The Multi-dimensional Unified Process for Inter-Organizational Collaborative Software Development

Research Abstract

Cyril S. Ku
Department of Computer Science
William Paterson University
Wayne, NJ 07470, USA
kuc@wpunj.edu

Vassilka Kirova
Alcatel-Lucent
600-700 Mountain Ave.
Murray Hill NJ 07974, USA
vassilka.kirova@alcatel-lucent.com

Thomas J. Marlowe
Department of Mathematics and Computer Science
Seton Hall University
South Orange NJ 07079, USA
thomas.marlowe@shu.edu

Norbert Jastroch
MET Communications GmbH
Eschbacher Weg 10
61352 Bad Homburg, Germany
norbert.jastroch@metcommunications.de

***Abstract*—Software development process is collaborative in nature, especially for large, complex, and innovative systems. From requirements acquisition, analysis and specification, software design, to deployment and maintenance; many teams with diverse expertise are involved in the process. Lately, the collaborative process has shifted from intra-organizational to become more inter-organizational and often inter-cultural and international. The Unified Process is one of the software development models that captures the incremental and iterative nature of software development. In this abstract, we explore the extension of the Unified Process from the traditional 2-dimensional model to a multi-dimensional model to specify the inter-organizational collaborative aspect of software development and management.**

***Keywords*-Collaboration, Collaborative Software Engineering, Software Development Process, Software Engineering, Software Engineering Model, Unified Process**

I. Introduction

Collaborative software development in recent years has become increasing popular due to many factors, including the advanced technology of computing environments, interfaces, internet, network communication, wireless, and in combination of economic factors. Many organizations take advantage of technology progress and gain competitive edge in less expensive labor markets and expertise by out-sourcing and/or off-shoring part of their software projects. Sub-contracting and academic-industrial collaboration are very common these days. Inter-organizational collaborative software development is a reality and software engineering must be collaboration-aware [6]. It is logical to think that all aspects of the software development process and their management need to incorporate and support inter-organizational collaboration.

Software engineering models are the abstraction of the development process, specifying what the process involves in terms of development stages (life-cycle phases) and artifacts [3, 8, 11, 12, 14, 15]. We have 1-dimensional models such as the Waterfall Model to the more modern 2-dimensional models of iterative and incremental such as the Spiral Model, Agile Process, Synchronize and Stabilize Model, etc. For object-oriented software design and development, UML [1, 8, 13] and the Unified Process [3] have become the *de facto* standard.

The Unified Process is also a 2-dimensional model. Since it is a popular model and we would like to make this model accommodate the inter-organizational collaboration aspect, we propose a multi-dimensional model for the Unified Process.

II. The Unified Process

The Unified Process is a development-focused structure with two fundamental dimensions: the technical (workflow or discipline) dimension focuses on development activities, and the business dimension on staging, timeline and progress. The workflows are standardly identified as (Business Modeling, Requirements, Analysis and Specification, Design, Implementation, Testing, Maintenance), and correspond, more-or-less, to the phases in the classical, waterfall model. There are two other workflows, at least in a general sense, Evolution and Reflection/Optimization. Minor evolutionary changes—whether adaptive, preventative, corrective, or perfective—fit well into the iterative incremental model, while major evolution is, rather than a workflow within the development process, a governor of transitions between product (or product line) iterations, resulting in the creation of new variants or versions.

The reflection and optimization workflow has similar characteristics—minor changes in the development process, or business rules, processes and practices, or risk and knowledge management can be incorporated (more or less gracefully) within the development process, but major changes—either in magnitude or direction—will probably have to wait for the transition between iterations. As a third possibility, identification of pervasive problems and the resulting changes may require revisiting and review or reworking of significant amounts of prior work.

Likewise, six distinct business phases can be identified: initiation, inception, elaboration, construction, transition, and maintenance and evolution. The middle four phases are the traditional analysis, design and coding phases identified in, for example, [3,8].

Initiation comprises preparatory activities by the parties to a software development venture—in particular, evaluation of enterprise models, expertise and preparedness, and development of either a request for proposals/bids (by the client) or a proposal for product development (by the developer), entailing a preliminary identification of product scope, and culminating in an agreement to explore development.

Finally, everyone agrees that maintenance and evolution is a necessary phase of the development process, requiring by far the most time and effort, but, since it cannot be scheduled in neat, time-boxed iterations with predefined deadlines and milestones, and because it is not, by definition, a front-end activity, it is often not covered with the others.

III. The Multi-dimensional Unified Process

A. Technical and Business Dimensions

The set of aspects/components in the technical and business dimensions of the MUP is (at this time) largely unchanged from the aspects mentioned above, although each aspect will have to be enhanced to accommodate collaboration. For example, in a collaborative venture, the set of partners involved needs to be stabilized either during initiation, or very early in inception.

B. Collaborative Dimension

Many facets of a project will be affected by collaboration [4, 5, 6, 10]. There will be some new activities, and some standard activities will need a very different approach in a collaborative setting.

1. Partner responsibilities, policies and practices [10]: Management issues, team organization, project scoping, decomposition of project/product, project structure, responsibility assignment, responsibility for client and supplier relationships, responsibility for maintenance and evolution.
2. Partner capabilities and resources: Resource assessment, acquisition and training; human resources, technical expertise, budget, hardware, software/application environment.
3. Software development process: Development process standardization, interface specification and first-class interfaces [9], testing responsibilities including interfaces and cross-partner integration testing.
4. Knowledge management and risk management [6, 10]: Collaborative risks and structures; integrated, collaborative and emergent knowledge, credit-use assessment; need for partner internal component and business knowledge for focus, debugging and maintenance, and quality assessment.
5. Quality assessment and consistency: Monitoring and metrics, software configuration management.

