To appear in Interacting With Computers

Socio-technical systems: From design methods to systems engineering

Gordon Baxter* and Ian Sommerville

School of Computer Science, University of St. Andrews, North Haugh, St. Andrews, Fife KY16 9SX, UK.

Abstract

It is widely acknowledged that adopting a socio-technical approach to system development leads to systems that are more acceptable to end users and deliver better value to stakeholders. Despite this, such approaches are not widely practised. We analyse the reasons for this, highlighting some of the problems with the better known socio-technical design methods. Based on this analysis we propose a new pragmatic framework for socio-technical systems engineering (STSE) which builds on the (largely independent) research of groups investigating work design, information systems, computer-supported cooperative work, and cognitive systems engineering. STSE bridges the traditional gap between organisational change and system development using two main types of activity: sensitisation and awareness; and constructive engagement. From the framework, we identify an initial set of interdisciplinary research problems that address how to apply socio-technical approaches in a cost-effective way, and how to facilitate the integration of STSE with existing systems and software engineering approaches.

Keywords: socio-technical systems; systems engineering; software engineering

* Corresponding author: phone +44 1334 463268; fax +44 1334 463278; email: gdb@cs.st-andrews.ac.uk

To appear in Interacting With Computers

1 Introduction

Socio-technical systems design (STSD) methods are an approach to design that consider human, social and organisational factors[1], as well as technical factors in the design of organisational systems. They have a long history and are intended to ensure that the technical and organisational aspects of a system are considered together. The outcome of applying these methods is a better understanding of how human, social and organisational factors affect the ways that work is done and technical systems are used. This understanding can contribute to the design of organisational structures, business processes and technical systems. Even though many managers realise that socio-technical issues are important, socio-technical design methods are rarely used. We suspect that the reasons for their lack of use are, primarily, difficulties in using the methods and the disconnect between these methods and both technical engineering issues, and issues of individual interaction with technical systems.

The underlying premise of socio-technical thinking is that systems design should be a process that takes into account both social *and* technical factors that influence the functionality and usage of computer-based systems. The rationale for adopting socio-technical approaches to systems design is that failure to do so can increase the risks that systems will not make their expected contribution to the goals of the organisation. Systems often meet their technical 'requirements' but are considered to be a 'failure' because they do not deliver the expected support for the real work in the organisation. The source of the problem is that techno-centric approaches to systems design do not properly consider the complex relationships between the organisation, the people enacting business processes and the system that supports these processes (Norman, 1993; Goguen, 1999).

We argue here that there is a need for a pragmatic approach to the engineering of socio-technical systems based on the gradual introduction of socio-technical considerations into existing software procurement and development processes. We aim to address problems of usability and the incompatibility of socio-technical and technical systems development methods. Our long-term research goal is to develop the field of *socio-technical systems engineering* (STSE). By this, we mean the systematic and constructive use of socio-technical principles and methods in the procurement, specification, design, testing, evaluation, operation and evolution of complex systems.

We believe that it is not enough to simply analyse a situation from a socio-technical perspective and then explain this analysis to engineers. We also must suggest how socio-technical analyses can be used constructively when developing and evolving systems. Many companies have invested heavily in software design methods and tools, so socio-technical approaches will only be successful if they preserve and are compatible with these methods. We must avoid terminology that is alien to engineers, develop an approach that they can use, and generate value that is proportionate to the time invested.

These are challenging objectives and, to achieve them, we must draw on research from a range of disciplines. There are at least four significant research communities that have explored and addressed socio-technical issues that affect the specification, design and operation of complex computer-based systems:

1. Researchers interested in work, in general, and the workplace. An interest in the design of work was the original stimulus for proposing socio-technical approaches. Mumford (1983) and Eason's (1988) research typify the approach of this community. The original objective was to make work more humanistic and the initial focus was on manufacturing systems. As computers have become pervasive in the workplace, however, the community has also examined the relationships between work and its computer-based support noting, for example, that the computer system can shape and constrain work practices (Eason, 1997)

[1] Here we use the term organisational to describe factors that are related to the company or business per se, whilst we use the term social to describe factors that are related to the relationships between people who work together within and across organisations.

2. Researchers interested in information systems. Information systems are large-scale systems that support the work of the enterprise and this community recognised at an early stage that socio-technical issues were significant (e.g., J. C. Taylor, 1982). This community has generally taken a broad perspective on the relationships between information systems and the enterprise rather than focusing on specific aspects of computer-supported work (e.g., Avison, et al., 2001).
3. Researchers interested in computer-supported cooperative work (CSCW). This community has focused on the minutiae of work arguing that the details of work, as understood through ethnographic studies, profoundly influence how computer-based systems are used. Suchman's seminal book (1987) which triggered work in this area, was followed by many ethnographic studies of systems in different settings (Ackroyd, et al., 1992; Bentley, et al., 1992a; Heath & Luff, 1992; Heath, et al., 1994; Rouncefield, 1998; Clarke, et al., 2003). Many of these were concerned with co-located work (e.g. in control rooms) and most did not consider wider enterprise issues that affect system requirements and design.
4. Researchers interested in cognitive systems engineering. This community, exemplified by the work of Hollnagel and Woods (2005; Woods & Hollnagel, 2006), has been primarily interested in the relationships between human and organisational issues and systems failure. Their main focus has been on control systems and health care and this community has not been much concerned with broader information systems.

Whilst these communities have had some mutual awareness, we believe that it is fair to say that there has been relatively little cross-fertilisation across communities. For example, in Mumford's (2006) review article, there are no references to the strands of work in CSCW or cognitive systems engineering, and few references to the information systems literature.

Sitting alongside these communities, with some awareness of socio-technical issues, is the HCI research community. Some areas of HCI have clearly been influenced by socio-technical ideas, including usability (e.g., Nielsen, 1993; Mayhew, 1999; Krug, 2005) and human/user centred system design (e.g., Gould & Lewis, 1985; Norman & Draper, 1986; Gulliksen, et al., 2003). Holistic design, for example, is identified by Gulliksen et al. (2003) as a key principle, and they note the need to explicitly consider the work context and social environment. More generally, much of the focus has been on sensitisation to socio-technical issues (e.g., Dix, et al., 2004 has a chapter on this topic). There has been little work on how these socio-technical issues might directly influence the design of an interface to a complex software system (understandably so: we believe this to be a significant research challenge). By the same token, some researchers in the ubiquitous computing community have been influenced by socio-technical thinking (Andy Crabtree, et al., 2006), although most research in this general area focuses on the development and evaluation of new technologies.

We believe that we need to integrate the work of these disparate communities under a common heading of socio-technical systems engineering. Our objectives here, therefore, are to summarise the contributions of the different research communities in this area, and to propose a practical vision for further developments. We do not provide a complete survey of socio-technical systems design (that would be impossibly long). Instead we present different perspectives on STSD, which we use as a basis for introducing a pragmatic framework for STSE that is deliberately limited in scope but which leaves room for the application of different STSD approaches. In this paper we have focused our discussions on organisational systems, but we believe that STSE applies to other types of systems based on Commercial Off The Shelf equipment and applications, for example, or domestic systems. After laying out our framework, we go on to propose a research agenda for socio-technical systems engineering where we identify research problems that need to be addressed to make STSE a practical reality.

Section 2 introduces the notion of STSD and Section 3 briefly discusses STSD approaches. Section 4 discusses shortcomings of these existing approaches. Section 5 introduces the notion of socio-technical systems engineering, identifying two main types of STSE activities. We conclude by identifying outstanding research issues that can be used to shape the discipline of socio-technical systems engineering.

