations of changes were always decided upon and managed by individual developers , or more precise the project lead.

The large benefit of any open system is to attract as many developers as possible to scrutinize and utilize the software contributing to a more efficient development process.

Lanzara and Morner (2005) contend that coordination and management of activities in the Open Source are upheld not exclusively but to a large extent by technology. However, the conditions and presuppositions of POS development deviated slightly, since it was an organizational approach, not depending on voluntary resources. Issues such as dead-lines and costs became critical.

Hewlett-Packard attempted to assimilate the advantages of a more open approach, i.e. bring in as many eyeballs as possible, while still considering the overall costs and maintaining the appropriate relationship with the customer. Delivering according to the demands of the customer, i.e. the customer relationship became as important as balancing the internal as well as the external resources available. The formal organization of a company in terms of management and control remained organizational in that sense, even though it tended to become more interactive as proposed by Lanzara and Morner (2005).

While Lanzara and Morner (ibid.) deliberately chooses to look at technology when trying to grasp how new software is developed in Open Source, I suggest we have to combine organizational issues and technology in order to comprehend POS.

POS development was considerably more open, and had also a more interactive approach than what HP had used in the past, only with very clear limitations on how resources were to be allocated.

### *Inscribing roles*

The POS projects relied heavily on technology for coordinating the development process. However, alongside with adopting Open Source technology and methodology a system of rules and roles supplemented the organization and management of the projects. The rules and roles helped to determine who could do what and how, and they co-existed alongside with the technology supplementing the POS technology. A manager on inscribing roles:

> We're going to control the project definition, the way the project's, we're going to control who has what privileges to the environment, meaning that we're going to have pre-defined roles, and those will be the only roles that are the users are allowed to be assigned to.

The process of inscribing roles for purposes of managing POS projects is metaphorically speaking similar to the process of collation a terminology used in theatres for activities related to rehearsing. However, POS processes were characterized by being considerately more emergent, rather than static as compared to rehearsing a play. Research by El Sawy and Marchrzak (2004) highlight that in most real time emergent processes user roles and work contexts are unpredictable.

The technique of inscribing roles and assigning them to developers was a way of enabling independent work within predetermined boundaries. Still it was recognized that such roles did not in themselves guarantee the quality of the code developed. This is the experience of a manager:

> We tried to set up the roles so that people could work independently, but when we integrate these tools, this system together, there are times when you really need to look at somebody's code because goodness, the API isn't working, and maybe they just didn't implement it.

Roles also helped to layer security, e.g. 'light-users' could see but not make changes. The managers particularly supported the POS solutions since it enabled control through establishing roles for developers to adhere to. And it

was envisioned as a possible path to enroll customers in the development process. This is a manager:

> One of the big things we liked in the CollabNet solution was that it had a lot of layers of security. We can really have a lot of different roles of who can change what. Who can see what. Even with a different pricing structure, which was good. We don't pay very much for the light users and we hope to have some day hundreds of thousands of light users. Basically every customer should be a light user. They have a say in how the products change with the time.

### *Inscribing rules*

It was perceived as necessary to inscribe rules on the out set of a project, in order to implement a structured and standardized approach that would make it possible for other teams to re-use ideas as well as code. This is a manager:

> And what we wanted to do, we wanted to start those rules right at the definition of a project, because just by defining the structure of the project, any defiance of that aligns with all this supporting infrastructure, then first off you're going to be able to leverage what other teams have done. You'll be able to drive new requirements back into these standards domains to make them grow and be better. They're going to be able to do something once and everyone gets to benefit from it, as opposed to if each team is independent, they're all going to have to fix the same mistake over and over again.

## POS a template of control

Managing and organizing activities the POS way encompassed activities aspiring to shape, guide and affect the conduct of the community of developers. Consequently it may be viewed as a regulatory system, or a process, with methods that inscribe knowledge. Methods of inscription are clearly identified by POS, e.g. rules, roles, language use and tools seemed to be shaping the knowledge construction of the community. Knowledge construction then became integral to the operation of power. As denoted by Foucault:

> The exercise of power perpetually creates knowledge and, conversely, knowledge constantly induces effects of power... (Foucault 1980:52).

Through the process of adopting POS, individuals as well as teams were forced to adopt a standardized approach towards software development. Standardization in this respect may be viewed as a tool subjugating the community of developers and corollary it also becomes the template of control.

Is it fair to look at POS as a project for the transformation of individuals? Or as Foucault (1980:44) said: serving as "an auxiliary to the penal system".[63]

Foucault pointed out that it is far more efficient and profitable in terms of the economy of power to place people under surveillance rather than to subject them to penalty. Furthermore he suggested that power is exercised within the social body, rather than from above it. It is a form of power that reaches:

> ...in to the very grain of individuals, touches their bodies and inserts itself into their actions and attitudes, their discourses, learning processes and everyday life (Foucault 1980:39).

Foucault did not perceive of power as being in the hands of one person exercising it alone, but he stressed that it may be viewed as a machine in which everyone is caught, those exercising power being caught as well as those subjugated. It is machinery that no one owns.

Openness of the POS work-processes and the contributions of individuals denote the problem of the visibility of bodies, individuals and things, under a system of centralized observation. Foucault, in his Discipline and Punish, elaborates Bentham's Panopticon, and observes that surveillance which both divides place while keeping the actions of the individual open, is a contraption for economically efficient control.

[63] Prison Talk, Interviewer: J.-J. Brouchier

Is it a valid claim to argue that POS had solved the problems of discipline posed by the large community of developers in the hands of a very few controlling the POS? The system clearly immersed developers in a field of visibility and yes it to a certain extent seemed to influence the opinions, observations and discourses of the community. And clearly, it seemed as it as Bentham and Foucault suggests restrain them from *harmful* acts. But then, what happens with creativity?

In the light of Bentham and Foucault, POS then perhaps is a great innovation for the easy and even necessary exercise of organizational power in physically distributed collaborative settings. Do we have to worry about who gets empowered by knowledge offered by the system and is it really significant to even worry about how the construction of social knowledge relates to the production and exercise of power?

The POS model of sharing knowledge promised a mobilization of collective potential to contribute more effectively to the overall result of the organization. The idea was to create mutual learning opportunities. In some aspects, this was still a primitive accumulation of explicit modes of knowledge, perhaps even at times reduced to mere information sharing. However, POS collaboration enabled new relationships to be formed more easily allowing a structure of outsiders being invited inside the very heart of research and development. Developers, external as well as internal, potentially became both subjects and objects of the research efforts of the community.

The POS project did not in itself eliminate the division of labor that traditionally had governed the different teams involved in collaborative software development at HP. The Silo- mentality[64] problem was still salient. It is perhaps suffice to recognize that many of the developers and managers that I

[64] Silo-mentality refers to instances where internal competition, lack of synergy, shortshighted solutions and poor communication prevails over collaboration and sharing between teams and divisions within an organization. It also reflects isolation, division, duplication of efforts, inefficiencies. Silo-mentality as laid out by those interviewed in this study encompass failure to coordinate activities, ie. constituting an internal functional barrier.

interviewed were convinced that the production, dissemination and implementation of knowledge supported and provided by POS significantly had increased efficiency, e.g. new products were produced at a faster pace, in parallel, and people were introduced quicker to the respective projects.

## *The legitimacy of contributions*

From what can be derived from this study, the technology utilized by POS, e.g. versioning control tools such as CVS, permit monitoring of who is doing what with the source code. And even though managers were able to monitor the contributions, it was still not possible to effectively estimate the legitimacy of the contributions.

Managers often brought up instances where they were concerned about the whether the contributions were really meeting the necessary requirements. And they had valid reasons to be concerned since even minor errors could cause major and costly defects in the products. And while POS were perceived as a presupposition for distributed collaboration as envisioned by high ranking officers in the organization, the centrality of information, and its visibility and openness frequently raised concerns in terms of issues related to validity and reliability of the information provided by POS. In interviews developers and managers used the terminology 'garbage in – garbage out'.

Managers and developers contended that an efficient development presupposes a highly structured way of coding information, encompassing not only the code itself but also structure in terms of designing and driving the development forward. The developers suggestively had to adhere to the same set of rules. Only in the beginning of the implementation of POS this was perceived as a problem. Issues related to legitimacy of contributions were perhaps considered the biggest concern of both managers and developers. This is a project leader and a manager:

> But it's just the free-form text area that maybe needs more structure and categorize whether this is an investigation or a product. So it

would take number 1, have structure put on it, number 2, that structure would have to be mandated that people will use this, and really, number 3, you'd have to police the projects to make sure that people are filling in the fields that are there, and that they're keeping them up to date. A project could go from investigation to the next step in the development process, and somebody never went back to switch that bullet. So if I'm working for a product that's shipping, their project is still listed as under investigation. And somebody needs to maintain that data. Garbage in, garbage out, I mean that's been around a long time, and that really applies here. If people aren't using the tools and adhering to the structure, there is no way to affect and automate the searching and categorization capabilities. You know the term garbage in garbage out. That's the only disadvantage that I'm finding. Just between the three different people we have working on it with three different projects, every individual will approach a project in a different way. So if you don't create that appropriate structure or definition around how you want to use this tool, you can all of a sudden get slight deviations in how people are using it and how information gets entered into it. Now all those differences actually start causing more work. If I'm working on one project, I do it this way. If I'm working on that one, I do it another way. Oh and on this project they put this data over here. Now it's becoming inefficient because people have to know three different structures for the tool. There's a hierarchy of how data is stored and how it's controlled. We tried to make an effort with Karl as the starting point of defining a consistent look of that project home page, make sure they all look the same, so there's consistency in where you go and how you store things. Then there's operational procedures of how you use it. Make sure that there's consistency there in how you're using it. And of course, there's added things like security of suppliers. We had to make sure that the system we put in for security was acceptable by all three of the suppliers we work with. Make sure it was consistent. If one didn't like it, we didn't do something independent for them. We learned what they wanted and then we folded it back to the other two. So, we want one system of controlling information, not three separate ones. That's probably the disadvantage of the tool. It's like the Web. Everybody can put

> information out there. It's tough finding the right information. At HP, as a manager, sometimes it takes me 30 minutes to find the one little thing I need. It's not so obvious where information is a lot of times. I think that's the one risk of something like this. If you don't have the information structure underneath it, the supporting processes of how you handle the information or secure it, then you can get massive deviations of how it's being utilized. That probably doesn't matter from team to team, but within a team I'd say is the problem. If that's the case, it becomes useless because nobody really knows where to go to get the information. It becomes more work to get the information, than to walk over to the isle way and ask somebody who knows.

It was also perceived as crucial to be able to trust the contributions of all the collaborating actors, whether they were internal or external. The tools, e.g. CVS, presumably would help managers as well as the developers to derive changes made to the code. However, CVS did not prevent mistakes to find its way in to the code and subsequently in to products.

Developers and managers frequently used terminology related to perceiving software as something that continuously is refined. This was at the same time perceived as a real problem when software was to be utilized in hardware products, such as e.g. ASIC's.

ASIC's are extremely costly to develop and to produce, which reportedly made it crucially important to catch the 'bugs' before the production process begun, or it could result in huge economic losses.

When the development was relying on different collaborating teams, it was perceived as important that the contributions were scrutinized. Developers as well as managers expressed great concern regarding problems pertaining the quality and reliability of information. Information quality and reliability were relying on structures and rules imposed by the system and by managers trying to construct a 'cookbook' that developers ought to adhere to.

The act of bringing new actors into the process conveys instances where developers and managers became increasingly concerned about issues related to whether code was legit or not. This is a developer:

> [I hear a lot about collaboration and your supposed to work with your team and your supposed to collaborate with third parties and bring in new people on your team and people come up from the outside is there anything that you feel that is it all good or what is good or perhaps not so good are you cautious about any of these things?]
>
> Yeah, I think in the course of this conversation I think I have mentioned a couple of maybe not in terms of... but for example this new person that was on that just became a member of this CDP community yesterday committed a file and I got notified about it last night. Now I'm really concerned that this file is legit. CDP and CVS just like our previous revision control systems do not substitute for common sense and when and what kind of changes need to be made. So unlike firmware that where firmware gets to be changed and oh yes I made a mistake I can change it back if an error isn't caught in an ASIC, it could result into a half million to a million dollars worth of error with us if the masks of all of the fabrication processes. So you think there's a bug we make the masks it fabricated to find it's not good all of that has to happen again that's about a half a million dollars right there. So I am concerned that we don't have... there still needs to be rules place on access and CDP seems to make access almost too easy. So I guess that's a good, that actually ought to be some control on access and there is to some limited degree because the project owner sets people up as developers or observers and so on and so forth. But there also needs to be other controls like after such and such date we can no longer accept changes except maybe certain ones and we don't want to just go and turn off everybody to being an observer then what needs to be more fine grained to regulations as to who does what. Or there needs to be some sort of a system where somebody reviews changes before their actually committed without there's concerns that CVS CDP makes changes almost too easy.

When considering POS management and organizing, it is necessary to take into consideration how information and communication technology is utilized within organizations and also about the implications, e.g. in terms of empowering the developers as well as creating necessary presuppositions for innovativeness. Is POS promoting logics of change implementing the necessary conditions for flux and transformation – or is it just another 'psychic prison' (cf. Morgan 1996).

