The Science of IS Development – Implications for Applications?

Joerg Evermann

School of Information Management

Victoria University Wellington

Box 600, Wellington, New Zealand

Abstract

Information Systems (IS) are understood as models or representations of an application domain. The first step in IS development is the analysis and description of this domain. A good understanding is important for effective software design and implementation. Scientific inquiry also aims towards an analysis, description, and understanding of a domain. In contrast to IS development and IS analysis, scientific inquiry is not a recent activity but is based on a 4000 year history. Its processes and methods are well accepted and arguably successful.

This paper suggests that IS development activities can and should be informed by the process of scientific inquiry. It explores the parallels between the two and shows potential implications of viewing IS development, and system analysis in particular, as a kind of scientific inquiry. From this, specific recommendations for IS development projects are derived. The paper is intended as a contribution to the discussion of a philosophical foundation for IS development.

Keywords : Philosophy of science, system analysis, IS development, IS implementation, IS project management

1 Introduction

Information systems (IS) are representations or models of an application domain (Iscoe *et al.*, 1991; Jackson, 1995; Wand and Weber, 1993). For example, a production planning and control (PPC) system is intended to represent, with its software structures and its data, the production facilities of the business. When things change on the shop floor, the PPC system must reflect this change. Hence, building an effective and efficient IS, which represents the domain, solves identified problems, and interacts with domain elements and users, requires a good understanding of the application domain.

IS development occurs as well defined projects. The development process begins with the analysis and documentation of the application domain and leads to the implementation and use of the final software system. Good system analysis has been argued to reduce project risk (Offen, 2002), enhance the completeness and consistency of software requirements (Iscoe *et al.*, 1991), and reduce costly later re-work (Boehm, 1988).

IS development and system analysis are relatively young fields, and there still exists much debate about its methods and processes (e.g. Brooks, 1986). In contrast, there exists a rigorous process of inquiry into a domain, which is much older, accepted in the community practicing it, arguably successful as measured by the usefulness of its results, and well regarded by its wider audience. This is the process of scientific inquiry.

This paper explores the parallels between the processes and artifacts of science and IS development. As both IS development, specifically system analysis, and scientific inquiry are concerned with the investigation and description of a domain, we suggest that much can be learned by applying well-accepted and successful scientific methods to the process of IS development and system analysis. This presents a new and different perspective on IS development and system analysis, and can provide theoretical foundations for IS development.

The paper also makes recommendations for project managers and system analysts. These are specific and operationalizable, and are based on transferring accepted, successful scientific practices to IS development. In summary, this paper can contribute to the discussion on foundations of system analysis and the IS development process.

Process

Science is defined by its process rather than its outcome. This process distinguishes it from other types of inquiry and makes science 'respectable' (Thagard, 1998; Casti, 1989). The process is iterative. In a simplified form, each iteration proceeds from a problem to a candidate solution or hypothesis, which is then tested and the test results evaluated (Bunge, 1998; Popper, 1968, 1972).

IS development is analogous to science: Starting with an initial business problem, e.g. frequent out-of-stock situations, the analyst forms a theory about the organization in the form of a conceptual or domain model, which expresses the analysts understanding of the domain (Arango, 1989; Mylopoulos, 1992). For example, for out-of-stock situation, the conceptual model may represent the procurement process, its participants, and information about warehouses and stock withdrawals. This conceptual model then forms the basis for subsequent software design and implementation. The resulting IS is understood to be not only a representation of the domain but also a solution to the out-of-stock problem. The IS is then tested to evaluate whether it is functional and solves the identified problem.

From this very general characterization of the scientific process and its analogy in IS development, we can draw a first implication. As repeated hypothesis formation and testing are a *normal* process in the sciences, IS development should similarly be expected and managed as an iterative process. Iterations are necessary for the development process, rather than failures of the process. With this expectation, project managers needs to facilitate the iterative nature of the process by supporting and guiding the evaluation of successive conceptual models and successive versions of the software system. Iterations need to be planned into the process, e.g. in terms of project milestones, budget, and staffing. Criteria that determine the activities and schedule of the iterations need to be established. The iterative nature of IS development has been recognized as useful for example to manage project risk (Boehm, 1988), and is found in evolutionary prototyping approaches (Floyd, 1984; Riddle, 1984) as well as in the more recent Rational Unified Process (Kruchten, 2002).

No scientific theory is entirely accurate. It will necessarily be revised and replaced by more accurate, more predictive, or more explanatory theories. The process of scientific discovery does not end, but proceeds along theories of increasing accuracy and explanatory power.

