

THE ADOPTION AND ADAPTATION OF OBJECT-ORIENTED METHODOLOGIES IN REQUIREMENTS ENGINEERING PRACTICE

Linda Dawson

School of Information Management and Systems, Monash University, Caulfield East, Victoria,
Australia

Phone: +61 3 9903 2415

Fax: +61 3 9903 2005

email: Linda.Dawson@sims.monash.edu.au,

Peta Darke

School of Information Management and Systems, Monash University, Caulfield East, Victoria,
Australia

Phone: +61 3 9903 2416

Fax: +61 3 9903 2005

Peta.Darke@sims.monash.edu.au

ABSTRACT

The move towards the use of object-oriented methods for information system development has led to the need for object-oriented approaches to requirements engineering. Research into current system development practices in object-oriented requirements specification is necessary for techniques and tools to evolve and improve. This paper describes a set of four case studies that examined the use of object-oriented methodologies in professional requirements engineering practice by experienced system developers. In these studies, it was found that the widely published and commonly available methodologies were rarely used in their entirety, if they were used at all. Rather, most consultants interviewed developed in-house methodologies based on selected parts of methodologies and notations described in the literature and their own experience of "what had worked for them in the past". The reasons for this development of in-house methodologies include cost constraints for commercial methodologies and personal preference for the flexibility in adapting parts of methodologies to suit a specific task or project. This research project confirms and extends the findings of existing research in the use of system development methodologies and in particular, contributes to research into both requirements definition and object-oriented development practice.

1. INTRODUCTION

There are many published object-oriented methodologies such as Object Modelling Technique (OMT) (Rumbaugh et al., 1991), OOSE (Jacobson et al., 1992), OPEN (Henderson-Sellers & Simons, 2000). There are also commercial methodologies available which can be purchased in many configurations such as Rational Rose (Quatrani, 1998) and Rational Unified Process (Jacobson et al., 1999). All these methodologies take a structured approach to requirements specification with the emphasis on deliverables.

The purpose of the research presented in this paper is to investigate the use of object-oriented methodologies in requirements engineering practice. In the case studies presented here, methodologies

were rarely used in their entirety, if at all, for the development of requirements specifications. Rather, most consultants who were interviewed developed in-house methodologies based on methodologies published in the literature and their own experience of “what had worked for them in the past”. The reasons for this development of in-house methodologies range from cost considerations of the commercially available packages to personal preference for the flexibility in adapting parts of methodologies to suit a specific task or project.

A system development methodology (SDM) has been defined as “...a systematic approach to conducting at least one complete phase (e.g. requirements analysis, design) of system development, consisting of a set of guidelines, activities, techniques and tools, based on a particular philosophy of system development and the target system” (Wynekoop & Russo, 1997, p. 48). It is generally recognised that the use of some kind of methodology is necessary to facilitate successful system development (Avison & Fitzgerald, 1995; Fitzgerald, 1998). Requirements definition is a critical phase in the system development process, and object-oriented methodologies have emerged as a significant class of SDMs (Graham, 1994; Henderson-Sellers, 1997; Vessey & Conger, 1994). Research into system development practices in object-oriented requirements definition is therefore needed to improve understanding of both requirements definition practice in general and the use of object-oriented methodologies in particular.

This paper presents the findings of a multiple-case study of the use of object-oriented methodologies for requirements definition in practice. The study examined how consultants use and adapt object-oriented methodologies and components of methodologies to produce their own customised methodologies for requirements specification. Qualitative data collection and analysis methods were used in the cases to allow in-depth understanding of methodology use in context. This study describes empirical research in the organisational context of the use of SDMs by experienced developers. It investigates a particular class of SDMs – object-oriented methodologies – of which few studies have been published (Wynekoop & Russo, 1997), and in the process describes the use of specifically object-oriented tools and techniques in practice, such as use cases and object class models. Section 2 of this paper provides background in related work. The research approach adopted is presented in Section 3. Section 4 presents the four case studies. A discussion of the findings is presented in Section 5.

