Modelling Organizational Issues for Enterprise Integration

Eric S. K. Yu[1] and John Mylopoulos[2]

[1] Faculty of Information Studies, University of Toronto, Toronto, Ontario, Canada M5S 3G6
http://www.fis.utoronto.ca/~yu
[2] Dept. of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 3H5

Abstract. Organizational and human issues are often crucial to the successful use of technology in organizations. Enterprise models that make these issues explicit can assist in analyzing issues, finding solutions, and in evaluating alternatives. This paper outlines the *i** modelling framework, in which organizations and work processes are modelled in terms of dependency relationships among strategic actors.

Keywords. Enterprise modelling, organization modelling, strategic dependencies, organizational analysis

1 Introduction

It is generally acknowledged that careful attention to organizational issues is crucial to the success of enterprise information systems. The most sophisticated technology becomes irrelevant or even harmful if it does not not meet human needs. Work context and organizational concerns are increasingly important as information technology is no longer used only to automate highly structured, repetitive tasks, but is offering support to practically all facets of work activities in enterprises today.

System designers have been able to deal with technical complexity with the help of systematic methods and models. Techniques such as functional decomposition and input-output analysis reduce complex systems into manageable subsystems. Organizational issues, however, often defy treatment by these conventional, systems-oriented techniques. For example, frameworks and technologies for enterprise integration (e.g., [7]) holds promise for achieving highly efficient and flexibile operations, overcoming barriers arising from the historical "islands of technology". Yet these technological advances could be ineffectual in the face of organizational and human barriers such as those among professions (e.g., engineering versus marketing), between workers and management, or between frustrated customers and the disinterested front-line worker. Technical systems should be viewed as ingredients and enablers in overall solutions that address human organizational concerns as well as enterprise objectives.

Organizational issues are hard to deal with using conventional systems techniques because of the need to address the fundamentally different nature of human and social relationships. Of course, these issues have been and are being widely studied in many well-established disciplines such as psychology, sociology, and management. The challenge, however, is to cast these issues into a form which can be analyzed and reasoned about during systems analysis and design. Systems issues such as functionality, performance, costs, reliability, etc., will need to be considered at the same time and in interaction with organizational issues such as human cooperation and conflict, power and politics, reward systems, individual differences and culture, as well as with business strategies.

As with the design of complex technical systems, appropriate modelling techniques can be invaluable in supporting the analysis and design tasks. To deal with organizational issues, we need modelling techniques that can express the richness of human, social, organizational relationships.

The i^* framework has been developed for modelling organizations, and to help reason about changes in relationships among strategic actors [9]. It adopts an agent-oriented perspective, which is receiving increasing attention in a number of research areas, including information systems requirements engineering [12]. In i^*, organizations are viewed as consisting of social actors who have freedom of action, but depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. The framework includes a Strategic Dependency model – for describing the network of relationships among actors, and a Strategic Rationale model – for describing and supporting the reasoning that each actor has about its relationships with other actors. These relationships are strategic in the sense that each party is concerned with opportunities and vulnerabilities and seeks to protect or further its interests. The models are formally represented in the conceptual modelling language Telos [5].

The framework as been described in detail elsewhere [9, 10] and in the context of several application domains, including business process reengineering [17, 14, 16], software processes [13], and requirements engineering [8, 11]. In this paper we give an overview of the framework and illustrate its relevance to enterprise integration.

For example, an insurance company may want to use enterprise integration concepts and techniques to link agents, appraisers, and claims managers so as to improve operations. Although one may be able to gain efficiency by automating some existing processes, even greater benefits could potentially be achieved by a more fundamental rethinking of business processes and relationships [3] while paying attention to broader organizational issues [2]. To do this one needs to understand the motivations, intents, and rationales behind the process steps and flow, the "whys" that underlie the "what." Enterprise activities and flows can usually be traced to the wants and desires of various actors, and how these are met by other actors.