2 Socio-technical Systems Design

The term *socio-technical systems* was originally coined by Emery and Trist (1960) to describe systems that involve a complex interaction between humans, machines and the environmental aspects of the work system—nowadays, this interaction is true of most enterprise systems. The corollary of this definition is that all of these factors—people, machines and context—need to be considered when developing such systems using STSD methods. In reality, these methods are more akin to philosophies than the sorts of design methods that are usually associated with systems engineering (Mumford, 2006). STSD methods mostly provide advice for sympathetic systems designers rather than detailed notations and a process that should be followed.

The term socio-technical systems is nowadays widely used to describe many complex systems, but there are five key characteristics of open socio-technical systems (Badham, et al., 2000):

- Systems should have interdependent parts.
- Systems should adapt to and pursue goals in external environments.
- Systems have an internal environment comprising separate but interdependent technical and social subsystems[2].
- Systems have equifinality. In other words, systems goals can be achieved by more than one means. This implies that there are design choices to be made during system development.
- System performance relies on the joint optimisation of the technical and social subsystems. Focusing on one of these systems to the exclusion of the other is likely to lead to degraded system performance and utility.

STSD methods were developed to facilitate the design of such systems. We have restricted our scope here to this class of systems, and do not consider deeply embedded systems, for example, where there is usually no social subsystem involved.

From its inception in the period immediately after World War II, by what is now called The Tavistock Institute, until the present day, there have been several attempts at applying the ideas of STSD. Some of these were successful, others less so (Mumford, 2006). The prevailing climate within a particular company (or sometimes within a country) affected attitudes towards the idea of STSD: where attitudes were positive this often led to the successful uptake of the ideas.

Mumford (2006) provides an historical overview of developments in STSD. The general aim was to investigate the organisation of work, with early work in STSD focused mostly on manufacturing and production industries such as coal, textiles, and petrochemicals. The aim was to see whether work in these industries could be made more humanistic. In other words, the intention was to move away from the mechanistic view of work encompassed by Taylor's (1911) principles of scientific management, which largely relied on the specialisation of work and the division of labour.

The heyday of STSD was, perhaps, the 1970s and the early part of the 1980s. This was a time when there were labour shortages, and companies were keen to use all means available to retain their existing staff. Apart from the usual cultural and social reasons, companies could also see good business reasons for adopting socio-technical ideas. The XSEL (eXpert SELler) system of the Digital Equipment Corporation (DEC), for example, was developed using STSD (see Mumford & MacDonald, 1989 for a retrospective view). It was an expert system designed to help DEC sales staff assist customers in properly configuring their VAX computer installations. This system was a success and at its peak the family of expert systems, including XSEL, that were being used to support configuration and location of DEC-VAX computers was claimed to be saving the company tens of millions of dollars a year (Barker & O'Connor, 1989). Of course, it is impossible to assess the contribution of STSD to this success but the example illustrates that socio-technical approaches can be used effectively in real systems engineering.

---

[2] Here Badham et al. are using the term social subsystem to refer to people, work context and organisations

By contrast, the latter part of the 1980s and the 1990s were possibly the low point in STSD's history. The adoption of lean production techniques and business process re-engineering dominated, and STSD was largely sidelined. Dankbaar (1997), however, suggested that these different methods (STSD, BPR, etc.) can all learn from each other. The late 1980s and early 1990s also saw the emergence of ethnographic studies of work, stimulated by Suchman's (1987) seminal research at Xerox PARC. These ethnographic approaches (e.g., Heath & Luff, 1991) highlighted the significance of socio-technical issues in the design of software-intensive systems (e.g., Blomberg, 1988).

The 21st century has seen a revival of interest in socio-technical approaches as industries have discovered the diminishing returns from investment in new software engineering methods. However, socio-technical ideas and approaches may not always be explicitly referred to as such (Avgerou, et al., 2004). The ideas appear in areas such as participatory design methods, CSCW and ethnographic approaches to design. Indeed, one of the key tenets of STSD is a focus on participatory methods, where end users are involved during the design process (e.g., Greenbaum & Kyng, 1991). However, these methods, all of which have their roots in STSD, differ in important respects. Participatory design, which covers a whole range of methods (e.g., see Muller, et al., 1993), often involves the users (or user representatives) effectively moving into the territory of the system developers for the duration of the project. By contrast, empathic design (Leonard & Rayport, 1997) and contextual design (e.g., Beyer & Holtzblatt, 1999), which reflect STSD ideas, adopt the inverse view and put the developers into the users' world as part of the development process.

The field of CSCW came about partly in response to a need to discuss the development of group support applications (Grudin, 1994), but it has implicit roots in socio-technical thinking. Bowker et al. (1997) make the link explicit, dealing with the socio-technical system and CSCW, as does the recent special issue of the journal *Computer Supported Cooperative Work* which deals with CSCW and dependability in health care systems (Procter, et al., 2006). The field of dependability[3] (Laprie, 1985; Avizienis, et al., 2004) is also intrinsically concerned with socio-technical systems, although this field sometimes uses the term 'computer-based systems' to refer to socio-technical systems.

STSD methods continue to be advocated for systems development and appear to be particularly suited to some application areas. Since the late 1990s, for example, STSD has been frequently advocated within health informatics for the development of health care applications (e.g., Whetton, 2005). Many such systems are under-utilised because they introduce ways of working that conflict with other aspects of the user's job, or they require changes to procedures that affect other people's responsibilities. One of the keys to developing systems that are acceptable to the users is a detailed understanding of the underlying work structures. In other words, what is required is a socio-technical approach (Berg, 1999, 2001; Berg & Toussaint, 2003).

Most recently, in the UK, the need for STSD has been highlighted by issues surrounding the National Health Service's ongoing National Programme for Information Technology (NPfIT; see Brennan, 2007 for a commentary on the programme). Even though many of the developments to date within the NPfIT have been imposed in an essentially top-down manner, there are still areas where there is a role for STSD, even if only at a local level (Eason, 2007).

Although the vast majority of applications have been implemented in the workplace, socio-technical ideas are equally applicable in other settings where technology is deployed. In recent years, there has been an increasing uptake of technology in the home, particularly as smart home technologies and assistive technologies. The requirements for home-based systems are somewhat different from those of workplace systems. Sommerville and Dewsbury (2007), for example, developed a model for the design of dependable domestic systems, which adopts a socio-technical view in which the system comprises the user, the home environment, and the installed technology.

[3] See also www.dirc.org.uk

3 Socio-technical Systems Design Approaches

Socio-technical systems design has been manifested in a wide range of different methods. Different traditions developed in different countries at different times have led to different approaches (see Mumford, 2006 for a fairly comprehensive historical review). The individual methods, to some extent, reflect different national cultures and approaches to work and work organisation. The consequence has usually been that each method is tailored to a particular market, which partly explains why there have never been any significant or successful attempts to integrate approaches to create a more general, standardised method of STSD.

There has been limited transferability of the available methods. In general, those who developed a method have had most success in applying it. Mumford's ETHICS (1983, 1995), for example, was mostly used in the USA when Mumford worked directly with organisations based there, such as DEC (see section 2).

As the nature of the different markets has changed, the methods have not always kept pace. In some instances, the methods have been reactively refined—ETHICS, for example has recently been paired with agile methods of software development (Hickey, et al., 2006). In most cases, however, there has not been any reconsideration of the role of the earlier fundamental notions of STSD. Whether this is because STSD is not deemed relevant to modern ways of working, or because there is simply ignorance of these approaches is an open question. STSD remains an active field of research and practice, although in many cases it is the ideas, rather than the original methods, that are being applied.