2. STUDIES OF THE USE OF METHODOLOGIES

The need to use system development methodologies (SDMs) in the system development process has been generally acknowledged in the literature (Avison & Fitzgerald, 1995; Chatzoglou, 1997; Fitzgerald, 1997), and the number of methodologies available for use, both commercial and in-house methodologies, continues to increase (Avison & Fitzgerald, 1995, Chap 7). However, several studies have indicated that the use of methodologies in system development projects is low (Bansler & Bodker, 1993). A recent survey of system development practice revealed that 60% of respondents were not using methodologies (Fitzgerald, 1998). Methodologies in this case included commercial formal SDMs, in-house methodologies based on commercial SDMs, and in-house methodologies not based on formal SDMs. Fitzgerald’s study also found that 79% of those not using a methodology did not intend to adopt one. Similarly, a survey reported in Chatzoglou and Macaulay (1996) indicated that 47% of respondents did not use a methodology. Various explanations for methodology non-use have been proposed, including developers’ lack of knowledge and slow rates of diffusion of new approaches. Some recent research has suggested, though, that experienced developers do know about methodologies and methods and that they select and change aspects of methodologies to suit their needs in an informed and pragmatic way (Fitzgerald, 1997). Given the importance of the system development process, it is essential that we know more about how methodologies are selected, adapted and used in practice (Wynekoop & Russo, 1997).

In their analysis of SDM research Wynekoop and Russo (1997) note that much existing research consists largely of either normative studies or surveys. They define normative studies as “...concept

development not based on empiricism or theoretical grounding, but on the author's speculations or opinion". Normative studies accounted for over half of the SDM research publications they analysed, revealing a lack of field research concerned with the adoption, use and usefulness of SDMs in practice. Empirical SDM research mainly includes surveys, which generally identify SDM distribution but not the processes by which they are selected, adapted and used in individual projects (Wynekoop & Russo, 1997). These processes are determined largely by specific organizational contexts, so that field research such as case studies and action research is necessary in order to fully understand these processes. Wynekoop and Russo's analysis indicated that there was "... little interpretive research and action research, and few practice descriptions or case studies".

Research into the use of SDMs has generally focused on the entire development process. There have been few studies of the use of methodologies in requirements engineering, and in particular in object-oriented requirements engineering. There have also been very few studies of the use of object-oriented methodologies as a class of SDMs (Wynekoop & Russo, 1997). Chatzoglou and Macaulay (1996) surveyed methodologies used for both system development and requirements capture and analysis. They found that more respondents (62%) used a methodology for requirements capture and analysis than in the rest of development, and that respondents were more confident about the quality of requirements gathered when a methodology was used. Developers in industry were much less likely to use a methodology during the requirements phase than those in consultancies and software houses. Although useful, the quantitative data of surveys cannot provide the in-depth understanding and contextual information essential for revealing how SDMs are selected, adapted and used in practice (Wynekoop & Russo, 1997).

3 RESEARCH APPROACH

The objective of this project was to investigate the use of object-oriented methodologies for the production of requirements specifications during the system development process. The research approach used was based on multiple case studies which involved taped semi-structured interviews with individual practising professional requirements engineers. Each participant was interviewed several times, providing empirical data which is interpretive and descriptive rather than normative or quantitative as found in many of the studies described by Wynekoop and Russo (1997). This approach provided rich, qualitative data which produced similar results to the studies discussed above. For example, in this study the qualitative data collected revealed that consultants using object-oriented approaches to requirements specification and system development are more likely to use and adapt object-oriented methodologies to produce their own customised methodologies for requirements specification.

As is common with this type of qualitative research, the cases were opportunistically selected in that the participants were used because they were available and willing to be part of the study. Participants were recruited through industry contacts. Some participants provided contacts for subsequent participants. The lack of available professionals working in the field of object-oriented requirements engineering in Melbourne, Australia where this study was undertaken means that there has been no attempt to select participants based on specific background characteristics. The common factor is that all the participants were experienced developers currently working in the field of object-oriented requirements specification.

4 THE MULTIPLE CASE STUDIES

The following sections describe the four case studies with particular reference to the system development experience of the consultant/analyst, the development philosophy of the consultant/analyst and the methodology used for the project undertaken. Section 5 of this paper discusses the concepts and issues that arose in the case studies.

The presentation of each case is based on illustrated narrative style, or an oral narrative told in the first person, as described by Miles and Huberman (1994) and Myers (1999) and as used in Fitzgerald (1997) and Urquhart (1998). This approach is described as (Miles & Huberman, 1994) "...each part of the sequence is followed by a series of illustrative excerpts [quotes from the transcripts]" which does not resort to explicit coding but looks for " ... key words, themes, and sequences to find the most characteristic accounts." Where transcript data is quoted directly the researcher's questions or interactions are shown as bold italic and the participant's as plain italic.