Zuboff argued already in the mid 90's that while information becomes omnipresent in organization, yet it has not been accompanied: "by a new social contract derived from a new moral vision" (1996:13). The missing link in the information economy was a new approach towards the process of organizing work. Zuboff contended that information must become available and open to all members of the organization leaving them as much freedom and: "the authority to express and act on what they can know" (1996:16).

The new social contract should clarify the role of the employee including their competences and skills, and lastly also specify exactly what they can do within the realm of the organization. Ostensibly, this contract leaves management in control and to a large degree unaccountable to employees.

## The role of management in POS

It was perceived as necessary to introduce new roles for management in POS. Managing distributed teams demanded a new attitude towards leadership and control. In this interview a high ranking officer describes how managers within HP typically used to manage by having very tight and detailed control over the projects. But he also depicts that POS requires managers to set clear goals and then trusting people to deliver. The manager:

> No we do have some problems in that realm. I would say a lot of it is managerial attitude I think is our biggest obstacle right now. Many of our managers have been used to an environment where they control aspect of development. We have had a number of development tools that they can literally come in every day and see what their team has

> done. It can be a bit challenging for a manager who is used to looking over their team. Having daily status meetings and updates and now to realize that their team is on the other side of the planet and they can't see them any more. They are very reliant on contributions of people who are not under their direct command and control. I think we have some cultural problems still to work through here.
>
> [What kind of leaders are needed. Are we talking about a new kind of leadership.]
>
> I think we are. A lot of managers at HP have managed like the engineer. In other words they came up through the engineering ranks. They basically want to maintain a tight control on everything. I think it is more trust and an empowerment model that we are moving toward. It is really not just trust and empowerment. It is spending a little bit more time getting the goals and objectives well outlined. Then trusting the people to deliver against those goals and objectives. I think it is going to take some time to help our management team understand how to get in around those goals and objective immediately. Then also to learn not to have to be as reliant upon day-to-day micro management of the way things go. I think that actually creates more creativity in the engineering teams. I used to joke with people. I've never been at a place where I felt I could trust the engineers more and where we actually trust them less. I think we've got a challenge there. We have a very capable, talented set of engineers and we don't let them use their heads. We over manage.

The high ranking officer also described how HP is still lacking a clear system for measuring the individual. He also denoted that POS presupposes managing by influence rather than controlling at the micro-level. He continues:

> [How do you manage people...]
>
> It is difficult in that there aren't clear metrics by which you measure people and there aren't clear rewards. Many of the people that are working on projects of mine or that I am interested in or that I champion are involved in other parts of the company. I can't directly

> tell them what to do. Nor can I reward them for a job well done. So a lot of it depends on relationships and it depends on being able to follow through and actually have impact from what people are doing. Being able to in fact preserve and manage the assets that cross-organizational teams create. I think is very important. The simple things that you learn in management school, in business school about managing people, much of it doesn't apply to managing a group where you really have to manage by influence rather than by control.
>
> [Do you do that through getting involved in people's projects?]
>
> You have to get involved. You have to get involved at multiple levels. You have to get involved at the researcher at the researchers level. You also have to get involved at the manager to manage as well. There is an added component of my job which has to do with working with the managers of the people who are doing research collaboratively researchers who work for me. It makes the job much more complicated than a straight management job.

## *The manager monitors and communicates*

The role of the manager was altered as POS was introduced. The role encompassed monitoring and communicating with developers that were physically distributed. Managers typically described that it was difficult to grasp what the teams were actually doing in spite of the tools that supposedly would allow them to monitor the projects. Some admitted that the tools were facilitating monitoring activities nevertheless they perceived it as necessary to actually communicate with people. This is a manager's view:

> It makes it more challenging to understand what the remote people are doing. Certainly having the tools in place gives you more reporting capabilities on seeing who is working on what. How their progress is done. I don't think anything has been done to replace actually talking to people. Weather it be here at work in Japan. You need to pick up the phone or hopefully one day look into a video camera and have a conversation to find out how that employee is doing. Do they like what they are doing? Are they being successful?

> Do they have roadblocks in their way? Are they performing well? Are you asking them to I don't think all these are great tools to augment that, but when it comes to managing a team, there is still a lot of one on one communication that has to happen.

## *Soccer Metaphor and the value of team work.*

Leading actors within the POS initiative typically perceived collaboration as something desired, putting particular emphasize on the role of leaders as enablers. The role of the leader in the POS setting then was envisioned as being able to appreciate and recognize individual differences and strengths for the benefit of the team as a whole. Typically qualities such as appreciating teamwork, being able to put the right person in the right place while at the same instigating trust was depicted as important characteristics.

This is one of the key players of CDP, and also one of the very early adopters of ideas that constituted POS:

> I know there's a better way we can do things, I've coached soccer for a number of years and I personally see the value of teamwork and the motivation and how one person who's not great at one thing can be great at something else, but then the compliment one another. Then when I coach the girls and when I get a girl who's really, really fast, but not much of skill, I'm going to work with her on that skill, but I'm not going to put her in the middle of that field because it's there that you have to be able to get through people, but I'll put her off to the side so that someone could kick the ball to her and she could get a good cross off, so I'm going to put them where they're going to be successful on the field. A girl that's really tall and just wants to jump in the mud, she's going to be the goalie. From that personal experience, we're not forcing the situation, you use the best people you have in the best situation, I think that can work here. Also for me, they're just printers and you look at people and they love what they're doing, they love their technology and I think they're really cool, I do all my Christmas cards on them, I do scrap booking so I have a digital camera, I love printing these kinds of pictures that

> really capture the memories, but I'm not personally energized by working on a printer and what I am energized by is improving the way we get our work done. And I really believe, in my experience, when you're working with people you trust and who in return trusts you and are given the gratitude to work with you, you can get so much more done. So that's where my motivation comes from. Then in terms of who motivates or mentors me, every now and then I run across a couple people who believe in this as much as I do, and that's what kind of reinvigorates me.

This manager also told me about how the 9/11- incident had completely altered her thinking and her attitude towards leadership. It had made her think more closely on collaborative endeavors and how important it was for her to make an impact in terms of introducing ideas of equality and rewarding collaborative behavior. The reward system ought in her view not only encompass top scientists inventing, but also appreciate those developers who do a great job improving the quality of already existing products, i.e. rewarding collaborative behavior. This is her story:

> [You said something about September 11th, what did it do to you?]
>
> At first it was horrifying, but about three days later I just wanted to quit HP, I was thinking that this doesn't matter, this doesn't help the world with what I'm doing. I want to be a teacher, I want to influence people's lives in a way that it'll be positive, so that's where I was and working my way back, it was like, can I improve the way we're doing things here. And part of it is that I look at engineers who are rewarded for doing the heroic thing and staying up all night to get something done, and I don't think that's a way, I guess looking at diversity, single young engineers can do that great, but folks that do have families and still want to be successful can't so I'm trying to look for ways for equality. I mean people can still be heroes, and they're going to be rewarded for it, that's never going to go away, but finding a way that people can get the most out of what they're able to do, not everybody is a superstar engineer, some people do a great job at verifying our products and writing test code that

> improves it, and those people don't get recognized, but yet it's just as valuable.

Yet another manager denoted the importance of recognizing interdependent relationships between different divisions and units within HP. He expressed an idea that in a sense saturated many of the discussions I had with key actors of POS, namely believing in the idea of the 'big family'. This is his view:

> We look to draft where it makes sense, and where there's a piece that doesn't fit our business, yes, we'll do something that's different. But then look to contribute that back, because you don't want to be different for the sake of being different, you want to be different because you have a different problem domain you're trying to solve. And then look to share your solution with anybody who may also stumble across that same problem domain. We're pretty big believers in the big family. The way we ran Blue Stone was trying to do things that are good for the family, not just good for you.
>
> [But can you see how the potential interdependencies in the future would be in the company where it could be a benefit that you're actually on the same; that you use CDP and perhaps other groups.]
>
> Sure, lots of big examples. The question is, is the company going to ensure that it actually has those interdependencies between its business. I mean there's a fierce sense of distributedness about HP. HP divisions don't like to be that tightly coupled with other HP divisions.

This manager also used the metaphor drafting which is a concept commonly utilized in biking competitions. Typically, it denotes a team biking together, helping each other, promoting the individual though facilitating the team effort. The team is more important than the individual in one sense, but it also encompasses helping the individual to reach his or her goals. The manager describes the importance of drafting like this:

> I'm actually trying to; I bike ride a lot. You ever watch bikers? They have this concept of drafting. We're trying to draft on the work other people are doing, and if we do find a reason that we have to take a

> lead, we're trying to let other people draft with us. So I'm not really looking to go left when everybody's going right. What I'm really looking to do is say what do they have in place that's consistent with what our goals are? Use it. If there's stuff that's not consistent, or, hey, stuff evolves quickly. Let's enhance it or take it to the next level, and recontribute it back. And now if they want to now take advantage of it, great. If not, or they don't need it, that's fine and okay. So for example, inside of HP there's a huge team that have a lot of affinity with Source Forge, there's a huge team that have a lot of affinity in CollabNet, there are organizations that have affinities to other foundational technologies. I actually explored them all.

## Conclusions

Individual developers that took part in POS development went through a process of becoming knowable about the possibilities and limitations of the system. The identity formation of the individual was provisional, depending on how familiar and comfortable they were with the system and especially coming to terms with the openness and visibility offered by the system. Developers and managers frequently addressed instances where disciplinary activities were operating to create more order in the development process, and corollary we may conceive that disciplinary activities have an impact on knowledge formation.

The POS was constructed to facilitate a progressively more open development process. This was perceived as a change of paradigms. In the past knowledge was less explicit residing to a larger extent concealed in individual developers. The introduction of POS conveyed exposing, i.e. making visible and more transparent the efforts of the community and the individual. Control in such a system relies on the principles of openness and visibility. The individual accomplishment and the individual capacities were up for scrutiny, not only by 'the manager' but also from peers. The individual's accomplishments were inscribed and notated.

We may assume theoretically, that the process provided by POS makes the individual as well as the community more easily calculable and manageable. The POS process facilitated distributed teams and individuals. And the system also made it possible to measure efficiency while alleviating a lot of administrative decision-making. We can perhaps also assume that while a system like POS encourages self-discipline, it may also perhaps on some instances be a little too efficient, jeopardizing creativity and explorative modes of knowledge construction.

One of the more striking features of POS relates to issues expressed by developers concerning what might be labeled anomic effects, i.e. they expressed great concerns regarding job insecurity as well as issues of marginalization and depersonalization. Some groups were actually unable to function properly, or as efficient as before, due to the introduction of POS and as a result felt alienated. Groups of developers (the San Diego Team) were left on their own figuring out how to deal with problems caused by the introduction of POS.

POS presupposed communication and development processes supported by a common language and a common toolbox. It left little room for local adaptations. Since the introduction of POS conveyed significant challenges for some teams in terms of adopting tools, it is relevant to questions how the tools were actually picked, and if the idea of having common tools (quest for the Tower of Babel) in fact preserved undesirable power relationships rather than liberating the community of developers.

In addition, the POS initiative was launched during a period when many employees were laid off, and outsourcing became even a more prevalent strategy, corollary the initiative was perceived as threatening to some groups and individuals.

The larger issues at stake when analyzing the introduction of POS in terms of management and organizing has to do with:

1. What will management be like in this 'post-hierarchical' form of organizing? We can assume that a new type of leader will emerge which is typically denoted by the examples that I have referred to in this chapter.
2. Whether we can assume that leadership will depend and rely on a meritocracy similarly to what we can see in Open Source projects?

If we highlight electronic surveillance in the form of the information panopticon, the technology becomes means as to reinforce imperative control. But if we believe that openness also encompass freedom as perceived and envisioned in the Open Source world of development we may balance the nightmarish vision with brighter prospects for the future.

My interpretation is that at least in theory it will be more likely with a larger share of trust balanced with security measurements, in companies like HP, i.e. in companies that have financial as well as intellectual resources to dominate, since being profitable also encompass the possibility to provide good working conditions and long-term relationships both with internal as well as external collaborators.

Only, many companies operate under different conditions, i.e. they have to operate in stiff competition with short-term performance goals. Perhaps the most striking difference between Open Source projects and POS projects, is that Open Source projects can afford long-term perspective, i.e. time and money constraints are non-existent.

# 8. SUMMARY AND CONCLUDING REMARKS

At this time I intend to make a few closing remarks and also recapitulate and delineate the overarching conclusions. I also provide a summary of the theoretical contributions related to this project. I also present implications and challenges for practitioners. Lastly, I will present a set of possible avenues for future research given the insights I have had resulting from studying the Progressive Open Source (POS) initiatives at Hewlett-Packard.

## A vision of openness

The Internet continues to change the way work is conducted, and it is protecting and ushering a new era of collaborative, participatory and global approach towards innovation. Openness is the hallmark of these new processes. Openness is enabled by the Internet - in fact the best example of standards is the Internet itself. And more importantly, openness has emerged as a viable strategy for organizations.

In this study I have particularly highlighted how the construction of a development project, POS, embraced openness and sharing, albeit encompassing a progressively more open approach towards bringing internal as well as external actors in to the process of innovation.

Furthermore I have touched upon some of the techniques and methods utilized by individuals and teams to infuse development practices with ideas

translated from Open Source software development. Openness as embraced by POS denotes sharing and having information in common. It also encompasses the strategy to open up the technology, through standardized tools and language use, in order to drive down operating costs for IT.

