Published in Proceedings of the 2nd Workshop on Multi-Dimensional Separation of Concerns (MDSOC) co-located with the 22nd International Conference on Software Engineering (ICSE), Limerick, Ireland, June 2000

Dimensions of Concerns in Requirements Negotiation and Architecture Modeling[1]

Paul Gruenbacher Alexander Egyed Nenad Medvidovic

Computer Science Department
University of Southern California
941 W. 37th Place, Los Angeles, CA 90089-0781
{gruenbac, aegyed, neno}@sunset.usc.edu

Abstract: *The development and refinement of system requirements into an architecture satisfying those requirements relies heavily on the successful collaboration of stakeholders with different backgrounds, expertise, and responsibilities. Stakeholders involved in this iterative process need comprehensible views that may be provided through multi-dimensional separation of concerns. Stakeholder objectives, constraints, and agreements captured in a requirements negotiation have to be organized e.g., by system features, system properties, or stakeholder contribution. On the other hand in software architecture modeling, components and connectors are the dominant dimensions of concerns used for decomposition. We will discuss dimensions of concerns used in the WinWin requirements negotiation approach and present the CBSP (Connector, Bus, System, Property) taxonomy supporting the classification and refinement of WinWin negotiation results. We will also discuss tools supporting this process.*

Introduction

Evolving system requirements into a viable software architecture is still mainly based on intuition with little available guidance. Requirements and the system architecture emerge in an iterative process involving multiple stakeholders with conflicting goals, needs, and objectives. An important success-factor is thus to provide stakeholder-relevant views on the evolving requirements and architecture models.

Multi-dimensional separation of concerns is a powerful concept supporting stakeholder-relevant views across life-cycle phases [5, 11]. During the collaborative development of requirements, the separation of concerns helps *organizing* negotiation artifacts (e.g., by feature, system property, stakeholder contribution, priority, etc.). This allows focusing on certain aspects needed for the development or evolution of a software architecture. In a complex, real-world problem hundreds of negotiation artifacts have to be taken into account by the software architect. While improved separation of concerns in the requirements eases the task of the software architect, the major challenge remains on how to evolve and refine requirements into a software architecture. The focus is clearly on *decomposition* by identifying components and connectors of the future system as well as their roles and responsibilities [9].

In this paper we will discuss multi-dimensional separation of concerns in the context of WinWin requirements negotiation approach and the evolution and refinement of WinWin artifacts into architectural models. We propose a set of dimensions of concerns used to organize architecturally relevant artifacts from a software architect's point of view and to support the refinement of high-level requirements into more operational entities.

WinWin requirements negotiation

The WinWin approach aims at making winners of all success-critical stakeholders involved in a project by providing a model that guides stakeholders in expressing and negotiating objectives. Stakeholders express their goals as *win conditions*. Constraints and conflicts among WinConditions are captured as *Issues*. Stakeholders propose *Options* to reconcile issues and to describe candidate solutions. If consensus is achieved stakeholder requirements are captured as *Agreements* [2].

[1] This research is sponsored by the Austrian Science Fund (Erwin Schrödinger Grant 1999/J 1764), by DARPA through Rome Laboratory under contract numbers F30602-94-C-0195 and F30602-99-C-0174, and by the Affiliates of the USC Center for Software Engineering.

Real-world WinWin negotiations may contain hundreds of artifacts. Multi-dimensional separation of concerns is thus vital to effectively take advantage of the gathered rationale. WinWin uses various dimensions of concerns to organize the negotiation space. WinWin artifacts are associated to elements in a taxonomy. The taxonomy helps to structure the end-user domain and provides a set of project-specific dimensions of concerns. The taxonomy is tailored to the needs of a project based on high-level taxonomy elements including system *capabilities* (application features and services), *interfaces* (to the user and other software and hardware systems), system *properties* ("non-functional" requirements), *project and process* (development environment, cost, schedule, etc.), as well as *evolution* artifacts (describing the long-term progression of the system) [3].

In addition WinWin supports further dimensions of concerns: In the EasyWinWin collaboration environment [6] stakeholders identify themselves, which allows organizing negotiation artifacts by *stakeholder contribution*. Stakeholders prioritize WinWin conditions by *importance* (business view) and *ease of realization* (technical view). These dimensions of concern help to define the scope of a project and to identify increments for development. Furthermore, the negotiation space can be separated by using individual artifacts (Win conditions, Issues, Options, and Agreements) and following the established links.