Case 1: Developing a Transaction Specification Methodology for Electronic Service Delivery for a State Government Authority

Overview:

The consultant in Case 1 used object-oriented methods to develop an in-house transaction specification methodology based on "lazy dog" templates of identified "common transactions" for multiple clients with similar needs. Each client organisation had their own end-clients and could do their own requirements definition using the generic transaction specification methodology with the assistance of an IT liaison person. The templates were called "lazy dog" because they were partly completed specification templates containing use cases (both scripts and graphs) and OMT class diagrams which the client (via the IT liaison person) could alter by addition, or striking through, of appropriate elements. The development team consisted of three members and it took nearly two years to complete the transaction specification methodology using six client organisations.

Philosophy:

The consulting organisation in this case provided IT consultancy and educational services to a broad range of clients, both from the public and private sectors. The organisation's philosophy was outlined on their web site as: "*We do not subscribe to a single, rigid methodology. Each assignment is treated as a unique challenge. We tailor our approach to meet the specific requirements of each client, drawing on a wide range of well-researched techniques and the combined experience of our consultants.*"

Consultant's Experience:

The consultant analyst interviewed for this case had been developing object-oriented systems for about 10 years. She was self-taught in object-oriented system design and had given courses on object-oriented system development. She felt that "*OO is a very natural way for me.*"

Methodology Used:

The methodology used was based on use cases (Jacobson et al., 1992) and a generic object model using Rumbaugh et al's (Rumbaugh et al., 1991) Object Modelling Technique (OMT) notation. Jacobson's full development methodology was not used in its entirety because it was considered too open-ended, with too many decisions for client organisations to make. The client organisations were given a general pattern for a transaction that could be configured to how the client/end client wanted it. These partial specifications were called "lazy dog" templates. "*There is one general methodology, one [generic] object model and there are a set of seven different templates, for each [common] transaction type that you can use and you can tailor the templates in what I call "lazy dog" templates – i.e. they are half filled out - it is not a blank form.*" The concept of "lazy dog" templates is used in engineering specification. The partial specification, or "lazy dog" template, was presented to the client organisation (specifically to the IT liaison person or team within the client organisation). This template could be configured to the client/end client's specific needs.

Case 2 A Fault Management System for a Telecommunications Organisation

Overview:

The project was a fault management system for managing planned and unplanned outages in a transmission network. It was a five-year project and involved two and a half years of serious development work for this consulting analyst. The project was funded incrementally and for the first stage the deliverables were a suite of requirements and analysis specifications. The requirements model was a use case model and there was also a prototype. The development team (including members from the consulting organisation and members from the client organisation) was mostly a team of 12 and peaked at a membership of 15. There were two subteams to do a lot of the early work and the consulting analyst supervised one of those subteams.

Philosophy:

The methodology used for system development was an in-house object-oriented method. It was based on other methodologies that members of the team were familiar with. *"We sampled from methodologies that we were familiar with. We used bits of other methodologies as appropriate.... five of the developers had significant experience of building similar systems elsewhere... What that meant was there were three or four people who were able to contribute to a methodology that picked up bits and pieces from a number of influences... They just all brought their biases and their interests and thoughts."* The development process was heavily influenced by the people who were available, and the fact that they had come with quite considerable industry experience in this kind of software development.

Consultant's Experience:

The consultant considers himself to be a very experienced developer who has spent more time than the average developer in requirements engineering. He had been doing object-oriented system development for about four years.

Methodology Used:

Models were based on use case scripts, OMT class models and interaction diagrams although the interaction diagrams were not used much until the design phase. There was a prototype as well which included the use case model. The use case model was categorised by the consultant as a dynamic requirements model, or a functional model. In this case the users only ever dealt with use case type models - they never had to understand the OMT model or interaction models. The object model and interaction model are models used only within the analysis team and understood by members of the team.

Case 3: A Generic Insurance Package

Overview:

The project described in Case 3 is a receipting system for a generic insurance package. The client was a software development organisation which develops packaged software for the financial industry. The participating consultant was a director and a partner of a small consulting organization consisting of three people. The members of the consulting organisation work as system developers in the object-oriented field but are also mathematicians " ... *with a particular view on life*". The project took about a year from commencement to first release.

Philosophy:

The consultant did not use any specific methodology in this project or any other projects. In his position before his current one he worked for an organisation which was " ... *not aligned with a*

particular methodology though one came across methodologies all the time so one used those techniques in various ways. His philosophy was to not use a methodology "... I haven't been, let's say, an advocate of any particular methodology from start to finish ... I don't believe in methods as such ... What I talk about is a underlying concept rather than a methodology. A methodology seeks to impose a concept ... I think methodologies and parts of methodologies are useful but they're just props and tools and can be picked up and thrown away as required ... I think a methodology is only as good as the deep understanding that people have of the concepts that it's built on ... a methodology is no good on its own... you need to have rigour and the diagramming notations and the steps in the methodologies give you that but you also need to play in the sandpit."

