Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2009 Proceedings

Americas Conference on Information Systems (AMCIS)

2009

Combining Diagrams to Enhance Understanding: Forging a Common Language for Different World Views

Kamal Masri
Simon Fraser University, kmasri@sfu.ca

Andrew Gemino
Simon Fraser University, gemino@sfu.ca

Drew Parker
Simon Fraser University, drew@sfu.ca

Combining Diagrams to Enhance Understanding: Forging a Common Language for Different World Views

Kamal Masri
Faculty of Business Admin.
Simon Fraser University
8888 University Drive,
Burnaby, BC, V5A 1S6
kmasri@sfu.ca

Andrew Gemino
Faculty of Business Admin.
Simon Fraser University
8888 University Drive,
Burnaby, BC, V5A 1S6
gemino@sfu.ca

Drew Parker
Faculty of Business Admin.
Simon Fraser University
8888 University Drive,
Burnaby, BC, V5A 1S6
drew@sfu.ca

ABSTRACT

The Unified Modeling Language (UML) has become the *de facto* standard in object oriented systems design. It has, however, been subject to considerable criticism by analysts due to its complexity and inability to communicate complex systems models. This paper introduces 'Modular UML,' a modified presentation and communication format of the UML to more effectively understand multiple UML diagrams as a conceptual model of a complex system. The challenges modular UML are designed to address, the process of developing a modular UML set of exhibits, and an example are discussed.

Keywords (Required)

UML, conceptual models, communication, work systems.

INTRODUCTION

Interest is increasing in knowledge sharing and its relation to IT enabled work in organizations (Bock, Zmud, Kim, & Lee, 2005; van den Hooff & Huysman, 2009). The role of knowledge management continues to expand as organizations seek competitive advantage through innovations in processes, products and services (Ndofor & Levitas, 2004). Much of this innovation is directed through changes in what Alter (2008) defines as "*work systems*" enabled by information technology. The capabilities and practices involved in planning and designing these IT-reliant systems of work can be viewed as a core property of the IS discipline (Alter, 2003; Benbasat & Zmud, 2003). Innovation in technology enabled work systems often involves cross-functional teams that require integration of technical/engineering knowledge with knowledge of business processes, governance and strategy (Jasperson, Carter, & Zmud, 2005). Researchers have recognized that managing knowledge through cross-functional teams can improve the ability to innovate (Eisenhardt & Tabrizi, 1995; Leonard & Sensiper, 1998), but there are difficulties in sharing meaning across knowledge boundaries in occupational communities (Bechky, 2003; Carlile, 2002).

This paper considers the usability of techniques through which knowledge of IT enabled work system elements are shared beetween the business/system analysts and the less technical work system stakeholder by considering the effectiveness of what Carlile (1997) defines as a "boundary object." We argue that the most natural boundary object, in the case of IT enabled work systems analysis, between business/system analysts and system stakeholders is a conceptual model of the work system.

There are a variety of techniques for developing conceptual models of work systems (Avison & Fitzgerald, 2006; Lang & Fitzgerald, 2006). One of the more popular methods is the Unified Modeling Language (UML). UML 2.0

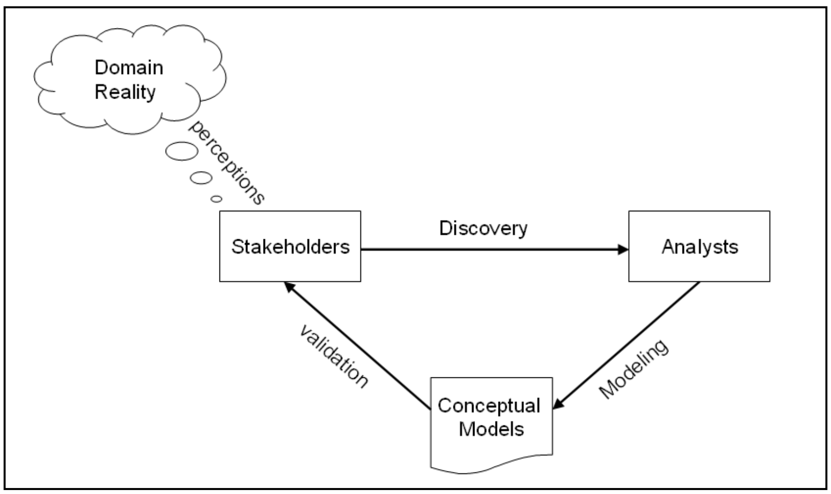
(Object Management Group, 2003) defines thirteen different diagramming/modeling techniques for describing elements in work systems. The complexities of the language and its many diagrams have sparked discussion about UML's usability and overall effectiveness (Dobing & Parsons, 2008; Dori, 2002; Erickson, 2008; Glezer, Last, Nachmany, & Shoval, 2005; Grossman, Aronson, & McCarthy, 2005; Siau & Cao, 2001).

