Large Scale Requirements Analysis as Heterogeneous Engineering

Mark Bergman[1], John Leslie King[2], Kalle Lyytinen[3]

[1] Department of Information and Computer Science, University of California, Irvine, mbergman@ics.uci.edu

[2] School of Information, University of Michigan, king@si.umich.edu

[3] Department of Information Systems, Case Western Reserve University, kalle@po.cwru.edu

Abstract

We examine how to improve our understanding in stating and managing successfully requirements for large systems, because the current concept of a system requirement is ill suited to develop true requirements for such systems. It regards requirements as goals to be discovered and solutions as separate technical elements. In consequence, current Requirements Engineering (RE) theory separates these issues and reduces RE to an activity where a technical solution is documented for a given set of goals (problems). In contrast, we advocate a view where a requirement specifies a set of mappings between problem and solution spaces, which both are socially constructed and negotiated. Requirements are emergent and need to be discovered through a contracted process, which likens to a "garbage-can" decision-making. System requirements thereby embrace an emergent functional ecology of requirements. This leads to equate requirements engineering with heterogeneous engineering. The admitted heterogeneity of technological activity avoids a commitment to social (or technological) reductionism. Requirements engineers need to be seen as "heterogeneous engineers" who must associate entities that range from people, through skills, to artifacts and natural phenomena. They are successful only, if built socio-technical networks can remain stable in spite of attempts of other entities to dissociate them.

Keywords

system requirements, functional requirements, system failures, heterogeneous engineering

1. Introduction

Information systems development (ISD) has remained a high-risk proposition despite huge advances in computing and telecommunications technologies. Information systems projects in general, and large information systems projects in particular continue to fail at an unacceptable rate (Abdel-Hamid et al. 1990; Drummond 1996; Mitev 1996; Myers 1994; Rosenwein 1997). While some portion of troubled ISD projects is turned around successfully, intensive research in the past has generated too little understanding in how to avoid failures in large systems development initiatives. From the growing incidence of failed projects, we conclude that advances in technologies are not sufficient to save large system projects. Instead, large projects remain susceptible to failure until we learn to understand how technological, organizational and institutional changes are interwoven in large systems and how system developers should accordingly state and manage requirements for such systems.

Consider the following example. On March 11, 1993 the world was shocked by the sudden cancellation of the Taurus project, which the London Stock Exchange had been developing for more than six years. Taurus was expected to be instrumental in the radical restructuring the securities trade, widely known as the Big Bang, by forming a backbone system for the London Stock Exchange. The project cost the Stock Exchange $130 million, and securities companies invested $600 million more (Drummond 1996). After years of alternating embarrassments and heroic efforts, Taurus was cancelled before a single module was implemented because the required functionality and performance could never be delivered.

Although Taurus was a very complex project, involving novel technologies and massive organizational and institutional scale, ineffective project controls allowed requirements to change continuously throughout the project. Moreover, management ignored clear warning signs about organizational and technical risks, whilst powerful interests pushed for Taurus' development despite confusion over the system's purpose and design. Simply, there was no understanding what the systems was supposed to do and what stakeholders it should serve. In the end, advocates held an almost superstitious faith in the project, dismissing objections and proposals for modifications and clearer statement of the requirements with comments like "...we have had all those arguments. Great idea but no, we have been arguing about it for twelve years, forget it" (Drummond 1996) (p. 352). With the benefit of hindsight, the Taurus failure could have been averted by adjusting its course based on a more delicate and politically sensitive requirements engineering. But this was not done despite a well known truism shared both in academia and industry that systematic requirements engineering is a keystone to a successful delivery of a large scale system. The failure of Taurus can be partly attributed to the dismissal of this well known fact, but we think there is more to learn. Taurus failure was also due to the fact that we poor knowledge about how to state and manage requirements for large systems that involve political and institutional elements.

Stating requirements for such systems is not just a technical exercise, but necessitates a new mind set which we call "heterogeneous engineering" after Hughes (Hughes 1979a; Hughes 1979b; Hughes 1987). Heterogeneous engineering sees all requirements specifications to be inherently heterogeneous due to the need to establish stable networks involving both social and technical elements through engineering (if the network is not stable the system fails!). As Law (Law 1987) (p. 112) puts this: "The argument is that those who build artefacts do not concern themselves with artefacts alone, but must also consider the way in which the artefacts relate to social, economic, political, and scientific factors. All of these factors are interrelated, and all are potentially malleable." Consequently, requirements engineers need to be seen as "heterogeneous engineers" who must successfully associate entities that range from people, through skills, to artefacts and natural phenomena.

In this paper we will examine the problem of stating and managing requirements for large system development initiatives qua "heterogeneous engineering." Our argument is twofold. First we will argue that failures like the Taurus disaster do not happen only because existing approaches to requirements engineering have not been adopted. In contrast, we argue that current requirements engineering techniques used alone will not do the job. This is because they are based on a fallacious assumption that business problems and political problems can be separated from technical requirements engineering concerns of how to specify a consistent and complete technical solution to a business problem. In contrast, large scale system development initiatives involve a simultaneous consideration of business, institutional, political, technical and behavioral issues. Second, based on behavioral theories of decision-making we argue, that solutions and problems are intertwined and addressed simultaneously during a requirements engineering process. Thus, requirements engineering can be understood only by using theories of behavioral and institutional decision making along with applied technical understandings, but not only through the lens of rational technical "engineering."

This paper is organized as follows. In section 2, we examine the received "view" of requirements engineering as outlined by the proponents of the current requirements engineering literature. Our conclusion is that though forming a necessary step to understand the difficulties involved in requirements engineering for technical change, the current view is insufficient alone to understand how technical requirements relate to the organizational and institutional "problems." Therefore, in section 3, we propose an alternative concept of requirements engineering which we call the functional ecology of requirements. In this view, requirements are not discovered but constructed as mappings between solution and problem spaces. The construction process involves a protracted "walk" between these spheres. Section 4 examines the implications of the functional ecology model. Section 5 concludes the paper by drawing some consequences for requirements engineering research.

2. Requirements Engineering Defined

The concept of a system requirement is relatively well known in the system and software engineering literature since mid 70's. The concept was originally conceived to involve the stating what the system is supposed to do before stating how the system produces the desired functionality (Ross 1977). The earliest concepts of system requirements can be traced back to work of Langefors (Langefors 1966) and some early attempts to develop high level system description languages[1]. One reason for separating the how and the what can be attributed to the desire to achieve what we call a "responsibility push-back". By this we mean the desire to relegate the failure to develop or implement the system to the prior environment, which gave rise to the definition of the system development task. Such attempts to move the "reasons" for failure to higher level system environments has been a continuing trend in software engineering and system development since the mid 70's. This has gradually shifted the interest of the software engineering and system development communities from implementation considerations (like "structured programming", "structured design") into problems of how to define what the system is expected to do and what this involves. This is currently called fashionably "requirements engineering" (RE) (Kotonya et al. 1998).