Consultant's Experience:

The consultant in Case 3 had extensive experience in using object-oriented methods to specify and build actuarial and insurance systems. OMT class models or interaction diagrams were used within the team and in the design phase. The consulting analyst had been doing requirements engineering for about 22 years and his background is in mathematical modelling. He believes that he has been doing object-oriented analysis longer than anyone else in Australia and also believes that he was one of the first commercial users of object-oriented systems, SmallTalk, in 1985. Although the consultant has not taken any formal courses in object-oriented system development, he has delivered them, including the first course in Australia.

Methodology Used:

In this project the team applied the use case concept. The same development team started with the requirements, moved onto design, and then became involved in other parts of the development. "*... the single thread through the whole thing has been the use cases and a lot of the objects are still there and they ... well they've got the same name but the way they're organised is quite different.*"

Prototyping, particularly using illustrative methods and tools like PowerPoint slides to mimic input screens, was seen as a way of enhancing requirements gathering and later acceptance of the requirements "*If you get, as part of the requirements gathering, a prototype you get much better sense of requirements. One of the things I've done on this project (and which I've actually done before) is I've used PowerPoint (before we had an interface developed) to simulate an interface and the thing is that it's not just having a picture of a screen, you can run a slide show and see how you interact with the screen and I actually threw away the text use cases when I got to the design phase and just did it all that way. The interface developers were using that as a guide.*"

Case 4: A Stockbroking System

Overview:

The consultant in Case 4 was a senior project manager for a software development organisation which developed generic packaged software systems. In this project a generic stockbroking package was being developed using an in-house object-oriented methodology. The anticipated number of users was 350 and the project team had up to 15 members. Models shown to users were based on prototypes, screen simulations and animations with use case models used mainly at the validation phase. The consultant's official title is Technical Development Manager. All system administrators report to him and he also acts as a system architect from a software perspective and so is responsible for all designs and all analysis of the software that the organisation develops.

Philosophy:

The consultant had not and does not use a complete proprietary or commercial methodology because he believes that they are too expensive and too complex "*But in most areas ... I've not seen anybody use the big methodologies. I think there are two reasons. (A) If you buy the professional ones they charge too much, which is also why I think why everyone talks about Rational, although we tend to use a competitive product, Select, mainly because it's a little cheaper. And even then I don't have as*

many copies as I should have because it's so expensive. They are VERY high cost and if you put on the process flow modelling, all the methodology on top of that ... What happens is that the cost of setting up a developer starts to become prohibitive and there's no return on that so either you escalate the price of your product to cover that high cost or you hope you work for a multi billion dollar company that can afford to simply write cheques and say 'yeah we will spend all this money'. The other reason certainly why we tend to use our own methodologies which have short cuts and work arounds and all sorts of different things and why even methodologies where you are supposed to follow them [in our case] there are odd documents missing and some are much shorter than they should be. There is simply not enough time to follow the whole box and dice and produce all of the documents. You produce those documents where, if you've got to get a user to sign off do those. Why, because that affects the bottom line and that's really what it's about."

Consultant's Experience:

The consultant was experienced in many methods, both object-oriented and non object-oriented, for specifying and building business systems. The consultant has been involved in object-oriented systems for about five years and has worked for the organisation for twelve and a half years. He has spent all of that time doing systems analysis and requirements engineering. The consultant had used non object-oriented methods before moving into object-oriented based systems. He used a relational database management systems (RDBMS) approach based on INGRES and before that he developed COBOL-based systems using traditional structured techniques. Data Flow Diagrams and Structured Systems Analysis and Design Method (SSADM) techniques were used in the RDBMS approach. He believes that although object-oriented approaches have certain advantages they also have some limitations for developing systems *"It [the object-oriented approach] has some advantages in some of its approaches and the encapsulation concepts work well, however, in many regards we were already doing that even back in the COBOL days by using ... proper use of subprograms and reusable code and ...modular design ... The problem with using DFDs and that is that the models were far too data-centric which was fine if you were doing a lot of retrieval but to do good transaction processing was quite awkward and you really did need very high levels of expertise to get it right in the RDBMS world. That is much less so [in OO] and therefore you can actually end up with much simpler solutions with OO techniques as long as you keep the propeller head so to speak away, you actually end up with systems which are very easy to understand and easy to maintain and that's the big plus."*