Potentially openness can cause transformations, e.g. openness will increase transparency and competition, but also for the worse, it can bring potential invasion of privacy. And as this study have depicted, openness seems to induce an increased self-awareness and self-discipline slightly altering the presuppositions for management and organizing.

But, what does openness really connote in the context and assumptions given by the digital economy? As this thesis indicates, works and processes are neither completely open nor closed – but resides somewhere on a spectrum between the two. And also, as the name Progressive Open Source indicates, accessibility to work and information seems to indicate the degree of openness. Intellectual property law provides the mean by which holders of these rights may close off information, thereby controlling access to and in extension also charging for the rights to copy, distribute and/or modify. With increasingly more openness facilitated by open standards, especially the Internet, it seems like the model relying on Intellectual Property rights and closing off information is under considerable pressure.

In this thesis the concept of openness relate to giving and providing access to information needed for making necessary progress in the development projects. Only, increased access, i.e. less secrecy, also encompasses the risk of providing adversaries with information, thereby potentially increasing their strength.

The model of openness and sharing is not new. In the practice of science in academia, openness has been suggested as necessary for the processes of trial and the elimination of error. The principle of openness comprise of freedom of access by all interested persons to the underlying data, and to the processes, as

well as the final results of research. Openness and the effort to make scientific information available rest on the assumption that society as a whole will benefit by increasing access to information. In such a 'perfect' world scenario scientists share problems and collaborate. One such promising example is open courseware, which is a phenomenon that is gaining momentum providing free and open educational resource available for anyone interested in self-learning.

In yet another dimension, openness relates to honesty, in the sense that all information ought to be out on the table. It also relates to trust, since being open puts people inevitably at risk for speaking up and having that same information coming back being used against them. Openness in communication includes sharing key information.

Whilst openness seems to be preferred for many good reasons, it usually also includes elements of fears. Typical fears that tends to have an impact on openness is fear of retribution, fear of coming across as foolish, fear of conflict and isolation, e.g. being ostracized.

In this study it was evident that sharing ideas were perceived as threatening for developers, particularly considering that the organization was going through financial difficulties resulting in massive lay-offs, coupled with substantial outsourcing efforts. These fears and similarly related may or May not be real, still it has an impact on collaboration in an organizational context.

When we bring openness into the context of software development and standards it connotes the process of ensuring that things made by different people will either work together or work in the same way. A standard is at the same time a blueprint and a set of plans that can be implemented. Building software according to a standard include complying to rules usually set up by a technical committee, and it means that no single vendor can arbitrarily change it, and that serves the goal of providing security to those who have chosen a particular standard.

Why is it important to standardize and why do HP choose to standardize their development processes? The overarching goal is to make sure that products, in particular all HP products, are compatible with each other. In short: standards make things work together. The strategic intent of HP was to guarantee that their products work together in order to build solutions that strives at solving real customer problems.

What does it mean for a standard to be open? According to Sutor (2006) openness and transparency go together, enabling the community of developers to be involved in the process. The process of standardizing must also be democratic in order to be labeled as open. The costs need to be low for the developers adopting the standard, and the licenses needs to be generous in terms of permitting. Open standards are crucial since it allows for software made by different people to work together.

What then is Open Source Software? To be explicit it is software where you can see, re-use and redistribute all or part of the source code. Perhaps the most prominent example of Open Source development is the Linux computer operating system. Unlike its proprietary counterparts, e.g. Windows and Mac OS, its source code is available to the public scrutiny and use. But, even though Linux is free to use, it rests on a model that contributions are evaluated by the leaders of the project. Openness is in this regard limited in order to ensure the quality and reliability of the system. It has been suggested that Open Source has fuelled innovation and increased innovation in the area of software development. But, either way, it surely has influenced the industry.

If we consider POS, we may think of it in terms of a hybrid, i.e. POS was a mix of Open Source ideas and methodologies whilst not quite leaving the realm of traditional proprietary software development. POS also encompassed the notion of standardizing languages and tools. As this study indicates, some languages tend to be better than others for helping developers accomplishing their respective tasks. And the quest for commonality in language use and

toolsets resulted in instances where developers found it difficult and sometimes even impossible to justify collaboration.

And while language use and toolsets were going through standardizing efforts as a result of introducing POS, some developers and teams were advocating local toolsets and language use, favoring a strategy which encompassed building bridge tools. The desired outcome of those advocating POS was to make sure that applications were created in order to be combined. In such an approach big applications can be factored into smaller parts or models, consequently they become easier to handle, and the whole can be dived and put together like a jigsaw puzzle.

With the introduction of POS, modules could be written by different developers, often residing in different parts of the world, being internal or external to the organization. The modular design was also important for re-use in different projects. POS and its precursor, the Owen project, proved this by being able to produce more products in parallel using the same code base.

POS was perceived as a new development paradigm, which aimed at supporting communities that develop code by maintaining the code and keeping it open to the trusted partners. This approach encompasses structuring development, thus making modules and libraries more available. This is similar to the complete Open Source approach, which enable many developers to contribute and ultimately also improving it, to the mutual benefit of everyone being able to take advantage of it.

## *Limits of openness*

But why is it that companies, such as HP, choose to implement POS, why not go all the way and Open Source everything? There are many reasons and I will discuss a few of relevance for this study.

Firstly, the software HP develop contains code originating from others, (often customers), and they may not be at all interested in giving their property

away. It became pertinent that even sharing information internally in the organization was not always feasible in practice, because HP was sometimes assisting customers that in turn were competing with each other.

Secondly, HP was achieving strategic advantage, allowing differentiating, and most importantly receiving revenue from the software and the products originating from the software, as is the case with e.g. printers and inkjet production.

Thirdly, there was no immediate community of developers outside the organization that were willing to contribute, leaving the company alone with spending resources on the development. This was particularly obvious with the Cooltown project, which aimed at releasing the code they developed completely open. It was also true for significant research efforts being conducted at the HP labs in Palo Alto.

Even though speed over secrecy was the guiding star of POS, it is evident that it was preferred that outsiders know as little as possible about the code, since it would help keeping competitors out of the way. The company tried to control openness by creating rules and roles to control the dissemination of sensitive information.

The implementation of POS is in itself was a form of organizing that involved many different aspects, e.g. being a device for knowledge sharing, facilitating geographically dispersed teams, comprising of efforts of standardizing language use and use of tools and methodologies, as well as management issues related to control.

In its quest for openness and innovativeness the company implemented POS, which as already pointed out may be perceived as a hybrid of ideas captured from prototypes and models from other contexts, internal as well as external. POS was in this aspect, both the result of wanting to create something new and unique while striving to protect and preserve its own uniqueness.

In the light of what has been revealed throughout this thesis, I hope that this case study can contribute to a better understanding of what it means to organize large corporate software development projects in a progressively more open context. Or maybe it is the other way around, how do ideas of openness contribute to processes and efforts such as POS?

To sum up, the combination of the Internet and digital information has provided the means for organizations to form new creative enterprises and also enabled new forms and processes for innovation. Information and communication technologies provide the foundation for new collaborative models of open innovation. And those who advocate more openness contend that openness will result in greater innovation than would otherwise have been achieved relying on a model that set out to protect Intellectual property rights. In hindsight we may derive whether this is true or not.

## Contributions for research

Translating ideas into a local context is a process developing new knowledge and in practice it is utilized and perceived as attractive by many corporations. HP was relatively early adopters of Open Source ideas and methodologies translating it in to their own particular paradigm, the POS.

Looking more closely at the process of introducing POS, we can discern certain features that showed how a process of translation and transformation may work in practice. Initially, as I have described in chapter 3, there was a phase of introducing concepts emanating from the Open Source bringing them in to the world of corporate software development – it was an instance of matching identifications and situations as described by Sevón (1996:53). The pioneer developers' modes of action can clearly be described as typical of the logic of appropriateness. The study shows that the developers acted consciously according to what expected. They were very clear-cut on what they wanted to achieve, instigating changes in the way software development was conducted

within the organization. They began to abandon the rules of the garage entering a model of open collaboration.

The study also showed that the transfer of ideas and methodologies from the Open Source context to the realm of corporate software development produces conscious as well as unconscious innovations leading to deviations from the original model (cf. a similar translation process described by Westney (1987:25). Looking at POS it becomes discernable that even though the pioneer developers driving the POS effort within HP perhaps desired to build a perfect replicate (stemming from Sirius development and Open Source methodologies) this can never happen of course. For obvious reasons, perfect information about a desired model is never available to those engaged in introducing change. And if we take a closer look at POS, it really consisted of at least four different and distinct initiatives, sharing many similarities, only having different rationale behind changing existing patterns in terms of openness and sharing.

In large-scale software development projects involving large corporations, such as HP, both informants and information-seekers see only (at least initially) their respective parts of the organization and relate firstly to their own subset of problems. Is this only an example of Silo-mentality, or is it perhaps a case of misinterpreting the overarching goals set up by management? In my view, it can probably be both.

Traditional organizations often consist of distinct business units and they still have to prove their existence by showing positive results. To give up local efficiency and productivity in order to enable the collaborative effort of the whole organization is a difficult equation leaving many issues still unresolved. Through interviews with developers and managers it became evident that some local business units were reluctant to join the collaborative effort, because they simply could not make a business case out of joining. It was argued that the sacrifices were too great to give up their local methodologies and tools and start utilizing the universal standardized methodologies and toolset offered by the larger initiative.

To assert this is quite straight-forward and uncomplicated. However, by identifying the particular set of factors leading to these departures, it also becomes possible to grasp some interesting features of how culture and organizational patterns interact leading to new forms of hybrid organizations. Corollary, when studying translations it becomes necessary to take in to consideration the restrictions for change in the local context.

In this study some of the more important restrictions stem out of the fact that communities in cyberspace, such as Open Source communities rely on voluntary resources whereas the corporation have to face restrictions in terms of costs related to delivering products on time, often stipulated in contracts and responsibilities towards share-holders, employees, customers and third party collaborators.

This study sets out to contributing to research and practice by denoting how complicated the process of translation is, as it starts its trajectory from a global idea about Open Source to a local application such as POS. Although a model of translation is a simple model theoretically it turns out much more complicated in practice.

Moreover, in this study I show that in terms of transferring ideas it is not irrelevant to consider who the motivating actors are, i.e. those who serve as igniting and driving forces when an organizational model is deliberately transferred out of the institutional environment in which it was originally developed.

Is it possible to always anticipate the outcome of a translation? As it turns out, most likely we can make predictions, but just as often, it is impossible to foresee the full consequence of a translation. And this is due to the fact that such processes are open-ended. The spreading models are continuously shaped and reshaped and it becomes impossible to tell when and if it comes to its conclusion. When looking at social translations, i.e. when social rules are

translated in to rules of action, it becomes impossible to determine when the goals are obtained, since it never really does.

In this thesis, I have looked at a particular organization, Hewlett-Packard, and four distinct efforts, all labeled under the same umbrella, the POS. I have tried to highlight what models were depicted for the purpose of imitation, and particularly what specific features of the model that were chosen and finally selected by the organization.

I have also sought to identify how the ideas were translated and why there were deviations from the original ideas. I have concluded that albeit there were particular reasons behind modifying the original ideas, the local interpretations somehow reflect the original ideas only locally adapted to the restrictions given by the context of corporate software development.

The idea of openness and the effort to standardize development had an impact both on the social environment internally, i.e. the conditions for conducting development were altered for the developers concerned and it also influenced external third party developers and their respective approaches towards contributing to the process.

## Relevance for practitioners

In this section I will summarize a set of contributions that I perceive as having implications for practitioners. It consists of an array of challenges and critical incidents that were uncovered during the study. This thesis sets out to contribute both to theory and practice. Corollary, I want to share the insights and experiences that emanated out of the discussions I had with the people actually working towards implementing a new model for supporting research and development.

At the time when HP introduced POS it was a decentralized company and it had an IT infrastructures built around supporting various businesses, i.e. IT

resources were de-centralized. The advanced technology was developed in different labs. It was acknowledged by management that knowledge sharing was not as efficient as it potentially could be.

HP has for good reasons been identified as a company with a very strong engineering culture, often resulting in innovative work. Only there was common awareness and understanding concerning the goal to ensure that innovative ideas must be more effectively brought into new products. This was highlighted through the baseball analogy a manager at the HP labs used. He stressed that when labs were pitching software there ought to be a business unit there to catch it and take it to market!

Managers were exploring different paths as how to build collaborative relationships between the labs and the business units, trying to commercialize the ideas, embed them in products and in the end make a profit. POS was envisioned as a way to utilize contributions more effectively alleviating resources and taking advantage of the existing synergies in development work.

The rationale behind implementing a common infrastructure for knowledge sharing was to make sure that the good ideas, and innovations were captured by the large array of business units in order to leverage these innovations commercially.

### *Common infrastructure encompass increased risk and vulnerability*

Introducing large collaborative projects, such as POS, which lean on a standardized and unified approach i.e. common infrastructure and common tools increases the risks for a company if and when the system for various reasons fail (i.e. the system goes down).

Down-time relates both to the reliability of the system, and also to scheduled maintenance of the system.

The introduction of POS was a major re-engineering effort, and it also encompassed investing in a more robust and reliable application delivery

infrastructure. The POS incurred significant growing pains when deploying the POS technology. In addition, HP also decided to buy the IT solution from an outside vendor, and they in turn were having reliability issues initially. The introduction of a common infrastructure conveyed greater dependence on one system, which in turn potentially increased the risks.

