2001

Reuse in Information Systems Development: Classification and Comparison

Danny Brash
Stockholm University, danny@dsv.su.se

Reuse in Information Systems Development: Classification and Comparison

Danny Brash

Department of Computer and Systems Sciences
Stockholm University, Sweden
danny@dsv.su.se

Abstract

There has been a trend in recent years towards an increasing acceptance of reuse as an approach to information systems development. This trend however has not been accompanied by an understanding of the underlying implications of this approach for the development process. We attempt to briefly highlight these affects by an initial categorization of the reuse approach according to two alternative taxonomies for reuse techniques followed by three philosophical frameworks from the literature. The affects on the functional property of reusability is then discussed leading to some initial conclusions regarding the application of reuse in information systems development.

Keywords

FA IS Development Strategies, FB IS Life Cycle Activities, FB03 Information Requirements Determination, FC Is Development Methods And Tools

INTRODUCTION

The trend in recent years has been towards systems development approaches, methodologies, methods and techniques based on some forms of the fundamental principles of object orientation, abstraction, encapsulation and inheritance. The rationale for object orientation is the belief that reuse in information systems development would increase efficiency, productivity and reliability of information systems. While object orientation has is perhaps the most popular approach to information systems development, reuse has in practice lagged behind. It could be argued that as a result of this situation, reuse itself has recently begun to appear as an approach towards information systems development, partially independent of object orientation. It is this emerging approach called reuse that is the topic of this paper.

BACKGROUND

As with all attempts with finding the "silver bullet" for Information Systems Development it must be recognized that each development situation is complex and often unique. Rather than throwing each new idea out to be replaced with a new one, it must preferable to understand and then take parts that appear to be relevant. In the case of information systems methodologies Hirscheim et al. 1998 state,

> *"In such a state(a methodology jungle), it is clear that the highest priority should not lie in developing new methodologies but in better understanding the existing stock and the collective methodology knowledge embedded in them."*

As noted by Fitzgerald (1999) practice has often led theory ISD. A wider understanding of reuse should assist developers in making choices concerning approaches, methodologies and techniques in ISD. At the same time, another academic motivation besides explaining practice is the more fundamental need to understand existing ISD practices rather than adding new methodologies, techniques and tools to the large existing collection (Fitzgerald 1997), (Bubenko and Wangler 1992).

According to Cater-Steel and Ah-Fock (2000) the main research issues concerning reuse in systems development are economic, performance and technology. The technology research has strongly focussed on functionality. Algorithms, languages, methods and techniques and tools have been developed without there being any evidence of more than sporadic success. There is a need to go beyond the cliche, "avoid reinventing the wheel", that implies that reuse is in principle always beneficial. We need to take a step back and understand the nature and philosophy of reuse before we can solve any problems needing functional solutions. Instead of providing a solution, the purpose of this paper then is to improve understanding of reuse in information systems development. This should assist developers and researchers in making informed decisions concerning reuse in information systems development.

METHODOLOGICAL INFLUENCES

The aim of this paper is then to bring some clarity and systemic understanding to reuse in information systems development. This will be achieved by using a series of classifications that are described in more detail later in this paper. These classifications should allow for some generalizations to be made concerning diverse reuse techniques.

This work has been influenced by two methodological/philosophical understandings. One of the influences, soft systems thinking, relates the need to elucidate problem situations in ill-defined domains rather than seek solutions for pre-defined problems. We will attempt then to understand the situation rather than solve a problem. This however only constitutes part of the rationale for using Soft Systems Methodology (SSM). We will not be using SSM itself since we are not looking at "Human Activity Systems" in a strict sense, nor are we to engage in action research, two of the key elements of applying SSM, (Checkland 1999).

Similarly to SSM, the second major influence, "critical systems thinking" is subjectivist in nature. With critical, we do not mean criticism as such but rather a questioning of the implicit assumptions underlying certain accepted phenomena, being reuse in this case.

Thirdly, since this work is based on understanding reuse techniques, the data used will be qualitative in nature, comprise of understanding and interpretations of literature deemed to be representative of the particular technique.