An insurance company wants to keep its customers happy so that they will continue to renew their policies. At the same time, it wants to minimize claims payout to claimants, and for this reason hires appraisers to keep repairs to the necessary minimum. Car owners want repair damages to be assessed fairly, and are likely to get body shops to give repair estimates that maximize the insurance payout. What information is collected and used by the claims representative (accident particulars, witness statements) and the appraiser (e.g., photographs of damage, multiple repair estimates) reflects the strategic interests of the various parties. In devising an effective enterprise integration solution, it is crucial to understand the interplay of strategic interests among organizational players.

2 The Strategic Dependency Model

A Strategic Dependency model is a graph, where each node represents an *actor*, and each link between two actors indicates that one actor depends on the other for something in order that the former may attain some goal. We call the depending actor the **depender**, and the actor who is depended upon the **dependee**. The object around which the dependency relationship centres is called the **dependum**. By depending on another actor for a dependum, an actor is *able* to achieve goals that it is otherwise unable to achieve, or not as easily or as well. At the same time, the depender becomes *vulnerable*. If the dependee fails to deliver the dependum, the depender would be adversely affected in its ability to achieve its goals.

For example, a car owner can have his car repaired by a body shop, even if he does not have the ability to do the repairs himself. However, he is vulnerable to the car not being repaired.

The model distinguishes among four types of dependencies – **goal-, task-, resource-** and **softgoal-dependency** – based on the type of freedom that is allowed in the relationship between depender and dependee. Three levels of dependency strengths are distinguished based on the degree of vulnerability. Actors may be differentiated into agents, roles and positions.

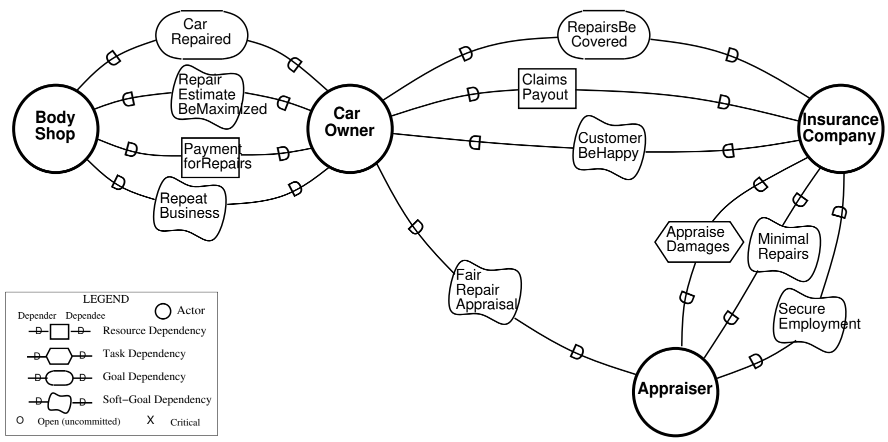

Fig. 1. *Strategic Dependency model of traditional auto insurance (the "as-is" arrangement)*

Figure 1 shows a Strategic Dependency model for a traditional automobile insurance business configuration. The car owner depends on the insurance company to reimburse for the repairs from an accident (ClaimsPayout). For this, car owner pays insurance premium in order to have coverage (RepairsBeCovered). The insurance company wants to offer good service to the customer in order to keep the business (CustomerBeHappy). To maintain profitability, the company depends on appraisers to appraise damages so that only the minimal necessary repairs are approved.

The car owner depends on the claims appraiser for a fair appraisal. However, the appraiser can be expected to act in the interests of the insurance company because of his dependence on the latter for continued employment. The car owner, in turn, can depend on the body shop to give an estimate that maximizes the car owner's interests, since the body shop depends on the car owner for repeat business. A detailed analysis of strategic dependencies would suggest what relationships one should focus on when developing enterprise integration solutions.

