CREWS Report 98-35

in Proceedings of the 11th Conference on Advanced Information Systems
Engineering, CAiSE'99, Heidelberg, Germany, June 14-18, 1999.

Linking Business Modelling to Socio-Technical System Design

by

Alistair Sutcliffe and Shailey Minocha

Centre for HCI Design
School of Informatics
City University
Northampton Square
London EC1V 0HB
United Kingdom

Tel: +44 (0)171 477 8411
Fax: +44 (0)171 477 8859

{a.g.sutcliffe, s.minocha}@city.ac.uk

Linking Business Modelling to Socio-Technical System Design

Alistair G. Sutcliffe[1] and Shailey Minocha[2]

[1]Centre for HCI Design, School of Informatics, City University
Northampton Square, London EC1V 0HB, UK
e-mail: a.g.sutcliffe@city.ac.uk
[2]Faculty of Mathematics & Computing
Open University, Milton Keynes MK7 6AA, UK
e-mail: s.minocha@city.ac.uk

Abstract Few methods address analysis of socio-technical system requirements. This paper describes a method for analysing dependencies between computer systems and users/stakeholders in the operational environment. Domain scenarios describing the system and its context are used to create an environment model based on the i* notation. A method is proposed to define business organisational relationships, according to the coupling between agents determined by types of event flows between them, and secondly, by operationalising transaction cost theory to obtain an a priori view of relationships according to the market context for a client and supplier. Coupling metrics are applied to assess the degree of dependencies between the system and the users. High-level requirements are suggested to deal with different types of organisational design. The method is illustrated with a case study of a service engineer support system.

1 Introduction

Few methods have emerged to analyse socio-technical system requirements, even though many problems in requirements engineering are known to have their origins in complex social problems [1], [2]. Ethnographic techniques have been applied to gather data on social issues and requirements do emerge from this process [3]. However, there is little generalisable knowledge, models or analytic methods that can be gleaned from ethnography, so the quality of requirements analysis is dependent on the practitioner's experience.

Some socio-technical models have been proposed for RE, notably the ORDIT project [4] which describes systems in terms of agents, task and roles. Socio-technical systems approaches advocate a human-centric analysis that investigates the impact of computer systems (the technical system) on people and considers ways in which technology can be design more effectively for people. However, few analytic techniques have been reported for such analyses, so the requirements analyst are still dependent on experience for interpreting such models. The Inquiry cycle [5], [6] uses scenarios to investigate barriers to effective use (called obstacles) that may arise in the social domain. However, the Inquiry cycle does not give detailed techniques for analysing socio-technical system dependencies. Analytic guidance has been given in the stakeholder analysis methods (e.g. [1]), which advise modelling requirements according to different user categories or viewpoints. A key problem in socio technical systems is how to structure and manage relationships between organisational units.

Williamson's theory on transaction cost analysis [7], [8] provides a principled analysis of inter-organisational relationships, yet it does not appear to have been applied to design of business processes and their technological support. A motivation for this paper is to attempt an initial operationalisation Williamson's theory as a method for business organisation design and explore its implications for requirements analysis.

This paper explores the problem of dependency analysis in socio-technical systems by proposing a method for modelling and analysing event flows between users and the intended system in order to derive high level requirements. This extends the work of [9] by addressing workflow problems via a coupling analysis derived from concepts in modular software design [10] and organisational theory [11].

The paper is organised in four sections. The next section introduces the method and this is followed by a more detailed description of two method stages: coupling and transaction cost analysis. A case study of service engineer support system (SESS) runs through these sections to illustrate the method. The paper concludes with a brief discussion.

2 Method Outline and Models

The CREWS-SAVRE (Scenarios for Acquisition and Validation of Requirements) method compares scenarios describing the domain with requirements specifications and models, focusing on events or information flows between the system and its environment. Analysis questions probe the dependencies between people and computers across a tentative system boundary. The system boundaries will change during a requirements investigation as alternative designs emerge; hence there is a single model of the intended system-environment upon which a boundary is imposed. The relationship between scenarios, use cases and the requirements specification is illustrated in Figure 1.