The main concept in the requirements engineering is the desire to repeatably create successful systems. The main interest in the requirements engineering literature is to explore the means to express and articulate the desire to develop the system, i.e. how to define features of the new systems, or how to change current systems that will solve an identified business need, want, or desire (Loucopoulos et al. 1995; Pohl 1996; Sommerville et al. 1997). Therefore, the requirements engineering literature has concentrated on developing tools and methods which answer questions like: Who's desire? How to express the desire? Who and what defines success and failure criteria for addressing the desire? For example, when doing user-centered design, the end-users of the new or changed system are expected to define success & failure criteria (Noyes et al. 1999; Pohl 1996). At the same time, knowledge of the current state of the art of system design can influence the choice of success & failure criteria. These can be seen as system design constraints and opportunities, which can also affect, i.e. change, the identified business wants, needs, and desires. In general, requirements in the received literature are seen to establish these success & failure criteria. The "received" definition is the IEEE standard 610.12 (Loucopoulos et al. 1995; Pohl 1996), which defines requirement as:

1. A condition or capability needed by a user to solve a problem or achieve an objective.
2. A condition or capability that must be met or possessed by a system or a system component to satisfy a contract, standard, specification, or other formally imposed document.
3. A documented representation of a condition or capability as in 1 or 2.

Accordingly, requirements engineering denotes a set of methodologies and procedures used to gather, analyze, and produce a requirements specification for a proposed new system or a change in an existing system.

3. The Functional Ecology of Requirements: The need for a conceptual model

In this section we will create a systematic conceptual model of RE from an emergent functional perspective. The term functional in the term suggests that any RE analysis is done in pursuit of practical objectives for a given task domain, such as to make task accomplishment more efficient and/or effective. We use the term emergent to capture the evolutionary view of how organizational goals, problems and solutions are constructed during the RE, as opposed to discovered, in alignment with a behavioral view of human decision making.

We will develop the model through a set of conceptual clarifications and definitions, which define exactly[2] the content of the major components of a requirements specification situation. These include the concepts of problem, solution, requirement, principal and goals. These are derived (though not explicitly) from a careful reading and analysis of the literature in institutional decision-making in complex domains. As with any conceptual model, our main goal is to define the major analytic relationships between these concepts. Constructing this conceptual model allows us to define more exactly what functional emergence means and why such emergence is inevitable, thus making large scale RE so hard to successfully do. We assert the model explains origins and sources of RE complexity. In turn, analysis of the model seeks to offer some means to understand the challenge we are facing on both conceptual and practical levels for constructing and stating adequate requirements. As we will show, this model enables us to pinpoint more exactly our major disagreements with the received IEEE definition. By developing rigorously such a vocabulary[3] and underlying model for discussing large scale RE in all its complexity, the conceptual model enables us later on to formulate research questions more systematically and to develop techniques that can help manage such processes.

Though, the suggested model is still in line with a dominating model of RE in which it is assumed that organizational goals are clear[4], it digresses from it in how organizations and actors approach these goals and what mechanisms they have at hand for accomplishing those objectives. A prevailing bias in the requirements engineering literature is the notion that requirements exist "out there" waiting to be captured by the systems analyst and refined then into a complete and consistent specification for the system that will be thereafter created (Davis 1993; Kotonya et al. 1998; Loucopoulos et al. 1995; Macaulay 1996; Pohl 1996). Consequently, the main focus has been on formalizing the content of the system that will deliver the solutions and how this meets the objectives of being complete and consistent. Continued disappointing experiences in large-scale system development suggest, however, that the challenge is a good deal more complicated. One point we want to make is that the content of RE may mean different things for different people, and it is dynamical due to ambiguity and uncertainty related to the goals of the stakeholders and the solutions, which can be brought to bear upon identified problems.

3.1. Requirements Analysis Framework

The crux of the ecological view is to adopt insights from the study of human decision processes and use this to inform our formulation of the RE framework. We draw on two major sources in this endeavor that digress considerably from the dominating "technical rationality" of RE. First, in any complex development initiative, including RE, we must take seriously Simon's theory of bounded rationality (Simon 1979; Simon 1982) in that we can never find an optimal, but at most a satisficing solution. Accordingly, RE processes should be analyzed and understood from the view point of heuristics, limited search spaces and the quest for increased intelligence during the RE process. This is what RE methods and tools seek to offer. But their problem is that they scale up poorly for large systems, and in addition they fail to recognize the type complexity inherent in large scale RE.

Second, we will draw upon the institutional and behavioral theories of decision making which have in the past decades studied complex organizational decision making processes involving complex social, business or political change (Lindblom 1979; March et al. 1976). These studies have shown that complex organizational decisions are not discrete events (i.e. bets) in which pros and cons are weighed and optimal decisions are rendered. Instead, organizational decision-making forms a protracted processes of iteration in which problems search for solutions, solutions search for problems, and decision makers search for decisions to be made (March et al. 1976). In the organizational theory, this is coined the "Garbage-Can Model." It is so named because of the relatively permanent nature of the cans in which different solutions, problems and decisions are "thrown" within an organizational arena. Available experience from many large scale RE (e.g., software engineering) initiatives coincide with this view. The development of World Wide Military Command and Control System (WWMCCS) of the 1970's and early 1980's formed a continued process of redefining this "can" over two decades. Hence, contrary to common assumptions underlying RE, RE decisions are implicated by solutions searching for problems rather than the other way around. The behavioral study of decision-making has thus benefited from the transformation of the "problem → solution" construction to a new and more evolutionary view of iteration represented as "solution → problem → solution."

We will next refine this model in relation to RE, and will therefore begin with an examination of what a "solution space" means in relation to requirements, followed by an examination of the mediating "problem space." This leads to articulation of the requirements analysis process as an iterative "walk" between the solution and problem spaces. The main components of the framework are depicted in Figure 1. The acronyms M and N in the figure describe how different components in the RE environment can be related to one another during a RE process (i.e. many to many).

3.2. Solution Space

The ecological view suggests that any RE process starts from an existing solution space, S_t, that will be affected by a proposed new or changed system (see Figure 1). We depict the continuous construction and movement of solutions by rotating arrows around the solution space. The "existing solution" space, that we call the Current Solution Space, is denoted as S_t. Fundamentally, this space embodies a history of solved social, technical and procedural problems and it constitutes the legacy (or competency) of previously solved organizational problems. This definition denies that solutions exist a-historically. Instead, they are socially constructed and legitimized. Capabilities to produce such solutions must be acquired and maintained in the surrounding socio-technical system. Therefore, the solution space is intimately related to the principals, i.e. a set of actors who have the capability to represent themselves as capable of arriving at solutions to an identified problem, or who possess specific skills that can result in specific solutions. The solutions are socially constructed also in the sense that the principal's must find solutions to fit to their problems and thereby accept the legitimacy of a specific solution to their specific problem. Principals have also incentives to create their own solutions (e.g., goals) so they can influence the social system in which they reside and obtain resources. Accordingly, many times solutions search for problems and not the other way round.

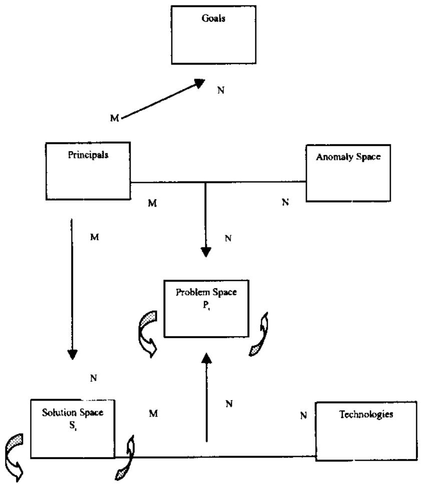

Figure 1: Requirements Analysis Framework

Working solutions form instantiations of one or more principals' successful attempts to adapt generic as well as custom technologies to suit to specific business or social problems. Hence, solutions embody new and novel ways of carrying out organizational tasks often with untried configurations of social arrangements and technical artefacts. Our concept of technology is thus a heterogeneous one in the sense that it covers both social and managerial innovations, and technical innovations that draw upon properties and laws of the physical world and which demand that the final solution is a socio-technical ensemble (Law 1987).