These issues are partly addressed by interoperability [2, 7], which applies to most or all of these facets and dimensions, as well as some mentioned below. However, interoperability is more focused on providing a platform and means of communication than in solving the underlying problems.

C. Management Dimension

Technical assessments in the previous sections interact with and are constrained by corporate and other issues affecting the individual partners [5, 6, 10]. On the one hand, legal issues, security and intellectual property concerns, management objectives, and systems objectives constrain the willingness and ability of the collaborators, while trust and familiarity, formal and informal communication frameworks, the existence of mediation/arbitration processes, and the like affect the readiness for collaboration and the success of a given collaboration. Partners will of course apply cost/benefit analysis, including opportunity and risk, in deciding whether to pursue a given collaborative venture.

D. Inter-Dimensional Relationships

There are, of course, multiple dependences between/among artifacts and activities in these dimensions--we will explore a number of these in the presentation and full paper. For one example, the initial partition of responsibilities and decision to proceed with both the venture and the collaboration is central to initiation and inception in the Business Dimension, to business modeling and requirements analysis in the Technical Dimension, to knowledge management and risk management in the Collaborative Dimension, and to security and intellectual property concerns, as well as cost/benefit analysis in the Management Dimension.

IV. Conclusion and Future Work

The Multi-Dimensional Unified Process provides a more realistic model of software development for complex products and projects, in an inter-organizational collaborative environment. It extends and generalizes the current Unified Process model in being sensitive to the demands and costs of collaboration, and to the inherent hierarchical nature of project decomposition in a collaborative venture. However, the full power of this approach may not be needed for every collaborative venture. The proposed approach is best suited for complex collaboration—projects with some or all of the following characteristics: substantial dependences and interactions between partners and between partner components, integration of knowledge management and risk analysis and management is critical, high innovation affecting interfaces or multiple components, and testing, debugging, metrics, and product evolution likely to require integration of changes in individual components and those in interfaces between partner components. For less complex or innovative, highly standardized or idiomatic, change-local products, a simpler approach may be effective, such as the plug-and-play model in Teichmann [16].

V. Acknowledgement

This research was supported in part by the ART (Assigned Released Time for Research) program, Office of the Provost, William Paterson University of New Jersey.

References Consulted

[1] A.Booch, G., Rumbaugh, J., and Jacobson, I., The Unified Modeling Language User Guide, Addison Wesley, 1998.

[2] European Commission: Enterprise Interoperability Science Base. http://cordis.europa.eu/fp7/ict/enet/fines-eisb_en.html, last accessed November 10, 2011.

[3] Jacobson, I., Booch, G., and Rumbaugh, J., The Unified Software Development Process, Addison Wesley, 1999.

[4] Jastroch, N., Kirova, V., Ku, C. S., Marlowe, T. J., and Mohtashami, M., “Software Engineering Must Be Collaboration-Aware,”

Proceedings of the 22nd International Conference on Software & Systems Engineering and their Applications (ICSSEA 2010), Paris, France, December 7-9, 2010.

[5] Jastroch, N., Kirova, V., Ku, C. S., Marlowe, T. J., and Mohtashami, M., "Adapting Business and Technical Processes for Collaborative Software Development," Proceedings of the 17th International Conference on Concurrent Enterprising (ICE 2011), Aachen, Germany, June 20-22, 2011.

[6] Jastroch, N., Kirova, V., Marlowe, T.J., "Collaboration, Knowledge and Interoperability — Implications for Software Engineering", submitted to CENT 2012.

[7] F. Koussouris, F. Lampathaki, S. Mouzakitis, Y. Charalabidis, J. Psarras: Digging into real-life enterprise interoperability areas - definition and overview of the main research areas. Contributed to the CENT 2011 Symposium, in: Proceedings of the 15th World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI 2011), Vol. II, Orlando FL, July 2011.

[8] C. Larman, **Applying UML and Design Patterns**. 3rd ed., Prentice Hall, Pub. 2004.

[9] T. J. Marlowe, V. Kirova, "High-level Component Interfaces for Collaborative Development: A Proposal", 2nd International Multi-Conference on Engineering and Technological Innovation (IMETI 2009), July 2009.

[10] M. Mohtashami, T. Marlowe, V. Kirova, F. Deek, Risk-Driven Management Contingency Policies in Collaborative Software Development, International Journal of Information Technology and Management, Volume 10 (2-4), p 247-271, 2011.

[11] Pfleeger, S. L. and Atlee, J. M., *Software Engineering: Theory and Practice,* Fourth Edition, Prentice Hall, 2009.

[12] Pressman, R. S., *Software Engineering: A Practitioner's Approach,* Seventh Edition, McGraw-Hill, 2009.

[13] Rumbaugh, J., Jacobson, I., and Booch, G., *The Unified Modeling Language Reference Manual,* 2nd Edition, Addison Wesley, 2004.

[14] Schach, S. R., *Object-Oriented & Classical Software Engineering,* Eighth Edition, McGraw-Hill, 2010.

[15] Sommerville, I., *Software Engineering,* 9th Edition, Addison Wesley, 2010.

[16] G. Teichmann, E.M. Schwartz, F.M. Dittes: Collaborative Engineering of Inter-Enterprise Business Processes. Special Issue on Collaborative Enterprise, Journal of Systemics, Cybernetics, and Informatics (JSCI), Vol. 9 (5), p. 57 – 64, December 2011.