WHAT IS REUSE?

Reuse in ISD can be described as; *The employment of a previously used and explicitly defined systems development artifact in another information systems development* process. There is also a key differentiation between development for reuse and development by reuse. Reuse is accomplished by firstly developing reusable artifacts, i.e. *intentional reuse*. This can be compared to the reuse of artifacts in an arbitrary unplanned way i.e. *ad hoc reuse*. In this paper, we will be looking at intentional reuse.

An artifact is an item that is employed in the ISD process such as an analysis model, a requirements specification or a piece of software code. A reuse artifact is intentionally developed for reuse. This then precludes completed and independent products/applications, such as word processing applications that are part of a larger applications. The types of artifacts for reuse in ISD has grown from initially being only software, then through object orientation, to currently include artifacts in other phases of ISD than just implementation.

Reuse efforts within information systems development originated in the software engineering community with its emphasis on formality and rationality. The aim has always been to make software development a science, concise and measurable. Part of this effort is the belief that software could be considered to be building blocks, that could stored in a box for future retrieval and (re)use. These fundamental understandings have led to some success, encouraging the belief that this could also be achieved in other domains (Kelly and Whittle 1995). After all who would argue with the well worn adage, "avoid reinventing the wheel".

CLASSIFICATION AND COMPARISON

Given that innumerable methodologies, methods, techniques, models and languages within information systems development have been created over the years, there has also been a parallel development in the attempts to compare and categorize these. This has been done with the objective of bringing clarity that could allow for some degree of rational choice amongst comparable alternatives as well as that of trying to understand the significance of specific contributions to system development projects.

In this paper, we will attempt to classify reuse in two dimensions, the technique oriented and philosophical. When IS development is discussed in terms of comparisons, it is common that the unit of reference is "methodology". When discussing reuse however, there are no explicit reuse methodologies. Explicit processes for reuse exist though in a series of techniques. These techniques together with aspects of object orientation imply that there are general principals or a philosophy of reuse. These techniques as well as philosophical perspectives are discussed in this section.

It should be emphasized that this work deals with applying reuse techniques to create and use artifacts in systems development. It will not deal with another major issue in reuse; that of compatibility between reusable artifacts. Hence, standards such as CORBA or UDDI will not be discussed. Another area which is not included in this work is "software process improvement", since the focus here is not only on reusing software, but also models, specifications etc throughout the development process.

Taxonomies of techniques

In this section, we will briefly describe two alternative taxonomies for describing and using reuse techniques. The taxonomy, in Table 1 below, should be useful for choosing a technique that matches an ISD phase in the form of a reusable artifact. The literature cited here is not exhaustive. It is rather an attempt to use literature that, where possible is well cited, or gives an overview of the functionality of the techniques. The techniques chosen have been done so on the basis of the following criteria: 1) They are explicitly described by their proponents as intended for reusing artifacts and 2) they are related to a stage(s) in systems development.

Phase / **Technique**	**Authors**	**Planning**	**Analysis**	**Design**	**Implementation**
Requirements Pattern	(Robertson, Robertson 1999)	Requirements specification	Use case model		
Domain Engineering	(Sutcliffe, Maiden 1994)		Domain analysis Domain model		
Analysis Pattern	(Fowler 1997)		Data model OO model		
Component based Development	(Allen, Frost 1998)		Component based business model	Components in framework	Software components
Design Pattern	(Gamma, et al. 1994)			Software Design	
Framework	(Johnson 1997)			Software Architecture	Software framework

Table 1: Reuse artifacts according to reuse technique and ISD phase.

In Table 1, the extent to which each technique covers the phases of systems development shows that component-based development includes more phases than any other technique. Even though a combination of requirements patterns, analysis patterns and design patterns would appear to constitute a common set of techniques in ISD, this is not the situation as they do not present a coherent methodology. The taxonomy in Table 2 below gives an overview of how each reuse technique is represented. As can be seen, there is no predominant representation method.