If e.g. a system is reliable 95% of the time, it means that during 5% of the time development slows down significantly. Even if a system has to be brought down, e.g. during maintenance, it has an impact on some teams depending on where they are located geographically. Since the goal of POS was to achieve 24/7 development it was perceived as difficult to schedule down-time in order to cause as little damage as possible. The vulnerability of the system increased as more developers adopted the common infrastructure. The perceived solution was to build in redundancy into the system. This encompassed having back-up servers, i.e. if one server were to fail, development could roll over to a back-up server.

Given the insights of this study we can perhaps suggest that:

1. There are risks involved in scaling up a common system, both in terms of reliability related to the infrastructure itself, and also in terms of the number of developers depending on the same system. The architecture of a common system has to support the number of developers in order to prevent scalability issues, with wide margin.
2. Strategies need to be in place to deal with expected as well as unexpected down-time.

### *Security issues related to POS*

POS relied on an extranet solution, i.e. it was a private network that utilized the same protocols as the Internet. It used network connectivity and largely also the public telecommunication system, albeit with tight security measures in place. In order to effectively share information with third parties a solution was in place which among other things also involved drilling holes through the firewall

of the collaborating partners. Security measures were extremely rigor when setting up POS. It turned out to be so rigorous that in fact it was difficult to actually bring some partners in. And some external partners were actually forced to adopt similar and/or equivalent technical solutions in order to match HP's POS solution.

### *Issues related to culture*

At the time when POS was introduced the company was going through a merger with Compaq. This in itself made developers and managers expect huge management and cultural challenges, i.e. clashes as a result of bringing a new organization and new teams together. HP perceived itself as being a company valuing technology contributions, whereas Compaq were (by HP) considered having more focus on the commodity product.

HP in itself has a history of being a distributed organization in the sense that the respective business units were very independent. Especially the HP Labs had a history of people being enabled to do things their own way, meaning the HP way (cf. Chapter 4).

Independent work and independent engineers also meant not trusting a centralized system. It was perceived as a result of going through constant changes, i.e. it is better to do things your own way than relying on centralized back-up systems. The researchers at the HP Labs that I interviewed were often very individualistic, and the projects were almost entirely single person projects. With projects evolving from individuals it tends to have an effect on how the projects are set up in terms of documentation, and the actual development would most likely be different if the projects were developed by teams. The reward system for researcher in the HP Labs was at this time focused on producing patents. They were rewarded for enhancing the innovations rather than rewarded for making ideas accessible to others. Basically the researchers were mandated to create but not to preserve.

The HP way was very deeply ingrained, in the sense that it was very often used to denote how things were supposed to be approached within the organization. Speaking to developers outside the organization or in the periphery it was suggested that the expression HP way was used to resist change.

In the timeframe of November 2001 the company was going through major work force reductions which created a lot of bitterness setting the mood at this particular time. The lay off policy of the past had been very conservative in the sense that it was almost unheard of.

It was acknowledged internally that the company was culturally divided between those being close to the HP kernel (Silicon-valley) and those being physically located further away from the centre. Developers residing outside of the Bay area realm described HP kernel as consisting of very nice and smart people, only passive/aggressive, meaning if you don't like me, you probably wouldn't help me.

It was also acknowledge by managers as well as developers that HP had a very consensus driven culture. In fact it was commonly recognized as the Achilles heal of HP that collaboration was very consensus driven, meaning that no action is taken until everybody agrees. However, in order to introduce POS, consensus had to be replaced by the term informed. This encompassed that the gatekeepers and the managers of POS had to take on the role of informing the community of how to relate to schedules and other important priorities. It was acknowledged that the consensus driven culture had to be replaced by a more centralized leadership, leaving somehow the egalitarian mode of the past. The focus had to be 'make a lot of people a little bit happy' as opposed to 'keeping the smaller group extremely happy'.

Perhaps the biggest cultural issue at the time of introducing POS was that HP was not (especially not by its own developers) considered to be a software company. It was not considered to have an effective software environment, but

identifying themselves as a hardware company (even though top management was making public claims contrary). The claim that HP was not at this time (2001 and 2002) an efficient software company was supported by the argument that HP mainly were selling things that are built, i.e. hardware, and only very little standalone software. Moreover, the company had at the time of this study no software executive reporting directly to the CEO, i.e. no one ultimately responsible for profits and losses, and research related to software. In addition, HP did not have a company-wide source repository and no company-wide sets of coding standards prior to POS.

What is more, in the domain of software development, a lot of the value-add were typically done by contractors and not by HP employees, indicating that the company at this point in time were considering software as an expense rather as a strategic investment.

And lastly HP had been a very dominant actor in the business for a long time. To use a sports analogy: it was very much a culture of try not to lose as opposed to planning to win.

Given what I have already indicated we may confer those independent entities within the HP organization, constituted a challenge when introducing POS. We may think of it in terms of a cultural shift, since in the past independence had been considered to be an advantage for HP. Whereas with the introduction of POS the focus was altered and encompassed giving up local efficiency in order to leverage the overall performance of the organization. However, at this point in time, the local subdivisions of HP were still responsible for their respective results, which may indicate that organizations attempting to implement solutions similar to POS perhaps need to take that into consideration, i.e. how to measure individual teams vs. the overall organizational performance.

### *Reluctance to change*

Some teams and individuals were very reluctant to change methodologies and tools in order to comply with POS. The business units still had to make a business case out of joining the initiative (as already indicated in the previous section). If they could not perceive any immediate gains locally for joining, the local management teams could or would not mandate their respective units to give up efficiency and productivity in order to contribute to the whole combined collaborative effort. Local business units that were growing and showing good results were not as interested in joining POS, especially if dependence on third party collaborators was insignificant.

The local teams still had to make a business case for them selves in order to justify joining the larger POS initiative. For some business units it was acknowledged locally that the common toolset of POS were in fact putting their teams at risk. It was not the collaborative mode specifically, but rather the perceived inefficiency of changing to less efficient set of tools and methodologies.

The introduction of POS presupposed a common toolset. But, as I have already indicated, the community of developers was fairly heterogeneous in the sense that they had a history of using many specialized tools. The very act of introducing a common toolset that they themselves have little of no influence over, inevitably lead to initial resistance.

The POS leadership were utilizing the support team to try to involve as many as possible, bringing in as much as possible of the concerns of the local teams. However, as indicated in Chapter 5, developers and managers working physically distributed had very little of no face-to-face communication with the POS leadership, which resulted in insignificant communication and little authenticity in terms of expressing all the implications that they were facing entering the POS initiative. For some teams, it meant jeopardizing significant revenue for every day that their work was delayed.

Changing tools, methodologies and language use also had significant impact on the individual developer in terms of learning. At the time of introducing POS, many teams had been going though lay-offs leading to significantly less slack. The standardized toolset was perceived as less efficient, and at the same time incurring the extra obstacle of learning. The individual developers also very often expressed that there was a perceived turmoil in terms of tools strategy. By way of rumor developers were expecting yet another generation of tools, which made some teams and individuals wanting to wait for the next generation of tools to minimize the pain of migrating.

The common approach, i.e. tools, methodologies and language use, threatened to become a 'religious war' for some teams. Individual units, teams and developers had in some cases existing tools that were very efficient leading to a large hurdle when shifting to a different set of tools. When I interviewed different teams it became evident that technical offerings must meet business needs also at a local level. The potential solution encompassed initiating co-existing strategies or bridges between different tool sets, at least initially.

### *Information overload and structure*

Since a large number of different teams and individuals began using the POS, before having implemented a straightforward approach, i.e. without following an exact 'cookbook'-example, the system contained a vast number of capabilities such as e.g. investigations, shipping information, and documentation related to projects and actual source code. It was evident that a project of this magnitude potentially could grind to a halt simply because it became too popular. Some teams, developers and managers, expressed that they had an overload of information while lacking a sufficient technique for leveraging information. The term garbage in – garbage out was frequently discussed. Other teams were concerned that the added complexity of the common approach potentially could limit usage. In short, no real knowledge management strategy in place on the outset, leading to great difficulties in terms of overview of projects. The suggested remedy was to implement templates and descriptions of projects in a

coherent and consistent fashion. Moreover, those templates had to be mandated, i.e. people had to be strongly recommended to actually follow those templates. It was suggested that a function of control had to be implemented in order to make sure that the projects were kept up to date both in terms of issues regarding information legitimacy and making sure that obsolete information was updated. And lastly, we may confer from this study that large teams collaborating tend to increase base line work, leaving significantly less time to do something innovative. It seems as if over head tends to increase with broad collaboration, bringing down the speed of development.

## There is always a better way to do things - ideas for future research

A research contribution like the one I have reported shares many similarities with other development projects, where aspects such as time, desires, intentions and commitment have significant importance and in its positive sense it both ignites and inspires. Only optimism initially in a project inevitably leads to pessimism in the end when having to discard interesting themes. And even though these positive aspects have manifested themselves largely throughout this research process, undeniably I have had to leave many important theoretical as well as empirical interesting topics unexplored.

In the future I would like to further study and develop theory about communication and knowledge transfer in open systems for collaboration across organizational boundaries. In particular I would like to study and analyze communication supported by common infrastructures and a standardized language use and its importance for problem solving, juxtaposing a set of conclusions derived from this study.

# REFERENCES


Alderman, Johan, (2001), Sonic boom: Napster, MP3 and the new pioneers of music, London: Fourth Estate.

Akrich, M. (1992), The de-scription of technical objects, In Bijker, W.E. and Law, J. (editors) Shaping technology/building society: Studies in sociotechnical change. MIT Press, pp. 205 – 224.

Akrich, M., & Latour, B., (1992), A summary of a convenient vocabulary for the semiotics of human and nonhuman assemblies. In Bijker, W.E. and Law, J. (editors) Shaping technology/building society: Studies in sociotechnical change. MIT Press, pp. 259-264.

Allen, T., (1977), Managing the flow of technology, Cambridge, MA: MIT Press.

Alvesson, M., Sköldberg, K., (1994), Tolkning och Reflektion, Studentlitteratur, Lund.

Amabile, Teresa, M., (1988), A Model of Creativity and Innovation in Organizations, Research in Organizational Behavior and Human Performance, 30: 41-47.

Arthur, W. Brian, (1994), Increasing returns and path dependence in the economy, Ann Arbor. University of Michigan Press.

Barab, S., A., & Duffy, T., M., (2000), From practice fields to communities of practice, In Theoretical foundations of learning environments, edited by D. Jonassen and S. Land, Mahwah, NJ: Erlbaum, pp. 25-56.

Barbrook, Richard, (1998), The High-Tech Gift Economy, First Monday 3.12, www.firstmonday.dk/issues/issue3_12/barbrook/index.html

Barley, Nigel, (1983), The Innocent Anthroplologist, Notes from a Mud Hut, Waveland Press Inc.

Barton, John, Kindberg, Tim, (2001), The Cooltown User Experience, HP Technical Reports, HPL-2001-22.

Barzelay, Michael, (1992), Breaking through Bureaucracy, Berkeley: University of California Press.

Bavelas, J., B., Hutchinson, S., Kenwood, C., & Matheson, D., H., (1997), Using face-to-face communication as a standard for other communication systems, Canadian Journal of Communication, 22, pp. 5-24.

Bavelas, J., B., & Chovil, N., (2000), Visible acts of meaning. An Integrated message model of language in face-to-face dialogue, Journal of Language and Social Psychology, 19, pp.163-194.

Bentley, R., Hughes, J.,A., Randall, D., Rodden, T., Sawyer, P., Sommerville, I., Shapiro, D., (1992) Ethnographically-informed systems design for air traffic control. In proceeding of the Conference on Computer Supported Cooperative Work, CSCW 92, ACM, New York, pp. 123-129.

Berger, Peter, & Luckmann, Thomas, (1966/1991), The Social Construction of Reality: A Treatise in the Sociology of Knowledge, Penguin Books, London, England.

Berger, Peter, & Luckmann, Thomas, (1995), Modernity, Pluralism and the Crisis of Meaning, Gutersloh: Bertelsmann Foundation Publishers.

Biggs, Maggi, (2001), ASPs bring business to the Web, InfoWorld, May 21, www.findarticles.comcf_0/m0IFW/21_23/74826807/print.jhtml

Brooks, Frederick, P., (1975), The Mythical Man-Month – Essays on Software Engineering, Addison-Wesley Publishing Company, California.

Brown, J., S., & Duguid, P., (1991), Organizational Learning and Communities-of-Practice: Towards a Unified View of Working, Learning, and Innovation, Organizational Science 2 (1): 40-570.

Brown, Steven, D. (2002), *Michel Serres, Science, Translation and the Logic of the Parasite*, SAGE, London, Thousand Oaks and New Delhi, Vol. 19(3):1-27.

Brunsson, Nils, & Jacobsson, Bengt, (2000), A World of Standards, Oxford University Press.

Callon, Michel, (1986), Some elements of a sociology of translation: domestication of the scallops and the fishermen of St Brieuc Bay. In Power, Action and Belief: A New Sociology of Knowledge?, ed. John Law, 196-233, London: Routledge and Kegan Paul.