This paper introduces the idea of 'Modular UML', a novel approach to presenting conceptual models. Structural and functional elements of UML models are combined into a single diagram. No change to the UML's current constructs is necessary. Modularization is accomplished by decomposing selected UML diagrams into modules, then combining related modules from various diagrams together. An example is shown in Figure 2. The objectives of this approach are to improve stakeholder (user or model viewer) understanding of UML diagrams during model reviews, and to improve communication between occupational communities in the analysis process.

CONCEPTUAL MODELS AS BOUNDARY OBJECTS

To understand the importance of conceptual models as boundary objects in the analysis process, it is important to recognize the difference in knowledge across the operational communities of work system analysis. Analysis of a work system can be viewed as a four phase process (Y.-G. Kim & March, 1995). In practice it is recognized as an iterative cycle without phases. Figure 1 - adapted from (Y.-G. Kim & March, 1995) and (Juhn & Naumann, 1985) - is a high level overview of this analysis cycle.

Figure 1: The Work System Analysis Cycle (Adapted from Kim & March, 1995)



Stakeholders can be viewed as executives, managers and users of the work system who work actively in the process and possess varying degrees of knowledge of business processes, the documents used to verify and perform work in the work system, along with the policies and strategies supporting the business processes. Analysts can be viewed as individuals who have knowledge of the capabilities of information technology, some basic knowledge of how the work system operates along with the knowledge of how to represent the work system elements in abstract models. Stakeholders are expected to convey their perceptions of the work system to the analysts. Analysts then work to document this knowledge and create conceptual models of the work system using diagramming techniques. Stakeholders and analysts then view the models to reconsider the work system and verify that the analysts have accurately interpreted stakeholder's views of the work system. The differences in knowledge across the two occupational communities, adapted from the analysis of work context provided by Bechky (2003), are summarized in Table 1.

Table 1: Key Differences in Work Context for Work Systems Analysis

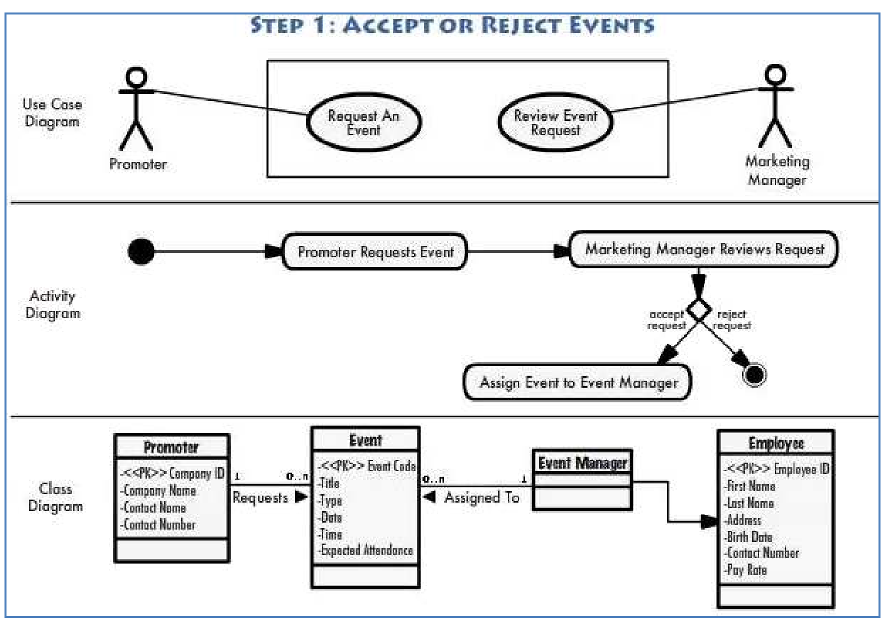
Work Context	Stakeholders	Analysts

Work	Actively participate in business process	Create diagrams and translate into requirements
Locus of Practice	Physical	Conceptual
Conceptualization of the Issue	Documents and flow of work	Information structure and process dynamics
Language	Language of the Business Process	Modeling Language

The challenge of effective work systems analysis is to find ways to integrate knowledge across these two occupational communities in a way that develops a high quality of pragmatic understanding of the work system across the two groups. The difficulty arises when the two communities begin to communicate. In her study of shared work contexts, Bechky (2003) noted that the lack of a shared work context suggests that analysts and stakeholders will have difficulty communicating because members of the different communities will describe the elements of the work system in a different way using the context that is familiar to them. She labeled this issue "decontextualization" and suggested that: *"decontextualization occurred when people from different groups met to discuss a problem, and brought different understandings of the problem to their discussion.*", the result of which was that a "*situation was presented in language that was assumed to be universal and unproblematic, but in fact the words were incomprehensible to those who did not share an understanding of the context of the situation*" (Bechky, 2006, p. 320). While the context of operational work differs for that of work system analysis, the notion of decontextualization clearly rings true when considering the communities involved in work system analysis.