In general there is a M:N (e.g. many to many) relationship between technologies and solutions. Hence, any given technology can be used to solve many types of problems and the same type of technology can be used to solve many problems. Moreover, any given problem can be solved or approached (in the spirit of socio-technical design) by the application of many types of technologies. This heterogeneity provides also a basis for the garbage-can model of organizational decision making: organizations can and often will throw several types of technologies into the "can" in their attempts to solve any given problem.

Organizations change over time, as do their solution spaces. A Local Solution Space, forms the current solution space and all locally accessible solution spaces that can be reached from the current solution space using available skills and resources offered by the principals[5]. A local solution space thus is a subset of a Global Solution Space, denoted GS that can be seen to be the union of all solutions, which can in principle be reached from the current solution space if all resources and skills were available. In other words, the global solution space is the space of all feasible solution spaces, including those not currently accessible from the local solution space and which require mobilization of all principals and technologies. Reasons for not being able to reach all of them can be due to lack of resources, lack of intelligence (i.e. this solution is not known or cannot be connected effectively to any known problem), cognitive bias, shifting goals or incompatibility with organizational goal(s) or political structure.

In general, a local solution space represents the range of all locally accessible solution spaces with regard to organizational resource limitations, but without regard to any particular proposed new product. A local solution space is a more general form of a product space (Davis 1993), but it contains the essential attributes and context of the product space.

3.3. Anomaly and Problem Spaces

The source of a problem is an anomaly, i.e. a known existing inconsistency between the current solution space and a desired solution space.[6] The set of all such inconsistencies we call an existing Anomaly Space. An anomaly is only a "potential" problem, because not all anomalies are attended by organizations as problems that need to be solved due to resource constraints and cognitive bias.

An anomaly becomes a problem only when it is observed and acted upon by a principal with a standing to act. Standing refers here to the power to define and legitimize an anomaly as a problem to be solved by collective action, and the demonstrated capability to mobilize means to address a defined problem[7]. This is normally defined in management statements that justify IT projects, in project goal specifications, or investment memos. A principal is thus assumed to wield organizational power, i.e. to have access to means by which she or he can influence the others and mobilize sufficient resources (Bacharach et al. 1980; Fairholm 1993; Pfeffer 1981). It is important to note that in large scale system development initiatives there are several or large numbers of principals who can obtain a standing in relation to problems identified. Moreover, it is important to understand that in large scale system development initiatives it is necessary to enroll a large number of principals to take a standing and agree on some level of problematization (Baier et al. 1986). Standing can be later on also held by groups as well as individuals at different stages of RE process, which relate to possible rewards, incentives or side-effects of the possible system solution. Lower-level participants in organizations hold such standings due to their knowledge, or access to unique local resources that are critical in proceeding in the project[8]. An example is a system analysts. Standing can and also often needs to be changed, and therefore it can easily drift during a large RE process (see e.g. Sauer 1994). In the RE literature, principals are called business stakeholders (Kotonya et al. 1998; Wieringa 1996).

Due to cognitive limitations, some anomalies are not recognized by actors with a standing, and thus are not acted upon. Similarly, anomalies can be observed by principals as problems, but they choose not to act upon them due to their resource constraints, or difficulty in defining a solution space which links with the problem (e.g., goal failure). Such processes of inattention relate normally to high political, functional, technical or implementation risks of moving to a new chosen solution space (Lyytinen et al. 1998b). Anomalies can also turn into problems at later stages of RE, or further down in the design process due to learning by doing. In the same vein, principals can later drop problems out of consideration and revert them to mere anomalies, or even beyond that if they change their goal sets, or observe high obstacles to move from the current solution space to the target space[9]. Thus the set of principals is not fixed but contextually emerging and negotiated.

Although the underlying causes of anomalies can spring from many sources, the conversion of an anomaly to a problem is a social process we call problematization. Problematization begins long before a recognizable problem space has emerged in RE. It begins with a principal's decisions standing to act or not act upon anomalies that turn them into problems. Often, these problematizations can start with the metaproblems of finding out what the problem is to which an existing or an emerging solution can be applied. During this activity, principals determine and apply legitimizing reasons to change an anomaly into a problem. Legitimate reasons can relate to the goals (see Figure 1), i.e. desirable properties of those solution spaces that can be reached from the current solution space. These goals are therefore not given or fixed, but instead are constructed and negotiated as a result of legitimizing the problematization. This process is by no means trivial exercise as any principal normally pursues several goals at the same time, and the same goal can be chosen by several principals. This results often in situations where the same problem space can relate to many different sources. Moreover, different principals can select them independently. In the same vein, these problems can be later on mapped to alternative new solution spaces, which means that several often contradictory, or supplementary change processes may be initiated to the same problem causes.

An important capability of a principal with standing is the power to define particular characteristics of the desired problem space.[10] These relate to general value statements and rationales underlying organizational action like increased control, competitive capability, shareholder value, or employee participation. Such features can be used to dictate who has a right to address the problem space, why this is regarded as the problem space, among several competing principals who are jockeying to a mandate to address the problem space. Moreover, as Fairholm suggests, such power entails 'the ability to gain aims in *interrelationship with others, even in the face of their opposition*'[11]. Altogether, it is the principals who define the problems and their sources, and by implication, their resulting solution spaces. Thus, they must be considered the most important RE stakeholders.

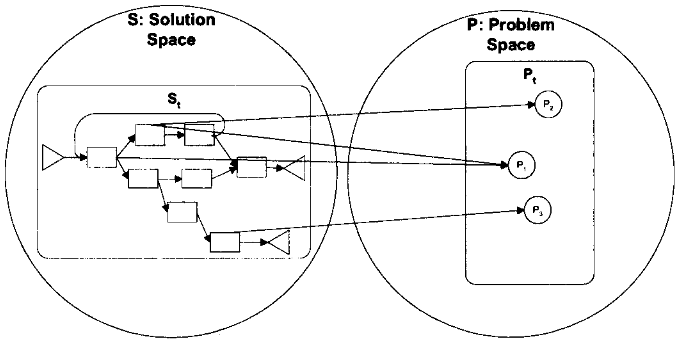

Figure 2: Solution and Problem Spaces

The space of all problems implied by a current solution space S_t is called the Problem Space, denoted here as P. A problem space (e.g., the space of all selected problems) is by definition always a subset of an anomaly space. Hence, a proposed system problem space, denoted by P_t, contains all of the recognized and chosen problems by all of the principals at time t[12]. This does not mean that elements of this set are consistent, non-contradictory or selected by following some overarching organizational "goal" set. What we say instead is that problems in P_t have to be contextualized into S_t[13] by some principals so that there is an observed need to change the current solution space. Accordingly, they can be later on associated with a proposed new system or system change by some principal with a standing.

Figure 2 shows a general relationship between S_t and P_t where the arcs represent connections to problems in P_t from their contextualizing source in S_t.

It is important to understand that multiple solution sources, as shown in Figure 2, can point to any one problem, and any one solution source can lead to multiple problems. This corresponds to the M:N relationship between the solution space and problem space as depicted in Figure 1. What this implies is that it is possible for a single problem to have multiple contextualizing sources. Also, a single solution source can contextualize multiple problems.