Callon, Michel, (1980), Struggles to Define What is Problematic and What Is Not: The Socio-logic of Translation, in K.D. Knorr, R. Krohn and R. Whitley (eds.), The Social Process of Scientific Investigation: Sociology of the Sciences, Vol. IV, Dordrecht: D. Reidel.

Callon, Michel, (1991), Techno-economic network and irreversibility, In J. Law, editor, A Sociology of Monsters. Essays on Power, Technology and Domination, Routledge, pp.132-164.

Castells, Manuel, (1996), The Rise of the Network Society, Malden, Mass: Blackwell Publishers.

Castells, Manuel, (2002), The Internet Galaxy, Reflections on the Internet, Business, and Society, Oxford University Press.

Chaiklin, S., & Lave, J. (eds), (1993), Understanding Practice: Perspectives on activity and context, Cambridge: Cambridge University Press,

Chatwin, B, (1988), The Songlines, London: Picador.

Christensen, Clayton, M., (1997), The innovator's dilemma when new technologies cause great firms to fail, Boston, Mass. Harvard Business School.

Clarke, I., Sandberg, O., Wiley, B., & Hong, T.,W., (1999), Freenet: A Distributed Anonymous Information Storage and Retrieval System. In ICSI Workshop on Design Issues in Anonymity and Unobservability.

Clark, Herbert, H., & Carlsson, T., (1982), Hearers and speech acts, Language, 58(2), pp. 332-373.

Clark, Herbert, H., & Marshall, C., R., (1981), Definite reference and mutual knowledge, In A. K. Joshi, B. L. Webber, & I. A. Sag (Eds.), Elements of discourse understanding, Cambridge, England: Cambridge University Press, pp. 10-63.

Clark, Herbert, H., & Schaefer, E., F., (1987), Collaborating on contributions to conversations, Language and Cognitive Processes, 2(1), pp. 19-41.

Clark, Herbert, H., & Schaefer, E., F., (1989), Contributing to discourse, Cognitive Science, 13, pp. 259-294.

Clark, Herbert, H., Wilkes-Gibbs, Deanna, (1986), Referring as a collaborative process, Cognition, 22, pp. 1-39.

Clark, Herbert, H., & Brennan, Susan, E., (1991), Grounding in Communication, in Perspectives on Socially Shared Cognition, eds. Resnick, Lauren, B., & Teasley, Stephanie, D., American Psychological Association, Washington DC.

Clark, Herbert, H., (1996), Using Language, Cambridge University Press.

Clegg, Stewart, (1998), Foucault, Power and Organizations, In Foucault, Management and Organization Theory, edited by Alan McKinlay and Ken Starkey, Sage Publications, London, Thousand Oaks, New Delhi.

Clegg, Stewart R., (1990), Modern Organizations: Organization Studies in the Postmodern World, Newbury Park, CA: Sage.

Clement, A., Wagner, I., (1995) Fragmented exchange: Disarticulation and the need for regionalized communication spaces, in Proceedings of ECSCW'95, Stockholm, Kluwer, pp. 33-49

Cohen, Andrew, L., Cash, Debra, Muller, Michael, J., (2000), Designing to Support Adversarial Collaboration, CSCW'00, December 2-6, 2000. Philadelphia, PA, 2000 ACM 1-58113-222-0/00/0012.

Cooper, Robert, Burell, Gibson (1988), Modernism, Postmodernism and Organizational Analysis: An introduction, Organization Studies 9/1, pp. 91-112.

Czarniawska, Barbara, (1997), Narrating the Organization, Dramas of Institutional Identity, The University of Chicago Press, Chicago and London.

Czarniawska, Barbara, (1999/2005), Writing Management, Organization Theory as a Literary Genre, Oxford University Press.

Czarniawska, Barbara & Sevón, Guje, (1996), Translating Organizational Change, Berlin, de Gruyter.

Daft, R., & Lengel, R., (1984), Information richness: A new approach to managerial behaviour and organizational design, Research in Organizational Behavior, 6, 191-233.

Dertouzos, Michael, L., (2001), The Unfinished Revolution, How to make technology work for us – instead of the other way around, Perennial, Harper Collins Publishers.

DeSanctis, G., & Gallupe, R., (1987), A foundation for the study of group decision support systems, Management Science, 33, pp. 589-609.

DiBona, Chris, Ockman, Sam & Stone, Mark, (1999), Open Sources: Voices from the Open Source Revolution, O'Reilly.

Dinkelacker, Jamie, Garg, Pankaj, K., Miller, Rob, Nelson, Dean, (2001), Progressive Open Source, HP Laboratories Palo Alto, HPL-2001-233, Hewlett-Packard Company. To appear in the *Proceedings of the 24 th International Conference on Software Engineering*, Buenos Aires, Argentina, May 2002.

Doheny-Farina, S., (1986), Writing in an emerging organization, Written Communication 3(2), April.

Doherty-Sneddon, G., Anderson, A., O'Malley, C., Langon, S., Garrod, S., & Bruce, V., (1997), Face-to-face and video mediated communication: A comparison of dialogue structure and task performance, Journal of Experimental Psychology: Applied, 3, 105-125.

Dougherty, Deborah, (1996), Organizing for Innovation, in Handbook of Organization Studies, edited by Clegg, Stewart, R., Hardy, Cynthia & Nord, Walter, R., Sage Publications.

Duncan, S., D., Jr., & Fiske, D., W., (1977), Face-to-face interaction: Research, methods, and theory, Hillsdale, NJ: Lawrence Erlbaum Associates.

Eco, Umberto, (1995), The Search for the Perfect Language, Blackwell Publishing.

Ekman, P, (1982), Emotion in the human face, 2nd ed, Cambridge, England: Cambridge University Press.

El Sawy, Omar, A., and Majchrzak, Ann, (2004), Critical issues in research on real-time knowledge management in enterprises, Journal of Knowledge Management, VOL. 8, NO.4, pp.21-37.

Feyerabend, P.K., (1975) Against Method, Verson, London.

Fillmore, C., (1981), Pragmatics and the description of discourse, In P. Cole (Ed.), Radical and pragmatics, pp. 143-166, New York: Academic Press.

Fogel, Karl, 1999, Open Source Development with CVS, Cariolis, The United States.

Frenkel, S., Korcynski, M., Shire, K., (1995), Reconstituting Work: Trends toward Knowledge Work and Info-Normative Control, Work, Employment and Society, 9:773-796.

Foucault, M., (1977), Discipline and Punish: The Birth of the Prison, Harmondsworth: Penguin.

Foucault, M., (1980), Power/Knowledge: Selected Interviews and other writings 1972-1977, Edited by Colin Gordon, Pantheon Books, New York.

Galegher, J., Kraut, R., E., (1990), Technology for intellectual teamwork: Perspectives on research and design, In Intellectual Teamwork: The Social and Technological Foundations of Cooperative Work, pp. 1-20, J. Galegher, R.E. Kraut, C. Egido (Eds.) Hillsdale, NJ:Erlbaum.

Gantt, Michelle & Nardi, Bonnie, A., (1992), Gardeners and Gurus: Patterns of Cooperation among CAD users, Hewlett-Packard Laboratories, Human-Computer Interaction Department, Palo Alto, CA, ACM 0-89791-513-5/92/0005-0107.

Geertz, Clifford, (1973), The Interpretation of Cultures, Selected Essays, Basics Books Inc., New York.

Giddens, Anthony; (1985). A Contemporary Critique of Historical Materialism (Vol.2): The Nation-State and Violence, Cambridge: Polity Press.

Giddens, Anthony, (1987), Social Theory and Modern Sociology, Cambridge: Polity Press.

Glaser, B., Strauss, A., (1967), The Discovery of Grounded Theory, Aldine, Chicago.

Goffman, E, (1959), The Presentation of Self in Everyday Life, Anchor Books, Doubleday.

Goffman, E., (1981), Forms of Talk, University of Pennsylvania Press.

Goodwin, C., & Goodwin, M., J., (1996) Formulating Planes: Seeing as a Situated Activity. In Middleton, D & Engeström, Y (eds.) Communication and Cognition at Work: Cambridge University Press, Cambridge pp. 61-95.

Grudin, J., (1988) Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces, In Proceeding of CSCW 1988, 85-93, Portland, Oregon:ACM Press.

Gundling, E., (1999), How to Communicate Globally, Training and Development (June), pp.28-31.

Habermas, J., (1973), Theory and Practice, Beacon, Boston, MA.

Hage, J., (1988), Futures of Organizations: Innovations to Adapt Strategy and Human Resources to Rapid Technological Change, Lexington Books, MA.

Hallowell, E., (1999), The human moment at work, Harvard Business Review, January- February, pp. 58-66.

Handfield, Robert, S., & Melnyk, Steve, A., (1998), The scientific theory-building process: A Primer using the case of TQM, Journal of Operations Management, 16(3/4): pp.321-339.

Handy, C., (1995), Trust and the virtual organization, Harvard Business Review, 73, pp. 40-50.

Hargadon, Andrew, B., (1998), Firms as Knowledge Brokers: Lessons in Pursuing Continuous Innovation, California Management Review, 40(3): pp. 209-227.

Heath, C., & Luff, P., (1991), Collaborative Activity and Technological Design: Task Coordination in London Underground Control Rooms, In Proceedings of the Second European Conference on Computer-Supported Cooperative Work, Kluwer, Dordrecht, pp: 65-80.

Heckscher, Charles, & Donnelon, Anne, (eds.), (1994), The Post Bureaucratic Organization, Thousand Oaks, CA: Sage.

Henderson, Rebecca, M., & Clark, Kim, B., (1990), Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms, Administrative Science Quarterly 35(1): pp. 9-30.

Hilton, J., L., & Darley, J., M., (1991), Effects of interaction goals on person perception, In M. P. Zanna (ed.) Advances in experimental social psychology, San Diego: Academic, pp. 235-267.

Himanen, Pekka, (2001), The Hacker Ethic and the Spirit of the Information Age, Secker & Warburg.

Hollan, J., & Stornetta, S., (1992), Beyond being there, In proceedings of CHI'92 Conference on Human Computer Interaction, (119-125), New York: ACM Press.

Holland, J.H., (1975), Adaptation in Natural and Artificial Systems, Ann Arbor, MI: University of Michigan Press.

Hoskin, Keith, (1998), Examining Accounts and Accounting for Management: Inverting Understandings of 'the Economic', in Foucault, Management and Organization Theory, edited by Alan McKinlay and Ken Starkey, Sage Publications Ltd.

Hutchins, Edwin, (1991), Organizing Work by Adaptation, Oranization Science, Vol 2., No. 1, Special issue: Organizational learning: Papers in Honor of (and by) James G. March, pp. 14-39.

Isaac, E., A., & Clark, Herbert, H., (1987), References in conversation between experts and novices, Journal of Experimental Psychology, 116, pp. 26-37.

Jarvenpaa, S., & Leidner, D., (1999), Communication and trust in global virtual teams, Organization Science, 10, pp. 791-815.

Jelinek, M., & Schoonhoven, C., (1990), *The Innovation Marathon: Lessons from High Technology Firms*, Oxford: Basil Blackwell.

Kaghan, W., & Phillips, N., (1998), Building the Tower of Babel: Communities of Practice and Paradigmatic Pluralism in Organization Studies, Organization, Vol. 5, pp. 191-217.

Kiesler, S., Siegel, J., & McGuire, T., W., (1984), Social psychological aspects of computer-mediated communication, American Psychologist, 39, pp. 1123-1134.

Krauss, R., M., (1987), The role of the listener: Addressee influences on message formulation, Journal of Language and Social Psychology, 6, pp. 81-97.

Krauss, R., M., & Fussel, S., R., (1990), Mutual knowledge and communicative effectiveness, In J. Galegher, R. E. Kraut & C. Egido (Eds.), Intellectual

Teamwork: Social and Technical Bases of Collaborative Work, Hilldale, NJ: Erlbaum.

Kraut, R., E., Lewis, S., & Swezey, L., W., (1982), Listener responsiveness and the coordination of conversation, Journal of Personality and Social Psychology, 43, pp. 718-731.

Kraut, R.,E., & Lewis, S.,H., (1984), Some functions of feedback in conversation., In H. Sypher and J. Applegate (Eds.), Understanding interpersonal communication: Social cognitive and strategic processes in children and adults, Beverly Hills, California: Sage Publications.

Kraut, R.,E,. Egido, C., & Galegher, J., (1990), Patterns of contact and communication in scientific research collaboration, In J. Galegher, R. Kraut, and C. Egido (eds.), Intellectual teamwork: Social and technological bases of cooperative work, Hillsdagle, NJ: Erlbaum, pp.149-171.

Kreiner, Kristian, Mouritsen, Jan, (2003), Knowledge Management as Technology: Making Knowledge Manageable, in The Northern Lights – Organization theory in Scandinavia, Barbara Czarniawska & Guje Sevón (eds.), Liber, Abstrakt, Copenhagen Business School Press.

Kreiner, Kristian & Mouritsen, Jan, (2003), Knowledge Management as Technology: Making Knowledge manageable, In The Northern Lights – Organization Theory in Scandinavia, Eds. Barbara Czarniawska & Guje Sevón, Liber Förlag.

Kreiner, Kristian, (2002), Tacit knowledge management: the role of artifacts, Journal of Knowledge Management, Volume 6, Number 2, pp. 112-123.

Kreiner, Kristian,& Tryggestad, Kjell, (2002), The co-production of chip and society: unpacking packaged knowledge, Scandinavian Journal of Management, 18, pp. 421-449.