Conceptual modeling, then, becomes a boundary object that can bridge discussion between the two occupational communities. The quality of the conceptual model is realized by the effective communication that is made through the model (Lindland, Sindre, & Solvberg, 1994). The UML has become the primary language used by systems analysts to develop this conceptual model, but it is not without criticism.

Figure 2: Combined UML diagram



UML AND CONCEPTUAL MODELING

Complexities of the UML and its many diagrams have sparked debate about UML's usability and overall effectiveness. For example, Hugos (2007, p. 23) described UML and its diagrams as filled with words "*designed to confuse and disengage the typical business user*". This describes a general sentiment about the complexity of UML as summarized by Ericksson (2008). Ericksson suggests that UML's continued dominance will depend, in part, on making it more comprehensible by "*...finding ways to allow UML to appear less complex, if not actually be less complex*" (Erickson, 2008, p. v) to users and developers.

A significant source of complexity attributed to UML is the number of diagrams and a lack of apparent connectedness of these diagrams. UML 2.0 has 13 diagrams divided into three classifications: structure, behavior, and interaction. It is argued there is a large number of constructs and symbols associated with these diagrams that lead to difficulties in learning and using UML (Dori, 2002; Siau & Loo, 2006). In addition, the lack of an apparent strong relation among the multiple diagrams can lead to difficulty of learning UML (Siau & Loo, 2006) and continues to be a major concern of using UML despite the limited research in this area (Batra, 2008).

The different diagrams are not considered rigid or central for compliance to the UML structure. Analysts are empowered to select diagrams they believe are appropriate to communicate with stakeholders (Fowler, 2004). In a survey conducted by Dobing and Parsons (2006), analysts reported a tendency not to use certain diagrams either because the diagrams are not well understood by analysts or the diagrams do not add sufficient value to justify the cost. The same survey revealed only 55% of analysts feel that UML was moderately successful, at best, in facilitating communication while 25% of analysts felt that UML was not successful at all (Dobing & Parsons, 2008). Dobing & Parsons concluded that *"the complexity of UML is a concern, suggesting more programs are needed to help IS professionals and their clients learn the language and how to use it more effectively"* (Dobing & Parsons, 2006, p. 113).

Perceived complexity of UML along with the research findings by Dobing and Parsons (2006, 2008) lend support to the suggestion that model complexity results in comprehension difficulties which, in turn, leads users to misunderstand information represented in the models (Gemino & Wand, 2003). Therefore, reducing the complexity of UML models is required to increase the effectiveness, usability, and to maintain or increase the dominance of UML.

Reducing the semantic and structural complexity of UML is difficult to accomplish. Kobryn (1999) suggests standardization is achieved through consensus which often leads to bloated specifications. The need for consensus to implement significant changes to reduce UML's complexity requires stakeholders who have invested heavily into the current UML (such as tool-vendors and users' installed base) to support large-scale changes. This is considered "*highly unlikely, if not impossible*" (Dori, 2002, p. 85). The increase to 13 diagrams in UML 2.0 from the original nine diagrams adds credence to Dori's statement. So, short term practical changes to UML must shift away from semantics and structure towards a behavioral perspective to achieve the necessary reduction of UML's overall complexity (Erickson, 2008).

Suggestions to reduce the complexity of UML models include combining structure and behavior elements to reduce complexity and increase usability of these models (Dori, 2002). Research in this area considered the use of Object Process Methodology (OPM) (Dori, 2001) as a feasible representation of combining structure and behavior. Other studies introduced the Functional and Object-Oriented analysis Methodology (FOOM) to combine structural elements from class diagrams with data flow diagrams (Shoval & Kabeli, 2001). FOOM studies yielded a partial advantage when compared with OPM (Kabeli & Shoval, 2005). However, both of these approaches suggest replacing UML altogether or, at the very least, amending the UML to include the described modeling techniques. Adoption of such techniques implies a fundamental change to UML which is unlikely to happen due to reasons highlighted by Dori (2002).