CBSP Dimensions of Concerns

Separating different concerns in a negotiation space is beneficial; the viewpoint of the software architect however is on filtering and refining architecturally relevant elements with the objective to develop a software architecture that is consistent with the requirements.

Typically, only a subset of the WinWin negotiation rationale is relevant for architectural considerations. We analyzed the results of a WinWin negotiation that had been carried out to elicit requirements and develop a release plan for a COTS groupware product. Developers, marketers, and executives created about 400 WinWin artifacts. We looked at the 108 Issues from the viewpoint of a software architect to identify the subset of the negotiation dealing with architectural concerns. 20 Issues were related to architectural elements (components and connectors). 15 Issues described properties of architectural elements (e.g., size of client software, security of connection, and so forth). 13 Issues described system-level properties. Three Issues were related to low-level design or implementation. Therefore about 50% of the identified Issues were architecturally relevant.

Based on this experience we developed the CBSP taxonomy of architectural concerns that supports classifying, characterizing, and refining negotiation artifacts. Each WinWin artifact is evaluated for its architectural relevance:

C-artifacts describe or involve a *C*omponent in an architecture.
Win condition: "Users access the system using a web-browser."

B-artifacts describe or imply a connector/*B*us in an architecture.
Win condition: "Interaction with 3rd party COTS products must be supported."

S-artifacts describe *S*ystem-wide features involving multiple components and connectors.
Win condition: "Capability to react to urgent cargo needs."

CP-artifacts describe or imply *C*omponent *P*roperties.
Option: "Reduce size of client software to avoid long download times."

BP-artifacts describe or imply *C*onnector/*B*us properties.
Win condition: "Secure transmission of data."

SP-artifacts describe or imply *S*ystem *P*roperties that should match the emerging properties of a system architecture.
WinConditions: "24/7 availability", "3 seconds maximum response time"

A team of architects analyzes the WinWin artifacts applying the CBSP criteria in a voting process supported by a groupware tool (see Figure 3). In this process, architects rate to what extent a WinWin

artifact fulfills above criteria (not, partially, largely fully). This classification process serves several purposes:

1. Revealing incomplete and puzzling WinWin artifacts: Negotiation artifacts are captured in natural language, which often leads to ambiguity, imprecision, and conflicting perceptions. Conflicting expert opinions in the vote spread may indicate that a WinWin artifact is poorly described and should be re-evaluated.
2. Determining architecturally relevant WinWin artifacts: The voting results help in identifying the subset of WinWin artifacts needed in creating or evolving the system architecture (e.g., all agreements covering system components). The process also points out WinWin artifacts requiring further architectural modeling and simulation to evaluate architectural feasibility and effects of proposed stakeholder requirements. This may involve rapid prototyping of the application, as supported by SAAGE [8, 9].
3. Refining WinWin artifacts: A WinWin artifact belonging to one or many CBSP dimensions can be evolved into smaller-grained WinWin artifacts following these dimensions. For example, a win condition "Ability to react to urgent cargo needs" can be refined into a component win condition "Cargo Monitoring and Planning" and a connector property win condition "Rapid, distributed interaction" (see Figure 1).
4. Improving stakeholder communication: The refinement activity also improves the interaction between the architect and stakeholders in the requirements negotiation process as it helps bridging the gap between high-level requirements and architectural elements through "intermediate" artifacts that can be understood by all stakeholders. The CBSP artifacts serve as a model connector [10, 7] between heterogeneous models [1].

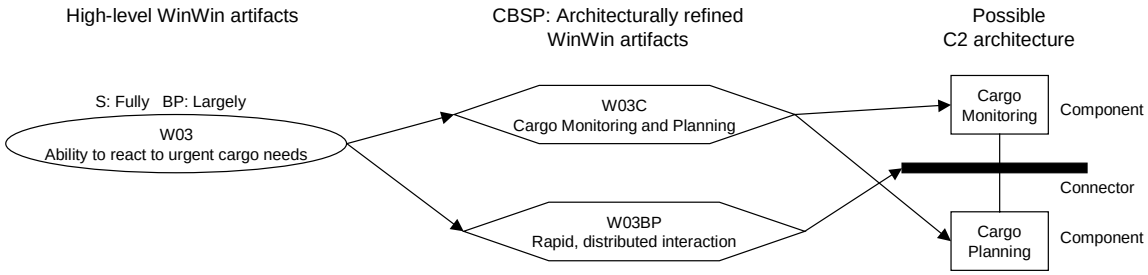

Figure 1: CBSP Refinement example

Figure 1 shows a simple example highlighting the interactions between requirements negotiation and architecture modeling. Win condition *W03* demands a system capability that would allow reacting to urgent cargo needs. The voting process indicates that the win condition should be refined into a component win condition *W03C* and a bus property win condition *W03BP*: The far right part of the figure suggests a possible architectural solution using the C2 architectural style [8]. *W03C* becomes relevant for the components "Cargo Monitoring" and "Cargo Planning". *W03BP* becomes relevant as a property of the connector between these components.