Representation Form / **Technique**	**Authors**	**Text**	**Conceptual Model**	**Software Code**
Requirements Pattern	(Robertson, Robertson 1999)	Requirements Specification	Use Case	
Domain Engineering	(Sutcliffe Maiden 1994)	Domain analysis	Domain model	
Analysis Pattern	(Fowler 1997)		Data Models OO Models	
Component	(Allen Frost 1998)		Business model	Software Component
Design Pattern	(Gamma, et al. 1994)	Pattern template		Software Code
Framework	(Johnson 1997)			Software Code

Table 2: Reuse techniques and representation forms

It is worth noting that there maybe a third type of taxonomy that differentiates reuse techniques. This maybe based on reuse techniques originating in differing research and professional communities. Each of these uses different frames of references and may be guided by different paradigms. One such grouping (software reuse, components) would be based on software engineering principles, where formalism documentation and measurement are key principles. Another grouping (design patterns, analysis patterns) is that of techniques based on the concept of patterns as developed by C.Alexander. The third grouping (requirements patterns, domain engineering) is one that is concerned with IS requirements determination. This taxonomy is not in the scope of this work, even if each grouping exists in terms of a different ISD paradigm and is a result of a different historical development within ISD.

PHILOSOPHICAL UNDERSTANDINGS OF REUSE

In this section, we briefly look at classifications of reuse in ISD in terms of three philosophical views of ISD taken from the literature; "themes" "approaches" and "paradigms" and see how reuse is related to these. These three all deal with categorizing ISD methodologies at a higher abstraction level. While "themes" are used to describe functional combinations of methodologies, "approaches" and "paradigms" relate more to philosophical similarities between methodologies or techniques. The latter two can be seen perhaps as a reaction to a perceived lack of progress described by Wynekoop and Russo (1993), of the former, i.e. comparisons on the basis of methodologies and techniques.

The latter comparisons, began to develop from the late 80's and are presented in Hirschheim and Klein (1989), Hirschheim et al. (1998), Iivari and Hirschheim (1996), Iivari et al. (1998), Iivari and Maansaari (1998) Iivari (1999), Iivari et al. (1999) and Iivari (2000). These articles have discussed aspects of classification of methodologies according to approaches, paradigms and/or philosophies. Hence similarities have been found according to underlying assumptions rather than common features

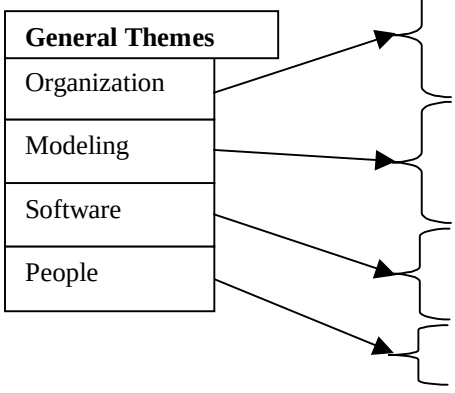
In the book "Information Systems Development, Methodologies, Techniques and Tools", Avison and Fitzgerald 1998 provide mainly a brief résumé of a number of ISD methodologies, techniques and tools based on essentially functional or procedural "themes".

Themes

In Avison and Fitzgerald (1998) the authors describe ISD in as being composed hierarchically as;

General themes → Themes → Methodologies → Techniques → Tools.

The components at each level are not mutually exclusive to specific components at a higher level. For example a specific technique may in practice be used in several methodologies. At the highest abstraction level, the following (Table 3) specific themes and general themes are presented. These themes are seen to have been in response to the failures of the traditional Systems Development Lifecycle (SDLC). In this section, we will relate reuse to these themes.