MODULAR UML

To address the concern of UML complexity, we suggest the use of a modular approach to UML. The theoretical foundations for this approach are an adaptation of Mayer's (2001) Cognitive Theory of Multimedia Learning. Modular UML is a presentation technique intended to facilitate understanding by reducing presentation complexity of multiple UML diagrams. The objective of using Modular UML is to improve communication and facilitate understanding of the domain between stakeholders and business/system analysis. Working with modular UML does not require analysts to learn yet another modeling technique. Instead, it uses knowledge about the system and business process to decompose the models into a set of steps. Stakeholders receive the full set of diagrams plus the set of combined models for review. The availability of the full models is important since stakeholders may want to review the complete diagram or to confirm linkages between steps.

We suggest that the Modular UML development comprises four distinct stages:

Stage 1: Create UML diagrams
Stage 2: Determine decomposition steps
Stage 3: Decompose models
Stage 4: Combine diagrams

At stage 1, the model developer (analyst) creates UML diagrams considered appropriate to model the system and used for communication with users. The diagrams are created according to UML specifications using standard UML constructs, notation, and procedures. Stage 2 requires the analyst to consider representing the system in 'chunks'. Each chunk should represent a major step (function or process) of the system. We suggest that a summary (white level) use case narrative is a natural starting point that provides a general overview of the system to assist in defining the major steps.

The analyst decomposes each UML diagram into modules in stage 3. Each major step identified during stage 2 must be represented by a module from each diagram to be used in the final exhibits. For example, if a system represented by three UML diagrams was decomposed into four distinct steps, then there would be twelve diagram modules, one for each step and for each diagram.

Stage 4 involves the analyst combining the diagram modules for each step onto one combined diagram. Here, the intention is to place the related diagram modules on the same display area. No visual connection such as boxes, lines, or arrows between modules is necessary in the combined diagrams.

An Illustrative Example: Human Resource Scheduling

We will use an illustrative model to demonstrate the process of creating combined diagrams. The system used in this example is a functioning human resource scheduling system designed and developed by the authors. The system continues to be used by an event hosting organization that owns and operates a large complex for hosting various events including professional sporting events. The system schedules over 500 part-time employees to host events over a two-week period. The analysis in this example generated business (high) level documentation. These include: a summary use case (Figure 3), a use case diagram (Figure 4), an activity diagram (Figure 5), and a class diagram (Figure 6).

Stage 1: Create UML Diagrams. The choice of which of the 13 possible UML diagrams depends on the usefulness of the diagrams as perceived by the analyst (Dobing & Parsons, 2008; Grossman, et al., 2005). Analysts therefore should choose the models they feel best communicate system information to users. Modular UML does not have any restrictions on which model to use although some diagrams are easier to decompose than others. For example, use case diagrams are easier to decompose than class diagrams.

The UML models selected for this example consisted of four of the top five most "popular" UML modeling techniques as noted in Dobing and Parsons (2008). Analysts rated, on a five point scale, use case narrative (4.00/5.00), activity (3.50/5.00), use case diagram (3.36/3.50), sequence (2.91/5.00), and class (2.90/5.00) as the top five diagrams used for verifying and validating requirements with client representatives (Dobing & Parsons, 2008, p. 10). The four diagrams for this example were intentionally selected from the top five listed to demonstrate use of modular UML with analyst-preferred diagrams.

Figure 3: Summary use case narrative

Summary (White level) Use Case

Use Case Name: Event Employee Scheduling
Business Actor: Promoter – wants to host an event at EntertainCo. facilities
Stakeholders: Marketing Manager, Event Manager, HR Department Scheduler, Event Employee, Accounting Department.
Trigger: Promoter contacts EntertainCo to host an event

Normal flow of Events at EntertainCo.:

1. Promoter contacts EntertainCo. regarding hosting an event
2. Promoter provides Event type, requested date, anticipated attendance, and desired venue setup
3. Marketing Manager reviews the request. The process for this event ends if the event is rejected.
4. If the request is approved, then the event is assigned to an Event Manager
5. The Event Manager is responsible for generating a seating plan based on expected attendance and venue setup.
6. Event Manager creates staff deployment for the event. The deployment provides a list of positions available that includes start and end times, employee classification, and employee skill.
7. The deployment is passed on to the Human Resource Department Scheduler who is responsible for producing a completed schedule.
8. The HR Scheduler generates an empty biweekly schedule from all the events to be held during the scheduling period. At the same time, Event employees are required to submit their bi-weekly availability for the scheduling period one week before the schedule is produced.
9. The HR Scheduler produces a two week schedule for all events to be held during the upcoming two week period using the following scheduling constraints
    a. No employee can be assigned work exceeding 30 hours per week
    b. Each employee must receive a minimum 12 hours break between shifts
10. The HR Scheduler posts the schedule on the company website and in the employee dressing room
11. During the scheduling period the HR Scheduler manages the schedule according to the "Manage Schedule Subflow".
12. The HR Scheduler forwards the final worked schedule which lists employees with actual worked hours to the accounting department following the conclusion of the event.
13. The accounting department updates the bi-weekly payroll.