Even though the notion of user participation lies at the heart of STSD, there has been a disappointing uptake of user-centred methods in general. Eason (2001), for example, found that none of the 10 most widely advocated methods (including socio-technical design) were in common use. Furthermore, even where the methods were being used, user involvement was still largely to assist in the development of a techno-centric system. Users were not seen as participants in an integrated systems development process to produce a system that took appropriate account of social and organisational requirements.

One area where user participation has been taken seriously is in software development using agile methods, such as extreme programming (XP), Dynamic Systems Development Method (DSDM), and Scrum (see Abrahamsson, et al., 2002 for a review and analysis of these methods). These methods incorporate at least some face-to-face user involvement—although in practice who plays the role of the user can often depend on who is available to talk to the developers—and use short iterative development cycles to develop evolutionary prototype solutions in a manner that takes account of local contingencies (e.g., see Boehm & Turner, 2004). However, agile methods are mostly concerned with end-user requirements, and make the simplistic assumptions that (a) suitable users are available to interact with the development team and (b) the user requirements are congruent with broader organisational requirements. While there are certainly interesting ideas emerging from agile methods, their focus on interaction with individual users does not address the need for broader socio-technical awareness in systems engineering.

In addition to the approaches covered by Mumford's (2006) extensive review, we have also identified several other approaches that encompass socio-technical ideas. We believe that these other approaches can also help inform the development of socio-technical systems:

1. Soft Systems Methodology (SSM; Checkland, 1981; Checkland & Scholes, 1999), which builds on ideas from action research, has its roots in systems engineering rather than the social sciences. SSM treats purposeful action as a system: logically linked activities are connected together as a whole, and the emergent property of the whole is its purposefulness. One of SSM's key features is its focus on developing an understanding of the problem (SSM uses the more generic term problematic situation). This understanding takes into account the roles, responsibilities, and concerns of the stakeholders that are associated with the particular problem. The understanding of the problem provides the basis for the solution, which again takes into account stakeholders' differing viewpoints. SSM explicitly acknowledges that the final solution is based on attempting to accommodate the views (and needs) of the various

stakeholders. We believe that problem understanding is one of SSM's principal strengths, but it can also be used to develop information models of the more technical aspects of a system. It has been used to evaluate existing information systems too (Checkland & Poulter, 2006).

2. Cognitive Work Analysis (CWA; Rasmussen, et al., 1994b; Vicente, 1999) was developed to analyse the work that could be performed by complex socio-technical systems. It is therefore a formative approach based on predicting what a system could do, in contrast to most approaches which are either normative (how work *should* be done) or descriptive (how work *is* done).
3. The socio-technical method for designing work systems (Waterson, et al., 2002) focuses on system design. It is used to identify tasks that have to be allocated to machines (and hence implemented using IT) and also considers those tasks that have to be performed by humans (both individually, and as teams). This method is designed for general use in function allocation and socio-technical work systems.
4. Ethnographic workplace analysis (e.g., Suchman, 1987; Hughes, et al., 1997; Viller & Sommerville, 2000; Martin & Sommerville, 2004) emphasises the situated nature of action, and has investigated how the results from ethnographic studies can inform the design of socio-technical systems. Ethnographic workplace analysis has largely focused on the operational issues that affect the functionality and use of a system. It has highlighted how workarounds and dynamic process modifications are commonplace and revealed the importance of awareness and the physical workplace in getting work done.
5. Contextual design (Beyer & Holtzblatt, 1999) is aimed at designing products directly from the designer's comprehension of how the customer actually performs work. It is founded on the notion that any system inherently embodies a particular way of working, which then largely dictates how the system will be used and how it will be structured. Contextual Design gives rise to activities that are focused on the front end of design, and, in particular, on customers and their work.
6. Cognitive systems engineering (Hollnagel & Woods, 2005; Woods & Hollnagel, 2006) deals with the analysis of organisational issues, and offers some practical support for systems design. CSE uses observation as a tool for analysing work in context, and uses abstraction on the results to identify patterns in the observations that occur across work settings and situations, thereby increasing the understanding of sources of expertise and failure.
7. Human centred design (International Standards Organisation, 2010), which follows principles such as basing the design upon an explicit understanding of users, their tasks, and the environments in which those tasks are carried out. It also includes as one of the four main design activities the understanding and specification of the context in which the system will be used, and explicitly refers to consideration of social and cultural factors, including working practices and the structure of the organisation.

STSD methods can be categorised based on the how well they deal with the three broad stages in the systems engineering lifecycle: analysis, design and evaluation. There are also some general sets of principles that provide abstract guidance for developing socio-technical systems, rather than directly supporting detailed aspects of systems development. These include Cherns' (1976, 1987) and Clegg's (2000) principles, which cover aspects such as power and authority (Cherns, 1987), and the fact that design should reflect the needs of the stakeholders (Clegg, 2000).

Table 1 indicates how some of the better-known approaches relate to the different phases of the systems engineering life cycle. All of the methods tend to be most strongly related to one particular phase of the life cycle, although they still provide some support for the other phases. Whilst several of the approaches offer support for most phases in the systems engineering lifecycle, our belief is that none of the approaches provide complete coverage for all of the phases.

Table 1. Relationship between socio-technical systems design approaches and the development phases of the systems engineering life cycle. A double tick (√√) indicates that a particular design

approach provides strong support for the associated phase of the life cycle; a single tick (√) indicates some support.

	General	Analysis	Design	Evaluation
Cherns' (1976, 1987) Principles	√√			
Clegg's (2000) Principles	√√			
Scandinavian approaches (e.g., Bjerknes & Bratteteig, 1995)		√	√	√
Dutch Integral Organisation Renewal (De Sitter, et al., 1997)		√	√	√
ETHICS (Mumford, 1983, 1995)	√	√√	√	√
Cognitive Work Analysis (Rasmussen, et al., 1994a; Vicente, 1999)		√√		
Socio technical method for designing work systems (Waterson, et al., 2002)		√	√	
Ethnographical Workplace analysis (Hughes, et al., 1992)		√	√	
Contextual Design (Beyer & Holtzblatt, 1999)	√	√√	√	
Cognitive systems engineering (Hollnagel & Woods, 2005)	√	√√	√	√
Human-centred design (International Standards Organisation, 2010)	√	√	√	√

4 Problems with existing approaches to socio-technical systems design

The development of STSD methods has identified and attempted to address real problems in understanding and developing complex organisational systems which, nowadays, inevitably rely on large-scale software-intensive systems. Despite positive experiences in demonstrator projects, however, these methods have not had any significant impact on industrial software engineering practice. The reasons for this failure to adopt and maintain the use of STSD approaches have been analysed in several places, and from several viewpoints (e.g., Mathews, 1997; Mumford, 2000, 2006). We summarise the main problems identified by these authors below, and also discuss other issues that have arisen in our own use of STSD methods.

4.1 Inconsistent terminology

There is considerable variation in what people mean by the term *socio-technical system* and this is inevitably confusing to potential adopters of these approaches. The term has its original roots in organisational and clinical psychology, in work carried out by the Tavistock Institute in the 1950s and 1960s. However, it is also often closely linked with the field of management science in the UK, where the ETHICS method (Mumford, 1983, 1995) was developed at the Manchester Business School.

Nowadays, many different fields have adopted the term, often using their own interpretation—sometimes focusing on the social system, sometimes on the technical, but rarely on both together. This may help to explain the somewhat disparate nature of the literature (e.g., Griffiths & Dougherty, 2001).

