Organizational Model to Derive Multi-Agent Architecture from Requirements

Lúcia R. D. Bastos and Jaelson F. B. Castro

Centro de Informática, Universidade Federal de Pernambuco, S/N, Recife PE, Brazil 50732-970
{lrdb, jbc}@cin.ufpe.br

Abstract. Understanding and supporting the interaction between software requirements and architectures remains one of the challenging problems in software engineering research. To narrow these problems, we are proposing a framework for deriving architectural model in concert with Multi-Agent System requirement specifications. Our framework advocates that a multi-agent system corresponds to the organisational structure, in which actors are members of a group in order to perform specific roles.

1. Introduction

Understanding and supporting the interaction between software requirements and architectures remains one of the challenging problems in software engineering research. In this paper we present an approach for integration of system requirements and software architectures. Our framework proposes that the MASs correspond to the organizational structure, in which actors are members of a group in order to perform specific tasks. An organization comprises groups, members, roles and interactions.

Organisational theories define an organisation as a whole consisting of agents performing tasks [3]. There is a fixed structure of agent tasks and agent communication. Organisation theory describes how practical organisations are actually structured, offers suggestions on how new ones can be constructed, and how old ones can change to improve effectiveness. Hence, the management principles from organisational theory can be applied in multi-agent systems.

Multi-agent systems (MAS) present a useful software engineering paradigm where systems are described as individual agents pursuing high-level goals. MAS approaches use role concepts for modelling social actors in organisational structure [3], and social agents in multi-agent systems [6], [7], [9]. Nevertheless, there is no agent framework or methodology that provides abstractions to support organisational groups, roles and interactions in building MAS following an architectural style.

This paper is structured as follows. Section 2 outlines the bank example. Section 3 outlines the definitions of our framework**.** Finally, Section 4 concludes the paper.

Proceedings of the CAiSE'05 Forum - O. Belo, J. Eder, J. Falcão e Cunha, O. Pastor (Eds.)
© Faculdade de Engenharia da Universidade do Porto, Portugal 2005 - ISBN 972-752-078-2

2. The Bank Example

Banks act as intermediaries between the people who have money to lend and those who have the need for money to carry out business transactions. A bank system can be described in terms of actors that administrate the bank and actors responsible for monitoring the customer accounts. Fig. 1 shows the organisational model of the bank example using the i* notation [8]. The i* (i-star) is the modelling framework for requirements analysis of the Tropos Project [2].

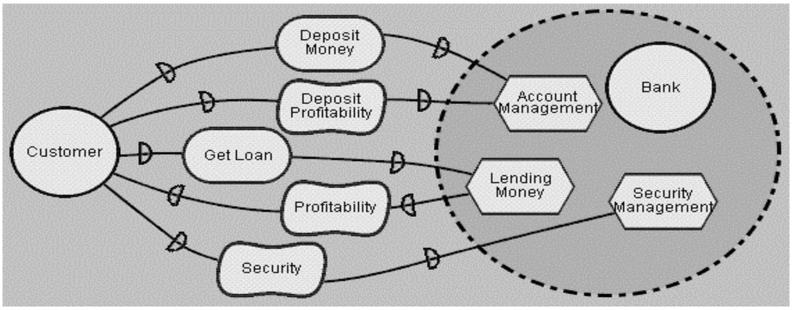

Fig.1. Bank work example in i* notation

In Fig.1, the Bank Actor depends on the Customer's money to invest it, and depends on Customer to request loans and to earn more money (Profitability). The Customer depends on Bank to hold (in deposit or in savings account) his money and to earn a certain amount of money called interest for depositing his money in a bank (Deposit Profitability) and letting the Bank use the money for new investments.

3. The SIRA Framework

The SIRA (***S***ystematic ***I***ntegration between ***R***equirements and ***A***rchitecture using Organisational Concepts) Framework emerges from the understanding that, within organisations, interactions aim at achieving some desired global goals, and the members are autonomous and heterogeneous. The Multi-Agent system has been described from the perspective of an organisation [1]. The SIRA approach is based on a set of complementary analysis models (as seen in Fig. 2): In the Organisational Model, the main goals are identified from the requirement model. A goal is refined into sub-goals or tasks so that each task is performed by a set of roles. The organisation of the social system consists of sub-groups, roles and interactions, as intended by the system and its environment. In the Assignment Model, roles are organised within sub-groups. Subgroups are related to components, based on their similarity with the architectural components. Finally, the Architectural Model describes the resulting architectural configuration, which is the allocation of sub-groups to architectural components.