The process of problematization uncovers frequently also other anomalies that are deemed problems by principals. This can trigger an iterative reconsideration of the current solution space and its anomalies resulting in a process called problem blossoming[14]. This iterative process can change the contents, and hence, the structure, of the current solution space (S_t) as well as the problem space (P_t). This process may have to be iterated as long as new affected areas of S_t are being discovered and the corresponding anomalies, and resulting problems, are constructed and organized into the current problem space. Once complete, or prematurely stopped by a principal with standing due to the fear of endless search, the resulting problem set is called P_t.[15]

3.4. Proposed Solution

A Proposed Solution, denoted as S_{t+1}, forms a new subspace of the solution space. A proposed solution by definition implies the reconciliation of S_t to P_t. In other words, each part of a proposed solution must be reconciled with one or more problems in P_t until all of the problems in P_t are addressed. The process of reconciliation, changing S_t into S_{t+1}[16] by solving for P_t, is called Solution Space Transformation. Finding this mapping forms the heart of RE. It involves specifying a mapping from a current solution space into a future solution space that is contextualized, or warranted, by the chosen set of problems. In other words, the analyst's job is at the intersection of the two solution spaces (along with technologies embedded in them) and the problem space. During this reconciliation process, constraints are seen as limitations of current organizational resources as well as limitations concerning the future IS, including people, artefacts, rules, processes and the like.

It is a custom desire in the RE literature to find an optimum path from S_t to S_{t+1}. This is, however, seldom the case in any given requirements analysis effort, because 1) the prospect of attaining global solutions is quite remote due to changing and shifting needs and goals of the principals, problem blossoming etc, and 2) because system analysts cannot locally foresee the impact of the chosen solution spaces or the difficulty of getting their due to their cognitive and resource limits. The task of the analyst is, instead, to find a traversable path from a current solution space to a new one that meets sufficiently the requirement of removing observed problems (Haumer et al. 1999). This needs to be accomplished also by identifying problems that will arise during the process of transformation.

A necessary outcome of the solution space transformation is to transform, and possibly expand, the local solution space, S. Transforming S means not only a changing, and hence a likely expanding, of some principals technical capability. It also means a changing, and presumptively expansion, of the organizational capability in the solution space. Hence, an expansion of S can reveal previously unavailable, but now realizable opportunities. The process can even expand a general solution space, and thus demonstrate organizational learning and innovation in the sense that new solution "frames" have been created (Lyytinen et al. 1998a) [17].

3.5. Redefining Requirements

Per our analysis, RE activity involves always a deliberate construction of an ecology that consists of two solution spaces and a problem space[18]. The objective of RE is to reconcile all essential aspects of the current solution space with regard to a problem space thus producing a specification for a particular solution space that can be achieved at some future time point t+x[19]. It is expected that this will mitigate or eliminate the identified problem space (though naturally this cannot be guaranteed). Due to the discovery of goals, problem blossoming and dynamics of the solution spaces, this is an iterative process: new information on both the solution space and the problem space is continually discovered, and consequently decisions need to be continually made to re-state both the solution space and the problem space in the direction of reconciliation. The RE specification is thus an outcome of a co-evolutionary process of discovery and decision, in which both the solution space and the problems space are iteratively constructed. This process is influenced by many constraints arising from the environment itself (e.g., physical laws, technical choices, legal considerations, institutional influences, organizational goals and capabilities, market forces). But, at the bottom, it remains a social process of negotiation and inquiry that is constrained by bounded rationality and limited organizational resources.

At this point of our treatise we can contrast this definition of RE with the "received" definition that is common to the requirements engineering literature. As previously stated a requirement as per this literature is:

4. A condition or capability needed by a user to solve a problem or achieve an objective.
5. A condition or capability that must be met or possessed by a system or a system component to satisfy a contract, standard, specification, or other formally imposed document.
6. A documented representation of a condition or capability as in 1 or 2.

In our terminology, item 1 focuses on meeting some expressed desire of a principal with standing, usually the client or the system's intended user. Such requirements have been called "functional" requirements (Loucopoulos et al. 1995; Pohl 1996). Item 2 departs from the issue of desire and addresses compliance with conditions set by social or technical environment. Such requirements have been referred to as "nonfunctional" requirements (Loucopoulos et al. 1995; Pohl 1996). Item 3 expects that a requirement needs to be represented in a document. In other words, if a requirement isn't written up, or equivalent, it is not a requirement.

A good summary of the requirements definition, accordingly, would be: a requirement specifies a written want, need or desire that solves a problem in the context of a set of constraints or a written constraint imposed by a formal document.[20]

We depart from this nomenclature in two ways. First, we see requirements not as solutions to problems, but as a set of relationships between solution spaces and a problem space. Solutions to problems are determined in design, not during requirements. As such, requirements are no more fixed than the evolving understanding of the characteristics of the two solution spaces and the problem space. Requirements in this sense cannot be discovered, but rather must be constructed by a search in the search space that covers all known mappings between the problem space and the two solution spaces[21]. Hence, requirements need to be represented by two sets of arcs, as shown in Figure 3, between the problem and two solution spaces. An analysis of the resulting requirements arcs postulates a conceivable alternative solution space for the identified problem set. More simply, once a stable set of requirements are known, one can postulate the specification for a new or changed technology, e.g. system. Still, as discussed earlier, we consider such a technology heterogeneous in construction, and thus includes technical, social, political, economic, and alike functionalities and constraints (Hughes 1987; Law 1987). We called this whole set of possible future solution spaces a conceivable solution space (S_V). S_V, by definition, contains the proposed solution, S_{t+1}. Any given element in the conceivable solution space can, consequently, become the focus of the system development or enhancement project, in future. Another way of reading our definition is that requirements link desired solution spaces to their original contexts and to problems embedded therein that truly need to be addressed by a future solution. Taken together, requirements become a contextualized set of rules and constraints through which a future solution can be constructed via design.

Second, we can apply insights from our definition of requirements to redefine some identified requirement types. We refer to what the literature calls a functional requirement as an objective requirement, denoted by R_O. An objective requirement is defined as: a want, need or desire that corresponds to a problem (e.g. P_t) as contextualized by a part or all of a current solution (e.g., S_t). A R_O is represented by an arc from a solution to a problem, as shown in Figure 3[22]. In this way, R_O can viewed as a relationship that comes from a solution looking for a problem. We refer to what the literature calls a nonfunctional requirement as a constraint, denoted by R_C. A constraining requirement, e.g., a constraint, is defined as: a restraining condition imposed upon a solution within S_{t+1} as contextualized by a problem within P_t[23]. A R_C is represented by an arc from a problem to a new solution, as shown in Figure 3[24]. The third part of the IEEE requirement definition remains unchanged, i.e. requirements must still be documented in some way for the principal(s).

Taken together, R_O and R_C fill in all arrows in the "solution → problem → solution" framework. Hence, it represents a model of the requirements as a functional ecology.