It is important that people involved in a specific systems development project have an agreed understanding of what is meant by the term socio-technical system. This particularly applies to the development team, in order to make sure that they focus on the appropriate social and technical aspects of the system and how these are interdependent and interact. The critical point is that there needs to be agreement about the social and technical elements of the system that need to be jointly optimised.

4.2 Levels of abstraction

Similar to the problems of terminology are problems in determining the appropriate levels of abstraction to use when analysing and describing socio-technical systems. Rather than using different terms to describe the same thing, though, here we are talking about people describing the same system but using different levels of abstraction, often based on the fact that they draw the system boundaries in different places. There is a tendency by some to decompose the system into separate social and technical systems. The depth of analysis for each of the (sub-)systems is then given different emphasis, with the focus often falling mostly on the technical aspects of the system (Eason, 2001).

Finding the appropriate level of abstraction is critical, but often not easy. Hollnagel (1998), for example, criticises the work on socio-technical systems for over-emphasising the context, which includes the organisational aspects, at the expense of neglecting the individual. He argues that current approaches cannot satisfactorily explain why humans perform erroneous actions and, hence, cannot be used in human reliability analysis. When this view is taken to the extreme, undesirable events are simplistically seen as the result of organisational failings, which stack the odds against the human operator, who is then portrayed as the innocent victim of these failings. In other words, it overlooks the fact that the context includes individuals, often working as part of a team, who through their own volition could still theoretically perform the correct action.

4.3 Conflicting value systems

In attempting to make sense of the literature, Land (2000) suggested that it can be divided into two basic categories. Each category is based on a set of values that underpins much of the thinking around socio-technical systems.

The first set of values is a fundamental commitment to humanistic principles. In other words, the designer is aiming to improve the quality of working life and job satisfaction of the employee(s). It is argued that increases in productivity will automatically follow, and that these will generate added value for the company. Early approaches to STSD were particularly concerned with ensuring that humanistic principles were considered during the design and deployment of new systems.

The second set is often described as managerial values. In this view, socio-technical principles are regarded as a means of helping to achieve the company's objectives (particularly economic ones). Humanistic objectives are perceived as having limited inherent value, but if their achievement leads to better employee performance, and the company benefits as a result, then all well and good. Approaches such as Contextual Design are primarily geared to the use of STSD as a means of building systems that provide more effective organisational support.

Ethnographic analysis can be considered as an intermediate category. Most work in this area has adopted an ethnomethodological approach where, it is claimed, the analysis of the work is not influenced by any particular theoretical framework or intended outcome. The extent to which such analysis is truly value-free is, of course, debateable.

Problems arise when these different sets of values come into conflict. The dichotomy between the first two categories helps to explain why, in some cases, managers and employees (as represented by trades unions, for example) can both be somewhat suspicious of socio-technical ideas, with the former applying managerial values, and the latter, humanistic values.

4.4 Lack of agreed success criteria

There has been significant theorising about the way to design socio-technical systems, but recent published examples of successful use in the design of software-intensive systems are comparatively scarce. Consequently, there has generally been little evaluation of the efficacy of using STSD approaches. Indeed, one of Majchrzak and Borys' (2001) major criticisms is that existing socio-technical systems theories are not specific enough to allow for empirical testing. Other reasons for the lack of evaluation include the predominant research emphasis on system design rather than evaluation and, in the UK at least, the difficulties of funding long-term, longitudinal research. Large scale complex IT systems often have a lead time that is measured in years, rather than months—in a hospital for example, it may take several years to introduce a new system throughout the organisation.

Another problem of assessing success is the difficulty in establishing evaluation criteria for the social elements of the system. Whilst benchmark tests can be used to determine whether the technical part of the system meets the appropriate criteria (response time, throughput, cost/benefit analysis), it is more difficult to determine if a system is a better fit to organisational needs, or that a system has increased the quality of working life of the staff. The latter often requires examining or measuring derived effects. So, for example, if a system claims to increase job satisfaction (as a first order effect), this might be measured by looking at the change in levels of absenteeism, improvements in health, and increases in productivity (Land, 2000). This evaluation is made harder by the fact that there are other, quite separate, influences on these factors and in many cases it may be impossible to link them directly to some new system.

Furthermore, the success (or otherwise) of the implementation is defined by a range of stakeholders, particularly operators, middle management and top-level management (Land, 2000). Each category of stakeholder is likely to have a different viewpoint on the system and different criteria for success.

Related to the lack of criteria for success is the absence of work that demonstrates the cost-benefits of STSD methods and tools. Similar problems have also affected other (related) fields such as HCI, and more generally, human factors/ergonomics. New methods may be perceived by managers and systems developers as simply adding extra time, effort and cost to what are already long and expensive development projects. Demonstrating the cost effectiveness of STSD methods should be an important goal, as is the need for them to integrate with existing system development processes.

4.5 Analysis without synthesis

Socio-technical design methods have mostly been used to analyse existing systems, but these methods are limited in the support that they provide for the more constructive synthesis where the results of the analyses are systematically used in the software design process. In other words, they have been used to critique existing systems that (may) have failed, but without always suggesting how the problems could be fixed by appropriate re-engineering of the system (e.g., see Kawka & Kirchsteiger, 1999). There are not many recorded examples of the successful use of these ideas in a prospective manner, particularly for the first instance of a new type of system. This may be due to the envisioned world problem (Woods & Dekker, 2000) which arises because of the difficulty of imagining or predicting the relation between people, technology and context in a domain that does not yet exist.

There are techniques that can be exploited in the construction of new systems ranging from the general notion of learning from past experience, to utilising existing components (appropriately

adapted to the situation at hand). Petroski (1986, 1994, 2006), for example, has documented how engineering has progressed as a discipline over the centuries by learning from its past failures. At a lower level, the work on patterns of co-operative interaction (Martin & Sommerville, 2004), offers a way of supporting the re-use of insights gained from previous fieldwork in new system design.

4.6 Multidisciplinarity

Some of the failings of STSD can be attributed to the multidisciplinary nature of system development. The need for several disciplines to be involved is widely accepted, but the borders between the disciplines have been largely maintained, despite efforts at creating interdisciplinary teams by involving domain specialists in the design process. The issue is mainly down to failures in understanding and communication, where one discipline does not fully understand what the other disciplines can do (Bader & Nyce, 1998), and hence does not ask them to deliver something that assists the system development processes. Dekker et al. (2003), for example, have suggested that practitioners of ethnography and contextual design fail to deliver products that can be used by other disciplines. Their argument is that some of the work carried out by ethnographers and those involved in contextual inquiry does not go far enough, because it essentially stops after collecting data, rather than analysing the data to ascribe meaning to it so that it could be more readily used by others. This was reflected in a report on cooperation between software engineers and sociologists, where it was found that differences in both language and culture were major barriers to multidisciplinary work (Sommerville, et al., 1992).

In general, the maintenance of boundaries between the various disciplines may be a result of the way that systems development has traditionally been perceived and carried out. Specialised individuals or teams were typically allocated responsibility for a particular stage of development, such as requirements analysis or user interface design, and were rarely involved with other developers. Rather than relying on specialised individuals (or teams), what is required is that an individual (or team) has a working knowledge and appreciation of what the other disciplines have to offer, and can communicate effectively with them.

