An Exploratory Study Comparing the Core Concepts of Information Systems Development and Software Engineering

Nik R. Hasan  
*University of Minnesota Duluth*, nhassan@d.umn.edu

Lars Mathiassen  
*Georgia State University*, lmathiassen@gsu.edu

Follow this and additional works at: [http://aisel.aisnet.org/amcis2008](http://aisel.aisnet.org/amcis2008)

Recommended Citation  
[http://aisel.aisnet.org/amcis2008/56](http://aisel.aisnet.org/amcis2008/56)
An Exploratory Study Comparing the Core Concepts of Information Systems Development and Software Engineering

Nik R. Hassan  
University of Minnesota Duluth  
nhassan@d.umn.edu

Lars Mathiassen  
Georgia State University  
lmathiassen@gsu.edu

ABSTRACT

The goal of this study is to apply a multidisciplinary approach towards the discovery of core concepts in the art and science of design. This study advances the intellectual body of knowledge for design science by uncovering common areas of agreement between information systems (IS) and computer science (CS) encouraging the development of new design theories within each individual field. This research avoids the trap of finding “yet another methodology” by merging the two dichotomous paradigms of design-as-natural-science and design-as-human-science, and by viewing the common concepts from these approaches through various philosophical lenses. These philosophical lenses ensure that the foundations for art and science of design will be capable of explaining the laws and theories of design and not merely reproduce a set of rules and procedures.

Keywords

Information systems development (ISD), software engineering, core concepts, design science, philosophy of ISD

INTRODUCTION

Despite numerous efforts, computing-related fields have continued to struggle to perfect the craft of designing software-intensive systems (Neumann, 2007; Standish Group International, 1994). One such pioneering effort was a NATO conference in 1968 that coined the term “software engineering” as a rallying cry for formalizing the craft of software development (Naur & Randell, 1969). The conference exposed the weak foundations software engineering stands on, and the absence of theory that it suffers to the present day. For example, the 50 or so experts in software-related fields that attended the conference could not agree on the distinction between the design of software and its production. Today, experts still cannot agree on such foundational concepts. According to Hevner et al. (2004), the knowledge base of design science— theories, frameworks, instruments, constructs, models, methods and instantiations—is the critical input into the process of building and evaluating designs. These foundational elements remain inadequate, lack theoretical bases and are “often insufficient for design purposes … designers must [therefore] rely on intuition, experience, and trial-and-error methods” (p. 99).

The goal of this study is to apply a multidisciplinary and philosophical perspective towards the establishment of the foundations for the science of design. This is performed by examining the core concepts of design as applied in two design-related fields, computer science (CS) and information systems (IS). By extracting the core design concepts from these fields, this study accomplishes two major objectives: (1) advancing the intellectual body of knowledge for design science by uncovering common areas of agreement and disagreement among the two fields that can be used to generate new design theories; and (2) discovering existing theories, constructs, models, methods and tools of high value that can be of immediate practical use for designers.

THE NEED FOR MULTI-DISCIPLINARY AND PHILOSOPHICAL APPROACHES

The Complexity of Design in Software-Intensive Systems

The process of design and systems development is similar to the process of building a house, only immeasurably more complex. McPhee (1996) lists general characteristics of design as (1) it starts with a need (a motivation for design) and requires an intention, (2) it involves some kind of transformation of a form that can be used to guide the implementation of an artifact, plan or process, (3) it involves a generation of new ideas (an element of creativity), (4) it satisfies a set of internal and external constraints derived from “functional and performance specifications of the artifact, limitations of the medium and process by which the artifact is rendered or produced and aesthetic criteria on the form of the artifact” (Mostow, 1985).
(4) it is a problem solving and decision making process, (5) the results of which becomes a scheme for implementing an artifact (a blueprint), and (6) it always involves diversity and evolution (change). The formalists in design base their work on theoretical foundations of mathematics and computing science while the pragmatists will use any other methods beyond the formal techniques as long as they work. Both approaches attempt to capture the complex conceptual constructs and somehow manage the transformation of these constructs into physical systems. Regardless of their inclinations, proponents of both approaches agree that software design is a human endeavor and designers will need to interact with users to determine the suitability of their design at some point in their processes.