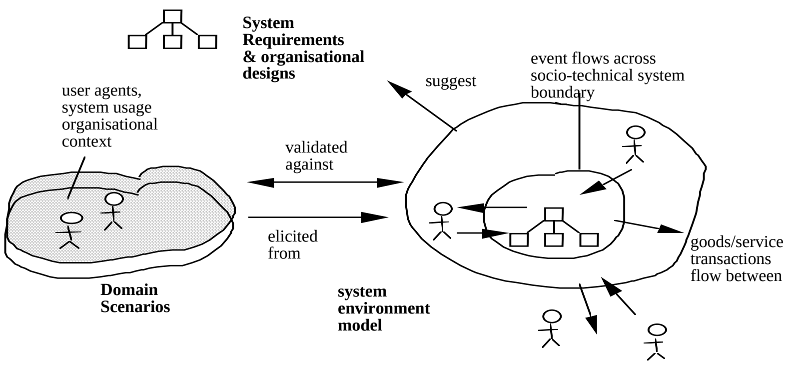

Fig.1. Relationships between Domain Scenarios, Environment Model and the Socio-Technical System Requirements

The method uses "domain" scenarios that contain descriptions of activities in manual systems, interaction with legacy systems, descriptions of agents, roles, and their organisation settings gathered from real-life examples. These are used to elicit facts for the system environment model.

2.1 CREWS-SAVRE Method Outline

The method stages are summarised in Figure 2. The method is iterative, so once the use cases and environment models have been created, all the other stages proceed concurrently.

Stage 1. Use Case and System Modelling

The system environment model is created as an overview model. Then use cases elaborate tasks and interactions between agents to provide more detail. Use case modelling follows standard OO procedures (see [12]), so it is not described further in this paper. Use cases are expressed as interaction diagrams [13] to map out the sequential dependencies of event flows between the agents and the system.

Stage 2. Inbound Event Analysis

Each use case is elaborated by comparing it with one or more domain scenarios and tracing events originating either from human agents or from other objects in the system environment. Each event inbound from the system environment indicates a requirement for a system action, as well as an action in the use case. In this manner the event analysis helps to refine the use cases by identifying events and system responses to deal with the events.

Stage 3. Outbound Event Analysis

Outbound validation is more difficult because the impact on a social system has to be judged. By their nature social systems are complex and unpredictable and the change introduced by a computer system frequently produces unanticipated and undesirable side effects. The outbound event flows are traced to their destination and then questions analyse the acceptability of the system output for the user. First, the domain scenario and use case are cross referenced to ensure output is generated when and where it is needed. Steps in the user's task that imply an information need are identified in the scenario; so if a user needs information at a particular point in the scenario a system output function should exist in the requirements specification.

Stage 4. Coupling Analysis

This assesses the dependencies between the social and technical systems. Coupling analysis is based on software engineering concepts and organisational theory [11], [14] that advises that control based coupling should be avoided. In human organisation control coupling occurs through lines of control, commands, and obligations to carry out activities in response to others. The motivation for avoiding control coupling is twofold. First, it decreases the flexibility of user-system interaction and decreases autonomy. Secondly, too much control imposes the system's goals on the user's working practices, and this may lead to system rejection. The system input and output event flows are classified and counted. The more commands there are, the closer the coupling between the social and technical systems will be. Closely coupled systems are reviewed to either change the design for more automation (computer autonomy) or to increase human control, and design the computer system for an advisory rather than a controlling role.