Similarly, no conceptual model or software system can be expected to be completely accurate in representing the organization and achieving all its aims. Instead, it will need constant adjustment and revision. Consequently, IS development should not be expected to be a one-off endeavour. Analogous to the way in which new scientific theories are continuously built on, and improve, old ones, analysis of a domain cannot be considered complete at the end of the first iteration of the development process (Sotirovski, 2001). This suggests that IS development should be viewed as a continuous and ongoing activity in the organization, rather than as being partitioned into discrete projects. Development, adaptation, and improvement of the IS must be actively managed throughout the period where the need for the system remains.

For example, as long as the company is ordering and warehousing items, there is a need for an inventory management system. Instead of analyzing the domain as part of a project effort outside of general inventory management group, we suggest instead that within that group a small team of analysts should continuously update the models of the domain. Depending on changes to these models, at certain points in time, the existing software system needs to be adapted, or a new system implemented. These latter activities may be done as discrete projects, as theory testing and instrument are resource intensive activities. However, the analysis of the domain should be an ongoing activity.

Rather than bringing together a team of analysts with diverse backgrounds for a short duration, analysts should be permanently assigned to manage and carry out the continuous investigation of the domain. Just as scientists work on a topic for a long period, domain or business analysts should not be moved from one project to another. Like scientists, they should be expected to increase their expertise in a particular application domain. In essence, this moves the system analysis activity from a dedicated organizational department into the line units.

The iterative nature of the scientific process requires it to be guided by the analysis of empirical data of scientific experiments. These data can point to specific aspects of a theory that may not be supported. Similarly, IS developers need to be able to perform such analyses. For example, the fact that inventory is not ordered in a timely fashion, may point to a fault in the model that describes the ordering process. In this case, the theory is not a good description of reality, and needs to be changed. Requirements traceability and requirements management (Jarke, 1998) become increasingly important for this purpose. They help trace back the failure of certain software features to achieve intended goals to elements of the conceptual model, which specified or determined those software features. Development projects need to establish and manage both forward and reverse traceability, putting in place appropriate tools and procedures.

In science, theories are tested based on a-priori acceptance criteria. They determine whether test results can be accepted as valid, either confirming or refuting the tested theory. These are not only criteria for statistical validity, but also criteria for acceptable research design, etc. Similarly, IS development and system analysis need to develop and establish acceptance criteria that determine whether a particular IS or software system is a valid representation of a domain. Such acceptance criteria need to extend beyond software acceptance testing to include acceptance testing of conceptual models, user acceptance, and delivery of business benefits. This in turn requires that user acceptance, business requirements, and business goals are measurable.

Theories

Scientific theories are more than mere sets of logical statements or a number of axioms. Theories, as tested and evaluated by science, are structures with a central axiomatic core and surrounding corollaries and derived laws (Lakatos, 1978). While the outermost elements can be changed with relative ease, scientists are reluctant to give up elements that are increasingly more central and serve as foundations for larger numbers of corollaries and laws.

In principle, an experiment tests the entire logically connected structure of a theory. In practice, when observation disagrees with prediction, scientists will attempt to find the 'weakest link' in the argument and adapt or discard it (Duhem, 1954; Quine, 1953). The weakest links are those that connect the fringes of the theoretical structure to empirical reality (Quine, 1953). However, re-evaluation of a particular statement of a theory may in turn require re-evaluation of connected statements. This iterative process continues until the whole theory structure is again logically consistent and coherent.

Drawing the analogy to IS development, the implemented software forms the outer fringe of the theory. This is where theory meets reality (the domain or organization). The software is constructed from a domain model by means of rigorous intermediate transformation steps (software engineering). When the software does not perform as expected, developers begin looking for failures. For example, when the IS does not solve the frequent out-of-stock situations, some software parameters may be set incorrectly (implementation), or some design decision is inadequate (e.g. overnight batch updates are too slow), or perhaps there was a failures in the domain analysis (e.g. intermediary warehouses were not considered).

Project managers need to establish principles that guide this Quinean process of recursive re-evaluation of theory elements. Well-documented criteria must be put in place to determine when changes need to be escalated to more central elements, such as conceptual models. Kuhn (1977) points out that such guidelines or criteria are necessarily social in their nature. Hence, project managers must put in place communication and negotiation procedures so that change escalation criteria can be communicated and agreed upon by the entire development team.