Specific Themes
The Systems Approach
Strategic information systems
Business process engineering
Planning approaches
Modeling
Process Modeling
Data Modeling
Object oriented systems development
Automated tools for information systems development
Prototyping
Software engineering and formal methods
Participation
Expert Systems

Table 3: Themes in ISD (Avison Fitzgerald 1998)

In terms of the themes in Table 3 above, the arguments often put forward for reuse are twofold; 1) economical, that is organizational resources do not need to be spent redoing the same activities and 2) reuse is an application of sound software engineering principles. In terms of the general theme of modeling, reuse is traditionally associated with object orientation firstly as a programming paradigm and more recently in terms of an approach to analysis and design in ISD particularly through the use of Unified Modeling Language (UML). Hence, reuse can be explained in terms of three of the general four themes in Table 3. However, reference in the literature to the fourth general theme, "people" is scarce.

The implications of the reuse approach not taking into consideration the "people" theme and in particular participation and user requirements, are of course far reaching but not surprising. When people are mentioned, it relates to assisting developers by providing them ready solutions to creating development artifacts. The other "people" in ISD, the users are conspicuous by their absence. This is a result of the fallback to an emphasis in all reuse techniques to describing ISD as purely an engineering or scientific discipline, requiring predetermined

inputs to gain expected outputs. However as is argued for in Checkland and Howell (1998), successful ISD is depends firstly on understanding the organization and its information needs.

The epistemology of a reuse approach is deterministic in that ISD process would only consist of matching knowledge in existing artifacts to any ISD situation that occurs. The main questions with a reuse approach are then those relating to how to represent, store and retrieve previously obtained "knowledge".

An interesting scenario could be that of a IS developer facing a set of user requirements with a repository of solutions in the form of for example, analysis patterns, i.e. conceptual models obtained from past modeling experiences. The developer has the option of amending the solution to fit the requirement or amending the requirements to fit the solution. Inevitably, there is risk that user requirements will not be met. Faced with the scenario of having a collection of analysis patterns ready for use, Fowler relates that whether he shows them to the customer depends on how he perceives the customer's expectations of how a developer should arrive at analysis models. That seems to be a rather difficult situation to be in for both sides! A summary of some of the literature concerning creativity in ISD and the leaving out of users from ISD is given in Fitzgerald (1994),

> *"To view development as an orderly progression from requirements analysis to a solution designed purely around those requirements is to miss the crucial synergy between user and developer".*

One of the key assumptions implied by a rational and systematic approach to ISD, inherent in reuse is that developers given a development situation where there is a possibility of choosing between the reuse of artifacts from a repository or creating new artifacts will make a rational and optimal choice. In a relatively small study of programmers that was carried out in such a situation, there was a tendency to anchor to the code rather than the problem description (Parsons, Saunders 1998). These cognitive aspects are unlikely to be controlled but at least they should be known.

Approaches

Another categorization of ISD methodologies in terms of approaches is described in Hirschheim et al. (1998), Iivari (1999) and Iivari et al.(1999):

> *"…an approach as a set of fundamental goals, guiding principles, concepts and principles for the ISD process for IS development, which drive interpretations and actions in information systems development. The **goal** specifies the general purpose of the approach. **Guiding principles** form the common philosophy of the approach which ensures that its methodology instances form coherent wholes. **Fundamental concepts** define the nature of an IS implicit in the approach as well as the focus and unit of analysis in ISD. **Principles of the ISD process** express essential aspects of the ISD process in the approach "*

We would argue that reuse is an implicit approach, without "methodology instances", but rather technique instances that do not form coherent wholes. This does not detract from our argument, but rather raises the question of the need for a reuse methodology.

In Table 4 below, a prescriptive view of the essentials of the reuse approach is given.

Essentials	**Reuse Approach**
Goals	To provide an approach that enables the reuse of artifacts to maximize economic effectiveness in ISD. Not to "reinvent the wheel".
Guiding Principles and beliefs	Seamless phases of ISD. ISD Artifacts are reusable in part or in whole given that the context of their development and deployment is known and documented.
Fundamental concepts	Flexibility of reusable artifacts. Development by reuse. Development for reuse. Intentional reuse. Traceability. Genericity. Adaptability. Procedural knowledge for developing and using artifacts.
Principles of the ISD process	Use of a repository of artifacts. Iterative comparison at every phase between situation at hand and artifact repository. Traceable flexibility in reusing artifacts.