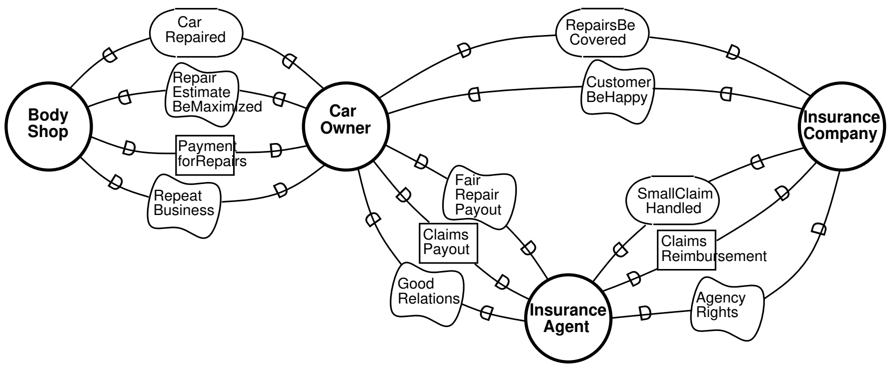

Fig. 2. *Strategic Dependency model for alternative 1 – "Let the insurance agent handle it"*

Figure 2 shows the Strategic Dependency graph for a new business configuration (adapted from a scenario described in [4] pp. 136-143). In this redesign, the insurance agent will do all the inquiry and payout, while the insurance company will concentrate on larger claims that have more significant impact on profitability. The agent gets to cement his relationship with the customer, while the customer is more likely to get a fair hearing from the agent about a fair payout amount. This keeps the customer happy, which is what the insurance company wants.

Shifting the claims handling responsibilities to the insurance agent means that the information needs of the insurance agent (and hence the enterprise information requirements of the insurance company) are also radically altered. Based on the new configuration of strategic dependencies, one could derive what information needs to be shared or sent among insurance agents and the insurance company, and how accurate and up-to-date they need to be.

The Strategic Dependency model encourages a deeper understanding of an organization and its business processes by focusing on intentional dependencies among actors, beyond the usual understanding based on activities and entity flows. It helps identify what is at stake, for whom, and what impacts are likely if a dependency fails.

Although a Strategic Dependency model provides hints about why a process is structured in a certain way, it does not sufficiently support the process of suggesting, exploring, and evaluating alternative solutions. That is the role of the Strategic Rationale model.

3 The Strategic Rationale Model

A Strategic Rationales model is a graph with four main types of nodes – **goal**, **task**, **resource**, and **softgoal** – and two main types of links – **means-ends links** and **task decomposition links**. A Strategic Rationale graph describes the reasoning behind each actor's relationships with other actors, thus revealing the internal linkages that connect external strategic dependencies.

A process is often depicted as a collection of activities with entity flows among them. For example, a claims handling process would include such activities as verifying the insurance policy coverage, collecting accident information, determining who is at fault, appraising damages, and making an offer to settle.

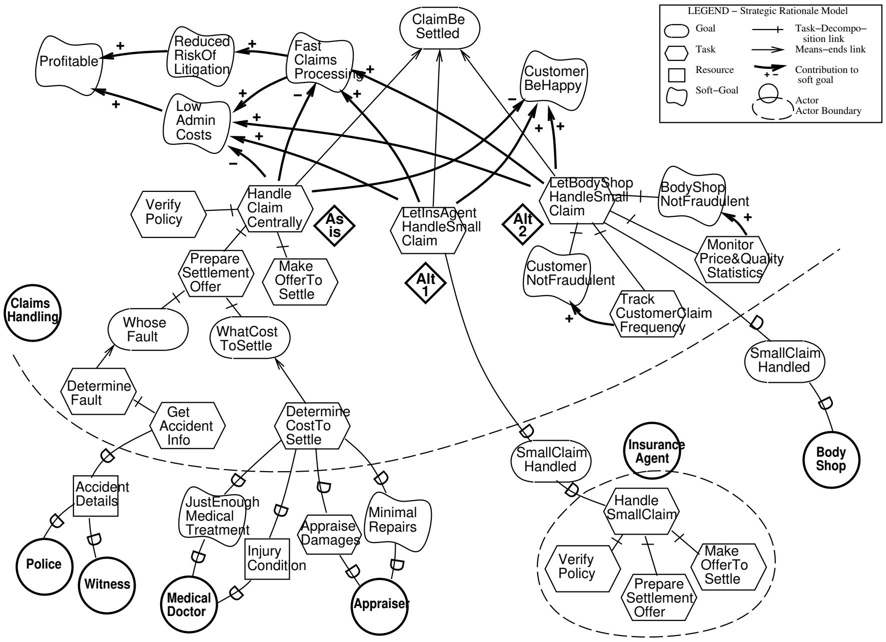

Fig. 3. *Strategic Rationale model to support reasoning about redesigning the claims handling process*