General design theory can also be categorized into either the “scientific” approach (natural science) or the social approach (human science) to problem solving (McPhee, 1996). The first often involves a rational, logical and analytical way of design, whereas the second is described as an artistic, intuitive and idiosyncratic approach to design. The scientific method of design is less likely to result in a creative design because there is little collaboration between the skills of the designer and the vision of the customer, but does offer many advantageous especially in relation to its “divide and conquer,” organized, systematic and disciplined decomposition of the larger, complex problem. The humanistic and social approach emphasizes the economic, psychological, sociological, organizational, philosophical, political and aesthetic issues surrounding software design. This approach acknowledges the interpretive processes that take place as software is developed, the flexibility and adaptability required in the rapidly changing and unstable tasks, process and environments and the recent changes overtaking the software industry such as the demands placed by the ubiquity of computers in society. Other reasons for choosing the humanistic approach includes better handling of the development process itself. Because constantly changing requirements is the nature of almost every software design project, the formalistic method is unable to handle the highly interactive, interleaved and loosely ordered tasks of software design. The humanistic approaches acknowledge that as social beings, human designers cannot be objective as in natural sciences because they are themselves a part of the design process, contributing their idiosyncrasies to the process of design. The process of design is seen as an interpretive process where meaning is discovered and agreed between human agents, and where meaning is understood in the context in which they are interpreted. This context includes the anticipated meaning or preconceptions that are carried by the participant in the design process and the learning process that takes place as a result of understanding the goals, constraints and requirements. Other authors in the information systems field that have written on the humanistic approach to software development include Boland et al. (1982) Lyytinen (1987) and Lee (1991).

The two different scientific and humanistic approaches can be found in different forms within the practices of two different fields, computer science and information systems. A brief discussion of how design is performed in these two fields follows:

**Design in Computer Science**

Much of the existing design science research in the CS field employs the engineering paradigm, hence, the term “software engineering.” The engineering paradigm seeks to apply mature scientific knowledge to solve technological problems. Hundreds of years of efforts between the natural sciences and craftsmanship made possible the highly successful field of civil engineering (Finch, 1951). This engineering field created its own repertory of concepts and rules by taking advantage of the mature analytical approaches and mathematics of the science of statics and strength of materials to create a highly successful and respectable discipline (Straub, 1964). The focus of the paradigm underlying civil engineering is therefore the reuse of existing knowledge and rules to solve problems faster than they otherwise could, in a primarily routine fashion. Such a paradigm does not always suit the requirements of software development. First, a mature knowledge base does not exist in software development, and even if it did, computer scientists seldom share that knowledge between themselves or between software projects (Shaw, 1990). Second, the bread and butter of engineering design is the reuse of prior solutions to solve familiar problems, whereas software development is a “wicked” problem (Rittel & Webber, 1973) that suffers from incomplete, contradictory, and changing requirements (Markus, Majchrzak, & Gasser, 2002). Such a problem requires an approach that produces innovative designs rather than routine designs.

**Design in Information Systems**

Much of the design science research in IS employs the behavioral-science paradigm, which treats technology as a black box (Hevner et al., 2004; Orlikowski & Iacono, 2001). This paradigm produces results that are often equivocal because they often ignore the technological component. On the other hand, this behavioral-science paradigm produces valuable insights into the environment of design science. This environment consists of the roles that people play, their capabilities and characteristics; and the link between the design and the strategy of the organization, its structure, culture and processes (Hevner et al., 2004). These elements contribute a large part to the relevance of the design, and without their consideration, may cause the design to fail.
A Synthesis of the Two Approaches

What is required for a solid foundation in design science is a merging of these two paradigms of design-as-natural-science versus design-as-human-science into an integrated foundational base that circumscribes both the engineering and human sciences. Design science needs to combine the rationalistic and formalistic approaches of the former with the more artistic, intuitive and idiosyncratic approach of the latter because often interpretive and constructivist approaches adapt well to rapidly changing and unstable tasks, process and environments.

RESEARCH METHODOLOGY

The research question this study seeks to answer is: What are the core concepts in the information systems development (ISD) and the software engineering (SE) literature and how do these compare in terms of focus and semantics? To answer this research question this study compares the core concepts discussed by the top authors in both fields. To extract the core concepts, this study refers to the field’s classic articles. Articles are considered classics because they have been cited at least four times a year since their publication (de Solla Price, 1963). This frequency of citations suggests that because the scholars in the field cite them often, they agree on the significance of those publications.