Figure 2 summarizes the dimensions of concerns discussed in this paper into three categories: The requirements category covers typical WinWin taxonomy elements [3] as well as stakeholder contribution and priorities. The CBSP dimensions are represented in the architectural relevance category. Individual artifacts may also serve as a means to separate concerns (see e.g., the negotiation rationale showing WinConditions, Issues, Options, and Agreements and their relationships).

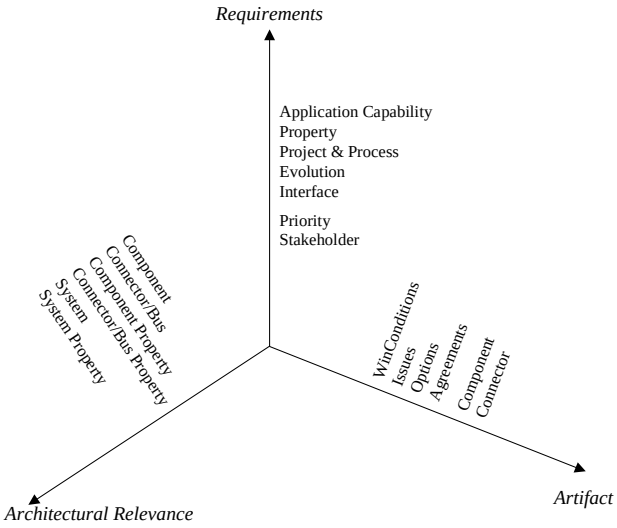

Figure 2: Categories of Dimensions of Concerns

Tool Support

We have developed and integrated tools supporting the process discussed above based on the EasyWinWin environment. EasyWinWin is a new implementation of the WinWin approach that aims to enhance the directness, extent, and frequency of stakeholder interaction [6, 4]. EasyWinWin is based on the COTS groupware product GroupSystems from GroupSystems.com and combines group productivity tools (electronic brainstorming, categorizing, voting, etc.) to improve the involvement of stakeholders and facilitate interactions in the requirements engineering process. EasyWinWin supports brainstorming of WinConditions, categorization and multi-dimensional prioritization of requirements, development and refinement of domain taxonomies, shared definition of terms, as well as negotiations and conflict resolution following the WinWin negotiation model. EasyWinWin supports multi-dimensional separation of concerns as discussed in this paper:

- An interface to the Rational Rose™ CASE tool is provided to support repository-based integration of negotiation results to better understand the different concerns in the negotiation space. This includes the creation of views and hypertext reports for different individual and combinations of concerns (e.g., features, system properties, artifact, stakeholder). The left part of Figure 3 shows a full WinWin rationale graph and two views automatically created by our tool that filters artifacts based on a selection of concerns: “WinWin negotiations initiated by stakeholder Developer”, “WinWin negotiations covering system evolution”.
- The voting tool has been customized in the EasyWinWin environment to support CBSP analyses of requirements negotiation results. The right part of Figure 3 shows voting results from applying the CBSP dimensions to a set of WinConditions. The different colors of the cells indicate the level of consensus among the experts.