In the Strategic Rationale model, we arrange these into a hierarchy of means-ends relationships and task decompositions (Figure 3). When a process element is expressed as a goal, this means that there might be different possible ways of accomplishing it. A task specifies one particular way of doing things (of accomplishing a goal), in terms of a decomposition into subtasks, subgoals, resources, and softgoals. In seeking ways to redesign a business process, goals offer potential places to look for improvement. An ambitious redesign effort needs to discover and rethink high-level goals – by asking "why" questions, rather than be content with solutions for low-level goals. Higher goals are discovered by asking “why” questions. Once sufficiently high-level goals have been identified, alternatives may be sought by asking “how else” the goals can be accomplished. In introducing enterprise integration solutions, designers should be careful not to merely *automate* existing processes. Innovative systems solutions often result from a more fundamental rethinking of business processes [3] and broader organizational issues [2].

In the auto insurance example described in [4], the reengineering team wanted to consider radical solutions, by identifying a high-level goal: that claims be settled. Unencumbered by current business thinking about how this goal should be accomplished, the team arrived at innovative proposals that involve new strategic business relationships with insurance agents and body shops.

Each alternative may have different implications for a number of quality goals, or “softgoals”, such as CustomerBeHappy, FastProcessing, and Profitable. A softgoal is one which does not have *a priori*, clear-cut criteria of satisfaction. Although some of these can be measured and quantified, a qualitative approach can be used at the stage of exploring the space of alternatives. Contributions to softgoals can be positive or negative, and are judged to be adequate or inadequate. The treatment of softgoals is based on a framework developed by Chung for dealing with non-functional requirements in software engineering [6].

By explicitly representing means-ends relationships, the Strategic Rationale model provides a systematic way for exploring the space of possible redesigns. Generic knowledge in the form of methods and rules can be used to suggest new solutions and to identify related goals [14, 16] .

4 Modelling and Analyzing Organizational Issues

The explicit modelling of strategic relationships among actors offers a systematic way for expressing and analyzing organizational issues. Consider a software development organization whose mandate is to develop and deliver quality software on time and on budget. While the operations of the organization can be described quite readily in terms of activities, entities, and input/output flows, many important organizational issues cannot be expressed in such terms. Some questions that one might ask such an organization may include:

- Is the system likely to be delivered on time and on budget?
- Are the needs of the end users likely to be met?
- Is the project team structure conducive to high quality software?

These questions are hard to answer using conventional enterprise models.

Figure 4 shows the strategic relationships among the customer, the end-user, the project manager, and the general manager in a hypothetical software project organization. The general manager depends on the project manager not to overrun the project budget and schedule. This dependency is likely to succeed because the project manager critically depends on the general manager for recognition of achievements. The customer’s dependency on the project manager for quick delivery of the system is also likely to succeed. Even though the project manager has no direct dependency on the