Garfield (1955) invented the science of citation analysis or scientometrics to reduce or eliminate citations of fraudulent, incomplete, or obsolete data by examining the papers that cited the data, instead of searching for papers that follow the original data. By listing out the papers that cited the data (a citation index) a new approach to controlling and analyzing scientific literature became possible. One of the major developments stemming from this methodology is its contribution to the sociology of science. It became possible for sociologists of science to predict who would be awarded the noble prize based on the number of times scientists were cited (Garfield, 1970a; Garfield & Malin, 1968), and to uncover core concepts that these scientists were inventing and using. For example the core concepts that led to the discovery of DNA were uncovered with the help of citation analysis (Garfield, Sher, & Torpie, 1964).

Bibliographic Coupling and Core Concepts

To find out if the top authors agree on the concepts they are discussing, a citation analysis method called bibliographic coupling is deployed. This method measures the relationship between two publications. If two articles cite the same reference in the same context, the two articles can be said to agree on the concept represented by the reference. If scores of articles cite the same reference in the same context, that concept can be called a core concept.

Because citing documents do not necessarily pinpoint the concept that is being cited, we use Small’s (1973; 1978) extension of Garfield’s citation analysis to link the cited authors to the concepts they are communicating. The citing document provides an unambiguous reference to a word, phrase, sentence or other units of text connected to a cited document that is embedded in its text. This unambiguous reference relates the concept which the citing document is discussing with the concept the cited document offers. In research, the cited concept provides meaning to the author’s text. At the same time, the author is imparting meaning to the sources by citing them. For example, when an article explains “supply chain management systems,” the author may decide to use concepts from Porter’s (1980) value chain analysis. In other words, the author is saying that value chain analysis is related to supply chain management. By identifying such linkages, this method extracts the concepts the authors intended to communicate because it constitutes the authors’ interpretation of the cited work.

It is important to distinguish between citation analysis methods such as bibliographic coupling and other quantitative methods that use keywords to analyze agreement on certain concepts. The problem with using keywords such as word or subject indexing and content analysis is that often the exact words or terms are not mentioned in the publication, even when the publication discusses a similar concept (Garfield, 1955). For example, many of the classic publications that led to the discovery of insulin did not even mention the term “insulin.” However, the phrase “internal secretions of the pancreas,” which was used by the scientists that discovered insulin described the same concept (Banting & Best, 1922). Other concepts that linked diabetes mellitus to pancreatic defect was known to scientists 30 years before insulin was invented (Garfield, 1970b). Citation analysis resolves this problem by linking the concept written in a publication to similar concepts written by other researchers regardless of what terms or words are used. Hence this study’s methods provide the best way of analyzing core concepts because it combines both citation linking and context analysis. Google’s PageRank system uses the same principles to deliver relevant search results for Internet users (Page, Brin, Motwani, & Winograd, 1998).

Bibliographic coupling is closely related to but not the same as co-citation analysis deployed by Culnan (1986; 1987; 1986) in the IS field. Like bibliographic coupling, co-citation analysis attempts to find relationships between two publications or articles even when they don’t cite each other. However, co-citation analysis measures the strength of the relationship using the number of citing works that cite two related works. Consequently, co-citation analysis monitors the frequency of
citations over time and measures changes in research focus and relationships. Bibliographic coupling measures the strength of the fixed relationship between two or more citing works that use similar references. The more similar their references, the stronger are their relationships (White & Griffith, 1981).

Bibliographic coupling and Small’s (1978) linking of concept to citations provide a way of finding the core concepts in any field, especially when applied to its classic publications. These classics contain the concepts that form the core of that field. For example, the field of psychology can be represented by the core concepts developed by researchers such as Freud, James, Holt, Piaget, Skinner, Cannon, Bandura and Rogers (Haggbloom, 2002). The concepts in psychology continue to be represented in a large extent by the same core topics that psychologists consider important (Griggs & Jackson, 1996; Griggs & Mitchell, 2002; Webb, 1991). Similarly, in sociology, a survey of 301 sociologists agreed on a list of core terms that represented the field and the concepts that they deem should be addressed systematically in their research as well as in their introductory courses (Babchuk & Keith, 1995; Wagenaar, 2004).