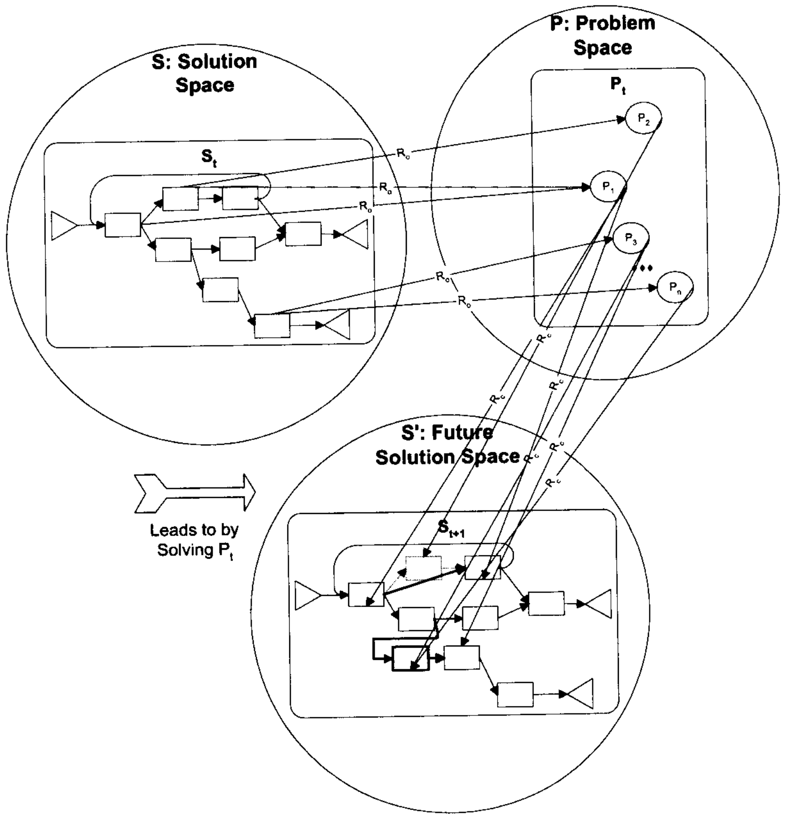

Figure 3: Solution Space Transformation – The Functional Ecology of Requirements Conceptual Model

4. Implications

An implication of the new requirements definition is that since solutions and problems are heterogeneous in nature, requirements are therefore heterogeneous in nature as well and RE should be conceived as heterogeneous engineering (Hughes 1987; Law 1987). This also means requirements of one type can be related to, and hence, can affect requirements of other types.

Another implication of the new requirements definition is, by transitivity, the principal(s) who own the problem(s) that are used to define a R_O or R_C are also the owners of that R_O or R_C. This point is needed to repair an ownership gap that appears between problems and requirements. Together, all of the arcs R_O and R_C form the essence of a requirements specification. A requirements specification is the set of statements selected for attention by the principal requirements owners. By induction, requirements owners are also the primary stakeholders of the system design. Also, by induction, since requirements are heterogeneous, the requirements specification is a heterogeneous, yet holistic, view of the future solution under consideration.

The activity of requirements analysis constructs a set of dynamic relationships between specific solution spaces and a specific problem space such that the specific principal's objectives are realized in the context of constraints. This definition is simple and clear, but its simplicity can be misleading. The challenge for researchers and practitioners alike is in the adjective specific that appears before the words "solution", "problem" and "principal". We argue that most requirements analyses fail first as a result of insufficient or incorrect specificity (i.e. level of detail and certainty) concerning the links between the solution and problem spaces. In fact, in many cases it can be extremely difficult to achieve such specificity due to limitations of bounded rationality, changing environmental conditions, limited resources, problem blossoming or political shifts among the stakeholders. Second, we argue that in large scale system development initiatives the requirements fail because it is difficult to stabilize the set of "specific" principals that take a common standing in relation to problems and later on to solutions due to political difficulties (see [Bergman, 2001 #123]), or cognitive problems.

As noted earlier, multiple solution parts can point to one problem and any one solution part can lead to multiple problems. Hence, as noted, there is a M: N relationship between S_t and P_t. This suggests that there can be a nonlinear number of R_o arcs between the nodes of the two spaces. Similarly, a problem can affect multiple parts of a future solution, while multiple problems can point to (i.e. are addressed by) the same part of a future solution. Hence, similar to R_o, there is a many to many relationship between P_t and S_{t+1}. This allows for a nonlinear number of R_c arcs in relation to their sources. Altogether, requirements at any given time can be represented as the set of all of the arcs, (R_o, R_c), that reflect the contextualized connections between the problem space and the current and future solution space.

The obtained final set of requirement arcs between S_t, P_t and S_{t+1} can be seen to form a network of interrelationships, e.g., a requirements web. Thus, any change to even a single part of S_t can affect a large number of problems in P_t. The change is nonlinear in size. Also, due to the reflexivity of the connections via problem blossoming from P_t to S_t, any such change can show up in other parts S_t itself. Accordingly, any change in P_t can result in a nonlinear set of new or changed connections to S_t and S_{t+1}. Therefore, a small change in S_t or P_t could result in a nonlinear, i.e. potentially explosive, change in the whole requirements set (R_o, R_c).

In addition, a change in one requirement could have a non-linear change effect on other requirements. This has been observed as the "cascade effect" of requirements change, and forms the source of a phenomenon what has been called "requirements shift" and "requirements creep" (Gause et al. 1989; Pohl 1996). This is a cascade effect because any requirement statement i.e. a pair of $<r_o^i,r_c^i>$ can influence or be dependent on a set of other requirements $<R_o^i,R_c^i>$ (Robinson et al. 1999) and the impact may cascade recursively. Thus, in large system designs, a non-linear number of requirements connections coupled with non-linear change effects can yield a very complex, non-linear delicate requirements web.

These problems can, often in turn, lead to "requirements paralysis," i.e. the rational fear of changing requirements due to the possible devastating impacts of the change[25]. This is well supported by the observed difficulties in the Taurus project to manage shifting requirements (Drummond 1996). The combination of requirements paralysis, bounded rationality and organizational resource limitations can create the impression that any serious attempt to change requirements, or to change their sourcing problems or solutions would result in incurring spiralling costs that are simply too high for the principals to pay. Many times this can lead into a path of escalation, e.g., increasing commitment to a failing course of action, when the project under consideration starts to falter (Keil et al. 2000).

Another implication of recursive, reflexive, non-linear requirements change is the following: determining requirements also determines stakeholders. Thus, shifting requirements shift our perceptions of who are the principals and who are the other stakeholders. These shifts can even change who can be a stakeholder. This means that the original principals of a project may not be the final principals of the project. The implications of this can have profound affects on the process of requirements determination in ambiguous and dynamic environments. Again, the demise of Taurus project associated with the incapability to manage the stakeholder set is a case in point.

5. Conclusions

The functional ecology of requirements i.e. the intricate linking and evolution of solution and problem spaces vis-à-vis a set of principals suggests that requirements are not solutions to problems. Rather, they establish links between solution spaces and problem spaces in a changing environment of environmental conditions, shifting principals, and the evolution of organizational knowledge defining what is a solution, and what is a problem. Accordingly, requirements emerge through the identification of anomalies in the existing solution space that are declared by a principal with standing to be problems, followed by an effort to identify new solutions that would reconcile those problems. Any effort to identify new links between solution and problem spaces can uncover information that alters perceptions of both the problem space and the solution space and the set of principals. Under this model, requirements are not waiting to be discovered by an analyst, but rather they are systematically constructed through an iterative and evolutionary process of defining and redefining the problem and solution spaces (see also (Iivari 1990a)). This construction process is by definition imperfect and often inefficient due to bounded rationality, organizational resource limitations, uncertainty due to changing environmental conditions, and changing views of principals with standing. Moreover, changes in the problem space can affect the solution space in nonlinear ways: with modest changes in one having the possibility of creating large changes in the other. The same holds in reverse. Thus, requirements analysis can be exceedingly difficult due to this instability, even in cases where organizational objectives are relatively clear.