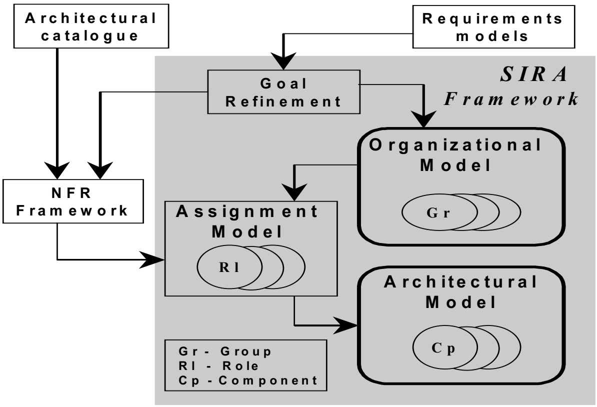

Fig. 2 - The *SIRA* Framework

3.1 Organizational Model

The SIRA Organisational model considers a group as a collection of roles whose behaviour co-operatively determines the accomplishment of organisational goals. SIRA Group is meant to describe systems as organisational group structure at the conceptual level. Groups can be refined into Sub-Groups, Member, Positions or Role in order to represent the social structure of an agent society. In this organisational model, any group is the form $G = \{SG, P, R\}$, where SG is a set of sub-groups SG $\{sg_1, sg_2, \ldots sg_m\}$ such that $i \neq j \Rightarrow sg_i \neq sg_j$ **and** $m \leq n$, and P is a set of positions P $\{p_1, p_2, \ldots p_m\}$ such that $i \neq j \Rightarrow p_i \neq p_j$, and R is a set of roles $R = \{r_1, r_2, \ldots r_n\}$ such that $i \neq j \Rightarrow r_i \neq r_j$. Rather than involving the entire group in a goal, the roles that address a goal are organised into sub-groups. Sub-groups consist of a subset of roles and are used to deal with local goals. Any sub-group is the form $SG = \{R\}$, where R is a set of roles $R = \{r_1, r_2, \ldots r_n\}$ such that $i \neq j \Rightarrow r_i \neq r_j$. Roles encapsulate an actor responsibility and are identified by grouping “related" tasks. The analyst, together with users and other stakeholders, allocates tasks and defines new tasks for specific roles within the organisation.

In SIRA, role identification is done by group tasks with i* means-end analysis, with goal as means [2]. In the Bank example, the Customer actor depends on the Bank actor to achieve his goal *Getting Loan*. Fig. 3 shows the goal refinement for the task *Lending Money*. It is one of the two major tasks of any Bank. *Lending Money* is refined in three sub-tasks: Customer Profile Management, Loan Request Analysis, and Loan Account Management.

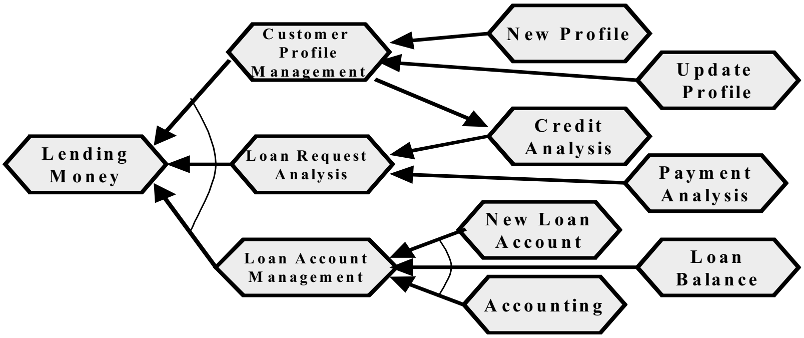

Fig. 3. A partial goal refinement for the task Lending Money

A social network is a graph (sometimes called a sociogram) that consists of a set of nodes (or actors) connected by edges (or relations or ties) [4]. A social graph is a set of relations performed and a set of roles played to fulfil a group goal. The nodes represent the actors and the arrows represent the relations among them. This representation makes it possible to distinguish between roles that interact and those that do not.