Pool of Classic Articles in ISD and SE

Bibliographic coupling begins by choosing a fixed set of articles that are considered classics. In order to be as representative of the field as possible, we choose the top journals from both fields. From IS, the following five journals based on the top six selected by the Association for Information Systems (2007) as their top journals form the source of the classic articles in IS. Articles from these journals were collected using the Web of Science database. The Journal of the AIS was not selected because it was not indexed by the Web of Science at the time of this writing.

1. European Journal of Information Systems
2. Information Systems Journal
3. Information Systems Research
4. Journal of MIS
5. MIS Quarterly

For the field of computer science, we selected the following journals that are top SE journals as agreed by CS scholars (Glass & Chen, 2005; Ren & Taylor, 2007; Tse, Chen, & Glass, 2005; Wohlin, 2008):

1. IEEE Transactions on Software Engineering
2. Information and Software Technology (IST)
3. Journal of Systems and Software (JSS)
4. Software Practice and Experience (SPE)
5. Software (SW), IEEE

<table>
<thead>
<tr>
<th>Organizational Alignment</th>
<th>IS evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements construction</td>
<td>IS use</td>
</tr>
<tr>
<td>User interface design</td>
<td>IS maintenance and evolution</td>
</tr>
<tr>
<td>Architectural design</td>
<td>Project organizing</td>
</tr>
<tr>
<td>Database Design</td>
<td>Supplier management</td>
</tr>
<tr>
<td>Software Design</td>
<td>People management</td>
</tr>
<tr>
<td>Design of user support system</td>
<td>Method management</td>
</tr>
<tr>
<td>Design of system controls and monitors</td>
<td>Risk management</td>
</tr>
<tr>
<td>IS testing</td>
<td>Performance management</td>
</tr>
<tr>
<td>IS implementation and acceptance</td>
<td>Software configuration management</td>
</tr>
<tr>
<td></td>
<td>Quality assurance</td>
</tr>
</tbody>
</table>

Table 1: Iivari et al's (2004) ISD Coding Terms

Because the IS journals are not specific to ISD, we identify ISD articles by performing a qualitative analysis (using NVIVO) on the titles and abstracts of these journals based on ISD codes supplied by Iivari et al. (2004) shown in Table 1. This process is not required for the CS journals because they are all SE journals. Having identified all articles in both IS and CS that are relevant to ISD and SE respectively, we use the Web of Science citation system to identify the most cited articles in
each field. These articles represent the articles that both fields agree are representative and authoritative of ISD in their respective fields. Using Small’s (1978) technique to identify concepts that these top articles study, we identify the top ISD concepts from each field.

RESEARCH RESULTS

We observe several interesting preliminary results. The five top journals in IS produced 166 articles that were cited at least 10 times or more. Iivari et al.’s (2004) coding terms included the term “acceptance.” The use of this coding term captured over 20 articles that referred to the Technology Acceptance Model (TAM) or user acceptance. These articles are not categorized as ISD articles because they mostly studied adoption-related issues rather than systems development or design. The qualitative analysis of the remaining set of articles using Iivari et al.’s (2004) coding terms resulted in only 14 articles that were cited more than 40 times and therefore qualify as a super classics (de Solla Price, 1963; Walstrom & Leonard, 2000) in ISD. These articles are shown in Table 2.

<table>
<thead>
<tr>
<th>IS Articles</th>
<th>Times Cited</th>
</tr>
</thead>
</table>
Several terms from the Iivari et al’s (2004) list did not appear in any of the titles or abstracts of the IS articles. These terms include “evolution,” “project organizing,” “supplier management,” and “configuration management.” Several articles that can be considered ISD articles were not identified by Iivari et al’s (2004) list including articles on business process reengineering, process redesign and usability. Such omissions suggest that we have not identified an exhaustive body of concepts that we can consider as ISD. The terms “testing” and “quality” only appeared once each time.

The citation analysis of CS articles from the five CS journals produced over 200 articles that were cited more than 10 times. The most highly-cited SE articles in CS received many more cites than articles from IS. Fourteen articles in CS were cited at least 140 times or more (Table 3).