Manage Schedule Subflow:

Event Manager Cancels Shift:-

1. Event Manager notifies the HR Scheduler regarding update to the original deployment
2. HR Scheduler removes the employees from the schedule and notifies the employees

Event Employee Cancels shift:-

1. Event employee contacts HR Scheduler to be removed from schedule
2. HR Scheduler removes employee
3. HR Scheduler adds a replacement employee
4. HR Scheduler notifies employee added to the schedule

Stage 2: Determine Steps. We use the business level use case narrative as our starting point to identify major segments of the business process. While we recognize there are issues with using use cases to model business processes (Odeh & Kamm, 2003), the use case narrative provides a linear, temporal description of the process under investigation. In this example, the process can be summarized at a high level in a series of four steps: "a request is made, the event is scheduled, the employees are scheduled, and the employees get paid." Of course, the previous simple description ignores all the details uncovered and documented in the use case and supporting models, but the brief description provides the major processes for decomposing the system. For example, workflow statements 1 through 4 in the use case narrative are noted as "step 1: accept or reject events." The rest of the steps are shown in Figure 7.

It is important to note that the concept of "step" is not well defined. In general, a step should constitute a complete concept or function in the system. The number of work flow statements comprising the step should not be a factor in deciding the number of steps. For example, step 4 in Figure 7 represents the complete workflow to update payroll consisting of only two statements. This is significantly different from number of use case statements covered in step 3 detailing how to create and update a schedule.

Stage 3: Decompose diagrams. A central condition at this stage is that each diagram must be decomposed into the same number of steps identified in stage 2 to maintain visual consistency and familiarity. Users will expect to see chunks of each model in every combination diagram or they may be distracted by searching for the "missing" components.

In our example, the first step describes the process of requesting and deciding whether to host an event. The two actors are the promoter and the marketing manager. Consequently, the diagrams are decomposed based on workflow processes involving the two principle actors along with activities and structure associated with them. Figure 8 shows the combined diagram for step 1.

The decomposition of the use case diagram includes the principle actors involved in step 1 only. The activity diagram is decomposed based on the activities associated with these principle actors. The activities are "promoter requests event" performed by the promoter and the "marketing manager reviews request", "accept or reject" decision node, and the "assign event to event manager" activities performed by the Marketing Manager. The class diagram is decomposed based on classes associated with step 1. The class module for step 1 includes the "promoter", "event", "event manager", and "employee" classes. The modules for the remaining steps are derived using the same process. Figures 9 to 11 show the combined diagrams for steps 2, 3, and 4 respectively.

Stage 4: Combine Diagrams. The final stage places the decomposed modules together on a page or screen. We suggest the following guidelines be used to create the combination diagrams: 1) identify the parent diagram of each module on every combination diagram, 2) use the same orientation and layout style for the modules as used on the parent diagram, and 3) clearly separate the modules.

Identifying the parent diagram provides users with a reference to quickly relate the module to the initial diagram. This becomes increasingly important when the diagrams are used with the MUI system described in the next section. Using the same orientation and layout style provides consistency and necessary visual cues to the user to easily relate the module to the parent diagram. This is consistent with results from Kim et al. (2000). For example, objects in the activity diagram module of step 2 (Figure 9) may have been rearranged horizontally from left to right; however, the vertical presentation matches the layout and orientation of these activities on the parent activity diagram (Figure 5). Finally, separating the modules by using a divider eliminates a possible additional source of complexity. For example, a user may mistakenly assume that the last activity in step 2 leads to the "shifts" class if a divider is not used to separate the activity module from the class module.