Scientific theories include statements about properties of the domain elements, including their dynamics. Traditional database driven IS development often neglects this aspect, and even object-oriented IS may be limited in the analysis of domain dynamics. However, simulation science can provide dynamic modelling and evaluation techniques on an abstraction level suitable for system analysis (Fishwick, 1995; Evans, 1988). Hence, we suggest that simulation models need to be employed during IS development and should become part of the domain model. Comparison of simulation results to the actual domain dynamics can help validate conceptual models. For example, while a UML activity diagram can be used to model the stock ordering process, a simulation tool can be used to simulate and compare the model to observed reality.

Paradigms

When a theory is viewed as an onion-like layering of logical statements or elements, the core of a theory is the scientific paradigm. It determines a scientist's perspective on the domain, and determines the ontology, the way we think and speak about the domain (Kuhn, 1996).

Similarly, in IS development, the conceptual model is a theory of the domain. It relies on the modelling language that was chosen. This, in part, determines the ontology, the terms in which the application domain is described. For example, the analyst may realize that the modelling language is ill-suited, e.g. the business can be better described in terms of objects and their behaviour rather than functional units and the processes they execute. An even more central aspect of the theoretical structure is the chosen perspective on the business itself. While the view of the business as a system of functional units is the dominant paradigm, it is not the only possible one. Hirschheim and Klein (1989); Hirschheim *et al.* (1995) have used a framework by Burrell and Morgan (1979) to explicate assumptions about the organizational context of IS development and Reed (1992) has identified five "analytical frameworks" of organizations, which can be likened to scientific paradigms.

First, organizations can be viewed as social systems with collective goals and institutional needs. When taking this perspective, information systems support and reflect the goal-oriented nature of the organization. Descriptions and models may focus on goal-structures, and business requirements, e.g. in the form of agent-oriented systems (Yu, 2001).

A second framework understands an organization as negotiated orders that are "created, sustained and transformed through social interaction" (Reed, 1992). These orders are temporary and continually reconstitute themselves. In this case, the analysis will focus on the negotiation processes (Chang and Woo, 1994), rather than the constituents and their fixed roles. Descriptions and models emphasize interactions (Auramaki *et al.*, 1988; Janson *et al.*, 1993; Winograd and Flores, 1986) such as advanced e-mail, instant messaging, workflow management systems, and other communication technologies.

Third, organizations are understood as power structures, which serve to maintain positions of dominance. From these macro-level considerations, implicit and unquestioned objectives such as 'rationality', 'profitability', 'efficiency', etc. are derived. In this case, descriptions and models will focus on identifying or defining control and command relationships between functional, organizational units. Projects must identify, explore, and manage their relation to such power position in order to gain acceptance.

Fourth, organizations as constructions of cultural artifacts are generated by the "values, ideologies, rituals, and ceremonies that express and make sense of participation"(Reed, 1992). Consequently, the culture "requires support and supplementation through symbols, myths and rituals." Business analysts can utilize these symbols and rituals as devices to effect change, gain acceptance, communicate with stakeholders and to solve problems (Hirschheim and Newman, 1991; Kendall and Kendall, 1993). When taking this perspective, the IS must be designed with these rituals and ceremonies in mind. It must support them with its user interfaces and interactions, and play an active role in them.

The fifth framework sees organizations as social practices in which they are "reproduced through the design and deployment of various administrative mechanisms by means of which managers attempt to realize effective regulation or control over the performance of work"(Reed, 1992). Thus, business analysis will focus on and administrative issues, e.g. by means of business rules.

It is important that IS development projects explicate the framework or frameworks they adopt. Each of the different perspectives or frameworks may lead to different requirements, can be expressed in different languages, and can be analyzed using different methods. Examples in (Checkland and Holwell, 1998; Dahlbom and Mathiassen, 1993; Hirschheim *et al.*, 1995; Winograd and Flores, 1986) show the strong influence of different organizational perspectives on IS development.

Paradigm Changes The assumptions about the nature of the organization or business form the core or the paradigm of the theory about the business. Consequently, these are hardest to change. In science, the failure of "normal science", i.e. the incremental and iterative changes to a theory, gives rise to a "scientific revolution" (Kuhn, 1996). The paradigm is discarded and with it the language and any theories of the previous paradigm. Failure of a paradigm becomes evident when too many phenomena are either left unexplained or require a large number of ad-hoc assumptions.