<table>
<thead>
<tr>
<th>CS Articles</th>
<th>Times Cited</th>
</tr>
</thead>
</table>
The SE articles from computer science appear to be much more focused on development and design than articles in IS. It is little wonder that our own practice of ISD borrows so much from CS. A cursory review of the terms in the titles of the articles suggests that the concepts studied in IS differs greatly from the concepts studied in CS. In fact there appears to be very little overlap in terms of concepts. As suggested by Iivari et al. (2004), the related body of knowledge of ISD focuses more on the organizational contexts in which the ISD is being performed. Although Iivari et al., suggest five knowledge areas of ISD: technical knowledge, application domain knowledge, organizational knowledge, IS application knowledge and ISD process knowledge, the ones that are highly-cited are only articles that address organizational knowledge, process knowledge (especially user participation and involvement), and a few on technical knowledge (CASE tools and usability). This five knowledge areas of ISD may need to be reevaluated based on the findings of this research.

A cursory survey of CS articles reveals a much richer set of concepts for SE. As expected from the engineering paradigm of computer science, the most common core concepts are measurement related, including measures of complexity of software, design measures, software estimation and effort prediction. Another common core concept that is mentioned by Iivari et al. (2004), but missing in the set of IS articles are architectural and configuration management and evaluation as well as development methods such as object-oriented paradigms in software development. In terms of life cycle phases, IS articles tend to focus on the ends of the phases, the earlier phases of planning and analysis as well as the latter implementation phases, whereas CS articles focus more on the middle phases of design, more on modeling and testing.

### Table 3: The Most Cited Articles in Software Engineering

<table>
<thead>
<tr>
<th>Page</th>
<th>Authors</th>
</tr>
</thead>
</table>

### Table 4: Cross-Fertilization of Core Concepts in IS and CS

<table>
<thead>
<tr>
<th>Core Concepts in IS Development</th>
<th>Core Concepts in CS Software Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Technology Fit</td>
<td>Complexity and measurement, requirements analysis, software metrics, software modeling</td>
</tr>
<tr>
<td>Handling incremental and radical changes/process reengineering</td>
<td>Software improvement, object-oriented development, reverse engineering</td>
</tr>
<tr>
<td>User participation and involvement</td>
<td>Versioning and implementation</td>
</tr>
<tr>
<td>Usability and performance</td>
<td>Software performance metrics</td>
</tr>
<tr>
<td>Strategic planning and alignment</td>
<td>Architectural design, configuration management, fault tolerance and distributed design</td>
</tr>
</tbody>
</table>

In terms of commonalities, both CS and IS lack emphasis on modeling whether it is about traditional modeling or newer models such as UML. Several articles on UML that were less cited were found in the CS set of articles, but none at all in the IS set. Table 4 shows the core concepts from both fields, matched according to the subject matter suggested by each concept. By qualitatively evaluating each pair of matching concepts, IS researchers can partner with CS researchers in interdisciplinary work and cross-fertilization studies. This cross-fertilization can lead to a synthesis of the different approaches from the two fields that circumscribe both the engineering and human sciences. For example, a topic which is currently being discussed by SE authors is the issue of planning for change. This topic is not a “traditional” SE topic, but CS authors have begun investigating ways in which software can be designed to be more flexible and open to changes (Germain & Robillard, 2005). IS authors have not “traditionally” addressed such topics as well. However, there is ample work in IS on organizational change and how technology could be designed to “fit” such change. Both fields can benefit from cross-disciplinary work in this area of study by combining the engineering and humanistic perspectives.
CONCLUSION AND FUTURE RESEARCH

This study performed a preliminary analysis of the most-cited articles in both IS and CS to uncover differences and similarities in the conceptual development of the science of design. Although the results are preliminary, they indicate vast differences between the two fields in terms of focus and emphasis. The results also confirm previous studies indicating that the IS field has not made much progress in systems development. A potentially significant contribution of this study is the vast amount of cross-disciplinary work and cross-fertilization that can be identified between development and design in IS and in CS, which will be very valuable towards improving systems development in general. This research will continue with a more detailed analysis using bibliographical citation methods. Using this method, articles that cite the list identified in this study will be collected in order to further isolate the concepts represented by these highly-cited articles. Consequently, it will be possible to uncover any new concepts that each field has forged within their respective design science domains.

REFERENCES