Figure 4: Use case diagram

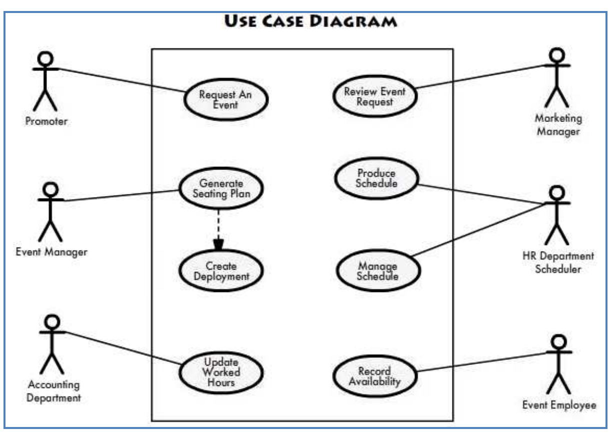


Figure 5: Activity diagram

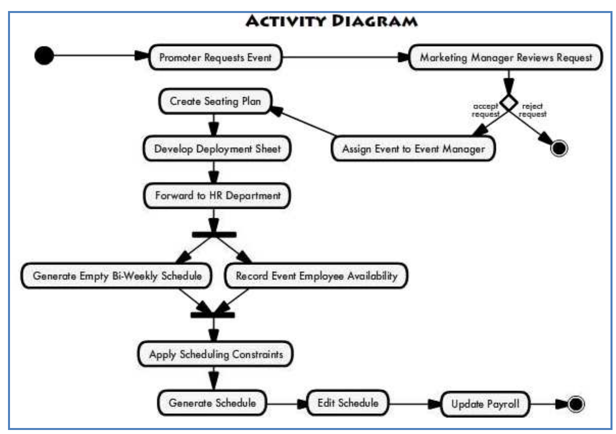


Figure 6: Class diagram

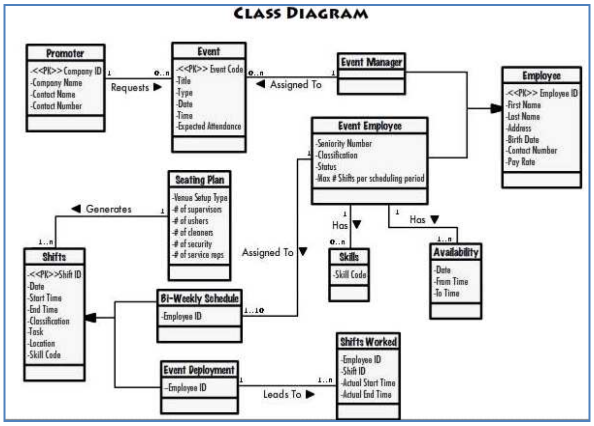


Figure 7: System decomposition into the 4 steps

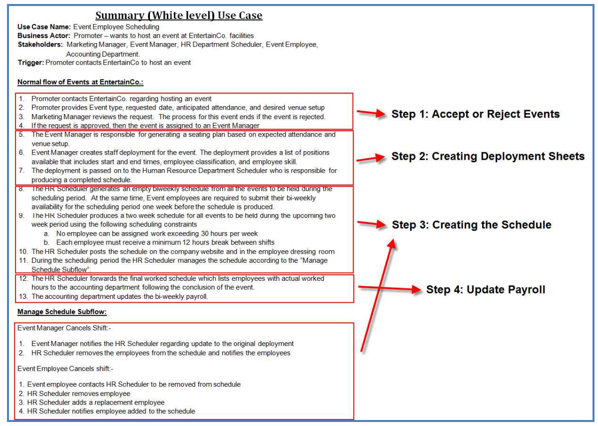


Figure 8: Combination diagram for step 1

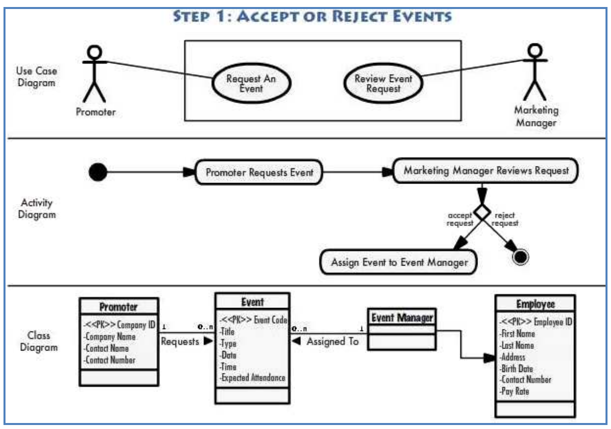


Figure 9: Combination diagram for step 2

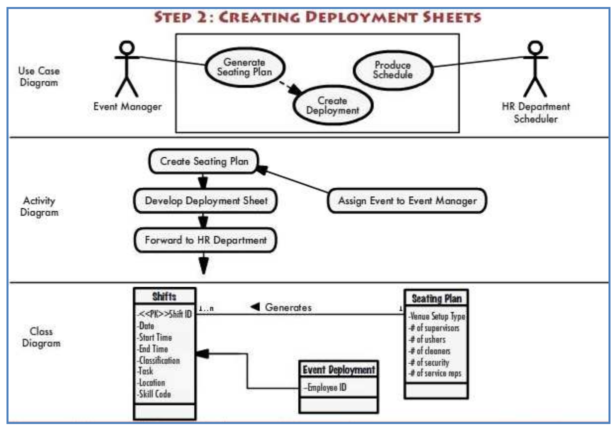


Figure 10: Combination diagram for step 3

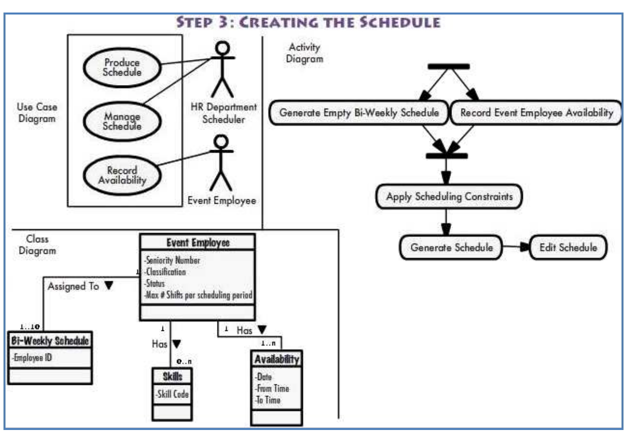
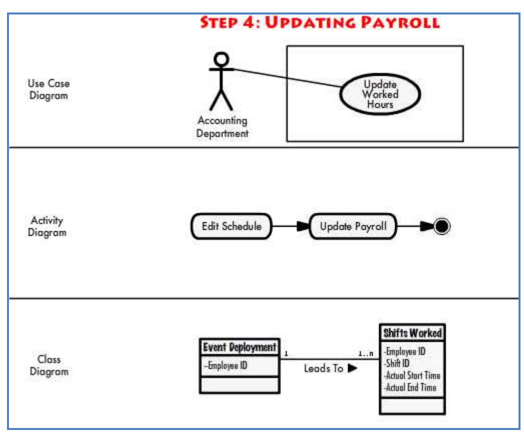


Figure 11: Combination diagram for step 4



CONCLUSIONS AND FUTURE RESEARCH

We have argued that conceptual models, such as those provided through the UML, provide important boundary objects that can be used by analysis and business stakeholders to overcome decontextualization and more effectively communicate about work systems. We have shown that the complexity of the UML, one of the more popular methods for developing conceptual models, may limit its ability to serves as a usable boundary object for work system analysis. As the UML continues to be seen as a standard for systems analysts, refinement will inevitably continue. This paper brings a suggested refinement, Modular UML, which proposes a novel combined presentation of UML model. Rather than proposing a new methodology, we propose a presentation format for existing UML diagrams that does not require changes to the UML specification. An illustrative example provides a demonstration of the potential of this technique. Preliminary experimental testing the effectiveness of modular UML on user understanding through a series of controlled trials provides support for the propositions presented in this paper (Masri, Parker, & Gemino, 2008).

We believe that a modular approach to UML can effectively address the issues of complexity associated with multiple diagrams in UML. While we see potential in a modular approach to UML, we also recognize the need to more formally develop a process for creating modules. We therefore look forward to future work in developing and extending these ideas.

REFERENCES

