Perspectives on IT Innovation in Organisations

Daniel Curto Millet
Candidate for M.Sc. In Analysis Design and Management of Information Systems
Information Systems and Innovation Group
Department of Management
London School of Economics

IT innovation is often considered a main actor for organizational transformation and can be an important asset for gaining competitive advantage (Ciborra, 2002a; Orlikowski, 1996). Yet, there is little agreement on how IT innovation happens. It is however critical to understand how IT innovation takes place in an organization so as to better facilitate its development. We identify and discuss several perspectives of how IT innovation happens and argue that they are not necessarily exclusive, simply casting a certain light upon a problem depending on the concerns of the practitioner.

1. Introduction

IT innovation has become increasingly important in organizations and has led many researchers to study its impact and its benefits (Ciborra, 2002c; Orlikowski, 1996; Orlikowski, 2000). However, there are many perspectives of how such innovation happens.

In this essay, we focus solely on understanding how IT innovation takes place in an organization, how it is developed or develops and what the different perspectives on the subject are. The difference between these perspectives is grounded on the difference in the underlying concerns from the various fields of study of the practitioners.

Technology innovation is often sold and advertised as being the result of the application of best practice engineering methods. This perspective often rests on the assumption that the use of rational techniques will result in a reliable and useful product (Stenning, 1986). However, this represents only one view that tends to see or idealise IT development as a clean and logical work. Other views cast a different light on how technology innovation takes place. These take a broader picture of the development of technology innovation and argue that innovation is not planned, but rather improvised (Ciborra, 2002c) or that it emerges through the use of technology (Lin and Cornford, 2000).

First, we identify and discuss four ways that drive the development of IT innovation: through rational process, through improvisation, through use and finally, the manner in which institutions can shape IT innovation. We then discuss some limitations of this literature review and then conclude.

2. IT Innovation in Organization: Differing Perspectives

In this section we will expose the differing viewpoints, starting with the engineering perspective, to then discuss the various sociological and socio-technical perspectives before finally considering the impact that institutions have on IT innovation.

2.1. Technology Innovation through Rational Process

An important view on innovation has its origins in the engineering world. This is significant, as engineers are the ones responsible for the creation of technological IT systems, and hence their view of technological innovation has implications into how systems are created. This, in turn, will have repercussions upon their concerns when engineering IT systems development.

Engineers tend to view IT innovation as being brought about by the construction of a software system by a rational semi-formal or formal structured process (Stenning, 1986). This is despite the impossibility for humans to ever follow a completely rational process (Parnas and Clements, 1986). Even if it is impossible to do so, Parnas argues that it is important to imitate rational processes. Rational processes are based upon the premise that the application of a certain methodology in the construction of a software system will increase the reliability of the software system and the productivity of the engineers involved (Stenning, 1986).

There are many different methodologies but all usually involve more or less the same tasks: requirements gathering and specification, some form of analysis and design, implementation and maintenance (Arlow and Neustadt, 2002; Boehm, 1988). The names assigned to these tasks vary from methodology to methodology. Boehm (1988), for example, refers to analysis as design and design as "detailed design". The methodologies are, however, semantically the same. More than just proposing a set of tasks, software systems development methods give guidance as to how the tasks should be carried out. Two such methods which stand at opposites are the waterfall-like models and iterative and incremental models. The former being a pure top-down approach where the various tasks are seen as steps while the latter is often seen as more flexible (Avison and Fitzgerald, 2003) by allowing the iteration between the tasks and building a system incrementally after each iteration.

The requirements task is the most interesting one for our purposes since it is this activity that will define technological innovation as viewed by engineers as it determines the purpose of the software. The requirements task tends to be viewed as the most difficult activity in building software, but also one of capital importance (Brooks, 1987). Carrying out requirements tasks involves many sub-activities which put together form the field of study of Requirements Engineering (RE).

There are many definitions of RE. In his paper, Nuseibeh (2000) provides a definition from Zave (1983) which describes RE as being "a branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems […] and the relationship of these factors". This idea is shared by Jackson who captures this notion of having the software, the *machine*, and the context

and environment, the *world*, as two different intersected entities (Jackson, 1995; Jackson and Zave, 1995). The world describes the problem (i.e. the requirements and goals) and the machine represents the actual program (i.e. the solution to be built in its own machine context). The most interesting idea is that requirements are solely contained in the world, and not in the machine. Hence, the purpose of the machine is only found in the world.

Out of the five main tasks, four are solution-oriented in that they are concerned only with the construction of the software. Only the requirements activity focuses on the problem domain, acting as a bridge between problem and solution. Technological innovation is thus considered a process grounded in the RE activity where the purpose of the system is identified. Yet the engineering view is always turned into bringing rationality and reasoning abilities in order to understand and create IT innovation. These five tasks highlight the main concerns that the engineers have to deal with: they must create an IT product that is both functionally useful and responds to required constraints in terms of quality of the product. Their concern is thus focused on the functions and on the quality constraints that the product *should* deliver.