Table 4: The Essentials Of The Reuse Approach based on Hirschheim et. al. (1998), Iivari (1999), Iivari et. al.(1999)

In the original version of this table several other approaches ISD were described. The fundamental difference that separates reuse as an approach from the other approaches, information modeling, SSM-based, speech act and trade unionist is that of creativity. With reuse, there is an attempt to first use or adapt previous solutions. The amount of reuse and the feasibility of a reuse in practice would then depend on a critical mass of artifacts

being available in the domain. The choice of using any reuse technique would need then to be made in every specific development situation as described in Fitzgerald (1997).

Another issue regarding the number of artifacts in any repository is that there must inevitability be a relationship between the number of artifacts and their complexity. At the base of a pyramid would be a relatively large number of small pieces of common software code. At the top of the pyramid there could be for example, a small number of unique and complex requirements patterns.

Paradigms

In addition, Iivari et al (1998), Iivari (1999) describe a set of nine bipolar, not mutually exclusive paradigmatic assumptions on which ISD approaches are based. In Table 5 below, we prescribe the paradigmatic assumptions of the reuse approach. These outline the underlying philosophies behind nine central phenomena in ISD.

Phenomena	**Reuse approach**
View of data/information	Descriptive facts
View of information systems	Technical System
View of human beings	Determinism with little voluntarism
View of technology	Mainly determinism with little human choice
View of organizations	Firmly structuralist
Epistemology	Strongly positivist
Research Methodology	Constructive: Conceptual development, Technical development Nomothetic: Field Studies, Formal analysis
Role of IS Science	Means-end oriented
Values of IS Research	Organizational effectiveness, Economic goals, Building on previous knowledge, Aiding developers

Table 5: Paradigmatic assumptions in a reuse approach, based on Iivari (1999) and Iivari et al. (1998)

The paradigmatic assumptions as described in Iivari (1999) and Iivari et al. (1998) are of three types:

- "Ontology-what is assumed to be the nature of IS
- Epistemology-what is human knowledge and how it can be acquired
- Research methodology-what are the preferred research methods for developing and continuous improving of the ISD approach and what are the modes of evidence by which they are justified
- Ethics-what are the values that ought to guide IS research"

It seems quite clear that applying reuse implies the acceptance of positivist epistemology where cause-effect relationships are stressed. The ontology of IS also seen to be strongly associated with realism, with reuse implied to be unquestionably objective, often with an economic justification. Human choice is limited to deciding between reusable artifacts within a repository, rather than questioning the reuse approach itself.

REUSABILITY

Following the previous brief discussion concerning the fundamentals of a reuse approach we turn to looking at the key functional property of reusable artifacts; reusability. If a reuse technique is to be used, an essential property needs to be flexibility in dealing with the knowledge contents in the artifact. This is because the dynamism of information needs and information systems will seldom allow for a perfect match between development situations and reusable artifacts. To successfully cope with dynamism, reuse techniques need to be capable of dealing with the following dimensions.

Dimensions of Reusability

Adaptability-Given that artifacts developed for reuse will seldom precisely match the situation at hand, the ability to adapt the artifact, trace the adaptation steps and results and describe the process used are essential. Reusable artifacts need to be described in such away that allows for encapsulation as with object orientation. In other words, there needs to be a differentiation between the content of the artifact and its relationship to other artifacts as well as the process of adaptation.

Genericity-Dealing with genericity can be done in alternative ways. One trend is that components are being designed for specific domains. Another trend is that model based artifacts tend to be more generic, being in effect guidelines. In any case, there should be a traceable relationship between generic and specific reusable

artifacts where these exist. There is also a dilemma in determining the degree of genericity of the artifact in that :

- The more generic an artifact is the more reusable it is across domains, but is at the same time more trivial.
- The more company-specific an artifact it is, the more relevant it may be, but is at the same time applicable in fewer development situations.

In both cases, it is insufficient to believe that only storing varying types of artifacts in a repository is sufficient to enable reuse as in sometimes the case with software reuse. This leads us to the following stipulative definitions of parameters for discerning the dimensions.