We believe that an articulation of the functional ecology of requirements captures one key reason for the persistent failures in system development, when the target systems operates in complex domains. The crux of the ecology is the recursive and reflexive relationship between solution spaces and problem space, and the fact that each can influence the other in nonlinear ways. This creates a situation in which "real" problems and their corresponding "real" solutions can be impossible to pin down with confidence. In such situations, the temptation is strong to abandon the analysis and shift to old, (at one time) reliable heuristics and learning by doing to implement one feasible solution without ever really pinning down its requirements and hoping that it will fit into the current ecology. Such attempts correspond to a random walk from the current solution space to an unknown new solution space. This signals a failure to state the requirements before the move but instead making the move and then finding out the requirements for the move. In other cases, principals can simply declare a particular problem/solution space alignment to be real and proceed to implementation. We call this the "early out strategy" and it was followed with disastrous consequences in the Taurus system. In the former case no specification is produced; in the latter it is likely that a bad specification will be produced and declared to be good by powers that be. Either can be damaging to organizational welfare, but the latter is often more destructive because resources are consumed in a futile effort to build a useful system from bad specifications and organizational confidence in system development suffers (Keil 1995; Markus et al. 1994).

As per the functionalist requirements model, care needs to be taken to minimize the size and impacts of possible requirements cascade affects. One possible way is to apply to concept of separation of concerns and modularization (Brooks 1995) at the heterogeneous level of the requirements web, not just at the technical level. Such constructs can act as a "firebreak" to the "wildfire" of cascade failures at the points of modular connection. This approach requires a more holistic approach in constructing a requirements specification. This corresponds to the construction of complex subassemblies as discussed by Simon (Simon 1996). Such subassemblies, as argued by Simon, allow for the creation of more stable and more highly complex systems. Complex subassemblies can be heterogeneous, i.e. a system composed of different types of parts. Indeed, this technique can allow for different types of possible future solution combinations that may be difficult to see without such constructions.

A second method to be gained from this requirements model is the ability to identify "killer problems" or what Hughes calls "reverse salients" (Hughes 1987). A killer problem occurs when a problem that must be addressed for the system under consideration to be successful cannot be adequately addressed within the current solution space. In other words, there is a gap between what exists and what is wanted by one or more principals, and it is too wide to cross within the current solution space. This can be due to either 1) the cost of crossing the gap being can be too high for the current principals to pay, which demands enrollment of new principals, 2) it is simply not known how to cross it, thus demanding novel technical solutions and organizational learning, or 3) it is difficult to cross it in that the resulting socio-technical system may not stabilize due to the radical nature and pervasiveness of change. Such killer problems are normally one problem or a set of interrelated problems- reverse salients- in the spirit of heterogeneous engineering that block the progress. By being able to separate such problem sets, it is easier to set the right targets for specifying requirement sets. Single problems are normally technical in nature, which may demand assessment of technical trajectories and capabilities, while the latter deal with heterogeneous elements in the solution space and combine functional, political or technical issues. The latter ones are normally extremely difficult to analyze and understand and are therefore often ignored. The main ways to deal with a killer problem is to either re-problem (change the project by choosing alternative, solvable problem(s) where possible) and correspondingly change the solution space; if possible, partition the problem into sub-problems which are solved one at a time and which enable learning and adaptation ("muddling through"); or end the project as too risky.

Since requirements are relationships between problems and solutions, it is very likely that there are killer requirements in large system development. These are accordingly, requirements that must be, but yet cannot be, adequately addressed by a current system under consideration. We argue that these are results of either improper relationships or representations of contextualized killer problems. An improper relationship is one that creates an objective or constraint, which cannot be met or makes it impossible to meet one or more separate requirements. In this case, it may be possible to either drop the requirement or rework it such that the connection between the solution and problem (objective) or problem and solution (constraint) can still be realized, and other conflicts mitigated. If such rework is not possible, this would indicate a gap that cannot be crossed between what is and what is wanted, i.e. a killer problem. Unfortunately, such analyses are not normally done and they rarely form part of a risk-assessment.

The ability to uncover additional problems by problem blossoming allows for the ability to discover killer problems, if they exist. Unfortunately, this could lead to a search in an exponential size search space. There are at least two methods that can help deal with this problem. First, if separation of concerns and modularization are well applied, then the search space is reduced to the complex components and the combined system itself. This will tend to isolate such problems within these subdivisions or at the system level, hence reducing the possible search space.

Second, while problems are discovered by analysts, they tend to be quickly categorized according to their understanding of how difficult it is to solve them. Most problems in new systems have therefore known solutions, otherwise the effort would be pointless. Some problems may not have obvious solutions, but usually there are known techniques to work around them. This leaves problems that are not known how to solve once they are uncovered. These are potential killer problems and they quite often relate to heterogeneous nature of the RE activity, i.e. how the resulting socio-technical system will stabilize. The list of these problems is likely to be rather small, but they are often "deadly enemies." As noted, these can also be some of the most tricky to either find, or correctly identify. Still, being able to perform problem triage to identify the most serious problems should be part of a system analyst's repertoire of RE techniques. We believe, that by applying the heterogeneous approach to system analysis would allow analysts to discover more of these potential killer problems. As such, it can be considered a risk reduction methodology.

These and other techniques needed to deal with heterogeneous RE are the subject of future research. By improving our understanding of the complexity and uncertainty involved in RE, we should see an overall reduction in failed systems and a likely increase in the production of successful systems. In the end, this is the main goal of requirements engineering.

6. Glossary

Anomaly – A known existing inconsistency between the current solution space and a desired solution space. An anomaly is only a potential problem, because not all anomalies are attended as problems that need to be solved.

Anomaly Space – The set of all anomalies associated with the system under consideration.

P – The space of all problems implied by a current solution space S_t. A problem space (e.g., the space of all selected problems) is by definition always a subset of an anomaly space.

P_n – A problem (1 of at least n problems) within P_t.

P_t – A Proposed System Problem Space is the space that contains all of the recognized and chosen problems by all of the principals at time t.

Principal – A person with standing. Standing refers here to the power to define and legitimize an anomaly as a problem to be solved by collective action, and the demonstrated capability to mobilize means to address a defined problem. A principal is thus assumed to hold organizational power, i.e. have access to means by which she or he can influence the others.

Problem – An anomaly that is observed and acted upon by a principal with a standing to act.

Problem Blossoming – The process of recursive problem determination. The process of problem determination (problematization) uncovers frequently other anomalies that are deemed problems. This can trigger an iterative reconsideration of the current solution space, respectively. This, in turn, can lead to a further discovery of new anomalies, which can be deemed problems. This recursive process continues until all related problems are constructed or prematurely stopped by one or more a principals.

R_c – A constraining requirement. A constraining requirement, e.g., a constraint, is defined as: a restraining condition imposed upon a solution within S_{t+1} as contextualized by a problem within P_t. A R_c is represented by an arc from a problem to a new solution.

R_o – An objective requirement. An objective requirement is defined as: a want, need or desire that corresponds to a problem (e.g., P_n within P_t) as contextualized by a part or all of a current solution (e.g., S_t). A R_o is represented by an arc from a solution to a problem. In this way, R_o can viewed as a relationship that comes from a solution looking for a problem.