Similarly, in system analysis, such a revolution discards the conceptual model, the modelling language, and the adopted perspective on the nature of the business. For example, instead of tackling the problem of long lead times by improving the efficiencies and control structures of each organizational unit, the analyst may instead decide to focus on the collaborative nature of the participants and their interactive needs. Hence, the organization is not viewed as a rigid system that must be optimized, but as an organic system of interacting agents. The ontology and the modelling language change from describing machinery, efficiency, and optimality, to describing actors, agents, communicative needs, and communication channels.

If important aspects of the organization cannot be explained or expressed in the conceptual model, or if this requires too many ad-hoc assumptions, then the perspective of the analysis needs to be redefined, i.e. a new organizational framework must be chosen. The challenge for IS project management is to recognize and facilitate necessary change, rather than oppose it (Dahlbom and Mathiassen, 1993). Continuous assessment of the current theories in terms of their acceptance by domain experts and their usefulness to IS design needs to be in place, together with guidelines on how changes are escalated from the conceptual model to the modelling language, and to the organizational perspective. How much ad-hoc modification should be allowed for the domain model before it becomes necessary to switch the perspective on the nature of the organization? How many aspects of the domain can be reasonably excluded from the conceptual model before it fails to accomplish its purpose?

Co-existing Paradigms A different interpretation of Kuhnian scientific paradigms views them not as successive and mutually exclusive, but as co-existing on different temporal, spatial, or organizational levels of abstraction. For example, Maxwellian electrodynamics remains valid and valuable for everyday electronic engineering, while quantum electrodynamics is evidently more accurate.

Drawing the analogy to IS development, it may be possible to apply different organizational paradigms (Burrell and Morgan, 1979; Reed, 1992) at the same time, each being applicable in a well-defined domain of abstraction. For example, examining an organization on the level of business units, one can make out functional structures and control relationships. When examining an individual department within a business unit, one may need to examine the dynamics of individual, goal driven actors and their activities. Hence, on different level of abstraction, different paradigms can be chosen.

The challenge for development projects is to clearly delineate the respective domains of the various organizational perspectives in terms of temporal, spatial, or organizational abstraction levels. In order to gain an understanding of the organization as a whole, the analysts need to develop points of integration between the perspectives and the associated domain models. For example, business process descriptions using the language of UML Activity Diagrams need to be integrated with models of spontaneous goal driven behaviour of agents. While the process model captures one aspect of the organization (one paradigm), the agent model provides, at the same time, another perspective on a different level of abstraction.

How does goal driven behaviour on the individual level give rise to business processes on the more abstract level? How can the two descriptions be reconciled? In science, such integration usually involves approximations. For example, Newton's theory of gravitation approximates of General Relativity for small distances. In the case of an organization, one can for example investigate how business process level descriptions can be considered an approximation of intelligent actor dynamics.

Choice of Theory

Choosing a new paradigm or theory is not always a rational activity (Kuhn, 1996). Many theories can account for the same observations. Moreover, there is no theoretic or scientific reason to choose between altering a theory in the fringes and replacing it entirely. These questions must be approached by "reasons of good sense" (Duhem, 1954), which include criteria such as accuracy, internal and external consistency, broad scope beyond the particular observations it was designed to explain (predictive), simplicity (or efficiency, elegance, beauty), and disclosure of new phenomena and relationships (Kuhn, 1977). A theory is preferable to another, if it has withstood more attempts at refutation (Popper, 1968; Lakatos, 1978). These guidelines, like the overall scientific process, are the result of social agreement (Laudan, 1990). Other criteria concern the relative importance assigned to the explanation of existing observations versus the confirmation of predictions (Snyder, 1994; Achinstein, 1998).

Consequently, in IS development, decisions regarding the choice of conceptual models or perspectives of the nature of an organization are not necessarily rational. Hence, they should not be managed as if they were rational. Instead of a-priori fixed criteria, project management needs to support the social nature of the decision making process. Criteria such as simplicity and elegance are important characteristics and need to be included. Experienced system analysts, like experienced scientists, will recognize these values in a theory, even if they defy precise measurement.

Based on initial criteria, project managers must initiate and manage the social process to arrive at shared final quality criteria for conceptual models and software. Quality frameworks, like those proposed by Krogstie *et al.* (1995); Moody and Shanks (1994, 1998); Schütte and Rotthowe (1998), can be used to begin this effort. However, they should be used only as a starting point for discussion, rather than as absolute metrics of model quality or theory quality.