Many alternative views, however, oppose this positivist perspective. We study one such a view in the next section.

2.2. Technology Innovation through tinkering and improvisation

One of the alternative views on technology innovation opposes the idea that innovation is carried out only through a rational process (Ciborra, 2002c). This in a way joins Parnas' idea that rational design processes are an ideal that will never be reached giving several reasons for this among which is the fact that IT is developed by humans (Parnas and Clements, 1986). This view is somewhat shared by Brooks who argues that there is no 'silver bullet' in software engineering which will improve the creation of IT (Brooks, 1987). Therefore, it would seem that rationale processes are unlikely to be an absolute method to the success of IT development. Ciborra, as opposed to Parnas however, argues *against* the need to imitate them. According to Brooks (1987), what distinguish successful IT products are not so much the methods used, but rather the skills of the designers. Ciborra takes this idea quite drastically forsaking the use of rational processes altogether, whereas Brooks simply argues that their sole use is no guarantee of success.

Furthermore, Ciborra argues that the world is complex and that it cannot be captured in a static picture (Ciborra, 2002b; Ciborra and Hanseth, 1998). This constitutes another criticism of the rational processes. What the world constitutes for Ciborra is not completely transparent. He often gives examples from a managerial and economic point of view, relating such concepts as management agendas and top-down managerial approaches to the way IT systems are developed. These would seem, a priori, as separate concerns entirely. However, we argue in this paper that they really are the same. In the engineering view, requirements belong to the *world* and the *machine* acts upon the *world* by way of specifications that it must render true (Jackson and Zave, 1995). As previously described, all the other activities of the software lifecycle are based upon the requirements and their subsequent satisfaction. The world that Ciborra refers to really equates to the world used in Zave's definition (1983) from which goals are drawn and which an IT system implementation would need to satisfy. World goals here would be, for example, aligning a business and IT strategy or focusing on the need for flexible infrastructure (Ciborra, 2002b; Ciborra and Hanseth, 1998). Of course, these are high level goals, but they nonetheless provide the rationale for the lower level requirement specifications that can be used and transformed into a working IT system. According to Ciborra then, a static picture of the world is impossible to define since the world is dynamic and cannot be captured into what engineers view as requirement specifications, which are static specifications by nature.

Instead of rational processes and planned structured methods, some authors argue that the ideas of tinkering and improvisation in IT systems are better able to cope with the dynamic picture that the world offers (Ciborra, 2002c; Elbanna, 2006). Tinkering takes place when an IT system comes to support a function that had not been planned during its creation. Also, they argue that the success of many IT products have actually originated from tinkering and improvisation and not from structured methodologies and rational thinking as management likes to advertise once the IT product has become a success (Ciborra, 2002c; Elbanna, 2006).

This idea of tinkering is not exempt of criticism. Ciborra, does not mention any structured methodology in particular, seemingly encompassing all the methods as being articulated from the management hierarchy in a very much planned top-down approach which, once articulated, encompasses the strategic agenda to be carried forward to the next stages of development, ending in a useful IT product. This, however, would represent one type of system development methodology, namely the waterfall model (Royce, 1970). Other methods are much more incremental and iterative in nature, allowing a learning process of what is doable by the system and in understanding what the desired real world goals and requirements are (Arlow and Neustadt, 2002; Boehm, 1988).

There also is an important difference in concerns. Ciborra (2002c) constructs his argumentation from an economic and managerial perspective, referring to IT artefacts as possible conduits towards sustainable competitive advantage if methodologies other than structured ones are used. The main concern here is the economic advantage coming from the innovation of an IT artifact. The engineers on the other hand must consider a plethora of widely different concerns stemming from the main goals for creating the IT system. They also must consider all the technical details of the IT artifact, whether it will be usable and whether it will satisfy the main goal. For this reason there is a lot of research in the line of what Stenning (1986) proposed: ways to validate that each step taken brings closer the creation of an IT artifact that is both reliable and in line with what is needed (Lamsweerde, 2000).

Ciborra gives various examples of successful IT products which came about through tinkering (Ciborra, 2002a). All of these examples have two things in common: firstly, they came to support a certain function that was not expected during the creation of the IT artifact; secondly, the idea to build the function came through the use of the IT product. Hence, the ability to hack and tinker seems to require an actual working IT product. Thus, Ciborra's idea of tinkering does not necessarily invalidate the use of structured methods as an

approach to create IT systems from scratch. Furthermore, it is much more likely that one will find new ideas while using a built IT system than to imagine it before it is constructed. This is interesting since it indicates that the real use of an IT innovation is revealed when in context, when it has to fuse with the working practices of the users. This perhaps is the main reason why improvisation can be seen to work well: the organizational settings are easily apparent when the new innovative function is developed. Hence, tinkering is the development of IT innovation in context, satisfying certain identified social or organizational need. As a result, tinkering is an interesting framework to support and explain the finding of new ideas that will be truly innovative but often seems to require an existing system.