Contextual Adaptability

This parameter explains the degree to which the context in which an artifact was originally created is described. When creating a reusable requirements pattern for example, the context could be at least the type of organization in which it creation was based, in which country does this organization exist, what is the rationale for and consequence of using the artifact, etc, etc. These types of descriptions are often used when for some techniques based the idea of a pattern (Patterns 2001).

The principle of contextual adaptability is that the more the context in which a reusable artifact is described, the easier it should be adapt it to the context at hand. The context is a reflection about the creation, development and use of an artifact that is apart from the artifact itself in the specific development situation. Without a contextual description, implicit assumptions underlying the creation and existence of artifacts will be hidden making the artifact less adaptable and hence less reusable. Hence, an artifact with high contextual adaptability should be more easily reusable while an artifact with low contextual adaptability consisting almost exclusively of the artifact itself would be less adaptable. The second parameter, situational genericity is related to contextual adaptability, with both simultaneously affecting reusability.

Situational Genericity

When developing reusable artifacts, a key concern is the level of abstraction of the artifact. An analysis pattern may for example be so general as to be trivial, while a section of software code may be very specific to a particular organization. Unlike contextual adaptability, there is no optimal solution i.e. high situational genericity or low genericity. In the case of situational genericity, it depends on the planned scope of reuse and is affected by the ISD stage in which the artifact is developed. For example, a bank would find very specific use cases for credit transactions reusable in differing systems or even the same system. A more generic use case would more suitable for organizations with broader domains or for developers of systems for a wide range of organizations. Situational genericity has then more to do with management of reuse in terms of policy and intent rather than ISD in the narrowest sense.

An optimal situational genericity of an artifact is quite simply then one that fits in with the intentions for which the artifact is developed. In other words, not to generic so as to be trivial for its intended purpose nor too specific so as to require considerable adaptation to any other purpose.

CONCLUSIONS

In this paper, we have presented two alternative taxonomies for reuse techniques and used three alternative frameworks for describing the philosophical underpinnings of a reuse approach to Information Systems Development. The implications of the understanding gained following these classification were then used to develop and expand the meaning of reusability as a functional property. The implications then of the emergence of reuse as an approach are not clear cut but can be summarized as:

1. Reuse based development is a regression towards a functionalist-positivist approach to systems development.
2. While ISD methodologies allow the systematization of ISD procedural knowledge, a reuse approach allows for the systematization of ISD product knowledge.
3. The impact of users is minimized in favor of increased developer discretion.
4. Creativity is seen to be uneconomic, unnecessary and implicitly riskier than using past results.
5. The key to successfully applying reuse techniques should be that they include the ability to adapt and manipulate ISD artifacts rather than just storage and regurgitation.

Future Research

As with other areas of research in information systems, there seems to be a need to move towards interpretive or critical research approaches, to gain understanding of proposed functional solutions. With reuse, this could

entail case studies of development by reuse. These could be of a cognitive nature such as in Parsons and Saunders (1998), sociological or organizational. There is a need in any case to come away from the notion that there are quick fixes, based on the belief that systems can developed by putting building blocks together.

The work presented here is part of a wider effort in attempting to understand underlying assumptions and the actual practice of reuse.

REFERENCES

Allen, P. and Frost, S.(1998) *Component-Based Development for Enterprise Systems: Applying the SELECT Perspective*™, Cambridge: Cambridge University Press; SIGS Books, 1998.

Avison D.E., Fitzgerald G. (1998) *Information Systems Development: Methodologies, Techniques and Tools.* Second edition. The McGraw Hill Companies.

Bubenko J.A. jr., Wangler B. (1992) *Research Directions in Conceptual Specification Development.* Conceptual Modeling. Databases and CASE: An integrated view of Information Systems Development. John Wiley, 1992.

Cater-Steel and Ah-Fock (2000) *Reuse: Can it Deliver Competitive Performance?* ACIS, Australasian Conference on Information Systems, December 2000, Brisbane, Australia.