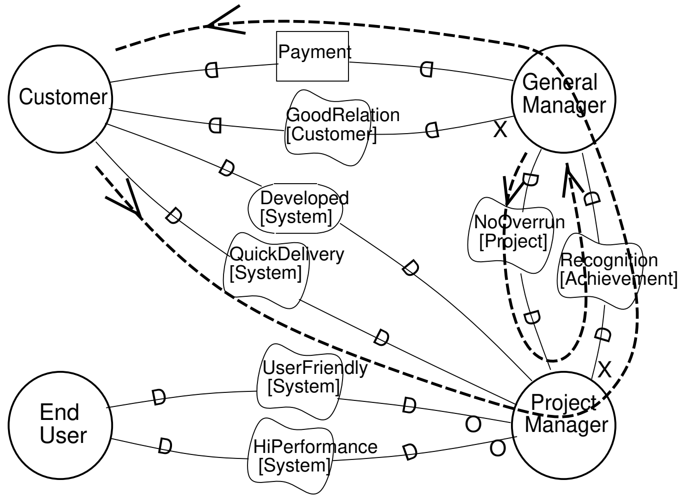

Fig. 4. *Example of loop analysis*

customer, there is an indirect dependency due to the project manager's dependency on the general manager for recognition, and the general manager's dependency on the customer for payment and for good relations. On the other hand, the end-user's desire for a user-friendly and high performance system may not be met if there are no dependencies from the project manager to the end-user.

One could call this kind of analysis "loop analysis", to determine whether there are reciprocal dependencies among actors, either direct or indirect. A formal analysis of the dependency network may not in itself offer useful conclusions, since the relative weights of the dependencies need to be taken into account. In the absence of numerical weights, a qualitative reasoning approach could be used to support systematic reasoning [6].

Figure 5 shows some of the strategic dependency relationships involving design engineers and quality assurance engineers in this hypothetical organization. Here, we distinguish among *agents, roles*, and *positions*. Agents *occupy* positions and *play* roles. A position usually *covers* a number of roles. *Actor* is the generic concept of which agent, role, and position are specializations.

Agents are physically embodied social actors or systems such as Jack and Jill. In this example, Jack is an instance of the agent class Design Specialist. Jill is an instance of QA Specialist. Distinguishing among agents, roles, and positions allows the enterprise modeller to identify and express relationships among them. For example, when a design specialist is hired and placed into the position of design engineer, the former (the agent) has an expectation or desire that the latter (the position) will provide opportunities to do creative, state-of-the-art design. However, there is also a dependency from the Monitoring Progress role of the project manager on the Design Engineer position not to add fancy features (no "gold-plating"). These two dependencies are in

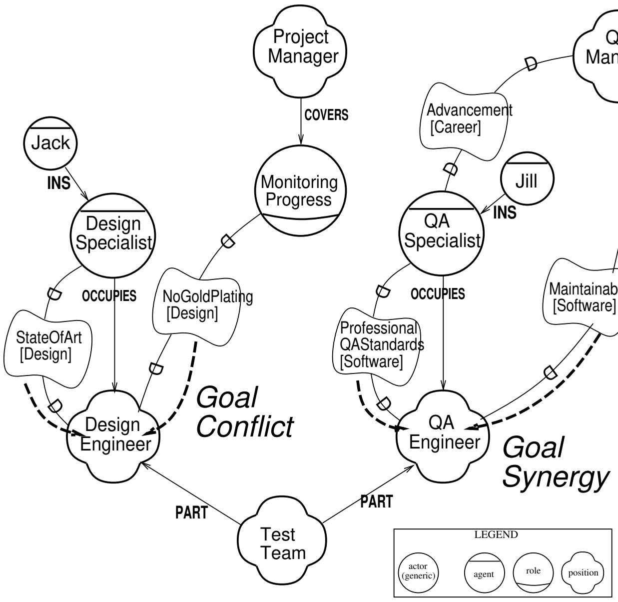

Fig. 5. *Example of node analysis*

conflict, suggesting that one or the other may have to give way, or that some negotiated resolution may be necessary.

In contrast, the QA specialist's desire to uphold professional QA standards is synergistic with the QA manager's dependency on the QA engineer to assure maintainable software. So these two dependencies are likely to be viable through mutual reinforcement. In these examples, we are analyzing the interactions among dependencies as they converge on a node in the dependency graph.

A software tool is being developed to support i^* modelling and reasoning. The tool assists the user in assessing the viability of goals and dependencies, by propagating contributions among goals and other intentional elements [9]. The propagation algorithm for softgoals is based on the qualitative reasoning framework for dealing with non-functional requirements in software engineering [1].