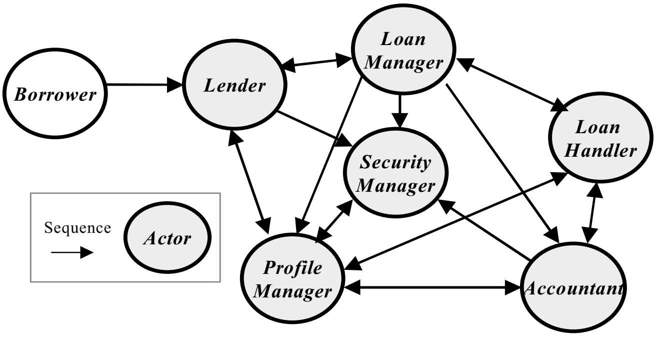

Fig. 4. Social graph for the task Lending Money

Fig. 4 shows the social graph for the task *Lending Money* in the bank example, which uses a directed graph with the actor relations. Each actor (or sub-groups of roles) is represented as a "circle”. The task Lending Money is performed with a set of six roles: Lender is the role performed to interact with Customer (Borrower Role); Loan Handler is a role performed to keep track of all information about loan account; Security Manager is the role performed to provide secure management; Loan Manager is the role performed to manage and control all strategic data; Profile Manager is the role performed to keep track of all information about customers; and Accountant is the role performed to keep track of all information about bank accounting and financial data.

3.2 Architectural Selection

The organisational catalogue from Tropos project are evaluated and compared with the non-functional properties of MAS architectures [2]. In the correlation catalogue for the organisational styles, some software quality attributes for multi-agent architectures were identified from a perspective of organisational styles [5], such as: predictability, security, adaptability, coordinativity, availability, integrity, modularity, aggregability.

In the bank example, one of the most important non-functional requirements when building information systems to be used on the bank domain is security, integrity and availability. In our case study, the selected architectural style is the Structure-in-5, because it contributes more positively for *satisficing* the softgoals security, availability and integrity in the bank domain. In this architectural selection, the Structure-in-5 style is an aggregate of five sub-structures [5].

3.3 Assignment Model

The SIRA Assignment Model proposes the social network analysis as a method for framing and describing the effects of organisational characteristics to relate multi-agent systems and organisational architectures. The Assignment Model addresses two issues:

1. How to Cluster Roles into Sub-groups? Our model clusters like and unlike roles into their respective sub-groups. Cluster analysis groups together the most similar actors, recalculates similarities, and iterates until all actors are combined. The cluster analysis enables to get a view of how the actors in one set are "approximately equivalent" and why different sets of actors are different. Based on the similarity analysis, the actors Loan Manager and Profile Manager are clustered in a sub-group named Service Manager. The current bank example has five sub-groups.
2. How to Match sub-group and architectural components? Our model examines the degree of structural similarity between sub-group and components. In the SIRA framework, the second major way of examining the similarity is to look for patterns of relations and then comparing the behaviour of organisational elements and components, i.e. each architectural component is a sub-group of actors. The components of such architectural organisation have to be *structurally equivalent or similar* to the SIRA Sub-group with respect to the set of roles that they bridge.

Table 1. Correlation Matrix between sub-groups and Structure-in-Five components

	Apex	**Mid. Agency**	**Oper. Core**	**Tech.**	**Sup**
Service Holder	++	--	++	-	-
Service Manager	--	++	--	-	-
Product Handler	++	--	++	-	-
General Manager	++	--	++		

Financial Manager	+	-	+	+	+

Table 1 shows the similarity correlation between Sub-groups and Components for achieve the goal Get Loan. The notation models respectively, strong (++) or partial (+) positive association, strong (--) or partial (-) negative associations.

3.4. The Bank Architectural Model

In bank example, a first architectural configuration is obtained from the sub-groups assigned to components of the *Structure-in-5* Style.