4.7 Perceived anachronism

Changes in ways of working at organisational, national and global levels were at least partly reflected in changes in attitudes towards STSD. In the late 1980s, for example, companies started to move towards lean production methods and business process re-engineering (BPR), often based on the use of new enterprise systems. The philosophy that underpins these methods ostensibly runs counter to many of the humanistic ideas behind STSD (e.g., Niepce & Molleman, 1998), and there were no attempts to try and adapt the STSD methods to the changing business management methods. It is somewhat ironic that it was BPR that made the explicit link to IT innovations, while the socio-technical systems community expended significant energy in the preceding decades on ideological debates (Mathews, 1997) rather than trying to keep pace with technical and organisational developments.

In addition, STSD approaches were largely developed during the 1960s and 1970s, before the advent of the personal computer, and widespread use of interactive computing systems. It was only in the 1980s, however, that HCI achieved widespread recognition as a separate discipline, with its inherent focus on the importance of the interaction between people and technology at the lowest level rather than just the design of the user interface. It explicitly recognised the importance of the roles of the social and technical aspects of work. Many STSD approaches, however, fail to take account of the work in HCI and hence have little to say about interaction design.

The failure to reflect developments in organisational methods and technology can make STSD appear rather anachronistic and unfashionable. This is particularly true when designing new systems that are based on innovative ways of working and novel technology.

4.8 Fieldwork issues

Although STSD methods such as participatory design prescribe the involvement of users, it is comparatively silent on issues such as which users to select, what level of experience in design they need and so on (Damodoran, 1996; Scacchi, 2004). More generally for fieldwork, there are problems with identifying the system stakeholders in the first place, before deciding which groups of stakeholders (and which individuals) should be involved. Traditional approaches involving an embedded ethnographer are expensive and prolonged, although notions such as 'quick and dirty' ethnography address this to some extent (A. Crabtree, 2003)

The key issue, perhaps, is the identification of the focus, extent and level of detail required in the fieldwork. This is not just a problem for STSD. Within HCI, for example, there have often been discussions about the pragmatics of using available methods, which are seen as overly time consuming and unwieldy. Discounted engineering (Nielsen, 1993) and lightweight methods (e.g., Monk, 1998) offer possible solutions.

4.9 Summary

The problems that we have identified all need to be solved if socio-technical approaches are to be accepted and effectively used by the systems engineering community. None of them are insurmountable, although the solution to some of the problems, such as the lack of agreed success criteria (4.4) will only emerge as people apply the framework. We have used the problems to inform the requirements for a discipline of socio-technical systems engineering, which we describe next.

5 Socio-technical systems engineering

In reflecting on the history of socio-technical methods, Mumford (2006) suggested that these methods continue to be relevant, arguing that there is still a role for humanistic, socio-technical ideas in the 21st Century. In addition to the humanistic arguments, we believe there is a strong pragmatic case for applying socio-technical approaches to systems engineering. Simply put, the failure of large complex systems to meet their deadlines, costs, and stakeholder expectations are not, by and large, failures of technology. Rather, these projects fail because they do not recognise the social and organisational complexity of the environment in which the systems are deployed. The consequences of this are unstable requirements, poor systems design and user interfaces that are inefficient and ineffective. All of these generate change during development, which leads to delays in the delivery of the system, and to a delivered system that does not reflect the ways that different stakeholders work.

We have noted that the system stakeholders inevitably have different concerns. The main concern of the system developers is usually whether the system meets the specified requirements. The main concern of the users is usually whether the system will help them do their job, without adversely affecting other parts of their work. The main concern of management is whether the system will generate added value to the organisation in a timely manner and whether it is compliant with regulatory requirements. Reconciling these different concerns is not a simple task.

We argue that these concerns can be addressed, at least in part, by evolving current socio-technical methods into a discipline of socio-technical systems engineering (STSE), in which a socio-technical approach pervades the entire systems engineering life-cycle. Our vision is for a discipline that combines the philosophies of the STSD approaches with the complementary methods identified in section 3. STSE has to be founded on the recognised strengths of socio-technical approaches but must also address the recognised problems in existing approaches (see Section 4). Furthermore, we have to take into account the barriers to introducing any new approach namely:

1. New methods require upfront investment for an unknown later return.
2. There is often a high entry cost in terms of tooling and training to use new methods.

3. The challenge of method usability–experience is required to improve method usability but if initial usability is poor, the methods will not be used.

These constraints mean that, whatever the academic credentials of new techniques and methods, it is hard to get practitioners to adopt them. If STSE is to become a reality, we need to recognise these barriers and develop approaches that minimise the costs of introduction and the associated risks.

In promoting STSE, our intention is to focus on the development of complex IT systems, as well as providing a more effective basis for analysing existing systems. In this way, we hope to overcome the tendency to simply analyse existing systems that has often affected STSD methods (see Section 4.5). Instead, we intend to use the results of the analysis to exploit what we have learned about socio-technical systems (including how they can go wrong, for example) and synthesise the results to help in designing better systems (Coiera, 2007; Walker, et al., 2008).

We consider a complex IT system to be a system that includes one or more networked, software intensive systems that is used to support the work of different types of stakeholder in one or more organisations. In general, we assume that these systems are 'systems of systems' involving databases, middleware and personal applications such as MS Excel. We make no assumptions about the technologies used to develop the system, but note that it is increasingly the case that such systems are constructed by configuring off-the-shelf ERP systems such as those provided by SAP (Pollock & Williams, 2009). Nowadays, new systems are rarely completely new, but instead incorporate and inter-operate with a wide range of existing systems. The costs of integration are likely to exceed the costs of developing the new components of the system (Hopkins & Jenkins, 2008).

We fully realise that in order for STSE to be successful we need to bring about something of a change of mind-set among systems engineers. This is no small task, because engineering per se has developed its own culture over a long period of time, and is often slow to change (Vincenti, 1993). It does, however, have a history of changing as a result of learning from failures (Petroski, 1986, 1994, 2006), so we intend to promote STSE by highlighting the socio-technical nature of system failures, and indicating the lessons that need to be learned. In this way we believe that we can help systems engineers become more aware of the usefulness of the social sciences, and hence make them more amenable to socio-technical ideas.

We strongly believe that if we want to make an impact on practical systems engineering, we have to start with existing systems engineering processes. Socio-technical considerations are not just a factor in the systems development process: social-technical factors have to be considered at all stages of the system life-cycle. While systems engineering processes differ considerably between organisations, we have observed four fundamental activities in all complex organisational IT systems development projects (Figure 1):

1. *Procurement* Decisions are made on what systems to reuse and what new systems to procure from internal or external suppliers. Some analysis will normally precede this, but this is rarely an in-depth analysis of the areas of the organisation where the system will be used.
2. *Analysis* Stakeholders in the system are involved in a process that results in requirements for the new components of the system that is to be introduced.
3. *Construction* The new components of the system are constructed and integrated with existing systems and databases.
4. *Operation* The system is deployed and put into use. Over time, changes to the system are proposed and the development activity continues to create new releases that are deployed and used.

--

Figure 1 Systems engineering activities (about here)

--

In Figure 1, we have deliberately avoided showing these activities as sequential. We believe that they are fundamental to all complex IT systems and that these activities interchange information. The

nature and extent of the information interchange varies considerably. For example, a military system may involve an extended analysis phase which culminates in the publication of a detailed requirements document. This is then input to the construction phase with a tightly controlled change management mechanism for feedback to the analysis phase. In contrast, agile development approaches interleave analysis and construction with informal requirements used to drive the construction of the system.