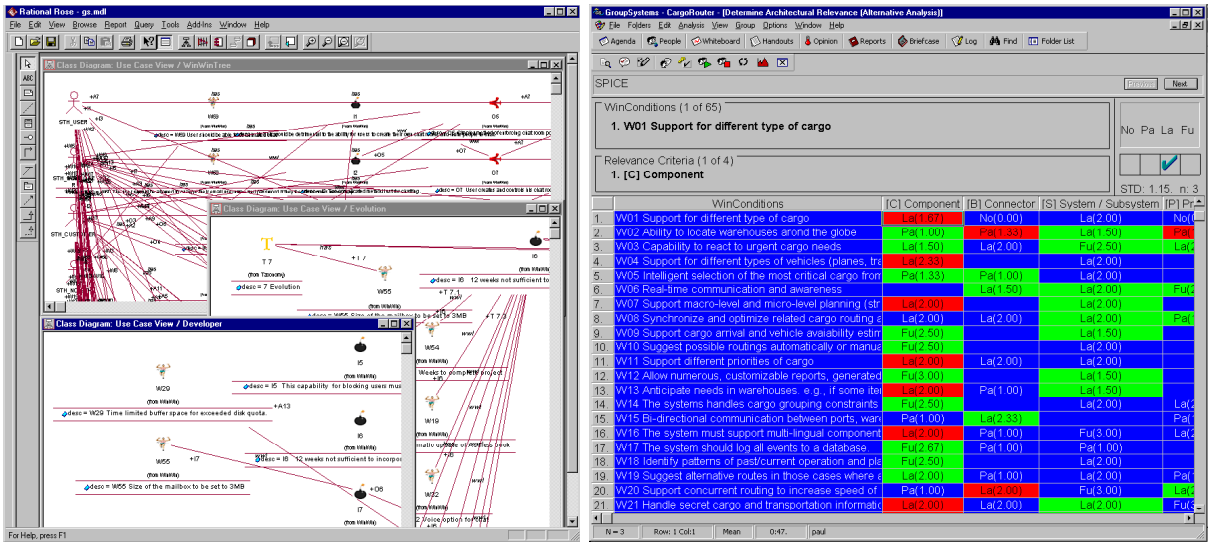

Figure 3: SOC using the EasyWinWin Rational Rose™ extensions and the CBSP voting tool

Conclusions

There is strong need to co-ordinate separated concerns across development activities [1] in order to avoid mismatches in system development. We have discussed the need of multi-dimensional separation of concerns in the context of requirements negotiation and architecture modeling. We have presented the CBSP taxonomy of architecturally relevant concerns to organize the available negotiation artifacts from the viewpoint of a software architect. We developed tools supporting the concept within the EasyWinWin environment, a new implementation of the WinWin approach enhancing WinWin with improved collaboration and prioritization features as well as CASE tool integration. As a next step in this research, we will combine the tools with SAAGE [8, 9] to enhance the integration with architectural modeling and analysis.

References

[1] Bayer, J. Towards Engineering Product Lines Using Concerns, Workshop on the Multi-Dimensional Separation of Concerns in Software Engineering at ICSE 2000.

[2] Boehm B., Egyed A., Kwan J., Port D., Shah A., Madachy R., Using the WinWin Spiral Model: A Case Study, IEEE Computer, 7:33–44, 1998.

[3] Boehm B., Port D., Escaping the Software Tar Pit: Model Clashes and How to Avoid Them, Software Engineering Notes, Association for Computing Machinery, pp. 36-48, January 1999.

[4] Boehm B., Gruenbacher P., Supporting Collaborative Requirements Negotiation: The EasyWinWin Approach, International Conference on Virtual Worlds and Simulation, San Diego 2000 (to appear).

[5] Clarke S., Harrison W., Ossher H., Tarr P., The Dimension of Separating Requirements Concerns for the Duration of the Development Lifecycle, 1st Int. Workshop on MDSOC at OOPSLA 1999.

[6] Gruenbacher P., Collaborative Requirements Negotiation with Easy WinWin, 2nd International Workshop on the Requirements Engineering Process, Greenwich London, September 2000.

[7] Hermann S., Mezini M., Dynamic View Connectors for Separating Concerns in Software Engineering Environments, Workshop on the Multi-Dimensional Separation of Concerns in Software Engineering at ICSE 2000.

[8] Medvidovic N., Rosenblum D.S., and Taylor R.N., A Language and Environment for Architecture-Based Software Development and Evolution. In: Proceedings of the 21st International Conference on Software Engineering (ICSE'99), pp. 44-53, Los Angeles, CA, May 16-22, 1999.

[9] Medvidovic N., Taylor R.N., A Classification and Comparison Framework for Software Architecture Description Languages. IEEE Transactions on Software Engineering, 26:1, January 2000.

[10] Medvidovic N., Gruenbacher P., Egyed A. F., Boehm B.W., Software Lifecycle Connectors: Bridging Models across the Lifecycle, Technical Report, USC-CSE, 2000.

[11] Tarr P., Osher H., Harrison W., Sutton P., N Degrees of Separation: Multi-Dimensional Separation of Concerns, Proceedings ICSE'99, pp 107-119, Los Angeles, CA, May 16-22, 1999.