Using Intentional Actor Modeling to Support the Evolution of Enterprise Software Architectures in Organizations

Daniel Gross, Eric Yu

Faculty of Information, University of Toronto, Canada

Abstract. The development and evolution of enterprise-wide software architectures is influenced by multiple stakeholders and decision makers in development organizations. Architectural design and evolution is embedded in a larger distributed network of organizational participants, who actively contribute to the identification, interpretation, delegation, reasoning, and enacting of decision processes. In this paper we argue that to adequately support architectural design and evolution in development organizations, it is necessary to deal with the inherently distributed and interrelated nature of organizational decision making in development organizations. This paper proposes treating architectural design and evolution as a distributed and interconnected decision process among organizational actors. The utility of the proposed approach is explored through a pilot study at an insurance company during an enterprise SOA evolution effort.

1 Introduction

Architectural decision making in software development organizations occurs within a larger context of organizational decision making. Multiple stakeholders, occupying various positions in the development, client and other third party organizations, are involved in decision making that directly or indirectly influences architectural development and evolution. For example, upper management scans and interprets the organizational environment, such as current customer demands, future market opportunities and the like, and decides on current and future strategies and goals of the organization. Upper management then hands off enterprise strategies to other decision making stakeholders in the organization, such as product management to exercise their know-how in turning strategy into product approaches. Similarly, upper management and product management rely on enterprise architects to develop architectural principles and guidelines for a sound enterprise-wide architecture that supports current and future strategic goals.

Decision making in development organizations is thus a distributed and interconnected phenomenon, in which upstream management decisions have influence on downstream architectural decision making, and where downstream decisions influence upstream goal achievement. An architectural modeling and analysis approach that supports architectural design reasoning and decision making

needs to support representing, capturing and analyzing the distributed and interconnected decision processes in development organizations[1].

2 Objective of Research

The objective of this research is to explore techniques for supporting enterprise architects in capturing, reasoning about and communicating the architectural design goals, principles and guidelines in the context of other organizational decision processes in the development organization. More specifically, this research proposes applying and adapting agent and goal modeling and analysis techniques to support enterprise architects in:

1. representing, capturing and analyzing design reasoning and decision making of designers and stakeholders from different contrasting viewpoints, such as from a component designer's point of view, and from the enterprise architect's point of view;
2. identifying higher level stakeholders and decision makers whose goals, priorities and choices give rise to, and influence the lower level goal and design reasoning and decision making of designers and stakeholders;
3. systematically analyzing how goals and design approaches compare and contrast when analyzed from different points of view and in relation to their links to goals, prioritizations and tradeoff making of higher level stakeholders in an organization

3 Scientific contribution

The main contribution of this research is the application and adaptation of i* [2] to represent, capture and analyze distributed architectural design and decision making in an enterprise development organization at different organizational levels such as the management level, the enterprise architecting level, the enterprise system architecting level and the component designer level; and the exploration of the utility of the proposed approach during a study at an insurance company (to simplify in this paper we use the terms agent and actor interchangeably).

While i* was originally proposed in the context of early requirements engineering, it has also been applied to representing, capturing and analyzing software architecture [3-5]. These approaches have focused on mapping agent and goal-oriented concepts to concepts in the software architecture domain. In these works, agents directly map onto components, dependencies across agents usually capture some quality constraints across interactions of components, and architectural design reasoning and alternatives are captured outside of agents, thereby assuming an anonymous and global designer of the architecture and all components. This underutilizes the i* approach given its ability to deal with distributed intentionality, autonomy and decision making in organizational settings [2, 6].

Furthermore, existing approaches do not link architectural design to relevant business and management decision making stakeholders in the development organization, or to client and third party organizations. The novelty of this work is that it views the representing, capturing and analyzing of such distributed intents and decision making across all such stakeholders in organizations as an essential part of the development and evolution of software system architectures.

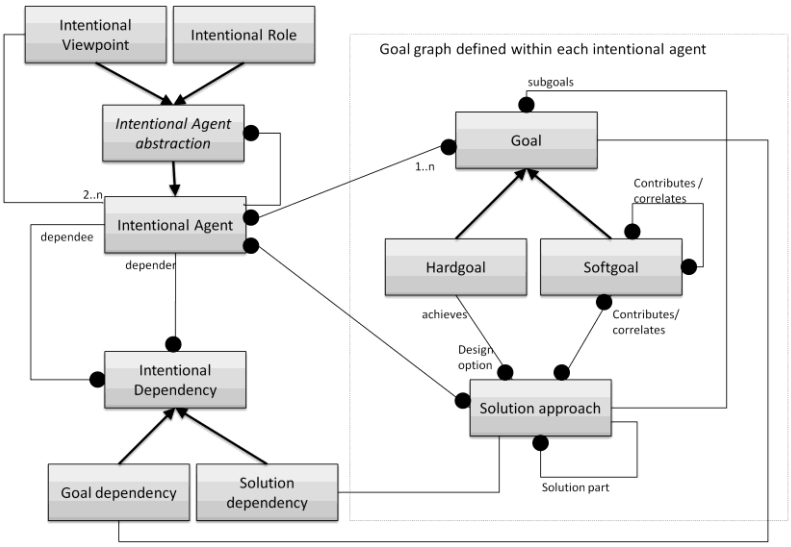

Fig 1: Modeling concept overview

This work can be seen as a significant extension of some early works of the authors [7, 8] in which stakeholders in development organizations contributed intents towards a design team involved in architectural reasoning and decision making. Here the work extends these into dealing with distributed business and architectural decision making in development organizations.