Checkland. P.(1999) *Systems Thinking, Systems Practice*. John Wiley and Sons.

Checkland P., Howell S. (1998) *Information, Systems and Information Systems. Making sense of the field.* John Wiley and Sons, 1998.

Fitzgerald, B. (1994) *The Systems Development Dilemma: Whether to Adopt Formalized Systems Development Methodologies or Not?,* in Baets, W. (Ed) Proceedings of Second European Conference on Information Systems, Nijenrode University Press, Holland, pp. 691-706.

Fitzgerald, B. (1997) *The Use of Systems Development Methodologies in Practice: A Field Study*. The Information Systems Journal, Vol. 7, No. 3, pp. 201-212.

Fitzgerald, B. (1999) *Systems Development Methodologies: The Problem of Tenses*. Information Technology & People

Fowler M. (1997) *Analysis Patterns: Reusable Object Models*, Addison-Wesley, 1997.

Gamma E., Helm R., Johnson R., Vlissides J. (1994) *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley. October 1994.

Hay D. C. (1995) *Data Model Patterns : Conventions of Thought* . Dorset House Publishing.1995.

Hirscheim. R., Iivari J., Klein H.K. (1998) *A Comparison Of Five Alternative Approaches To Information Systems Development*. Australian Journal of Information Systems, Volume 5 Number , September 1998.

Hirschheim R., Klein. H.K. (1989) *Four Paradigms of Information Systems Development*. Social Aspects of Computing. ACM. Volume 32 Number 10. October 1989.

Iivari, J. (1999) *Information systems development as knowledge work: The body of systems development process knowledge*. 9th Conference on Information Modeling and Knowledge Bases, 1999.

Iivari J. (2000) *Information Systems Development as Knowledge Work: the body of systems development process knowledge*. 11th Information Modeling and Knowledge Bases. IOS Press, 2000.

Iivari J., Hirschheim R. (1996) *Analyzing Information Systems Development: A comparison and Analysis of Eight IS development Approaches*. Information Systems, Vol.21 No.7 pp.551-575, 1996.

Iivari J., Hirschheim R. & Klein H.K. (1998) *A Paradigmatic Analysis Contrasting Information Systems Development Approaches And Methodologies.* Information Systems Research, Vol. 9, No. 2, pp. 164-193, 1998.

Iivari, J., Hirschheim, R. Klein. H.K. (1999) *Beyond Methodologies: Keeping up with Information Systems Development Approaches through Dynamic Classification*. Proceedings of the 32nd Hawaii International Conference on System Sciences, 1999.

Iivari, J. Maansaari J. (1998) *The usage of systems development methods: are we stuck to old practices?* Information and Software Technology 40 (1998) 501-510.

Johnson R. (1997) *How frameworks compare to other object oriented techniques*. Communications of the ACM, October 1997/Vol 40, No.10.

Kelly T.P. and Whittle B.R. (1995) *Applying lessons learnt from Software reuse to other domains*. The Seventh Annual Workshop on Software Reuse. 28-30 August 1995; St. Charles, Illinois, USA.

Sutcliffe, A.G. and Maiden N.A.M. (1994) *Domain Modeling For Reuse*. 3rd Intl. Conference on Software Reuse: Advances in Software Reusability, Rio de Janeiro, Brazil, November, 1994.

Parsons J., Saunders C. (1998) *Anchoring and Adjustment: Impedients to Reuse in Object-oriented Programming?* 8th Annual Workshop on Information Technologies and Systems, WITS'98, Jyväskylä, Finland, 1998.

Patterns (2001) *The Patterns Home Page.* http://www.hillside.net/patterns/patterns.html

Robertson, S. and Robertson, J. (1999) *Mastering the Requirements Process*. Addison-Wesley Pub Co, 1999;

Wynekoop J. L., Russo N.L. (1993) *Systems Development Methodologies; unanswered questions and the research-practice gap*. 14th International Conference on Information Systems. Orlando Florida, 1993.

COPYRIGHT