Kuhn, Thomas (1962/1996), The Structure of Scientific Revolutions, Third Edition, The University of Chicago Press, Chicago, IL.

Kyng, M., (1991), Designing for cooperation-cooperating in design. Communications of the ACM, 34(12): pp. 64-73.

Lanzara, Giovan, Francesco, & Morner, Michèle, (2005), Artifacts rule! How organizing happens in open source software projects, In Actor-Network Theory and Organizing, Edited by Barbara Czarniawska & Tor Hernes, Liber & Copenhagen Business School Press.

LaPlante, Alice & Seidner, Rich. (1999), Playing for Profit: How Digital Entertainment Is Making Big Business Ouf of Child's play, New York: Wiley.

Latour, Bruno, (1986), The Powers of Association, In Power, Action and Belief. A new sociology of knowledge? John Law (ed), Routledge & Kegan Paul, London, pp 264- 280.

Latour, Bruno, (1986/1998), Förbindelsens makt, I Artefaktens återkomst – Ett möte mellan organisationsteori och tingens sociologi, Nerenius och Santérus Förlag.

Latour, Bruno, (1987), Science in Action, Cambridge, MA: Harvard University Press.

Latour, B., (1993), We have never been modern, Harvard University Press, Cambridge, Massachusetts.

Latour, Bruno, (1996), Aramis or the Love of Technology, Harvard University Press.

Latour, Bruno, (1999a), On recalling ANT, In Actor network theory and after, edited by John Law and Johan Hassard, Oxford, Blackwell.

Latour, Bruno, (1999b), Pandora's hope, Essays on the Realit of Science Studies, Harvard University Press, Cambridge, Massachusettes, London, England.

Lave, J. & Wenger, E., (1991), Situated learning: Legitimate peripheral participation, Cambridge: Cambridge University Press.

Law, John, (1991), Introduction: monsters, machines and sociotechnical relations, A Sociology of Monsters: Essays on Power, Technology and Domination, Law, J. (Ed), Routledge, London.

Law, John, (1992), Notes on the Thoery of the Actor-Network: Ordering, Strategy and Heterogeneity, Systems Practice 5(4): pp. 379-393.

Lessig, Lawrence, (1999), Code and other Laws of Cyberpace, Basic Books.

Lessig, Lawrence, (2001), The Future of Ideas, Random House, New York.

Lewicki, Roy, J., Bunker, Barbara, B., (1996), Developing and Maintaining Trust in Work Relationships, In Roderick M. Kramer and Tom R. Tyler (eds.) Trust in Organizations, Thousand Oaks, CA: Sage.

Lewis, D.,K., (1969), Convention: A philosophical study, Cambridge, MA: Harvard University Press.

Lincoln, Y., & Guba, E., (1985), Naturalistic Inquiry, Sage, Beverly Hills, California.

Lindblom, Carl, E., (1990), Inquiry and Change, The Troubled Attempt to Understand and Shape Society, New Haven: Yale University Press.

Lipnack, J., & Stamps, J., (1997), Virtual Teams: Researching across space, time, and organizations with technology, New York: John Wiley and Sons.

Lloyd, P and Boyle, P, Eds., (1998), Web-Weaving: Intranets, Extranet, Strategic Alliances, Oxford: Butterworth-Heineman.

March, J.G., (1991), Exploration and Exploitation in Organizational Learning, Organization Science, Vol.2, No. 1, pp. 71-87.

March, J. G., (1994), A Primer in Decision Making - How Decisions Happen, New York: The Free Press.

March, J. G. (1999), The Pursuit of Organizational Intelligence, Blackwell Publishers, Oxford

Margolis, Howard, (1987), Patterns, Thinking, and Cognition, A Theory of Judgement, The University of Chicago Press.

Markus, M., L., Conolly, T., (1990), Why CSCW applications fail: Problems in the adoption of independent work tools, In Proceedings of CSCW 1990, pp. 371-380, Los Angeles, CA: ACM Press.

Maruyama, Magoroh, (2005a), Prevalent orthodoxy vs. heterogram analysis, APROS 11, Asia-Pacific Researchers in Organization Studies, 11 th International Colloquium, Melbourne, Australia, 4-7 December 2005.

Maruyama, Magoroh, (2005b), Architectural configuration, APROS 11, Asia-Pacific Researchers in Organization Studies, 11 th International Colloquium, Melbourne, Australia, 4-7 December 2005.

Matusov, E. (1993) Intersubjectivity without agreement, Mind, Culture, Activity: An International Journal.

Mauss, Marcel, (1950/1990), The Gift: The Form and Reason for Exchange in Archaic Societies,(transl. W.D. Halls), New York: W.W. Norton.

McMaster, T., Jones, M.C., Wood-Harper, T., (1997), Designing stakeholder expectations in the implementation of new technology: Can we ever learn our lesson? In Kyng, M. and Mathiassen, L., Computers and Design in Context, MIT Press, Cambridge.

Melian, Catharina, Ammirati Burles, Cathy, Garg, Pankaj, Sevón, Guje, (2002), Building Networks of Software Communities in a Large Corporation, HP Laboratories Technical Report, HPL-2002-12 January 23rd, 2002, Software Technology Laboratory, HP Laboratories Palo Alto, California.

Meyerson, Debra, Weick, Karl, E., Kramer, Roderick, M., (1996), Swift Trust and Temporary Groups, In Roderick M. Kramer and Rom R. Tyler, (eds.), Trust in Organizations, Thousand Oaks, CA: Sage.

Mills, Kevin, L., (1999), Introduction to the Electronic Symposium on Computer-Supported Cooperative Work, ACM Computing Surveys, Vol. 31, No.2, June.

Mockus, Audris, Fielding, Roy, T., & Herbleb, James, (2000) A Case Study of Open Source Software Development: The Apache Server, Bell Labs, 263 Shuman Blvd., Naperville, IL 60566 USA, Proceeding of the 22 nd International Conference on Software Engineering, Limerick, Ireland, June 2000.

Moody, Glyn, (2001), Rebel Code, Inside Linux and the Open Source revolution, Perseus Publishing, Cambridge, Massachusetts.

Moon, Jae Yun & Sproull, Lee, (2002) Essence of Distributed Work: The Case of the Linux Kernel, in Distributed Work, edited by Pamela J. Hinds and Sara Kiesler, The MIT Press, Cambridge, Massachusetts, London England, pp. 381-404.

Moran, T., P., Anderson, R., J., (1990), The workaday world as a paradigm for CSCW design, In Proceedings of the Conference on Computer Supported Collaborative Work, pp. 381-394, Los Angeles, CA: ACM Press

Morgan, Gareth, (1996), Images of Organization, SAGE Publications, Inc.

Morley, I., & Stephenson, G., (1969), Interpersonal and interparty exchange: A laboratory simulation of an industrial negotiation at the plant level, British Journal of Psychology, 60, pp. 543-545.

Mukerji, Chandra, (1998), The Collective Construction of Scientific Genius, In Cognition and Communication at Work, Edited by Yrjö Engeström and David Middleton, Cambridge University Press.

Muller, M, Kuhn, S., (1993), Introduction to special issue on participatory design, Communications of the ACM, 36: pp. 24-28.

Nardi, Bonnie, A., & O'Day, Vicki, L., (1999), Information Ecologies: Using Technology with Heart, Cambridge, MIT Press.

Nardi, Bonnie, A., Whittaker, Steve, Schwartz, Heinrich, (2001), NetWORKers and their Activity in Intensional Networks, Forthcoming in a special issue of Computer-Supported Cooperative Work on activity theory and design, guest edited by Bonnie Nardi and David Redmiles 2001.

Nardi, B., Whittaker, S., and Bradner, E., (2000), Interaction and Outeraction: Instant Messaging in Action, Proceedings CSCW 2000, Philadelphia.

Nardi, Bonnie, A., Whittaker, Steve, Schwarz, Heinrich, (2001), NetWORKers and their Activity in Intensional Networks, Forthcoming in a special issue of Computer-Supported Cooperative Work on activity theory and design, guest edited by Bonnie Nardi and David Redmiles 2001.

Nardi, Bonnie, A., (1995), Context and Consciousness, Activity theory and Human-Computer Interaction, The MIT Press, Cambridge, Massachusetts, London, England.

Nardi, B., Whittaker, S., & Bradner, E., (2000), Interaction and outeraction: Instand messaging in action, Proceedings of CSCW 2000, Conference on Computer Supported Cooperative Work (79-88), New York: ACM Press.

Nardi, Bonnie, A., Whittaker, Steve, (2002), The Place of Face-to-Face Communication in Distributed Work, In Distributed Work, edited by Pamela J. Hinds and Sara Kiesler, The MIT Press, Cambridge, Massachusetts, London, England.

Neumann, Peter, G., (2000), Robust Nonproprietary Software, IEEE Symposium on Security and Privacy, Oakland, CA May 15-17.

Newman, R., Newman, J., (1993), Social writing: premises and practices in computerized contexts, in Computer-Supported Collaborative Writing, Springer-Verlag.

Nonaka, Ikujiro, (1994), A dynamic theory of organizational knowledge creation, Organization Science, 5(1): pp. 14-37.

Nonaka, Ikujiro, & Takeuchi, Hirotaka, (1995), The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation, Oxford University Press, Inc.

Nohria, N. and Eccles, R., (1992), Networks and Organizations: structure, form, and action Cambridge: Harvard Business School Press.

O'Conaill, B, Whittaker, S., & Wilbur, S., (1993), Conversations over videoconferences: An evaluation of the spoken aspects of video mediated interaction, Human-Computer Interaction, 8, pp. 389-428.

O'Connor, Gina & Rice, Mark, P., (2001), Opportunity Recognition and Breakthrough Innovation in Large Established Firms, California Management Review 43(2):pp. 95-116.

Olson, G., M., & Olson, J., S., (2000), Distance matters, Human-Computer Interaction, 15, p. 139-179.

Olson, J., S., Teasley, S., Covi, L., & Olson, G., (2002), The (currently) unique advantages of collocated work, In P. Hinds and S. Kiesler, Distributed work, Cambridge, MA: MIT Press , pp. 113-135.

Oravec, J., (1996), Virtual Individuals, Virtual Groups, Cambridge: Cambridge University Press.

Orlikowski, W.J., (1992), Learning from Notes: Organizational Issues in Groupware Implementation, MIT Sloan School Working Paper 3428-92.

Orlikowski, W., (1996), Improvising Organizational Transformation Over Time: A Situated Change Perspective, Information Systems Research, Vol 7, No 1.

Packard, David, (1995), The HP Way, How Bill Hewlett and I Built Our Company, Harper Collins, New York.

Parsons, T., (1956), Suggestions for a sociological approach to the theory of organizations, Administrative Science Quarterly, Vol 1, 1956, pp 63-85.

Parsons, T., ( 1960), Sturctures and Process in Modern Societies, Free Press, Glencoe, IL.

Patton, M., (1990), Qualitative Evaluation and Research Methods, Sage, Beverly Hills, California.

Pettigrew, A.M, (1985), The Awakening Giant; Oxford, UK: Blackwell Publishers

Pettigrew, A.,M, (1990), Longitudinal field research on change: theory and practice, Organization Science, Vol. 1 No. 3, pp.267-292.

Pfeffer, J. & Salancik, G., (1978), The External Control of Organizations: A Resource Dependence Perspective, Harper & Row, New York, NY.

Polanyi, M., (1962), Personal Knowledge: Towards a Post Critical Philosophy, Routledge, London.

Polanyi, M.,( 1967), The Tacit Dimension, Anchor, Garden City, NY.

Popper, K., (1972), Objective Knowledge: An Evolutionary Approach, Clarendon, Oxford.

Popper, K., & Eccles, J.C., (1984), The Self and Its Brain: An Argument for Interactionism, Springer, New York, NY.

Quinn, James Brian, (1992), Intelligent Enterprise, The Free Press.

Raymond, Eric, S., (2001), The Cathedral and the Bazaar: Revised Edition, O'Reilly, CA.

Rehn, Alf, (2001), Electronic Potlatch, A study on new technologies and primitive economic behaviors, KTH INDEK, Doctoral Disseration.

Richards, Lyn, (2005), Handling Qualitative Data, A Practical Guide, Sage Publications, London, Thousand Oaks, New Delhi.

Rogers, Everett, M., (1962, 1995), Diffusion of Innovations, New York, Free Press.

Rose, Nick & Buchanan, Nichola, (2001), A&M Records v Napster: the case and its consequences for copyright owners in the music industry, Copyright world 2001:112, pp. 13-15.

Rushkoff, Douglas, (1999), Playing the Future: What we Can Learn from the Digital Kids, New York: Riverhead Books.

Rutter, M., (1987), Communicating by telephone, Oxford: Pergamon Press.

Sahlin-Andersson, Kerstin, (1996), Imitating by editing success. The construction of organizational fields, In Translating organizational change, edited by B. Czarniawska and G. Sevón, Berlin: de Gruyter,pp. 69-92.

Sahlin-Andersson, Kerstin, Sevón, Guje, (2003), Imitation and Identification as Performatives, In The Northern Lights – Organization theory in Scandinavia, Barbara Czarniawska & Guje Sevón (eds.), Liber, Abstrakt, Copenhagen Business School Press.