2.3. Technology Innovation through usage

Another view from the socio-technical perspective on technology innovation in organization is that technology innovation does not simply happen, but rather changes through use (Ciborra, 2002d; Lin and Cornford, 2000). These authors use different terms to explain this phenomenon: Ciborra uses the word "*drifting*" while Lin and Cornford use the phrase "*emergent phenomena*". These terms are equivalent in that they refer to a difference in the usage of a technology from what was planned to its actual use in organizations. This means that technology is, at least partly, designed in its usage.

According to Lin and Cornford (2000), not only does technology influence organizational settings, but the inverse is also true: organizational settings influence how technology innovation takes place. The meaning of technology innovation thus really takes form in the workplace when the planned innovation is put against the expectations and working practices of the humans in the organizational setting which will define the technology innovation (Lin and Cornford, 2000).

This has several implications. Firstly, the emergence phenomena would imply that organizations can have an indirect and unplanned impact upon IT innovation. This is opposed to the direct and planned impact which can be brought about by rational processes as viewed by the engineering perspective. Thus, it might be interesting to investigate how rational design processes can accommodate this view (Lin and Cornford, 2000).

Secondly and following from the first point, there is a double relationship between the technology and the organization in terms of change. This is in stark contrast with the engineering view which perceives change as technologically driven (Orlikowski, 1996).

Thirdly, since organizational settings have an impact on the design of the IT innovation through its use, then the institutions must also have an impact upon the development of IT innovation.

2.4 Technological Innovation and Institutions

Yet another possible factor that can determine how technological innovations are developed in organizations is *institutions* (Orlikowski and Barley, 2001). Institutions are "norms, rules, beliefs and taken-for-granted assumptions" (Orlikowski and Barley, 2001) cited from Barley (Barley and Tolbert, 1997). These institutions have an impact upon the organizational settings in which these factors are created. These in turn will have an impact upon how technology is developed and used inside an organization. A good example of how institutions can affect technological innovation comes from Orlikowski and Barley (2001) where they evoke questions of intellectual property, individual privacy, organizational control, national sovereignty and corporate boundaries and their impact upon e-commerce.

It is also interesting to note that different countries will have different institutional beliefs which will affect how IT products are developed (Thanasankit, 2002). Culture and institutions thus inevitably affect the way IT creation is brought about. It would therefore seem that the creation of IT does not entirely stem from conscious effort, whether planned and rational or through bricolage, but that unconscious elements such as culture and institutions also affect the way IT innovation is developed.

3. Conclusion

Different perspectives on the development of IT innovation were reviewed. The engineering perspective prescribes the use of as rational processes as possible, and albeit acknowledging that rational processes are idealizations, one should strive to follow them as closely as possible (Parnas and Clements, 1986).

Critiques of these rational methodologies argue that technology innovation is unplanned and that strict processes cannot capture a dynamic world (Ciborra, 2002c). However, improvisation is often carried out of an already working IT artifact, and thus does not necessarily invalidate the engineer's point of view when creating systems from scratch.

Another view argues that technology innovation is constantly created via its use (Lin and Cornford, 2000). Thus, the development of technology is not a one-off creation, but rather an ongoing learning process.

Institutions also have an impact upon the development of IT innovation, affecting the way development has to cope with cultural differences and how emergence phenomena take place (Orlikowski, 1996; Thanasankit, 2002).

These different perspectives stem mainly from the different concerns practitioners hold. The engineers focus on solving technological complexity and uncertainty in order to provide quality IT products that are useful. Their scope is narrow (Orlikowski and Barley, 2001), but so can be the sociologists' view when it comes to understanding technology (Orlikowski and Iacono, 2001). This essay does not point out which perspective is best, as none really is; they are simply different lenses to see the world in a certain interested way. Sociologists argue that engineers see the world abstractedly, but the same argument can be said about the sociologists who see technology abstractedly. Rather, the different practitioners should foment understanding and find ways to accommodate the different perspectives.

The political and institutional impact on IT innovation could benefit from a further discussion, especially in what concerns requirements and how these are dealt with.

Two important questions stem from the idea of emergence phenomena. First, how can a rational process accommodate

this? Second, what is the relationship between emergence phenomena and the requirements engineering phase which defines what the IT product should do. An IT product with poorly conducted requirements engineering might be more likely to develop emergence phenomena than a well conducted one.