This work also extends i* with *intentional viewpoints*, a novel intentional actor type, and an adaptation of the intentional role concept. Different intentional roles support capturing the reasoning about different non-overlapping design responsibilities. Intentional viewpoints adapt the intentional role concept to reasoning about overlapping design responsibilities. This is analogous to the use of (non-intentional) viewpoints [9] in the usual non-intentional settings, where multiple model fragments are created by different modelers to describe the same phenomenon of interest.

Overlaps between intentional viewpoints are identified when they include overlapping hardgoals (see figure). Given the usual informal design discussions in organizations, is it however difficult to provide clear rules to determine when two hardgoals overlap. Instead, we propose some general guidelines. Given two hardgoals g1 and g2: we consider the goals overlapping if g1 in some way implies by g2. We

suggest looking at the following (non-exhaustive) list of criteria: if g1 is an instance of g2; if g1 is a subtype of g2, if g1 is part of g2; if g1 is an implementation of g2. In the case study we observed that architects and designers adjusted their terminology for each other during discussion. We therefore used the same hardgoal when representing each intentional viewpoint.

The intentional viewpoints concept illustrated in this research captures design reasoning, while non-intentional viewpoints present different design perspectives, which is the result of decision making. Such viewpoint models do not include the design intents and reasoning that lead to design outcome (hence non-intentional models).

4 Conclusions

The utility of the proposed approach was observed during a pilot case study at an insurance company. Intentional viewpoints were used to clearly show and contrast the reasoning of two stakeholders – the SOA architect on the one hand, and the designer of the "Consumer" component on the other. The Consumer component designer is concerned about local design simplicity and maintainability of the consumer component, whereas the SOA architect aims to achieve maintainable, evolvable and scalable enterprise systems, being responsible for the broader enterprise-wide design problem. By placing these intentional viewpoint models side by side the designers felt they were able to communicate better with each other over the design issue at hand. Furthermore, when management issues and reasoning where included (in the form of a network of "management" agents), linking them to the intentional viewpoints of the enterprise architect and the component designer, software design reasoning could be understood within the context of higher level management strategies and prioritizations. These linkages were also considered useful by stakeholders at the study site and were seen as contributing to SOA governance within the Enterprise.

5 Ongoing and Future work

The next steps in this ongoing pilot study are to further analyze design discussions and extend the modeling technique where appropriate; to identify modeling simplifications so that detailed agent and goal oriented design modeling and reasoning can be distilled to a few representative agent and goal elements specifically adapted to the explanatory needs of different types of stakeholders (maintainers, designers, reuse managers, middle and upper management, etc.); to put these simpler models to test in communicating and explaining discussion points with designers and stakeholders; and to obtain relevant feedback and, where necessary, ideas for improvements.

Technical features being explored include developing a systematic approach to compare and contrast the reasoning captured in different intentional viewpoints, while taking into account the different scopes and levels of abstraction each intentional viewpoint may present; the inclusion of different intentional agent types, such as

intentional roles and intentional positions, as well as other knowledge structuring mechanisms across agents, in particular inheritance and instantiation linking between intentional agent types [2], and how these are combined with agent types to support dealing with larger scale capturing and documenting of architectural decision discussions and decision making in development organization; the integration of intentional architectural agent modeling with non-intentional architecture modeling approaches, and the representation of relevant non-intentional knowledge in a notation neutral manner [10]; and appropriate tool support in enterprise organization settings.

References

1. Curtis, W., et al., *On building software process models under the lamppost.* Proceedings of the 9th international conference on Software Engineering, 1987: p. 96--103.
2. Yu, E., *Modeling Strategic Relationships for Process Re-Engineering*, in *Department of Computer Science*. 1994, University of Toronto: Toronto.
3. Gross, D. and E. Yu, *Dealing with system qualities during design and composition of aspects and modules: an agent and goal-oriented approach*, in *Proceedings of the 1st International Workshop on Traceability in Emerging Forms of Software Engineering*. 2002a.
4. Kolp, M. and J. Mylopoulos. *Software Architecture as Organizational Structures*. in *Proceedings ASERC Workshop on "The Role of Software Architectures in the Construction, Evolution, and Reuse of Software Systems*. 2001. Edmonton, Canada.
5. Grau, G. and X. Franch, *On the Adequacy of i* Models for Representing and Analyzing Software Architectures.* Proceedings of the First International Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM'07), 2007: p. 296-305.
6. Yu, E., *Agent Orientation as a Modelling Paradigm.* Wirtschaftsinformatik, 2001. **43**(2): p. 123-132.
7. Chung, L., D. Gross, and E. Yu, *Architectural design to meet stakeholder requirements*, in *Software Architecture*, P. Donohue, Editor. 1999, Kluwer: San Antonio, Texas, USA. p. 545-564.
8. Gross, D. and E. Yu, *Evolving System Architecture to Meet Changing Business Goals: an Agent and Goal-Oriented Approach*, in *Proceedings of the First International Workshop From Software Requirements to Architectures (STRAW 2001) at the International Conference of Software Engineering*. 2001: Toronto, Canada.
9. Nuseibeh, B., J. Kramer, and A. Finkelstein, *A framework for expressing the relationship between multiple views in requirements specification.* IEEE Transactions on Software Engineering 1994. **20**(10): p. 760-773.
10. Gross, D. and E. Yu, *Resolving artifact description ambiguities during software design using semiotic agent modeling*, in *12th International Conference on Informatics and Semiotics in Organisations*. 2010: Reading, UK.