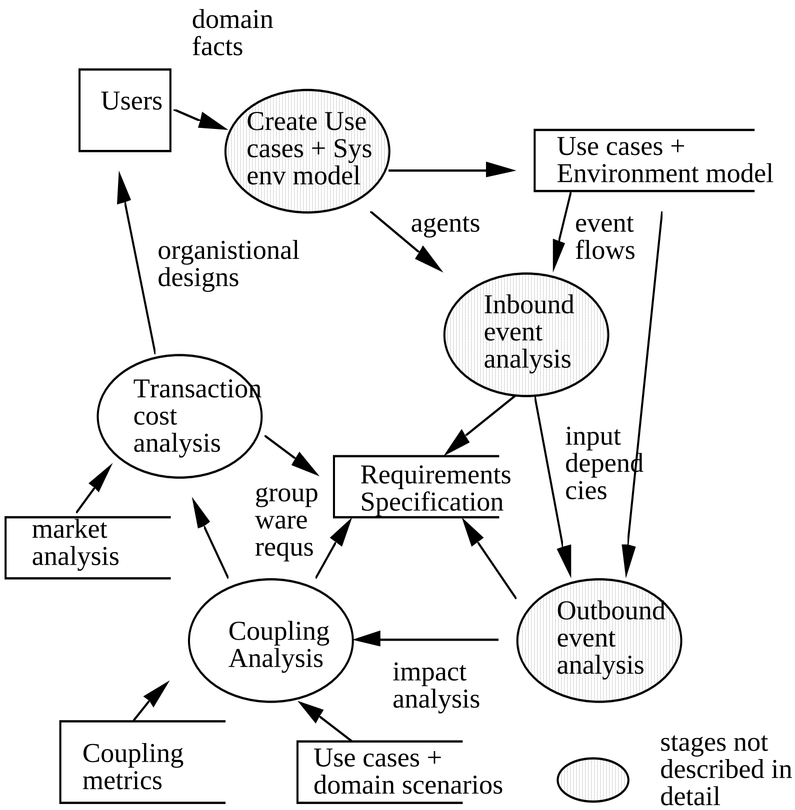

Fig. 2. CREWS-SAVRE method stages and associated models

Stage 5. Transaction cost analysis
The market context of the organisations is analysed according to the nature of the goods or services that flow between them. The good/services flows are characterised in terms of volume, values and specificity leading to predictions from Williamson's theory about the type of inter-organisation relationship that should be designed. The technological implications of the inter-organisational design are then investigated.

2.2 System Environment Model

Domain scenarios, captured as narratives or in other media, provide the basis for creating the system environment model. This is an enterprise level model which evolves through different versions as requirements analysis progresses. Rather than invent yet another modelling notation, we have adopted the notion of i* model. This consists of goals, soft goals (non- functional requirements), tasks, agents, and resources (see [9]). The two additions we make are to first, add a socio-technical system boundary, and secondly, relationships to model dependencies between tasks, agents, and objects:

- responsibility: models the association between an agent and an action/task or goal which it has the duty to carry out, e.g. < agent, is-responsible-for, goal | action |

task>;

- authority: describes the relationship between two agents in which a dominant agent has authority over the behaviour of a subordinate, or ability of an agent to permit another agent to initiate task or consume some resource; e.g. <agent (x), has-authority-over, agent (y) [task | resource]>;
- accountability: models the relationship by which the achievement of goal or task by an agent is assessed or monitored, e.g. <agent (x), held accountable for, goal,| task, by agent (y)>;
- capability: an agent has the necessary knowledge and abilities for carrying out an action/task, e.g. <agent , has-capability-for, action | task>;

An example of a system environment model is illustrated in Figure 3. This introduces the case study of the SESS (see the use case diagram in Figure 4). Four agents are involved; the controller who receives calls from customers and then allocates the service calls to engineers and schedules their work. The case study focuses on the controller's task and the dependencies between the controller and the engineers in his/her district. The engineer is accountable to the controller for reporting progress on customer calls, and the controller has authority to direct the engineers work.

3 Coupling Analysis

System requirements are discovered by assessing coupling between the required system and its users/stakeholders with the environment model and the use cases. Coupling analysis commences by a qualitative analysis using the scale illustrated in Table I. Events flowing between agents, represented in the context diagram of the top level use case, are counted and each event is assigned a coupling factor. This analysis is performed separately for human to human and human to computer communication as the implications are different for each case. Information coupling is low and makes few impositions on the recipients; however, command coupling places more constraints on actions depending on the type of command. Commands may constrain an agent's freedom to act or take decisions. For instance, the system might set a strategy that dictates how stakeholders must act. Command couples are rated for the strength of the obligation they impose on the recipient agent (optional, mandatory commands, commands with constraints) and the restriction on freedom of action imposed on the recipient. Where high levels of command coupling are apparent these indicate areas of possible conflict and system failure that should be investigated.