1. Alter, S. (2003) 18 Reasons Why IT-Reliant Work Systems Should Replace "The IT Artifact" as the Core Subject Matter of the IS Field, *Communications of the Association for Information Systems, 12*, 23, 366-395.
2. Alter, S. (2008) Defining information systems as work systems: implications for the IS field, *European Journal of Information Systems, 17*, 5, 448-469.
3. Avison, D. and Fitzgerald, R. (2006) Information Systems Development: Methodologies, Techniques and Tools (4th ed.), McGraw Hill, Berkshire, UK.
4. Batra, D. (2008) Unified Modeling Language (UML) Topics: The Past, the Problems, and the Prospects, *Journal of Database Management, 19*, 1, i-vii.
5. Bechky, B. A. (2003) Sharing Meaning Across Occupational Communities: The Transformation of Understanding on a Production Floor, *Organization Science, 14*, 3, 312-330.
6. Bechky, B. A. (2006) Talking About Machines, Thick Description, and Knowledge Work, *Organization Studies, 27*, 12, 1757-1768.
7. Benbasat, I. and Zmud, R. W. (2003) The Identity Crisis Within The IS Discipline: Defining And Communicating The Discipline's Core Properties [Article], *MIS Quarterly, 27*, 2, 183-194.
8. Bock, G.-W., Zmud, R. W., Kim, Y.-G. and Lee, J.-N. (2005) Behavioral intention formation in knowledge sharing: examining the roles of extrinsic motivators, social-psychological forces, and organizational climate, *MIS Quarterly, 29*, 87-111.
9. Carlile, P. (1997). *Transforming knowledge in product development: Making knowledge manifest through boundary objects.* University of Michigan, Ann Arbor, MI.
10. Carlile, P. (2002) A pragmatic view of knowledge and boundaries: boundary objects in new product development, *Organization Science*, 442-455.
11. Dobing, B. and Parsons, J. (2006) How UML is used, *Communications of the ACM, 49*, 5, 109-113.
12. Dobing, B. and Parsons, J. (2008) Dimensions of UML Diagram Use: A Survey of Practitioners, *Journal of Database Management, 19*, 1, 1-18.
13. Dori, D. (2001) Object-Process Methodology Applied to Modeling Credit Card Transactions, *Journal of Database Management, 12*, 1, 4-14.
14. Dori, D. (2002) Why significant UML change is unlikely, *Communications of the ACM, 45*, 11, 82-85.
15. Eisenhardt, K. and Tabrizi, B. (1995) Accelerating adaptive processes: product innovation in the global computer industry, *Administrative Science Quarterly, 40*, 1, 84-110.