5 Conclusion

The i^* framework for modelling strategic actor relationships outlined in this paper offers a way for explicitly modelling organizational issues and for analyzing them. This kind of modelling and analysis can be of considerable help in identifying enterprise integration solutions that can address the needs of organizations that have complex technical and human organizational environments.

The modelling of strategic actor relationships is complementary to many existing enterprise modelling techniques [7]. Efforts are under way to relate the i^* framework to other models [10], and in particular, to agent-oriented specification languages and process simulation languages [15, 17].

Acknowledgements. Financial support from the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Centre of Ontario are gratefully acknowledged.

References

1. K. L. Chung, *Representing and Using Non-Functional Requirements for Information System Development: A Process-Oriented Approach*, Ph. D. Thesis, also Tech. Report DKBS-TR-93-1, Dept. of Comp. Sci., Univ. of toronto, June 1993.
2. T. Davenport, Saving IT's Soul: Human-Centered Information Management, *Harvard Business Reivew*, March/April, 1994, pp. 119-131.
3. M. Hammer, Reengineering Work: Don't Automate, Obliterate, *Harvard Business Reivew*, July/August, 1990, pp. 104-111.
4. M. Hammer and J. Champy, *Reengineering the Corporation: A Manifesto for Business Revolution*, HarperBusiness, 1993.
5. J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis, Telos: Representing Knowledge about Information Systems, *ACM Trans. Info. Sys.*, 8 (4), 1991.
6. J. Mylopoulos, L. Chung, B. Nixon, Representing and Using Non-Functional Requirements: A Process-Oriented Approach, *IEEE Trans. Soft. Eng.*, 18 (6), June 1992.
7. F.B. Vernadat, *Enterprise Modelling and Integration: Principles and Applications*, Chapman and Hall, London, 1996.
8. E. Yu, Modelling Organizations for Information Systems Requirements Engineering, *Proc. 1st IEEE Int. Symp. on Requirements Engineering (RE'93)*, San Diego, Calif., USA, Jan. 1993.
9. E. Yu, *Modelling Strategic Relationships for Process Reengineering*, Ph.D. thesis, also Tech. Report DKBS-TR-94-6, Dept. of Computer Science, University of Toronto, 1995.
10. E. Yu, "Models for Supporting the Redesign of Organizational Work," *Proc. Conf. on Organizational Computing Systems (COOCS'95)*, Milpitas, California, August 1995, pp. 225-236.
11. E. Yu, Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering, *Proc. 3rd IEEE Int. Symp. on Requirements Engineering (RE'97)*, Anapolis, Maryland, USA, Jan. 1997.
12. E. Yu, Why Agent-Oriented Requirements Engineering, *Proc. 3rd Workshop on Requirements Engineering For Software Quality (REFSQ'97)*, Barcelona, Catalonia, June 1997.
13. E. Yu and J. Mylopoulos, Understanding 'Why' in Software Process Modelling, Analysis, and Design, *Proc. 16th Int. Conf. on Software Engineering*, May 1994, pp. 159-168.

14. E. Yu and J. Mylopoulos, From E-R to 'A-R' – Modelling Strategic Actor Relationships for Business Process Reengineering, *Int. Journal of Intelligent and Cooperative Information Systems*, vol. 4, no. 2 & 3, 1995, pp. 125-144.
15. E. Yu, P. Du Bois, E. Dubois, and J. Mylopoulos, From Organization Models to System Requirements – A 'Cooperating Agents' Approach, *Proc. 3rd Int. Conf. on Cooperative Information Systems (CoopIS-95)*, Vienna, Austria, May 1995, pp. 194-204.
16. E. Yu and J. Mylopoulos, Using Goals, Rules, and Methods To Support Reasoning in Business Process Reengineering, *Int. Journal of Intelligent Systems in Accounting, Finance and Management*, vol. 5, pp.1-13.
17. E. Yu, J. Mylopoulos, and Y. Lesperance, AI Models for Business Process Reengineering, *IEEE Expert*, August 1996, pp. 16-23.