Methodology Used:

The methodology used in this project was an in-house methodology based on UML notation but not the complete Rational development method. *" ... there may be an ITT (invitation to tender) or something of that nature which we start off with ... out of that document we do business requirements. We have a business rules document. And from that we go to a top level design, detailed design and then again from that there are a number of different levels of testing which are to be put in place through integration testing, system testing, and then user acceptance testing."* Prototyping in the form of a GUI prototype for the users was used in the project. *"We actually do a prototype and then work through the users with that and then gain sign off at that level."*

An integrated development tool called ModelWorks had been used in this project. It is an active modelling animation tool which allows developers to describe the business processes and model them using the modelling tool and then animate the model. It is possible to build the skeleton of an application or a prototype as the analysis is being undertaken.

5 ANALYSIS AND DISCUSSION

None of the consultants in this study used an entire formal methodology for requirements specification. All four used some of the OMT or UML notations and variations of use cases in their own in-house methodologies. All analysts interviewed agreed that they developed a customised methodology or "conceptual toolkit" of methods, techniques and notations that they had built up over

a number of years and a number of projects. Table 3 summarises the philosophies and methodologies used in the four cases.

Case No	Philosophy	Consultant's Experience	Methodology Used
Case 1	<i>"We do not subscribe to a single, rigid methodology. Each assignment is treated as a unique challenge."</i>	15 years in system development 10 years in OO system development	In-house based on OMT and use cases. Templates and generic OMT class model which was edited by clients.
Case 2	<i>"We sampled from methodologies that we were familiar with. We used bits of other methodologies as appropriate"</i>	12 years in system development 4 years in OO system development	In-house from team members experience based on OMT models and use cases. Prototyping based on a use case model.
Case 3	<i>"I haven't been, let's say, an advocate of any particular methodology from start to finish ... See, I don't believe in methods as such ... What I talk about is a underlying concept rather than a methodology. A methodology seeks to impose a concept"</i>	22 years in system development 13 years in OO system development	In-house methodology based on ER, OMT, use cases, ad hoc diagrams, prototypes etc. Prototyping using PowerPoint simulations for clarification with users.
Case 4	<i>"I've not seen anybody use the big methodologies. I think there are two reasons. (A) If you buy the professional ones they charge too much, ... The other reason [is] ... there is simply not enough time ...[to] produce all of the documents."</i>	12 years in system development 5 years in OO system development	In-house based on ER, UML notation and use cases. Prototyping using an animation simulation package.

Table 3 Summary of findings

These findings support and extend the findings of other studies which were discussed in Section 2 above. The consultants in this study developed and used in-house methodologies based on published methodologies and notations. As was found in the study by Fitzgerald (1998) the use of specific methodologies in their entirety was low. The consultants studied here were all very creative about "mixing and matching" tools and techniques taken from specific methodologies to suit their own particular philosophy and approach to system development and requirements engineering. As experienced developers they were familiar with available SDMs but rejected the use of a single published or commercial method in favour of their own customised SDM based on their own experiences and philosophy. This suggests that teaching and training in the use of methodologies should not be specific to a single, entire methodology but should concentrate more on the selection

and aggregation of appropriate tools and techniques not just for the particular project but also tailored to the style of a particular analyst/consultant.

The in-house methodologies used in the four cases fit Wynekoop and Russo's (1997) definition of a methodology presented in section 2 above. The customised methodologies or "conceptual toolkits" are systematic and consist of a set of techniques and tools based on a particular philosophy of system development. The implication is that methodologies provide structure for the requirements engineering process and are based on the developers' own experiences and development philosophies. This further implies that methodologies are important to these consultants. This is contrary to the findings that Fitzgerald (1998, p 326) reports that SDMs' "...most evident contribution seems to be as a framework for the use of tools and techniques". The consultants in this study constructed and used customised methodologies which were much more than a simple framework.

6 CONCLUSION

The research project described in this paper has confirmed and extended existing research into the use of system development methodologies. It provides rich qualitative data about the activities of practicing professional requirements engineers and contributes to a greater in depth understanding of professional requirements engineering practice on which to base future research.

Although this study was specific to object-oriented system development it has implications for the use of methodologies in general. This study has confirmed that the use of the published methodologies in professional practice is low and that this is not due to lack of knowledge or training in system development methodologies but rather that professional developers develop their own customised methodologies because they consider standard methodologies to be too complex, expensive and/or inflexible and unable to meet their needs. Also these customised methodologies or "conceptual toolkits" are not ad hoc but constructed from the tools and techniques available in the published methodologies.