S_t – A Local Solution Space is the current solution space and all locally accessible solution spaces that can be reached from the current solution space using available skills and resources offered by the principals.

S_v – A Future Solution Space is an alternative local solution space postulated from the problem space analysis for the problem set P_t.

S_t – The Current Solution Space, denoted as S_t. Fundamentally, this space embodies a history of solved social, technical and procedural problems which constitutes the legacy of previously solved organizational problems.

S_{t+1} – A Proposed Solution is a subspace of the future solution space, S', that includes the reconciliation of S_t to P_t.

Solution Space Transformation – The process of reconciliation, i.e. changing S_t into S_{t+1} by solving for P_t.

Notes

1. For a good historical analysis, see Couger and Knapp (Couger et al. 1974).

2. By exactness we mean here that content of the categories introduced and the nature of relationships embedded in the model are defined in an analytically exact way so that it can be used as a basis for developing techniques and deriving systematically research questions. To this end we will use simple set theoretic notations to when introducing our basic concepts.

3. Some of the definitions and analyses may look somewhat complex. Therefore we have included a short glossary of terms and their definitions at the end of the paper.

4. We have expanded later this model to cover also situations where goal congruence cannot be assumed (see Bergman et al 2002).

5. Those who are knowledgeable in possible world semantics (or Kripke semantics, see e.g. (Dowty et al. 1981) can see an immediate similarity with the set of solution spaces that can be reached from the current solution space and the concept of the accessibility relation R from any given possible world to other possible worlds. The difference is that due to organizational learning the set of possible solutions spaces accessible from the current solution space is not fixed, but changes over time.

6. This is similar to DeTombe's simple definition of a problem (DeTombe 1994). It is also in alignment with the definition used in the Requirement Engineering literature (Haumer et al. 1999; Kotonya et al. 1998).

7. This formulation does not exclude the possibility that the principal does not have these skills and capabilities available when the time to act is in. We must, however, believe that there is some belief among actors' involved in such capabilities under the rational model, otherwise the principal should choose not to act at all. Within a political scenario, this is not necessarily the case. This suggestion is also derived from the idea that it is actors with solutions looking for problems rather than the other way round. Therefore the demonstrated capability is important in any RE process.

8. Latour (1991) call such contingencies or situations as "passage" points that are governed by "gatekeepers (Latour 1991).

9. These are known in the IS literature as abandoned projects or the process of de-escalation (Keil 1995; Keil et al. 2000).

10. cf. Foucault's 'those who have ability to define 'truth' are those who have power' (Foucault et al. 1980).

11. Organizational Power Politics, pp. 22

12. The issue of 'who gets to be a principal' is as important as 'what is the problem.' This issue is discussed throughout the rest of the treatise. A more in-depth treatment of this issue is beyond the scope of this paper.

13. A problem is understood within the context of the socio-technical (e.g., organizational) ecology in which the solution space resides. Each problem is constructed, legitimized, and owned by one or more principals who reside in this ecology. Hence, all problems are ecologically situated (Suchman 1987), socially constructed (Berger et al. 1966; March et al. 1976) and owned. Contextualization includes all of these concepts.

14. For the sake of clarity, each anomaly in Figure 2 has been defined into a problem.

15. Problem blossoming is similar to an aspect of Checkland's Soft Systems Methodology (SSM) (Checkland 1990). However, problem blossoming is focused on problem discovery and identification of likely impacted parts of S_t. SSM, in contrast, focuses on the whole process of systems development. Problem blossoming and SSM share the basic components of iterative learning and discovery, as well as a recognition of the ability to change the current solution and problem spaces as per new insights.

16. This space, S, is similar to Davis' product space defined as "the range of all possible problem solutions that meets all known constraints." Software Requirements: Objects, Functions, and States, pp. 42

17. We are using here the concept of frame as defined by Bijker (Bijker 1987). Bijker uses the term frame to denote a new aggregate of concepts and techniques employed by a community of problem-solvers in its problem solving. Changes in frames embody "revolutions" and discontinuities in technology evolution. Such discontinuities are a reflected in such elements as goals of technology use, theories and concepts, problem-solving steps and tools, and organizational and managerial principles related to problem solving practices. An example of this is moving from structured programming to object-oriented programming.

18. In theoretic terms, a solution space is the subset of the Local Solution Space (S) that is affected by as well as affecting the system change. In practical terms, a solution space can be represented as a model of this space, for instance a (richly detailed) workflow or equivalent model.

19. For clarity purposes, t+x is replaced by t+1 throughout the rest of the paper. This is used to indicate the time of an operational future solution space.

20. Loucopoulos & Karakostas, as well as most of the requirements engineering literature, transforms a capability into functionality and a condition into a constraint (e.g., nonfunctionality). Functionality represents a want, need or desire of one or more principals in S_t for a new capability to be made available in response to one or more of their problems in P_t. This corresponds with Gause and Weinberg's view on requirements engineering: a process to discover what people desire (Gause et al. 1989). It is also in line with Graham's view, which focuses on what is needed by people (Graham 1998). A constraint represents a condition on a future solution in response to a problem.

21. This same observation has been confirmed by researchers advocating the spiral model of software development which emphasizes the evolution and learning of requirements and the dynamic nature of mappings between requirements and implementations (see e.g. (Boehm 1988; Iivari1990a; Iivari 1990b)).

22. As discussed earlier, there could be many arcs from a solution source to a single problem. Still, each arc is an individual requirement. Also, the arrow on the arc (which is informational only) indicates the node pointed to is contextualized by the node pointed from. In this case, a problem is contextualized by a solution.

23. A constraint does not originate in a formal document. It is rooted in one or more problems that are implied in the formal document. Since, by definition, these contextualizing problems come from a formal document, they must have principal representation and, thus, are part of P_t. But, these problems are usually not clearly and specifically identified in most formal documents. This means they cannot be accurately represented within P_t. This is a source of potential requirements failure. The possibility exists of not truly solving the problems that the formal document wanted addressed by the project under consideration even if the constraints (or objectives) specified in a formal document are well met. In turn, this allows for the increased probability of an incorrect future solution, resulting in eventual deployment failure. Altogether, this highlights the importance of constraints as traceable relationships, not just stated restrictions.

24. The lightened and bolded parts of S_{t+1} corresponds to proposed changes in the solution space S_t in response to the problems in P_t.

25. cf. Bak's adding a grain of sand causing an avalanche in a sand pile (Bak 1996).

References

Abdel-Hamid, T.K., and Madnick, S.E. "The Elusive Silver Lining: How We Fail to Learn from Software Development Failures," Sloan Management Review (32:1) 1990, pp 39-48.

Bacharach, S.B., and Lawler, E.J. Power and politics in organizations, (1st ed.) Jossey-Bass, San Francisco, 1980, pp. xviii, 249.

Baier, V., and March, J. "Implementation and Ambiguity," Scandinavian Journal of Management Studies (4:May) 1986, pp 197-212.

Bak, P. How nature works: the science of self-organized criticality Copernicus, New York, NY, USA, 1996, pp. xiii, 212 , [218] of plates.

Berger, P.L., and Luckmann, T. The social construction of reality; a treatise in the sociology of knowledge, ([1st] ed.) Doubleday, Garden City, N.Y.,, 1966, pp. vii, 203.

Bergman, M., King, J.L., and Lyytinen, K. "Large Scale Requirements Analysis Revisited: The need for Understanding the Political Ecology of Requirements Engineering," Requirements Engineering Journal:Submitted for Publication) 2002.