When new business systems (or systems of systems) are introduced, this is often in conjunction with a change process where there is a goal of (usually) implementing significant changes to the business or its processes. Segarra (1999), for example, highlighted the importance of making sure that IT developments and business change were integrated in the manufacturing of aircraft and cars in Europe. The organisational change process has a structure comparable to the development process, as shown in Figure 2. While this change process should (and to some extent does) take into account social and organisational issues, the changes are often deliberately disruptive because the organisation wants to impose process change. There is likely to be a reluctance to invest in understanding existing processes and their fit with the organisation because these processes are seen as obsolete and due for replacement.

This attitude can lead to serious problems because existing processes have been adapted by the people involved to take particular organisational and workplace concerns into account. A failure to understand the details of actual processes may mean that replacement processes are less suited to the work as it is really done and, hence, are considerably less efficient than current processes.

--

Figure 2 The organisational change process (about here)

--

A major problem in many organisations that we believe is an important contributor to system failure is that there are often only weak connections between change processes and system development processes (although see Segarra, 1999 for one attempt at integrating business processes and IT innovations). There are separate change and systems engineering teams, with the principal communication between them being a requirements document or a set of process workflows. Those involved in the change process may be unaware of technical factors that limit the flexibility of the system that is being developed. Those involved in the development process may have no real understanding of the ways that the proposed workflows will be instantiated in practice, nor of the environment where the system will be deployed.

Proponents of STSD have regularly referred to the process of design as being a socio-technical system itself. However, as noted above, there are actually two distinct processes that often only communicate infrequently. In the worst case, they are linked at the start of the project, when some form of requirements are gathered, and at the end of the project when the system is delivered. In the interim period, both processes are operating simultaneously, usually at different rates, and rarely interchanging information, even though the operation of one often has an impact on the operation of the other. The organisational issues being addressed by the change team are not communicated to the systems engineering team; the technical issues that constrain organisational change are not fed back to the change team.

Our vision of STSE is that it can serve as a means to bridge the system development and change processes as shown in Figure 3. The application of this approach should feed information to the development team about socio-technical issues and provide support for using this information constructively in making design decisions in a timely manner. Similarly, STSE should provide the change team with cost-effective approaches to socio-technical analysis and provide information to them about technical factors that constrain the possibilities of change.

--

Figure 3 Socio-technical systems engineering (about here)

--

To realise our vision we need to improve communications between system stakeholders about socio-technical issues, and provide constructive support for using information about socio-technical factors in both technical systems design and organisational change processes. We therefore envisage two types of STSE activities:

1. *Sensitisation and awareness activities* These are concerned with sensitising stakeholders across the system to the concerns of other stakeholders, and with convincing stakeholders of the value of a socio-technical approach. For example, engineers involved in designing the system database might be made aware of the fact that collecting complete data in some settings may be practically impossible.
2. *Constructive engagement* These activities are concerned with integrating STSD approaches into the practical systems development and change management processes in an organisation. The nature of the constructive engagement varies depending on the development or change activities that are involved.

We discuss below in a little more detail what we mean by sensitisation and constructive engagement. Rather than just noting that we need to take account of the social and technical factors and their interdependencies, we explicitly identify who needs to be made aware of which factors, and provide a focus for the activities that are needed to integrate STSD approaches into the engineering life cycle. We note here, however, that identifying appropriate approaches to sensitisation and constructive engagement and integrating these into development and change processes are the key challenges facing STSE researchers.

As well as bridging the change and system development processes, STSE can inform the change and systems development processes of broader organisational goals and constraints. It therefore acts as an information bridge between the wider organisation and specific projects to develop new complex IT systems.

This notion of STSE as a means of linking and coordinating change processes and systems engineering processes is pragmatic and deliberately limited. Our intention is to provide a framework through which we can use socio-technical approaches in practice and convince practical engineers of their value. While a broader notion encompassing humanistic work practices or organisational re-design could be adopted, we believe that our less ambitious approach has a better chance of adoption. Our approach is less threatening to existing management and can be introduced in an incremental way. If we can succeed in a limited way, we will then be in a better position to extend the scope of STSE.

We cannot and do not claim that this deliberately limited view of socio-technical systems engineering solves all of the problems that we identified in section 4 of this paper. However, by rooting the approach in the language of business, by explicitly linking to the notion of change management and by proposing close interaction between development and change management teams, we believe that we address some of these problems including inconsistent terminology, lack of agreed success criteria (success is related to the success of the change proposals), analysis without synthesis, multidisciplinarity and perceived anachronism. Other work that we are involved with is concerned with using responsibilities as an abstraction to represent work (Lock, et al., 2009; Sommerville, et al., 2009). This focuses on appropriate abstractions for STSE and may be incorporated into the approach described here at some later date.

5.1 Sensitisation and awareness

The primary aim of sensitisation activities is to ensure that system stakeholders, including the development engineers, are made aware of the socio-technical issues that may affect the design and use of the system. In short, they have to be convinced that adopting a socio-technical approach is worthwhile and persuaded to actively participate in the process. Based on our experience, we have noted several types of sensitisation activity:

1. Sensitising system engineers to the notion that socio-technical factors should be considered during system design, and to the cultures of the organisation's different stakeholder groups. In large organisations, different parts of the organisation may have their own cultures and there is a need for better cross-organisational understanding of these.
2. Sensitising those involved in procuring a new, complex IT system to the socio-technical considerations that may influence the design and use of the system.
3. Sensitising system stakeholders to the socio-technical issues that, almost inevitably, are a source of conflict with other stakeholders.
4. Sensitising system stakeholders to the notion that an analyst will be studying their work with a view to a deeper understanding of it, rather than to assess or audit what they do. Here, concerns such as snooping and reporting to management have to be addressed.
5. Sensitising stakeholder groups to the different world views of other groups, perhaps from different disciplines, in the organisation. For example, accountants think about financial transactions in one way and are concerned about ensuring accounting regulations are followed; users of financial data may think about these transactions in a totally different way, reflecting their own management responsibilities.
6. Sensitising management and other system stakeholders to the real technical constraints that limit what is possible with a software system.

The need for sensitisation varies depending on the people in an organisation and the organisation itself. In line with the pragmatic nature of STSE, activities are selectively employed as circumstances dictate. It is clear from our extensive experience in ethnographic studies, however, that sensitisation is essential if the later stages of systems engineering are to succeed. Failure at an early stage will inevitably mean that key system stakeholders will not understand the impact of socio-technical factors on systems and why systems design is not simply a technical process.

A key issue here, of course, is how to we achieve sensitisation in practice. The academic literature is of little help because, naturally, existing socio-technical studies have already crossed this barrier and have convinced companies and other organisations to become involved in these studies. Clearly, practitioners rarely read academic papers and appealing to the canon of work on socio-technical systems is unlikely to be an effective approach. There are three possible approaches that we have previously investigated and that we believe have some potential here:

1. *Taking engineers to the workplace*. The idea of bringing users to the software development team is one that is widely accepted (e.g. in agile methods) but we believe that taking software developers into the workplace, even for a short time, can reveal to them the complexity of work and the difficulties faced by system users. This approach is one that we have found to be successful in a number of different situations (Bentley, et al., 1992b; Lock, et al., 2008).
2. *Workplace vignettes*. Of course, the practicalities of achieving this can be daunting, so we have explored the notion of 'ethnographic vignettes', textual and video descriptions of situated work, that highlight socio-technical issues for engineers and managers. (Clarke, et al., 2003; Martin, et al., 2006).
3. *War stories.* War stories are short illustrative descriptions of problematic situations (Orr, 2005) that have arisen and how these have been addressed. We have catalogued a set of war stories relating to problems that arose in the development and deployment of an electronic patient record system. (Martin, et al., 2004; Mackie, 2006).