Scientific theories cannot be proven; they can only be disproven (Popper, 1972). Hence, theories must explain observed phenomena as well as make falsifiable predictions. For IS development, this has two implications.

First, the conceptual model should give rise to predictions, i.e. it cannot be purely descriptive. Some of the possible predictions need to be explicated and tested.

Second, testing of the conceptual model (and predictions based on it), or the software system, must attempt to falsify it. Test plans, test protocols and test procedures must reflect this aim. Project managers must ensure that all aspects of the theory, not only the final implemented software, are tested. Statement and predictions of the conceptual model can be tested and attempted to be falsified by actively looking for counter-examples within the domain. Analogous to peer review in the scientific process, the conceptual model should be reviewed and critically assessed by analysts and organizational staff *outside* the development team before final acceptance.

Models and Scientific Realism Two scientific theories or paradigms may specify different entities in the domain, i.e. different ontologies. In IS development, one conceptual model may suggest that the domain consists of functional units that are integrated into processes. Another conceptual model may represent the same domain as a set of interacting actors with goals and constraints. As different ontologies can account for the same observations, there is no criterion that could tell us whether the underlying ontological entities actually exist (van Fraassen, 1980; Laudon, 1981). For example, actual observations show only human actors behaving in various ways. The notion of functional units and processes is superimposed on these observations, as is the notion of goals and constraints. We cannot physically point to a functional unit or a goal.

Consequently, the often implicit requirement that models and modelling languages be isomorphic to, or be true representations of, the application domain (e.g. Wand and Weber, 1993), must be given up. No absolute criteria for correctness of a model can be given, only one of adequacy. Similarly, modeling languages also may only be found adequate, or inadequate, for a particular purpose. Model quality criteria need to focus on such social adequacy in the construction of the model and its problem solving abilities, rather than on its correspondence to reality (Schütte and Rotthowe, 1998). For example, a process-centered description may be as good as an object-oriented or agent-oriented one; two different object-oriented models may capture the world equally well.

Scientific Instruments

Theories are tested by building instruments. In the process of testing successive conceptual models during IS development, the actual IS (the software system) plays the role of instrument. It is derived from and based on the central aspects of the theory that it is intended to test. Similar to an electron microscope embodying the central aspects of quantum physics, an IS embodies the central beliefs about the business or organization as described by the conceptual model.

Using a finished IS as experimental instrument is not desirable for two reasons. (1) It is a very expensive instrument, available only after considerable amount of work has been invested. (2) It is generally not a simple tool; hence, any problems discovered may be with the instrument, rather than the theory. Thus, cheaper and more direct testing of theories (conceptual models) is desirable.

Prototype approaches (Floyd, 1984; Riddle, 1984) provide both a cheaper and more direct instrument, as they focus on one particular aspect of the theory that is tested. Because they are simpler than the final IS, it is easier to demonstrate their correctness as theory testing instrument. Rather than using prototypes in an evolutionary way or for requirements gathering, prototypes need to be specifically aimed at testing one or more particular statements in the conceptual model. These prototypes must be as simple as possible, so that their correctness can be demonstrated. Every important statement of the conceptual model should be tested by prototyping. To achieve this requires suitable management processes, budgeting, scheduling, and staffing considerations.

Social Aspects

Scientists are mutually dependent on each other for sharing of ideas, instruments, etc. (Longino, 1990). The community of scientists admits new members by training; one cannot simply declare to be a scientist. Moreover, this community is embedded in the larger society, depending on its values and support, monetary and otherwise (Longino, 1990).

This has three implications for IS development as scientific inquiry. First, IS development must be a team effort and good communication must be ensured. Projects need to ensure exchange of ideas and instruments, e.g. in the form of meetings, notes, public models, formal discussions, and informal exchanges. Mechanisms must be provided to facilitate exchange also of incomplete, initial, and informal ideas, not only of finished deliverables. Analysts cannot use strict division of labor and compartmentalized thinking. To maintain consistent theories or models, they must be aware of related and competing work. This exchange of ideas and instruments should extend beyond the scope of a particular project, just as it is in science. Best practice approaches, reference models (Scheer, 1994), analysis and design patterns (Eriksson and Penker, 2000; Larman, 2002) and open source software (Raymond, 2001) are examples of IS development approaches embodying these requirements.

Second, the argument for user participation in the process becomes problematic. In scientific studies, non-scientists are sources of information or experimental subjects. However, non-scientists are generally not involved in theory construction, although they may participate in instrument building and theory testing (engineers).