Schlager, M., Fusco, J., & Schank, P., (2002), Evolution of an on-line education community of practice, In Building virtual communities: Learning and change in cyberspace, edited by K. A. Renninger and W. Shumar, New York: Cambridge University Press, pp. 125-158.

Schutz, A., (1944), The Stranger, An Essay in Social Psychology, American Journal of Sociology, Vol. 49, No. 6, pp. 499-507.

Schutz, A., (1970), On Phenomenology and Social Relations, University of Chicago Press, Chicago, IL.

Schutz, A., & Luckmann, T., (1973/1985), The Structures of the Life-world, Northwestern University Press, Evanston.

Schwartz, Peter, Leyden, Peter, & Hyatt, Joel, (1999), The Long Boom: A Vision for the Coming Age of Prosperity, Cambridge, Massachusettes, Perseus Publishing.

Self, C., L., (1992), Computer-based conversations and the changing nature of collaboration, in J. Forman (Ed.), New visions of collaborative writing, Boynton/Cook Publishers, Portsmouth, NH.

Serres, M, (1982), *The Parasite*, Baltimore, MD: Johns Hopkins University Press.

Sevón, Guje and Kreiner, Kristian (1998), Constructing R&D Collaboration, Copenhagen Business School Press.

Sevón, Guje, (1996), Organizational imitation in identity transformation. In Translating organizational change, edited by B. Czarniawska and G. Sevón, Berlin: de Gruyter

Shacklett, Mary, (2007), Linux Inside Hewlett-Packard – Hardware, software, neutrality, and expertise makes a winning combination, Linux Magazine, January 2007.

Shapiro, D., Sheppard, B., H., Cheraskin, L., (1992), Business on a Handshake, Negotiating Journal 8(4): 365-377.

Schelling, T., C., (1960), The Strategy of Conflict, Oxford: Oxford University Press.

Short, J., Williams, E., & Christie, B., (1976), The social psychology of telecommunications, New York: Wiley.

Simon, H., (1977), The new science of management decision, NJ: Prentice Hall

Siviter, Douglas, Petre, Marian, Klein, Bruce, (1997), Harnessing technology for effective inter- and intra-institutional collaboration, ITiCSE'97 Working Group Reports and Supplemental Proceedings, ACM1-58113-012-0/97/0010.

Smith, M., A. & Kollock, P., (eds) Communities in Cyberspace, London: Routhledge, 1999.

Snyder, M., (1984), When belief creates reality, in L. Berkowitz (ed.), Advances in experimental social psychology, Orlando, FL: Academic, pp. 247-305.

Spender, J., (1990), Industry Recipes, Wiley, New York, NY.

Sproull, L., & Kiesler, S., (1992), Connections: New ways of working in the networked organization, Cambridge, MA: MIT Press.

Sproull, Lee, S. & Goodman, Paul, S. (1990), Technology and Organizations: Integration and Opportunities, in Technology and Organizations, eds. Paul S. Goodman & Lee S. Sproull, San Francisco: Jossey-Bass, pp. 254-265.

Sproull, Lee, S., Subramani, Mani, Kiesler, Sara, Walker, Janet, H., & Waters, Keith, (1996), When the Interface Is a Face, Human-Computer Interaction, Volume 11, pp. 97-124.

Star, S., L., (1995), The Cultures of Computing, Blackwell Publishers, Oxford.

Stewart, T.A., (1997), Intellectual Capital, The New Wealth of Organizations, Nicholas Brealey Publishing, London.

Strauss, A., (1987), Qualitative Analysis for Social Scientists, Cambridge: Cambridge University Press.

Strauss, A., Corbin, J., (1990), Basica of Qualitative Research, Sage, Newbury Park.

Suchman, L. & Trigg, R., (1991), Understanding practice: video as a medium for reflection and design, In Greenbaum, J. & Kyng, M. (eds.), Design at Work: Lawrence Erlbaum Associates, Hillsdale, NJ, pp.65-90.

Sutor, Robert, S., (2006), Open Standards vs. Open Source, How to think about software, standards, and Service Oriented Architecture at the beginning of the 21st century, www.sutor.com/newsite/blog-open

Tankjær, Christian, (1999), Performing a Transnational Region – The Importance of 'Open House Strategy,MPP Working Paper No.8/99, Department of Management, Politics and Philosophy, Copenhagen Business School, ISBN:87-90403-61-4.

Tankjær, Christian,(2000), Øresund as an Open House Strategy by Invitation, in Invoking a transnational metropolis, (eds.) Per Olof Berg & Orvar Löfgren, pp. 165-190, Lund, Studentlitteratur.

Tarde, G. (1903/1962), The Laws of Imitation, translated by E.C. Parsons with introduction by F. Giddings, New York, Henry, Holt and Co.

Toft, Peter, Coleman, Derek, and Ohta, Joni, (2000), "A Cooperative Model for Cross-Divisional Product Development for a Software Product Line," in Patrick Donohoe, Ed., *Software Product Lines:Experiences and Research Directions*, Kluwer Academic Publishers.

Toupin, Laurie Ann, (2001) Collaborating for profit. (Collaboration software), Design News, www.findarticles.com/cf_0/m1068/16_56/78055859/print.jhtml

Townley, Barbara, (1993), Foucault, Power/Knowledge, and Its Relevance for Human Resource Management, The Academy of Management Review, Vol. 18, No. 3, pp-518-545.

Tuomi, Illka, (2001) Internet, Innovation, and Oper Source: Actors in the Network, firstmonday.org/issues/issue6-1/-tuomi/index/html

Utterback, James, M. (1994), Mastering the Dynamics of Innovation, Boston, MA: Harvard Business School Press.

Van de Ven, A., H., Delbecq, D., & Koening, R., Jr., (1976), Determinants of coordination modes within organizations, American Sociological Review, 41 (2), pp. 322-338.

Van de Ven, A., H., Angle, H.,L., & Poole, M., S., (1989/2000), Research on the Management of Innovation, The Minnesota Studies, Oxford University Press.

Van Maanen, J., (1988), Tales of the Field: on Writing Ethnography, Chicago: University of Chicago Press.

Victor, B. and Boynton, A., C., (1998) Invented here: Maximizing your organization's internal growth and profitability, Boston: Harvard Business School Press.

Von Krogh, G, & Roos, J., (1995), A perspective on knowledge, competence and strategy, Personnel Review, Vol. 24, No 3, pp. 55-76.

Vygotsky, L., S., (1925/1982), Consciousness as a problem in the psychology of behaviour, In Collected Works: Questions of the Theory and History of Psychology, Moscow: Pedagogica.

Walther, J., (1994), Anticipated ongoing interaction versus channel effects on relational communication in computer mediated interaction, Human Communication Research, 20, pp. 473-501.

Warner, R., M., & Sugarman, D., B., (1986), Attributions of personality based on physical appearance, speech, and handwriting, Journal of Personality and Social Psychology, 50, pp.792-799.

Wayner, P., (2001), Free for All: How Linux and the Free Software Movement Undercut the High-Tech Titans, New York, Harper Business.

Weber, M., (1947), The Theory of Social and Economic Organization, New York: The Free Press.

Weber, M., (1978), Economy and Society: An Outline of Interpretative Sociology, Berkeley: University of California Press.

Weber, Steven, (2004), The Success of Open Source, Harvard University Press, Cambridge, Massachusetts, and London, England.

Weick, K., (1976), Educational Organizations as Loosely Coupled Systems, Behavioral Science, 19, pp 357-367.

Weick, K., (1979; The Social Psychology of Organizing, 2nd ed., McGraw Hill, New York.

Weick, K., (1982), Management of Organizational Change among Loosely Coupled Elements, In P.S. Goodman & Associates (Eds.) Change in Organizations, Jossey-Bass, San Francisco, California, pp.375-408.

Weick, K., (1995), Sense Making in Organizations, Sage, Beverly Hills, CA.

Westney, D., Eleonor, (1987), Imitation and Innovation: The transfer of westerns organizational patterns to Meiji Japan, Cambridge, Massachusettes, London, Harvard University Press.

Wildeman, L., (1998), Alliances and Networks: The Next Generation, International Journal of Technology Management, 15, 1/2, pp. 96-108.

Wilson; D.C, (1992), A Strategy of Change: Concepts and Controversies in the Management of Change, London, UK: Routledge.

Zahra, S. & Covin, J., (1995), Contextual influences on the corporate entrepreneurship-performance relationship: a longitudinal analysis, Journal of Business Venturing, 10: pp. 43-58.

Zeleny, Milan, (1989), Knowledge as a new form of capital: Division and reintegration of Knowledge, Human Systems Management, 8 (1): pp. 45-58.

Zuboff, S., (1988), In the Age of the Smart Machine, Basic Books, New York.

Zuboff, S., (1996), The Emperor's New Information Economy, In *Information Technology and Changes in Organizational Work*, edited by W. Orlikowski, G. Walsham, M. Jones and J. DeGross. London: Chapman & Hall.

# APPENDIX

## Template for interviews

Below are the questions that served as a point of departure for the interviews I conducted with developers, managers and third party developers. Depending on the interviewees' answers, the continuous interview developed differently from one occasion to another. I allowed the interviewees to develop their answers, and frequently added additional questions for reasons of clarification, whenever I perceived it as necessary for my own understanding. Consequently the below questions may serve only as an illustration of the questions and discussions that took place during the course of the interviews.

**Background information about interviewees**

- **Categories**

Male /Female

Age

Educational background

Years within Hewlett-Packard

Position within Hewlett-Packard

Geographical Location

Adopter categories

Innovators (CDP/CS users from the outset)

Early adopters

Early majority

Late majority

Laggards

- **Introductory questions**

How do you perceive CDP/CS (i.e. what is innovative about it? How would you describe CDP/CS in your own words?

How did you learn about CDP/CS?

What made you decide to start using it (or not using it?)

What are the consequences in your work of using CDP/CS?

What are the advantages/disadvantages of using CDP/CS in your individual situation (for the workgroup)Have you extended your network to include people outside your formal workgroups as a result of CDP/CS?

- **Success story**

Will you share with meone (at least one) successful project where you have utilized CDP/CS? Preferably one including 3rd party collaborators

- **Characteristics of Innovation**

Relative advantage (To what degree do you perceive CDP/CS as a better tool for collaborative work than those you have used before?

Compatibility. (The degree to which CDP/CS is perceived as being consistent with existing values, past experiences, and needs of potential adopters

Complexity (The degree to which an innovation is perceived as difficult to understand and use)

Trialability (The degree to which CDP/CS may be experimented with on a limited bases)

Observability (The degree to which the results of an innovation are visible to others)

- **Motivation**

What is your personal motivation for using collaborative tools such as CDP/CS?

Is the reward system, i.e. your salary or incentive program, tied to the team effort or to the individual effort?

Do you feel that individual and/or team recognitions are important or unimportant?

What are the effects on leadership in a collaborative environment, e.g. who takes on the role of leader (module designer, architect etc?)

Do you feel that your contributions are important to the HP community? Does CDP/CS make your contributions visible and accessible to the comunity?

What are the pros and cons of working in a collaborative setting?

Do you have suggestions for improvements?

- **Communication**

How do collaborative tools, such as CDP/CS effect communication?

What kind of information/knowledge is hard to share in a virtual setting? What kind of sharing can only take place within the team in a 'physical setting'?

- **Third party interaction**

What experiences do you have of collaboration including third parties? Please exemplify, or describe a particular project?

What are the benefits of third party collaboration, and is it challenging e.g. for commercial reasons, customer relations reasons, Intellectual Property reasons etc.?

What do you perceive as challenging when interacting with third parties?

How important is face-to-face interaction with third parties? E.g. when setting up a project – can you describe how work is actually conducted and/or coordinated? Is it easy to get them to adapt to the tools and the technology?

Do you have to assist third parties, e.g. in terms of training or other related issues?

What is the positive impact of CDP/CS in the relation to third party involvement in the development process?

- **Support**

What level of support do you receive when working with CDP/CS. – Is it sufficient? Suggestions for improvements? (System support)

Do management support collaborative work?+

What kind of support have you recieved from the CDP core team?

Do you have suggestions for improvements?

- **Security and Trust**

What are the advantages respective disadvantages of making contributions accessible to a larger community?

Do you feel comfortable making your ideas and your contributions accessible to a larger community of developers?

What can you as a manager/HR-officer do to encourage and develop trust in the workgroup?

Do you have any particular concerns about security issues and issues of trust that you would like to share with me?

- **Strategy**

What are the possible benefits to HP from utilizing collaborative tools such as CDP/CS?

Are you aware of any other companies that try to adopt similar tools for collaboration?

How important do you perceive third party involvement to be in the process of innovation?

Do you believe that collaborative tools, such as CDP/CS will be important for HP in the future?

What possible business opportunities for HP do you acknowledge resultning from collaborative experiences?