16. Erickson, J. (2008) A Decade and More of UML: An Overview of UML Semantic and Structural Issues and UML Field Use, *Journal of Database Management, 19*, 3, i-vii.
17. Fowler, M. (2004) UML Distilled: Applying the Standard Object Modeling Language (3rd ed.), Addison-Wesley, Boston, MA.
18. Gemino, A. and Wand, Y. (2003) Evaluating modeling techniques based on models of learning, *Communications of the ACM, 46*, 10, 79-84.
19. Glezer, C., Last, M., Nachmany, E. and Shoval, P. (2005) Quality and comprehension of UML interaction diagrams-an experimental comparison, *Information and Software Technology, 47*, 10, 675-692.
20. Grossman, M., Aronson, J. E. and McCarthy, R. V. (2005) Does UML make the grade? Insights from the software development community, *Information & Software Technology, 47*, 6, 383-397.
21. Hugos, M. H. (2007) Five Diagrams Beat A Victorian Novel, *Computerworld, 41*, 39, 23-23.
22. Jasperson, J., Carter, P. E. and Zmud, R. W. (2005) A Comprehensive Conceptualization of Post-Adoptive Behaviors Associated with Information Technology Enabled Work Systems, *MIS Quarterly, 29*, 3, 525-557.
23. Juhn, S. and Naumann, J. (1985). The Effectiveness of Data Representation Characteristics on User Validation. *Paper presented at the International Conference on Information Systems*, Indianapolis, Indiana.
24. Kabeli, J. and Shoval, P. (2005) Comprehension and quality of analysis specifications--a comparison of FOOM and OPM methodologies, *Information and Software Technology, 47*, 4, 271-290.
25. Kim, J., Hahn, J. and Hahn, H. (2000) How Do We Understand a System with (So) Many Diagrams? Cognitive Integration Processes in Diagrammatic Reasoning, *Information Systems Research, 11*, 3, 284-303.
26. Kim, Y.-G. and March, S. T. (1995) Comparing Data Modeling Formalisms, *Communications of the ACM, 38*, 6, 103-115.
27. Kobryn, C. (1999) UML 2001: a standardization odyssey, *Commun. ACM, 42*, 10, 29-37.
28. Lang, M. and Fitzgerald, B. (2006) New Branches, Old Roots: A Study of Methods and Techniques in Web/Hypermedia Systems Design [Article], *Information Systems Management, 23*, 62-74.
29. Leonard, D. and Sensiper, S. (1998) The Role of Tacit Knowledge in Group Innovation, *California Management Review, 40*, 3, 112-132.
30. Lindland, O. I., Sindre, G. and Solvberg, A. (1994) Understanding quality in conceptual modeling, *IEEE Software, 11*, 2, 42-49.
31. Masri, K., Parker, D. and Gemino, A. (2008). *Combining UML Diagrams to Enhance Understanding of Conceptual Models*. Paper presented at the ASAC 2008.
32. Mayer, R. E. (2001) Multimedia Learning, Cambridge University Press, New York, New York.
33. Ndofor, H. A. and Levitas, E. (2004) Signaling the Strategic Value of Knowledge, *Journal of Management, 30*, 5, 685-702.
34. Odeh, M. and Kamm, R. (2003) Bridging the gap between business models and system models, *Information and Software Technology, 45*, 15, 1053-1060.
35. Shoval, P. and Kabeli, J. (2001) FOOM: Functional and Object-Oriented Analysis and Design of Information Systems: An Integrated Methodology, *Journal of Database Management, 12*, 1, 15-25.
36. Siau, K. and Cao, Q. (2001) Unified Modeling Language (UML): A Complexity Analysis, *Journal of Database Management, 12*, 1, 26-34.
37. Siau, K. and Loo, P.-P. (2006) Identifying Difficulties in Learning UML, *Information Systems Management, 23*, 3, 43-51.
38. van den Hooff, B. and Huysman, M. (2009) Managing knowledge sharing: Emergent and engineering approaches, *Information & Management, 46*, 1, 1-8.