Bijker, W. "The Social Construction of Bakelite: Toward a Theory of Invention," in: The Social Construction of Technological Systems, W. Bijker, T. Hughes and T. Pinch (eds.), MIT Press, Cambridge, MA, 1987, pp. 159-190.

Boehm, B.W. "A spiral model of software development and enhancement," Computer (May) 1988, pp 61-72.

Brooks, F.P. The mythical man-month: essays on software engineering, (Anniversary ed.) Addison-Wesley Pub. Co., Reading, Mass., 1995, pp. xiii, 322.

Checkland, P. Systems thinking, systems practice J. Wiley, Chichester Sussex ; New York, 1981, pp. xiv, 330.

Checkland, P., and Scholes, J. Soft systems methodology in action Wiley, Chichester, West Sussex, England ; New York, 1990, pp. xv, 329.

Couger, D., and Knapp, J. Systems Analysis Techniques John-Wiley & Son, London, 1974.

Davis, A.M. Software requirements: objects, functions, and states, (Revision. ed.) PTR Prentice Hall, Englewood Cliffs, N.J., 1993, pp. xxviii, 521.

DeTombe, D.J. Defining complex interdisciplinary societal problems: a theoretical study for constructing a co-operative problem analyzing method: the method COMPRAM Thesis Publishers, Amsterdam, 1994, pp. xiv, 439.

Dowty, D.R., Wall, R.E., and Peters, S. Introduction to Montague semantics Kluwer Boston Inc., Boston, MA, 1981, pp. xi, 313.

Drummond, H. "The Politics of Risk: Trials and Tribulations of the Taurus Project," Journal of Information Technology (11) 1996, pp 347-357.

Fairholm, G.W. Organizational power politics: tactics in organizational leadership Praeger, Westport, Conn., 1993, pp. xxiii, 230.

Foucault, M., and Gordon, C. Power knowledge: selected interviews and other writings, 1972-1977, (1st American ed.) Pantheon Books, New York, 1980, pp. xii, 270.

Gause, D.C., and Weinberg, G.M. Exploring requirements: quality before design Dorset House Pub., New York, NY, 1989, pp. xvii, 299.

Graham, I. Requirements engineering and rapid development: an object-oriented approach Addison Wesley, Harlow, England ; Reading, MA, 1998, pp. xvi, 271.

Haumer, P., Heymans, P., Jarke, M., and Pohl, K. "Bridging the Gap Between Past and Future RE: A Scenario-Based Approach," RE'99, IEEE Computer Society, Limerick, Ireland, 1999, pp. 66 - 73.

Hughes, T. "Emerging Themes in the History of Technology," Technology and Culture (20:4) 1979a, pp 697-711.

Hughes, T. "The Electrification of America: The system builders," Technology and Culture (20:1) 1979b, pp 124-161.

Hughes, T. "The Evolution of Large Technological Systems," in: The Social Construction of Technological Systems, W. Bijker, T. Hughes and T. Pinch (eds.), MIT Press, Cambridge, MA, 1987, pp. 51-82.

Iivari, J. "Hierarchical spiral model for information system and software development. Part 1: theoretical background," Information and Software Technology (32:6) 1990a, pp 386-399.

Iivari, J. "Hierarchical spiral model for information system and software development, Part 2: design process," Information and Software Technology (32:7) 1990b, pp 450-458.

Keil, M. "Pulling the Plug: Software Project Management and the Problem of Project Escalation," MIS Quarterly (19:4) 1995, pp 421 - 447.

Keil, M., and Montealegre, R. "Cutting your losses: Extricating Your Organization When a Big Project Goes Awry," Sloan Management Review (41:3) 2000, pp 55-68.

Kotonya, G., and Sommerville, I. Requirements engineering: processes and techniques J. Wiley, Chichester; New York, 1998, pp. xi, 282.

Langefors, B. Theoretical Analysis of Information Systems Studentlitteratur, Lund, Sweden, 1966.

Latour, B. "Technology is society made durable," in: A Sociology of Monsters: Essays on Power, Technology and Domination, J. Law (ed.), Routledge, London, 1991.

Law, J. "Technology and Heterogeneous Engineering: The Case of Portuguese Expansion," in: The Social Construction of Technological Systems, W. Bijker, T. Hughes and T. Pinch (eds.), MIT Press, Cambridge, MA, 1987, pp. 111-134.

Lindblom, C.E. "Still Muddling Through," Public Administrative Review (39) 1979, pp 517-526.

Loucopoulos, P., and Karakostas, V. System Requirements Engineering McGraw-Hill Book Co., London, UK, 1995.

Lyytinen, K., Mathiassen, L., and Ropponen, J. "Attention Shaping and Software Risk- A Categorical Analysis of Four Classical Approaches," Information Systems Research (9:3) 1998b, pp 233-255.

Lyytinen, K., Rose, G., and Welke, R. "The Brave New World of Development in the internet work computer architecture (InterNCA): or how distributed computing platforms will change systems development," Information Systems Journal (8:3) 1998a, pp 241-253.

Macaulay, L. Requirements engineering Springer, London; New York, 1996, pp. xiii, 202.

March, J.G., and Olsen, J.P. Ambiguity and choice in organizations Universitetsforlaget, Bergen, 1976, p. 408.

Markus, M.L., and Keil, M. "If We Build It, They Will Come: Designing Information Systems that Users Want to Use," Sloan Management Review (35:4) 1994, p 22.

Mitev, N.N. "More than a Failure? The Computerized Reservation Systems at French Railways," Information Technology & People (9:4) 1996, pp 8-19.

Myers, M.D. "A Disaster for Everyone to See: An Interpretive Analysis of a Failed IS Project," Accounting, Management and Information Technologies (4:4) 1994, pp 185-201.

Noyes, J.M., and Baber, C. User-centered design of systems Springer, London; New York, 1999, pp. xii, 222.

Pfeffer, J. Power in organizations Pitman Pub., Marshfield, Mass., 1981, pp. xiv, 391.

Pohl, K. Process-centered requirements engineering Wiley, New York, NY, 1996, pp. xviii, 342.

Robinson, W.N., Pawlowski, S.D., and Volkov, V. "Requirements Interaction Management, Unpublished Working Paper, Department of Computer Information Systems, Georgia State University," 1999.

Rosenwein, M. "The Optimization Engine That Couldn't," OR/MS Today (24:4) 1997, pp 26-29.

Ross, D. "Structured Analysis (SA): A Language for Communicating Ideas," IEEE Transactions on Software Engineering (3: 1) 1977, pp 16-34.

Simon, H.A. "Rational Decision Making in Business Organizations," American Economic Review (69:4) 1979, pp 493-513.

Simon, H.A Models of Bounded Rationality MIT Press, Cambridge, MA, 1982, pp. 160-176.

Simon, H.A. The sciences of the artificial, (3rd ed.) MIT Press, Cambridge, Mass., 1996, pp. xiv, 231.

Sommerville, I., and Sawyer, P. Requirements engineering: a good practice guide John Wiley & Sons, Chichester, England ; New York, 1997, pp. xi, 391.

Suchman, L.A. Plans and situated actions: the problem of human-machine communication Cambridge University Press, Cambridge Cambridgeshire ; New York, 1987, pp. xii, 203.

Wieringa, R. Requirements engineering: frameworks for understanding Wiley, New York, NY, 1996, pp. xvi, 453.