Similarly, in IS development, users are sources of information, e.g. as user focus groups, or subjects of inquiry, e.g. interface development testing. However, participation in the domain analysis should be limited to trained analysts while software construction, i.e. the building of the instrument, can and should involve software engineers.

Analysts must understand the implications of analysis being a scientific process of theory development. Just as scientists are trained to become accepted in their social community, user participants in system analysis need to be trained in system analysis. Similarly, user representatives on project management teams and steering committees must understand the implications of IS development as a scientific processes, to ensures realistic expectations and understanding of the process and outcomes.

Third, the reliance on the wider society, i.e. the organization and general business staff, for support is critical. IS development projects must maintain communication with the rest of the organization. They need to share information about current conceptual models to gain and judge the support for those models and the entire project. This cannot be postponed until the IS is fully constructed and about to be rolled-out.

For example, focusing on command and control may be inappropriate and discussions with the future user base shows reluctance to support any models with these features. Such lack of support may be voiced directly, but may also show itself through indirect means, e.g. the withholding of information, of funds, or of personnel, all of which the project depends on. This is similar to the way in which scientific research relies on, and can be controlled by, access to support, such as information, funding, and personnel.

Communication requires language that is understandable both to the analysts as well as to the wider community. When the conceptual modelling language that is used in a project is not suitable for communication with non-analysts, the domain understanding must be translated to a different, simpler form. Communication also requires a managed process of both dissemination of the results of the analysis, e.g. in the form of project newsletters, and, more importantly, community input that may influence the values of the development project, e.g. in the form of informal "town hall meetings" or more formal and structured exchanges.

Discussion

The previous sections have shown many parallels between scientific inquiry into a domain and IS development, specifically system analysis. However, there are also differences between scientific inquiry and IS developments.

First, an IS is generally built in response to a particular business need. This is different from the environment in which scientific theories are developed. While development of theories reflects primarily explanatory needs, businesses have a need for the IS to play a role in adapting to changing external influences (Porter, 1980), which can range from government intervention (taxes, subsidies, etc.) to the emergence of new competitors, markets and technologies.

However, although scientific theories are explanatory in nature, the explanations they provide are used by engineers to develop useful solutions to problems. Similarly, system analysis can be explanatory in nature, but the conceptual models are used by software engineers to develop useful solutions. Just as good scientific theories can lead to successful engineering, good domain analysis can lead to successful software engineering and hence to successful IS.

Second, unlike science, which is assumed to operate in a relatively stable environment, IS development is an inquiry into often constantly changing domains. However, rapid change may be an indication that the models and theories about the business are not on a suitably abstract level. For example, while the specific business processes may change, the structure of the business and the behaviour of its agents may be stable, so that rapid change and perceived volatility is an indication of an unsuitable ontology, organizational perspective, or level of description.

Information systems will become part of the domain. Their purpose is not only to represent the domain but also to influence and change the domain. For example, in Business Process Re-engineering (Hammer and Champy, 1994), information systems are used as catalysts and enablers for considerable change in a business or organizational domain. In contrast

Scientific Inquiry	IS Development
Theory	Conceptual (Domain) Model, Software model
Instrument	Software system, Prototype
Paradigm	Organizational framework
Ontology	Modelling language
Theory building	System analysis
Instrument building	Software engineering
Theory testing	Prototyping and evaluation
Scientists	System analysts
Engineers	Software engineers

Table 1: Parallels between scientific inquiry and IS development

scientific theories and instruments do not effect substantial changes within the domain.

Conclusion

This paper set out to explore the parallels between IS development, especially system analysis, and scientific inquiry. We have drawn parallels between activities and artifacts of the two fields. These are shown in Table 1. From these parallels, implications have been derived that have the potential to improve IS development efforts.

Science has a long tradition and there is a good and well-accepted understanding of it. It appears informative to use the identified parallels to inform the process of IS development, especially the system analysis activities. This leads to a number of recommendations for the management of IS development projects. Most important, IS development should not be viewed as a one-off project, but instead is an ongoing, iterative process, that is in place as long as the need for the IS exists. Furthermore, system analysis is primarily a social activity that relies on communication, shared assumptions and values, and agreements. It needs the support, and must be based on the values, of the organization in which it is situated.