The implications for practice and, potentially for SDM vendors, is that developing complex or prescriptive methodologies may not be appropriate for addressing the requirements of professional practice. Further work on developing specific tools and techniques that can be combined to provide customised in-house methodologies may be more useful to professional practice. It is essential to further investigate which tools and techniques are considered useful by practising professionals, and what makes those tools and techniques useful, so that new and improved tools and techniques may be developed.

These issues will also need to be considered in the education and training of IS professionals. Students may need to be encouraged to start to build their own conceptual toolkits and so need to be exposed to as many useful tools and techniques as possible. The consultants interviewed in this study were experienced system developers. Although there is some published research on novice developers (Chaiyasut & Shanks, 1994; Sutcliffe & Maiden, 1992), investigating how less experienced developers learn and carry out system development tasks and how they develop their own conceptual toolkits would be of benefit in better understanding and improving system development tools, techniques, and methodology development in the future.

REFERENCES

- Avison, D. E., and Fitzgerald, G. (1995) *Information Systems Development: Methodologies, Techniques and Tools*, (2nd ed.), McGraw Hill
- Bansler, J., and Bodker, K. (1993) A Reappraisal of Structured Analysis: Design in an Organizational Context, *ACM Transactions on Information Systems*, 11, 2, 165-193.

- Chaiyasut, P., and Shanks, G. (1994) Conceptual Data Modelling Process: A Study of Novice and Expert Data Modellers, In *Proceedings of the First International Conference on Object Role Modelling*, Magnetic Island, Australia, 310-333.
- Chatzoglou, P. D. (1997) Use of Methodologies: An Empirical Analysis of Their Impact on the Economics of the Development Process, *European Journal of Information Systems*, 6, 256-270.
- Chatzoglou, P. D., and Macaulay, L. A. (1996) Requirements Capture and IS Methodologies, *Information Systems Journal*, 6, 209-225.
- Fitzgerald, B. (1997) The Use of Systems Development Methodologies in Practice: A Field Study, *Information Systems Journal*, 7, 201-212.
- Fitzgerald, B. (1998) An Empirical Investigation into the Adoption of System Development Methodologies, *Information and Management*, 34, 317-328.
- Graham, I. (1994) *Object Oriented Methods*, Addison-Wesley, Wokingham, UK.
- Henderson-Sellers, B. (1997) *A Book of Object-Oriented Knowledge*, (2nd ed.), Prentice-Hall, Upper Saddle River, NJ.
- Henderson-Sellers, B., and Simons, A. J. H. (2000) The Open Software Engineering Process Architecture: From Activities to Techniques, *Journal of Research and Practice in Information Technology*, 32, 1, 47-68.
- Jacobson, I., Booch, G., and Rumbaugh, J. (1999) *The Unified Software Development Process*, Addison Wesley Longman, Reading, MA.
- Jacobson, I., Christerson, M., Jonsson, P., and Overgaard, G. (1992) *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley/ACM, New York.
- Miles, M. B., and Huberman, A. M. (1994) *Qualitative Data Analysis: An Expanded Sourcebook*, (2nd ed.), Sage Publications Inc, Thousand Oaks, CA.
- Myers, M. (1999) Qualitative Research in Information Systems, <http://www.auckland.ac.nz/msis/isworld/>.
- Quatrani, T. (1998) *Visual Modeling with Rational Rose and Uml*, Addison-Wesley, Reading, MA.
- Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. (1991) *Object-Oriented Modeling and Design*, Prentice-Hall, Englewood Cliffs, NJ.
- Sutcliffe, A. G., and Maiden, N. A. M. (1992) Analysing the Novice Analyst: Cognitive Models in Software Engineering, *International Journal of Man-Machine Studies*, 36, 719-740.
- Urquhart, C. (1998) Analysts and Clients in Conversation: Cases in Early Requirements Gathering, *Proceedings of the Nineteenth International Conference on Information Systems*, Helsinki, Finland, 115-127.
- Vessey, I., and Conger, S. (1994) Requirements Specification: Learning Object, Process, and Data Methodologies, *Communications of the ACM*, 37, 5.
- Wynekoop, J. L., and Russo, N. L. (1997) Studying System Development Methodologies: An Examination of Research Methods, *Information Systems Journal*, 7, 47-65.