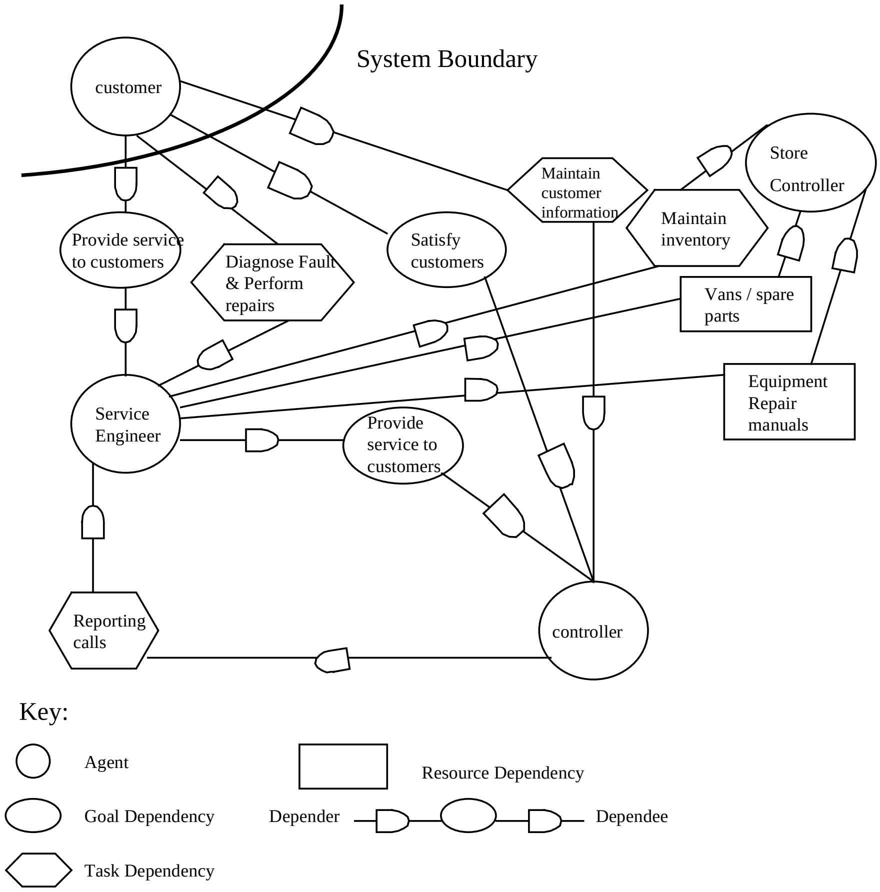

Fig. 3. System Environment model for the SESS, following the i* modelling notation

Table I Coupling Analysis Levels

Weighting	**Event Flow- Input (I) or Output (O)**	**Implications for users / stakeholders**
1	O- discretionary information	discretionary use
2	I- discretionary input	discretionary input- may lead to performance problems
3	O- decision-related information	agents needs information to take decisions
4	O- essential information	necessary for task or user action
5	O- indirect command	warning or message that requires attention, and possibly action

6	I- mandatory input	system needs data to continue
7	I- report command	agent must report completion of a task
8	O- command action	agent must carry out an action
9	O- command + constrained actions	agent’s way of working is controlled by the system
10	O- command + multiple constraints	command dictates the type/ sequence of another agent’s actions
11	O- command + constraints on many agent’s actions	one agent’s command controls several other agents

Where high levels of command coupling are apparent these should be investigated to reduce commands where possible. Table II summarises some implications of pathological coupling and possible remedies. When coupling scores are high, commands imposed by the system on the user should be reduced; for instance, by reallocating the work so only the user is responsible. Increasing autonomy of agents and decomposing the system into sub systems and also reduce coupling.

Table II Coupling Analysis Implications

Problem	**Possible Solutions / Generic Requirements**
Agent’s ability to respond to commands	check agent's responsibilities, capabilities, workload
Agent's willingness to obey commands	investigate agent’s motivation, responsibility and accountability
High coupling scores	Reduce commands, increase local decision making, sub-divide system
Many constrained commands	increase user training, increase local responsibility, control by objectives
Many report commands	monitor by objectives, gather events automatically, increase local responsibility
Many essential inputs	use defaults, reference files, integrate and share databases
Many commands / agents	investigate timing and schedule work, split responsibility
One agent creates many commands	review responsibility and authority, investigate work schedule

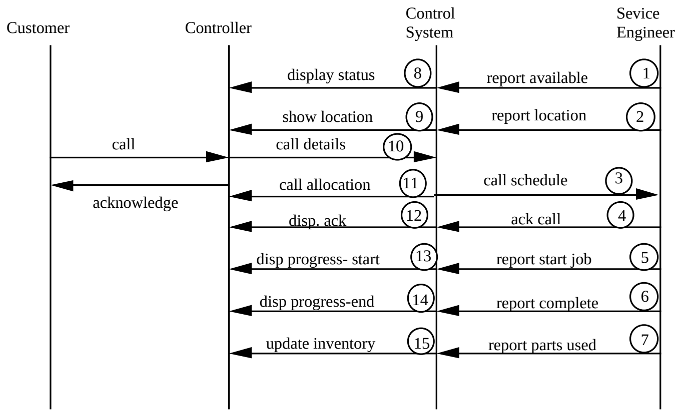

Fig. 4. Use case interaction diagram for the service engineer system (scenario 1). The numbering on the dataflows are used in the subsequent analysis.