Some of the proposed recommendations can be found already in one analysis method or another, while others are novel. Some are well-accepted principles while others may appear controversial. In conclusion, this paper advocates a pragmatic approach to suggesting a philosophical foundation for system development practices, by basing it on the well-accepted process of scientific inquiry. The recommendations proposed in this paper should not be considered absolutes, but rather as starting points for debate. It is hoped that this paper can be part of a fruitful discussion on the process and methods of IS development and system analysis.

References

Achinstein, P. (1998). Explanation v. prediction: Which carries more weight? In M. Curd and J. Cover, editors, *Philosophy of Science – The central issues*. W.W. Norton and Company, New York, NY.

Arango, G. (1989). Domain analysis: From art form to engineering discipline. In *Proceedings of the 5h International Workshop on Software Specification and Design, IWSSD-5, Pittsburgh, PA*, pages 152–159.

Auramaki, E., Lehtinen, E., and Lyytinen, K. (1988). A speech-act-based office modeling approach. *ACM Transactions on Office Information Systems*, **6**(2), 126–152.

Boehm, B. (1988). Understanding and controlling software costs. *IEEE Transactions on Software Engineering*, **14**(10), 1462–1477.

Brooks, F. P. (1986). No silver bullet. In H.-J. Kugler, editor, *Proceedings of the IFIP Tenth World Computing Conference*, pages 1069–1076.

Bunge, M. A. (1998). *Social Science under Debate: A Philosophical Perspective*. University of Toronto Press, Toronto.

Burrell, G. and Morgan, G. (1979). *Sociological Paradigms and Organizational Analysis.* Heinemann, Exeter, NH.

Casti, J. (1989). *Paradigms Lost.* Avon Books, New York, NY.

Chang, M. K. and Woo, C. C. (1994). A speech-act-based negotiation protocol: Design, implementation, and test use. *ACM Transactions on Information Systems*, **12**(4), 360–382.

Checkland, P. and Holwell, S. (1998). *Information, Systems and Information Systems: Making Sense of the Field.* John Wiley & Sons, Chichester, West Sussex.

Dahlbom, B. and Mathiassen, L. (1993). *Computers in Context: The Philosophy and Practice of Systems Design.* Blackwell Publishers, Oxford.

Duhem, P. (1954). *The Aim and Structure of Physical Theory.* Princeton University Press, Princeton. Translated by P. Wiener.

Eriksson, H.-E. and Penker, M. (2000). *Business modelling with UML: Business patterns at work.* John Wiley & Sons, New York, NY.

Evans, J. (1988). *Structure of Discrete Event Simulation: An Introduction to the Engagement Strategy.* Elis Horwood Ltd, Chichester.

Fishwick, P. (1995). *Simulation Model Design and Execution: Building Digital Worlds.* Prentice-Hall, Englewood Cliffs, NJ.

Floyd, C. (1984). A systematic look at prototyping. In R. Budde, K. Kuhlenkamp, L. Mathiasssen, and H. Zullighoven, editors, *Approaches to Prototyping.* Springer Verlag, Berlin.

Hammer, M. and Champy, J. (1994). *Reengineering the Corporation: A Manifesto for Business Revolution.* Allen & Unwin, St. Leonards, NSW.

Hirschheim, R. and Klein, H. (1989). Four paradigms of information systems developement. *Communications of the ACM*, **32**(10), 1199–1216.

Hirschheim, R. and Newman, M. (1991). Symbolism and information systems development: Myth, metaphor and magic. *Information Systems Research*, **2**(1), 29–62.

Hirschheim, R., Klein, H., and Lyytinen, K. (1995). *Information Systems Development and Data Modeling: Conceptual and Philosophical Foundations*. Cambridge University Press, Cambridge.

Iscoe, N., Williams, G. B., and Arango, G. (1991). Domain modeling for software engineering. In *Proceedings of the 13th International Conference on Software Engineering ICSE 91, Austin, TX*, pages 340–343.

Jackson, M. (1995). The world and the machine. In *Proceedings of the 17th International Conference on Software Engineering ICSE 95, Seattle, WA*, pages 283–292.

Janson, M. A., Woo, C. C., and Smith, L. D. (1993). Information systems development and communicative action theory. *Information & Management*, **25**, 59–72.

Jarke, M. (1998). Requirements tracing. *Communications of the ACM*, **41**(12), 33–36.

Kendall, J. and Kendall, K. (1993). Metaphors and methodologies: Living beyond the systems machine. *MIS Quarterly*, **17**(2), 149–172.

Krogstie, J., Lindland, O. I., and Sindre, G. (1995). Towards a deeper understanding of quality in requirements engineering. In *Proceedings of the CAISE'95 Conference*, pages 82–95.