Another important question which emerges from this work concerns the link between these perspectives and organizational change. In her paper, Orlikowski (1996) argues for a perspective which considers organizational transformation as being emergent. This perspective implies that there can be a mutual influence between technology and the organization, arguably leading to an evolution or accommodation of both rather than radical change. It would thus seem interesting to investigate how technology innovation can influence organizational change, and to what degree.

References

Arlow, J. and I. Neustadt (2002) "What Is the Unified Process—Chapter 2" in *Uml and the Unified Process*Addison Wesley, pp. 22-39.

Avison, D. and G. Fitzgerald (2003) "Where Now for Development Methodologies?" *Communications of the ACM,* **46 (1),** pp. 79-82.

Barley, S. R. and P. S. Tolbert (1997) ""Institutionalization and Structuration: Studying the Links between Action and Institution,"", *Organization Studies,* **18 (1),** pp. 93-117.

Boehm, B. (1988) "A Spiral Model of Software Development and Enhancement", *Computer,* **21 (5),** pp. 61-72.

Brooks, F. (1987) "No Silver Bullet—Essence and Accidents of Software Engineering", *Computer,* pp. 10-19.

Ciborra, C. (2002a) "Bricolage—Chapter 3" in *The Labyrinths of Information: Challenging the Wisdom of Systems*Oxford, pp. 29-53.

Ciborra, C. (2002b) "Gestell—Chapter 4" in *The Labyrinths of Information: Challenging the Wisdom of Systems*Oxford, pp. 55-82.

Ciborra, C. (2002c) *The Labyrinths of Information: Challenging the Wisdom of Systems,* Oxford University Press,

Ciborra, C. U. (2002d) "Dérive—Chapter 5" in *The Labyrinths of Information: Challenging the Wisdom of Systems*Oxford, pp. 83-101.

Ciborra, C. U. and O. Hanseth (1998) "From Tool to Gestell: Agendas for Managing the Information Infrastructure", *Information Technology and People,* **11** pp. 305-327.

Elbanna, A. R. (2006) "The Validity of the Improvisation Argument in the Implementation of Rigid Technology: The Case of Erp Systems", *Journal of Information Technology,* **21** pp. 165-175.

Jackson, M. (1995) in *Proceedings of the 17th international conference on Software engineering*ACM, Seattle, Washington, United States.

Jackson, M. and P. Zave (1995) in *Proceedings of the 17th international conference on Software engineering*ACM, Seattle, Washington, United States.

Lamsweerde, A. v. (2000) "Requirements in the Year 2000: A Roadmap", *Communications of the ACM*.

Lin, A. and T. Cornford (2000) *Sociotechnical Perspectives on Emergence Phenomena,* Springer-Verlag, Surrey, England.

Nuseibeh, B. and S. Easterbrook (2000) "Requirements Engineering: A Roadmap". In *Proceedings of International Conference on Software Engineering (ICSE-2000),* Limerick, Ireland, 4-11 June, ACM Press.

Orlikowski, W. (1996) "Improvising Organizational Transformation over Time: A Situated Change Perspective", *Information Systems Research,* **7 (1),** pp. 63-92.

Orlikowski, W. (2000) "Using Technology and Constituting Structures: A Practice Lense for Studying Technology in Organizations", *Organization Science,* **11 (4),** pp. 404-428.

Orlikowski, W. and S. R. Barley (2001) "Technology and Institutions: What Can Research on Information Technology and Research on Organizations Learn from Each Other?" *MIS Quarterly,* **25 (2),** pp. 145-165.

Orlikowski, W. and C. S. Iacono (2001) "Research Commentary: Desperately Seeking The "It" In It Research—a Call to Theorizing the It Artefact", *Information Systems Research,* **12 (2),** pp. 121-134.

Parnas, D. L. and P. C. Clements (1986) "A Rational Process: How and Why to Fake It", *IEEE Transactions on Software Engineering,* **SE-12 (251-256)**.

Royce, W. (1970) "Managing the Development of Large Software Systems: Concepts and Techniques".

Stenning, V. (1986) "Software Engineering Present and Future", *Proceedings of Contemporary approaches to the software engineering process.*

Thanasankit, T. (2002) "Requirements Engineering-Exploring the Influence of Power and Thai Values", *European Journal of Information Systems,* **11 (2),** pp. 128-141.

Zave, P. (1983) "Classification of Research Efforts in Requirements Engineering", *ACM Computing Surveys,* **29 (4),** pp. 315-321.

About the author

Daniel Curto Millet graduated with Honours from University College London with an MEng degree in Computer Science. He has participated in research on recommender systems at UCL. His final year research project concerned the specification of measurable quality goal patterns for requirements engineering. Daniel is now an MSc student in ADMIS at the LSE and his dissertation concentrates on power relations and requirements specification in the open source software community.