We cannot claim that these are complete solutions to the problems of sensitisation and there are real practical difficulties in presenting both vignettes and war stories. However, the availability of social media such as YouTube, may offer some opportunities to make this information widely and easily accessible.

5.2 Constructive engagement

Constructive engagement activities provide a means of integrating STSD approaches into the systems engineering and the organisational change processes, and synchronising the two processes at appropriate points. The precise nature of the constructive engagement will vary from project to project, largely determined by which particular activities in the development and change processes are involved. Here we discuss three types of constructive engagement.

5.2.1 Problem definition

Software design methods are geared towards developing a solution to 'the problem', so if that 'problem' is not understood, applying the methods will generate an inappropriate solution. The nature of the identified problem, though, is rarely simple because each group of stakeholders has its own viewpoint about what it really is. Instead of there being one single problem, there is usually a set of overlapping problems with conflicting characteristics. Indeed, some of these 'problems' may be no such thing – some stakeholders may be perfectly happy with the *status quo* and their 'problem' is that a new system is being imposed on them because of the requirements of other stakeholders.

STSD approaches have recognised that understanding 'the problem' that the system is intended to address is one of the keys to success, which is why many STSD methods are oriented towards analysis and problem understanding. Using an STSD approach will therefore help the stakeholders to focus on the nature of the problems and issues and come to some agreement about what these really are. It will also help systems developers to understand the *real* problems—rather than what they *perceive* as being the 'problem'—their system is supposed to solve.

The alignment of the systems engineering and organisational change processes during problem definition is facilitated by organising, presenting and analysing the process and environmental issues using a coherent framework. The result should be a description of the work context that has been agreed by the stakeholders, accompanied by a set of corresponding requirements based on work performed in that context. These requirements, in principle at least, will define: the purpose of the system within the wider organisational context; the practicalities of its use in its operational environment; and the functionality it provides to system users. Achieving an appropriate balance between these different requirements forms the basis for the construction of a system that will be acceptable to, and used by the end users, as well as delivering the expected benefits to the stakeholders.

In practice, however, expressing what is really required by system stakeholders as a set of requirements means losing some of the richness that is typical of socio-technical analysis. Requirements can state broad functionality, but the way that the functionality is realised and the ways that the system presents information to stakeholders cannot be described using requirements statements. We know that HCI design, for example, depends on prototyping and experimentation; other aspects of STSD such as support for cooperation and collaboration must also be explored and discovered rather than pre-determined.

5.2.2 Constructing the solution

We use the term construction rather than design and implementation because approaches such as agile development and configuration of ERP systems do not distinguish between these activities. The key to success lies in ensuring that the engineers involved in systems construction are aware of socio-technical issues—particularly the interdependence of technical and organisational aspects—and the realities of the environment in which the system will be used. It is also important that there is agreement within the organisation about which methods will be used during development. In this way we can alleviate design and implementation decisions that make it more difficult to incorporate the system into everyday, routine work.

Getting the construction right is not simply a matter of writing *better* system requirements. In the same way that requirements for a user interface cannot adequately express the richness of the interaction with a particular system, social and organisational complexity cannot be simply distilled into 'social' or 'cooperation' requirements. System requirements are still needed to provide engineers

with a broad understanding of what has to be constructed. The agile approach of involving end-users as 'owners' of requirements is a good one but needs to be extended to take into account a broader set of system stakeholders.

An unavoidable constraint on construction is the need to fit with existing procurement and systems engineering processes. For good reasons, organisations are very reluctant to make radical changes to these processes, so STSE has to integrate with them rather than be presented as a new, additional approach. If the procurement process does not consider usability then it should be extended to include it. If it is left to the supplier to decide on the levels of usability, these will be determined by the time and resources available during development, rather than seen as a requirement that has to be met (e.g., see Artman, 2002).

The human-centred design methods that have been developed in the field of HCI provide one way of making sure that technical and social aspects are considered together. The use of prototyping, for example, allows users to think about how they would use the system, and offer feedback on the way that the system will look and feel before the final system is delivered. It also provides a way of synchronously linking the systems development and organisational processes.

5.2.3 Evaluation

The evaluation of a socio-technical system involves assessing the deployed system to understand how well it has met the expectations of its stakeholders. In the ideal world, where perfect knowledge of the future was available, it would be possible to lay out all the criteria for evaluation during the analysis of the system, when the system goals are set. In reality, systems are frequently oversold with inflated expectations of how they will perform in a situation that often is unknown during the construction stage—Woods and Dekker's (2000) envisioned world problem—with the net effect that the final system fails to satisfy those expectations. It is therefore important to recognise that the nature of evaluation changes as the design and the organisation evolve, and that the expectations of the stakeholders will also change accordingly.

Human-centred design approaches advocate evaluation throughout the development process and in the longer term (International Standards Organisation, 2010). Full systematic evaluation of a deployed system is rare, however, partly because organisational issues get marginalised (e.g., Doherty & King, 2001). The original system stakeholders may have moved on, and the new stakeholders may have different expectations, based on their experience of the deployed system. Some stakeholders also take a fatalistic approach: they see themselves as being stuck with the system, so there is no point in complaining about it. Other stakeholders who are in a position to complain, simply refuse to use a system that they do not like, and disassociate themselves from it.

Nevertheless, we argue that there is a place for lightweight evaluation as part of the STSE cycle. This should not be seen as a means of criticising the original stakeholders or requirements, but rather as a constructive activity that leads to a more effective operational system. Essentially, the evaluation should be concerned with 'filling in the gaps' in the analysis of the system which may arise because of incompleteness or incorrectness, or because of subsequent organisational change. In other words, when new requirements arise, or existing requirements change on the organisational side, or when problems arise with satisfying the original requirements on the systems development side, these need to be assessed in their own right, and in terms of the wider development project. This is because they are likely to change the shape of the delivered system, and hence the nature of the evaluation of whether the system meets its goals.

We see the one of the primary roles of evaluation as being its contribution to the process of 'domestication' (Williams & Edge, 1996) where the system gets bedded into the organisation. Domestication is the activity of familiarisation with new software and changing both the software and business processes so that the software becomes an integral part of everyday work. The types of questions asked during evaluation are therefore not 'does this work?' but 'how can we make this work?' This may, of course, lead to change proposals and further iterations of the analysis and construction activities. However, the changes required may be process changes that people carry out to fit the system into their normal work practice.

6 An STSE research agenda

In this paper, we have briefly reviewed several methods for developing socio-technical systems and suggested why these methods have not entered the mainstream of system design practice. Based on this and on our own extensive experience—both authors have over 15 year's experience of working with industry, understanding industrial concerns and transferring research results into practice—we have proposed a pragmatic framework for socio-technical systems engineering. We believe that this framework can be used as a basis for integrating socio-technical analysis and practical, technical systems engineering. We have deliberately designed it as a means of linking organisational change processes and technical systems development and make no claims that our framework provides complete coverage of all socio-technical issues.

The framework is based on almost 20 years of experience of attempting to integrate social and organisational insights from workplace studies into the systems engineering process. The key lesson that we have learned from this work is that there cannot be one simple way to achieve this and that a variety of different techniques, appropriate to the organisations involved should be adopted. We believe that the framework we propose provides a basis for focusing socio-technical analysis around real business concerns and hence increasing the probability of uptake. It establishes a general model that will, inevitably, be instantiated in different ways in different organisations.