Kruchten, P. (2002). *The Rational Unified Process: An Introduction*. Addison-Wesley, Reading, MA.

Kuhn, T. (1996). *The Structure of Scientific Revolutions*. The University of Chicago Press, Chicago, third edition.

Kuhn, T. S. (1977). Objectivity, value judgment, and theory choice. In *The Esssential Tension: Selected Studies in Scientific Tradition and Change*. University of Chicago Press, Chicago, IL.

Lakatos, I. (1978). *Philosophical papers*. Cambridge University Press, Cambridge, NY.

Larman, C. (2002). *Applying UML and patterns: An introduction to object-oriented analysis and design and the unified process*. Prentice-Hall, Upper Saddle River, NJ.

Laudan, L. (1990). Demystifying underdetermination. In C. W. Savage, editor, *Scientific Theories, Vol. 14*. University of Minnesota Press, Minneapolis, MN.

Laudon, L. (1981). A confutation of convergent realism. *Philosophy of Science*, **48**, 19–49.

Longino, H. E. (1990). *Science as Social Knowledge: Values and Objectivity in Scientifi Inquiry*. Princeton University Press, Princeton, NJ.

Moody, D. and Shanks, G. (1994). What makes a good data model? evalutating the quality of entity relationship models. In *Proceedings of the 1994 International Conference on Conceptual Modelling ER'94*, New York, NY. Springer Verlag.

Moody, D. and Shanks, G. (1998). Improving the quality of entity relationship models - experience in research and practice. In *Proceedings of the 1998 International Conference on Conceptual Modelling ER'98*, New York, NY. Springer Verlag.

Mylopoulos, J. (1992). Conceptual modeling and Telos. In P. Locuopoulos and R. Zicari, editors, *Conceptual Modeling, Databases and Cases*. John Wiley & Sons, Inc, New York et. al.

Offen, R. (2002). Domain understanding is the key to successful system development. *Requirements Engineering*, **7**, 172–175.

Popper, K. (1968). *The Logic of Scientific Discovery*. Harper & Row, New York, NY.

Popper, K. (1972). *Objective Knowledge*. Clarendon Press, Oxford.

Porter, M. (1980). *Competitive Strategy: Techniques for Anlyzing Industries and Competitors*. The Free Press, New York, NY.

Quine, W. v. O. (1953). Two dogmas of empiricism. In *From a Logical Point of View*. Harvard University Press, Cambridge, MA.

Raymond, E. S. (2001). *Cathedral and the Bazaar*. O'Reilly, Sebastopol, CA.

Reed, M. (1992). *The Sociology of Organisations: Themes, Perspectives and Prospects*. Harvester Wheatsheaf, Hemel Hempstead, Hertfordshire.

Riddle, W. (1984). Advancing the state of the art in software system prototyping. In R. Budde, K. Kuhlenkamp, L. Mathiasssen, and H. Zullighoven, editors, *Approaches to Prototyping*. Springer Verlag, Berlin.

Scheer, A.-W. (1994). *Business Process Engineering. Reference Models for Industrial Enterprises*. Springer Verlag, Berlin, 2nd edition.

Schütte, R. and Rotthowe, T. (1998). The guidelines of modeling - an approach to enhance the quality in information models. In *Proceedings of the 1998 International Conference on Conceptual Modelling ER'98*, pages 240–254.

Snyder, L. J. (1994). Is evidence historical? In P. Achinstein and L. J. Snyder, editors, *Scientific Methods: Conceptual and Historical Problems*. Krieger Publishing Company, Malabar, FL.

Sotirovski, D. (2001). Heuristics for iterative software development. *IEEE Software*, **18**, 66–73.

Thagard, P. R. (1998). Why astrology is a pseudoscience. In M. Curd and J. Cover, editors, *Philosophy of Science – The central issues*. W.W. Norton and Company, New York, NY.

van Fraassen, B. C. (1980). *The Scientific Image*. Clarendon Press, Oxford, UK.

Wand, Y. and Weber, R. (1993). On the ontological expressiveness of information systems analysis and design grammars. *Journal of Information Systems*, (3), 217–237.

Winograd, T. and Flores, F. (1986). *Understanding Computers and Cognition: A New Foundation for Design*. Addison Wesley, Reading, MA.

Yu, E. (2001). Agen orientation as a modelling paradigm. *Wirtschaftsinformatik*, **43**(2), 123–132.