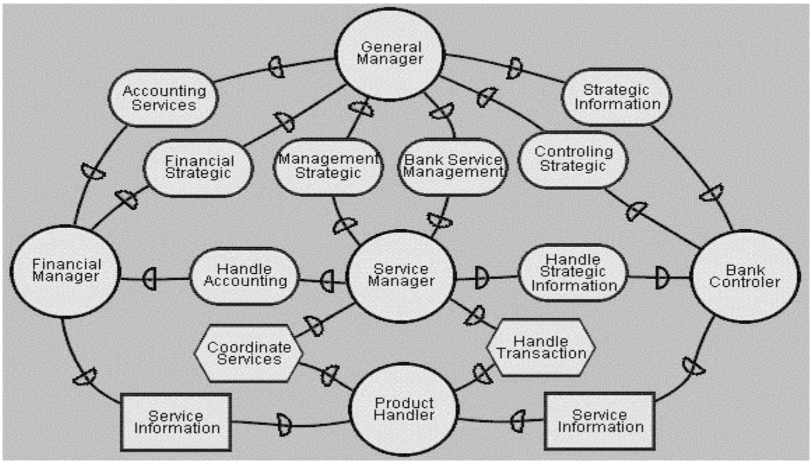

Fig. 5. The Bank Architecture

Based on the results presented in Table 1, the bank architectural configuration is shown in Fig. 5. The bank group elements and architectural components are related as follow:

1. The *Operational Core* is a component that carries out the basic tasks and procedures directly linked to the production of products and services. The sub-group is the Product Handler.
2. The *Strategic Apex* is a component that makes executive decisions to ensure that the organisation fulfils its mission. The sub-group is the General Manager.
3. The *Middle Agency* establishes a hierarchy of authority. The sub-group is the Service Manager.
4. The *Technostructure* serves the organisation by standardising work processes, outputs, and skills. The sub-group is the Bank Controller.
5. The *Support* provides specialised services outside the basic operating workflow. The sub-group is the Financial Manager.

4. Final Considerations

Our organisational model advocates that a system corresponds to an organisational structure, in which actors are members of a group, playing roles in order to perform specific tasks. Roles can be used both as an intuitive concept in order to analyse requirements in multi-agent systems as well as a behavioural structure in order to derive coherent software architectures. Our architectural model describes the MAS at a macroscopic level in terms of a manageable number of sub-groups.

5. References

1. Bastos, L. and Castro, J.: Systematic Integration between Requirements and Architecture. Third International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS´04), Endinburg, Scotland, May 2004. LNCS 3390: ´Software Engineering for Multi-Agent Systema III´, pp. 85-103.R.Choren et all. (Eds.), Springer-Verlag.
2. Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J. and Perini, A.: “TROPOS: An Agent-Oriented Software Development Methodology”. In Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers Volume 8, Issue 3, Pages 203 - 236, May 2004.
3. Fox, M., Barbuceanu, M., Gruninger, M. and Lin, J.: “An Organization Ontology for Enterprise Modelling”. Simulating Organizations: Computational Models of Institutions and Groups, M. Pritula, K. Carley & L. Gasser (Eds), Menlo Park CA: AAAI/MIT Press, pp. 131-152. 1996.
4. Hanneman, R. (2005): “Introduction to Social Network Methods”. On-line textbook. January. http://faculty.ucr.edu/~hanneman/SOC157/TEXT/TextIndex.html.
5. Kolp, M., Giorgini, P. and Mylopoulos, J.: “Organizational Patterns for Early Requirements Analysis”. 15th International Conference on Advanced Information Systems Engineering (CAiSE'03), Velden, Austria. June 2003.
6. Trzebiatowski, G. L. and Münch, I.: “The Role Concept for Agents in Multi-Agent Systems”. Modelling Artificial Societies and Hybrid Organizations. Workshop at KI2001, the Joint German/Austrian Conference on Artificial Intelligence. Vienna, 19-21, 2001.
7. Wood, M.F. and DeLoach, S.A.: “An overview of the multiagent systems engineering methodology”. First international workshop, AOSE 2000 on Agent-oriented software engineering, p.207-221, January 2001, Limerick, Ireland.
8. Yu, Eric and Mylopoulos, John.: “Understanding Why in Software Process Modelling, Analysis, and Design”. Proceedings of 16th International Conference on Software Engineering, May 16-21, 1994, Sorrento, Italy, pp. 159-168.
9. Zambonelli, F., Jennings, N.R., and Wooldridge, M.: “Developing Multiagent Systems: The Gaia Methodology”. ACM Transactions on Software Engineering and Methodology, 12(3): pp. 317-370. 2003.