The fact that the framework does not exist in isolation from its instantiation and situated use means that an empirical evaluation of the framework is not currently practical. Separating the value of the framework from its instantiation (essential for empirical framework evaluation) is, in our view, impossible. We have qualitatively evaluated our ideas through discussions with industrial collaborators and have received positive feedback from them.

In outlining our framework for STSE we have been particularly influenced by work on ethnographic workplace analysis and on cognitive systems engineering. The STSE framework is also compatible with Resilience Engineering (Hollnagel, et al., 2006). In particular, STSE addresses the way that people use everyday workarounds to keep systems running, and how people often intervene to mitigate the effects of failures that could otherwise have serious adverse consequences. Furthermore, the framework is also consonant with human-centred design approaches (International Standards Organisation, 2010), although our framework makes explicit the relationship between system development and organisational change.

We believe that the different socio-technical design methods have much in common and our notions of the basic activities of STSE allow any method of socio-technical analysis to be used. Methods of analysis, in our view, are not the issue. Rather, research in STSE should address the engineering problems of applying socio-technical approaches in a cost-effective way and integrating STSE with existing systems and software engineering processes.

Research in this area requires an interdisciplinary approach and may involve computer scientists, software engineers, HCI designers, psychologists, sociologists and human factors specialists. We believe that all of these areas still have much to learn from each other. We would advocate the use of techniques such as action learning (Revans, 1982) here, so that people can learn to know what things they do not know about, and to ask people in similar positions questions so that they can explore and overcome their ignorance.

Some of the most important areas are:

1. *STSE processes* Our model of STSE is based around the notions of sensitisation and constructive engagement. The research issues here relate to the specific activities that might be involved in the STSE process to manifest these notions and how these can be integrated with systems engineering process activities.

   How can requirements be made richer to incorporate information about socio-technical processes? In reality, the model of system development where systems are built to a specification of requirements is not going to change for complex systems. Nor, in our view, should it change. However, current requirements documents are usually impoverished

descriptions of how work is done and what is really needed. We need to develop guidance for requirements writers that allows them to express a richer picture of the socio-technical systems to the engineers responsible for systems development.

How do we transfer knowledge and experience from one organisation to another? The issue here is discovering how to separate the essential (what applies to all organisations in a sector) from the accidental (the specific ways in which an organisation works). We will then be in a position to transfer process knowledge across organisations.

What tool support is effective in supporting STSE processes? We need to make use of existing tools—both software engineering tools and Web 2.0 tools—that support collaboration and communication (wikis, social networks, and so on). We need to know more about how to deploy existing tools for distributed project support, how to use these tools to support problem solving, how to integrate technical and social tools, etc.

2. *Modelling and abstraction* Modelling and abstraction is fundamental to software engineering, with models of different types being used by engineers to communicate. The practical use of socio-technical approaches has to acknowledge this by providing a means of modelling, and by integrating with existing approaches. Examples of research issues in this area are:

   What models and abstractions are useful when thinking about systems design and interaction in a distributed multi-organisational system? The abstractions currently used in technical system modelling (e.g. use-cases, objects, etc.) do not seem to us to be sufficient to represent socio-technical considerations.

   Can current approaches to system modelling (e.g. the UML) be adapted to reflect socio-technical considerations? What are the benefits and problems of adopting this approach?

   Can organisations be meaningfully modelled to provide useful information for socio-technical systems design? This is a longer term issue which involves extending the scope of our framework beyond the change process in organisations to consider broader issues of organisational politics and dynamics.

3. *Integrated Human-Centred Design* The importance of effective human-centred design is now generally recognised, if not universally practised (Woods, et al., 2007). However, most methods of socio-technical analysis have paid little attention to those areas of design relating to individuals (Hollnagel, 1998). Furthermore, there is a tendency in the engineering community to identify all human, social and organisational issues as problems of the human interacting with the technology (such as "finger trouble"). In doing so, they ignore the relationship between individual interaction and the social organisation of work, and particularly how the latter can influence the former. Research issues here include:

   How can we integrate methods of socio-technical analysis with methods that support HCI design and evaluation? Many HCI methods have focused on the individual whereas socio-technical methods focus on the organisation and groups within the organisation. We need to develop practical process guidance that allows organisations to use these methods together and to integrate their results.

   How can we use the interface to highlight relevant socio-technical issues, such as awareness of work? The CSCW research community has addressed this issue and there have been a range of proposed techniques to support awareness (e.g., Gross, et al., 2005). Much of this depended on special purpose systems and has been overtaken by the use of web-based systems. This work should be extended and developed to reflect modern interaction and to take organisational rather than situational considerations into account.

   How can evaluation methods be extended to take organisational issues into account? Current approaches to evaluating HCI design are often based around the individual using the proposed interface. However, the organisational setting where work is done has a profound influence on the use of systems, and we need to extend evaluation methods to consider how organisational

considerations affect the use of an interface. This is particularly relevant when things go wrong and the system has to support coping behaviour.

4. *Organisational learning* In many cases, the socio-technical problems that affect a system are not new. They have occurred before but the organisation has no means of learning from these problems or, indeed, from the problems of comparable organisations. We believe that we have to revisit the notion of organisational memory (Walsh & Ungson, 1991) with a view to supporting the organisational learning process and thus reducing the chances of mistakes being repeated. Research issues in this area include:

   How can different types of knowledge be captured at low cost and maintained in an accessible way? The problem of low-cost knowledge capture was, we believe, one reason why many attempts to implement organisational memory systems in the 1990s were ineffective. Capturing knowledge for the future distracts people from their everyday work so we need to discover techniques that capture information from normal work activities with minimal intervention from the people involved in these processes.

   How can the use of organisational memories and other support for organisational learning be embedded in the STSE process? Organisational memories and learning from experience can only be effective if they are actually used. We need to invent ways of easily accessing such information as part of routine processes and ensuring that the information can be updated with accounts of practical usage experience.

   How can we deploy modern tools and technologies (wikis, Google etc.) to develop a workable organisational memory system? People are becoming increasingly familiar with Web 2.0 collaboration tools. Using these as a basis for organisational learning means that initial barriers to tool use are lowered. We are convinced that using these web-based systems is the most effective way to reduce the costs of collecting and using organisational information. To do so, however, we need to investigate how to structure these tools to maintain long-term information about an organisation and its processes.

5. *Global systems* Existing approaches to STSD are virtually all based on an assumption that systems are located within a coherent organisation where the system stakeholders have similar cultural values and assumptions. However, there is now an increasing trend to create global systems, which may involve several disparate organisations that are located around the world. Similarly, the teams involved in complex systems engineering projects are geographically distributed across timezones and cultures. Research issues in this area of global systems and the globalisation of systems engineering include:

   How should socio-technical systems design methods evolve to cover work that is not co-located? The evolution of socio-technical methods to address differences in organisational and social culture that cause problems to be understood and addressed in different ways.

   How can fieldwork techniques evolve to collect information about everyday practice at remote sites? Many STSD methods rely on interaction with end-users either through interviews or direct observation of work. This direct interaction is often impractical when users are distributed across the world. Methods of information collection about work practice have to evolve to cope with this situation.

   How are electronically mediated computer systems integrated with everyday work? Interaction of distributed teams is normally mediated by electronic systems. While there have been many studies of the use of systems such as email (e.g., Bellotti, et al., 2003), we need to understand how teams work around the problems that they encounter when using such systems. We also need to understand how social networks and social media can be used effectively in professional situations to support socio-technical systems engineering.

We are under no illusions about the problems of introducing new methods and approaches or the length of time required to introduce them into an organisation. However, we are convinced that the