# EFI, The Economic Research Institute

Published in the language indicated by the title. A complete publication list can be found at *www.hhs.se/efi*. Books and dissertations can be ordered from EFI via e-mail: EFI.Publications@hhs.se

## Reports since 2002

### 2007

#### Books

Wijkström, Filip och Torbjörn Einarsson. *Analysmodell för sektors-överskridande statistik. Fallet vård och omsorg.*

#### Dissertations

Valiente, Pablo. *Re-innovating the Existing. A Study of Wireless IS Capabilities to Support Mobile Workforces.*

### 2006

#### Books

Thodenius, Björn. *Organisering av kunskap. En studie av Wallenberg Consortium North.*

Wijkström, Filip och Torbjörn Einarsson. *Från nationalstat till näringsliv? Det civila samhällets organisationsliv i förändring.*

Wijkström, Filip, Einarsson, Stefan och Larsson, Ola. *Staten och det civila samhället. Idétraditioner och tankemodeller i den statliga bidragsgivningen till ideella organisationer.*

Östman, Lars. *Lysande ögonblick och finansiella kriser. Dramaten under ett sekel.*

#### Dissertations

Argenton, Cedric. *Quality provision in duopoly.*

Beckerman, Carina. *The Clinical Eye. Constructiong and Computerizing an Anesthesia Patient Record.*

Borglund, Tommy. *Aktievärden i fokus – internationell påverkan på intressentrelationer vid förvärv och fusion.*

Breman, Anna. *The Economics of altruism, paternalism and self-control.*

Edquist, Harald. *Technological breakthroughs and productivity growth.*

Eklund, Jana. *Essays on Forecasting and Bayesian Model Averaging.*

Frostenson, Magnus. *Legitimitetskontrollen – en studie av etiska värderingars roll i gränsöverskridande förvärv och fusioner.*

Gaspar, Raquel M. *Credit Risk and Forward Price Models.*

Gustafsson, Peter. *Essays on Trade and Technological Change.*

Hopkins, Elisabeth. *Is a higher degree of local currency pricing associated with lower exchange rate pass-through? A study of import pricing in 51 Swedish industries.*

Kling, Ragnar. *Developing Product Development in Times of Brutal Change.*

Langenskiöld, Sophie. *Peer Influence on Smoking: Causation or Correlation?*

Lychnell, Lars-Olof. *"Och fungerar det inte, gör vi på något annat sätt" — En klinisk fallstudie av IT-relaterat förändringsarbete i småföretag*

Meitz, Mika. *Five Contributions to Econometric Theory and the Econometrics of Ultra-High-Frequency Data.*

Mendicino, Caterina. *Financial Market Imperfections, Business Cycle Fluctuations and Economic Growth.*

Ovanfors, Anna. *Essays on Nonlinear Time Series Analysis and Health Economics.*

Paltseva, Elena. *Essays on Commitment and Inefficiency in Political Economy.*

Rogberg, Martin. *Den modeföljande organisationen. Om acceptansen av TQM och andra populära managementmodeller.*

Silvennoinen, Annastiina. *Essays on Autoregressive Conditional Heteroskedasticity*

Sjögren, Ebba. *Reasonable Drugs. Making Decisions with Ambiguous Knowledge.*

Slinko, Irina. *Essays in Option Pricing and Interest Rate Models.*

Wilander, Fredrik. *Essays on Exchange Rates and Prices.*

## 2005

### Books

Andersson, Per, Susanne Hertz and Susanne Sweet (eds). *Perspectives on market networks – boundaries and new connections.*

Charpentier, Claes. *IT inom omsorgen. Förväntade effekter av införande av IT-system för utförarna inom äldre- och handikappomsorgen.*

Dembrower, Maria. *Entreprenörskap i industriella nätverk.*

Lind, Johnny och Göran Nilsson (redaktörer). *Ekonomistyrningens metoder, sammanhang och utveckling – En vänbok till Lars A Samuelson.*

Samuelson, Lars A. *Organizational governance and control – a summary of research in the Swedish society.*

Svedberg Nilsson, Karin, Roger Henning och Karin Fernler (eds). *En illusion av frihet? Företag och organisationer i regelsamhället.* EFIs Årsbok 2005. EFI/Studentlitteratur.

**Dissertations**

Andersson, Martin. *Making a Difference – Project Result Improvement in Organizations.*

Arvidsson, Per. *Styrning med belöningssystem – Två fallstudier om effekter av belöningssystem som styrmedel.*

Bems, Rudolfs. *Essays in International Macroeconomics.*

Berg-Suurwee, Ulrika. *Nya trender, nya nämnder – effekter av en stadsdelsnämndsreform inom kultur och fritid.*

Björkman, Hans. *Learning from members – Tools for strategic positioning and service innovation in trade unions.*

Bodnaruk, Andriy. *Essays in Empirical Corporate Finance and Portfolio Choice.*

Clapham, Eric. *Essays in Real Estate Finance.*

Dareblom, Jeanette. *Prat, politik och praktik – Om individers möten med strukturer i en kommunal satsning på kvinnors företagande.*

Fromm, Jana. *Risk Denial and Neglect: Studies in Risk Perception.*

Hjelström, Anja. *Understanding International Accounting Standard Setting – A Case Study of IAS 12, Accounting for Deferred Tax.*

Hortlund, Per. *Studies on Swedish Banking 1870-2001.*

Lindahl, Therese. *Strategic and Environmental Uncertainty in Social Dilemmas.*

Linnarsson, Håkan. *Alliance for Innovation. A structural perspective on new business development in cooperative ventures.*

Madestam, Andreas. *Developing Credit Markets.*

Nilsson, Roland. *The Market Impact of Short-Sale Constraints.*

Nordfält, Jens. *Is consumer decision-making out of control? Non-conscious influences on the consumer decision-making process for fast moving consumer goods.*

Nordin, Fredrik. *Externalising Services – Walking a Tightrope between Industrial and Service Logics.*

Parmler, Johan. *Essays in Empirical Asset Pricing.*

Simbanegavi, Witness. *Price Discrimination, Advertising and Competition.*

Thodenius, Björn. *Användning av ledningsinformationssystem: en longitudinell studie av svenska storföretag.*

Tolis, Christofer. *Framing the Business – Business Modelling for Business Development.*

Östberg, Per. *Corporate Disclosure and Investor Recognition.*

## 2004

### Books

Ahrne, Göran och Nils Brunsson (red). *Regelexplosionen.*

Lind, Johnny. *Strategi och ekonomistyrning. En studie av sambanden mellan koncernstrategi, affärsstrategi och ekonomistyrning.*

Lind, Johnny och Walter Schuster (red). *Redovisningens teori, praktik och pedagogik. En vänbok till Lars Östman.*

Sevón, Guje och Lennart Sjöberg (red). *Emotioner och värderingar i näringslivet.* EFIs Årsbok 2004.

Wijkström, Filip and Stefan Einarsson. *Foundations in Sweden – Their scope, roles and visions.*

### Dissertations

Anderson, Anders. *Essays in Behavioral Finance.*

Balsvik, Gudrun. *Information Technology Users: Studies of Self-Efficacy and Creativity among Swedish Newspaper Journalists.*

Blix, Magnus. *Essays in Mathematical Finance – Modelling the Futures Price.*

González Gómez, Andrés. *Nonlinear dynamics and smooth transition models.*

Grönqvist, Erik. *Selection and Moral Hazard in Health Insurance: Taking Contract Theory to the Data.*

Ivarsson Westerberg, Anders. *Papperspolisen – varför ökar administrationen i moderna organisationer.*

Jutterström, Mats. *Att påverka beslut – företag i EUs regelsättande.*

Jönsson, Kristian. *Macroeconomic Aspects of Capital Flows to Small Open Economies in Transition.*

Larsson, Pär. *Förändringens villkor. En studie av organisatoriskt lärande och förändring inom skolan.*

Lagerwall, Björn. *Empirical Studies of Portfolio Choice and Asset Prices.*

Malmsten, Hans. *Properties and Evaluation of Volatility Models.*

Marshall, Cassandra. *Dating for Innovation. Creating and Recognizing Opportunities through Collaborative Interorganizational Relationships in Fluid Environments.*

Mattsson, Susanna. *På gränsen mellan ordning och oordning – tingens betydelse vid marknadsombildningar. En studie av svenska postväsendets ombildning under 1990-talet.*

Nilsson, Charlotte. *Studies in Environmental Economics: Numerical Analysis of Greenhouse Gas Policies.*

Nilsson, Hans. *Medborgaren i styrsystemet – beskrivning av VAD och HUR i styrning av kommunal verksamhet.*

Nystedt, Jens. *Competition, Regulation and Integration in International Financial Markets.*

Pajuste, Anete. Corporate Governance and Controlling Shareholders.

Richtnér, Anders. *Balancing Knowledge Creation. Organizational Slack and Knowledge Creation in Product Development.*

Salabasis, Mickael. *Bayesian Time Series and Panel Models – Unit Roots, Dynamics and Random Effects.*

Sandberg, Rickard. *Testing the Unit Root Hypothesis in Nonlinear Time Series and Panel Models.*

Skallsjö, Sven. *Essays on Term Structure and Monetary Policy.*

Strikholm, Birgit. *Essays on Nonlinear Time Series Modelling and Hypothesis Testing.*

Söderström, John. *Från Produkt till Tjänst. Utveckling av affärs- och miljöstrategier i produktorienterade företag.*

Talia, Krim. *The Scandinavian Currency Union, 1873–1924 – Studies in Monetary Integration and Disintegration.*

## 2003

### Books

Lundahl, Mats (editor). *Globalization and Its Enemies.* EFIs Årsbok 2003.

Sundgren, Bo, Pär Mårtensson, Magnus Mähring and Kristina Nilsson (editors). *Exploring Patterns in Information Management. Concepts and Perspectives for Understanding IT-Related Change.*

### Dissertations

Andersson, Henrik. *Valuation and Hedging of Long-Term Asset-Linked Contracts.*

Bergman, Malin. *Essays on Human Capital and Wage Formation.*

Damsgaard, Niclas. *Deregulation and Regulation of Electricity Markets.*

Eklund, Bruno. *Four Contributions to Statistical Inference in Econometrics.*

Hakkala, Katariina. *Essays on Restructuring and Production Decisions in Multi-Plant Firms.*

Holgersson, Charlotte. *Rekrytering av företagsledare. En studie i homosocialitet.*

Ivaschenko, Iryna. *Essays on Corporate Risk, U.S. Business Cycles, International Spillovers of Stock Returns, and Dual Listing.*

Lange, Fredrik. *Brand Choice in Goal-derived Categories – What are the Determinants?*

Le Coq, Chloé. *Quantity Choices and Market Power in Electricity Market.*

Magnusson, Peter R. *Customer-Oriented Product Development – Experiments Involving Users in Service Innovation.*

Meisiek, Stefan. *Beyond the Emotional Work Event Social Sharing of Emotion in Organizations.*

Mårtensson, Anders. *Managing Mission-Critical IT in the Financial Industry.*

Nilsson, Göran. Processorientering *och styrning – Regler, mål eller värderingar?*

Sandberg, Robert. *Corporate Consulting for Customer Solutions Bridging Diverging Business Logics.*

Sturluson, Jon Thor. *Topics in the Industrial Organization of Electricity Markets.*

Tillberg, Ulrika. *Ledarskap och samarbete – En jämförande fallstudie i tre skolor.*

Waldenström, Daniel. *Essays in Historical Finance.*

Wallén, Ulrika. *Effektivitet i grundskolan i anslutning till en stadsdelsnämnds-reform.*

Ögren, Anders. *Empirical Studies in Money, Credit and Banking – The Swedish Credit Market in Transition under the Silver and the Gold Standards, 1834–1913.*

## 2002

### Books

Schuster, Walter. *Företagets Valutarisk – En studie av horisontella och vertikala styrprocesser.*

Sjöstrand, Sven-Erik och Pernilla Petrelius. *Rekrytering av koncernstyrelsen – Nomineringsförfaranden och styrelsesammansättning med fokus på kvinnors ställning och möjligheter.* EFI/SNS Förlag.

Löwstedt, Jan och Bengt Stymne (red). *Scener ur ett företag – Organiseringsteori för kunskapssamhället.* EFIs Årsbok 2002. EFI/Studentlitteratur.

### Dissertations

Barinaga, Ester. *Levelling Vagueness – A Study of Cultural Diversity in an International Project Group.*

Berglund, Johan. *De otillräckliga – En studie av personalspecialisternas kamp för erkännande och status.*

Bolander, Pernilla. *Anställningsbilder och rekryteringsbeslut.*

Damjanovic, Tatiana. *Essays in Public Finance.*

Ekman, Mattias. *Studies in Health Economics – Modelling and Data Analysis of Costs and Survival.*

Heyman, Fredrik. *Empirical Studies on Wages, Firm Performance and Job Turnover.*

Kallifatides, Markus. *Modern företagsledning och omoderna företagsledare.*

Kaplan, Michael. *Acquisition of Electronic Commerce Capability – The Cases of Compaq and Dell in Sweden.*

Mähring, Magnus. *IT Project Governance.*

Nilsson, Mattias. *Essays in Empirical Corporate Finance and Governance.*

Schenkel, Andrew. *Communities of Practice or Communities of Discipline – Managing Deviations at the Øresund Bridge.*

Skogsvik, Stina. *Redovisningsmått, värderelevans och informationseffektivitet.*

Sundén, David. *The Dynamics of Pension Reform.*

Ternström, Ingela. *The Management of Common-Pool Resources – Theoretical Essays and Empirical Evidence.*

Tullberg, Jan. *Reciprocitet – Etiska normer och praktiskt samarbete.*

Westling, Gunnar. *Balancing Innovation and Control – The Role of Face-to-face Meetings in Complex Product Development Projects.*

Viklund, Mattias. *Risk Policy – Trust, Risk Perception, and Attitudes.*

Vlachos, Jonas. *Risk Matters – Studies in Finance, Trade and Politics.*