A large number of constrained commands indicates that users are not being trusted to carry out their work without direction. Increased training and giving users responsibility should be considered. A large number of reporting commands indicates excessive system monitoring, the necessity of this should be questioned to see whether it benefits the users. Analysis of the convergence of many commands on one agent should trigger a review of responsibilities, workload, and operational schedule.

Case study: Two scenarios are analysed, each represents a different management policy for controlling the system. In the first scenario, control is centralised and all customer calls are sent to a controller who allocates calls to service engineers. The engineers have to report their location and availability to the controller, as well as their progress when undertaking repairs, and other activities such as replenishing spares from the stores. The controller system schedules the work of the engineers who have to obey commands and displays the engineers progress to the controller. The controller's role is to enter details of the customer's call and monitor the performance of the automated call dispatch system. Coupling analysis for this scenario is shown in Table III. Column 1 & 2 give the coupling between the system and the engineer, while columns 3 & 4 give coupling from the controllers viewpoint. The event numbers cross reference to the use case diagram in figure 4.

In this scenario the coupling between the engineer and the controller (see Figure 4) and between the controller and the automated call dispatch system is high. The system commands for call allocation constrain the controller's choice, and in turn the schedule is passed onto the engineer who has no discretion in his or her work. Coupling between the controller and the system could be reduced from 51 to 33 if the system's function is changed to a decision support role (see Figure 5) in which the system displays the engineer's status, locations and customer calls, leaving the controller to decide on the allocation.

Table III Coupling analysis for the SESS (Scenario 1)

Engineer - control system Event	Coupling	Control-system - controller Event	Coupling
1: Availability- I	6	8. Eng availability-O	6
2: Location -I	6	9. Eng location -O	6
3. Call schedule O	8	10.Cust call details-I	6
4. Ack call- I	6	11. Eng-call allocation-O	9
5. Start job- I	6	12. Display acknowledge-O	6
6. Finish job-I	6	13.+14. Report process (start and end)-O	12
7. Report part used- I	6	15. Update inventory-O	6
Total	44	Total	51

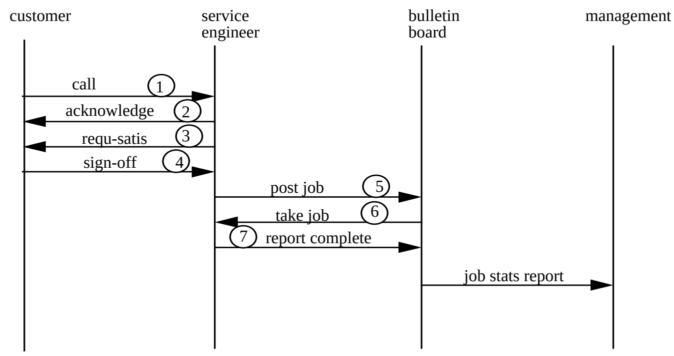

Fig. 5. Use case for the bulletin board (scenario 2) version of the SESS

This reduces the Engineer call allocation coupling to 3 (discretionary decision) and makes the ‘Update progress input’ unnecessary, hence saving 12 points overall. However, the coupling between the engineer and the controller would remain the same unless some autonomy is granted to the engineers. The high coupling in this scenario indicates a possible brittleness in the system, and more autonomy is desirable. Furthermore, the controller has to handle all the engineers in a branch and this indicates a problem of too many commands converging on one agent.

In the second scenario, a decentralised system is investigated. Small groups of engineers co-operate to handle customer calls in their area. Each engineer is assigned a district and customers call the engineer they have been assigned to directly (see Figure 5). When an engineer receives more calls than he can deal with he posts them onto a bulletin board which is shared with all engineers in the branch. An engineer with no calls who is reasonably close by is expected to take the call, otherwise the customer has to wait. Limited reporting of completed calls is carried out for management statistics. The coupling analysis for this scenario is shown in Table IV.

Table IV Coupling analysis for the Workgroups Scenario version of SESS

Engineer - customer events	**Coupling**	**Engineer - bulletin board events**	**Coupling**
1.Call-I	6	5.Request -I	2
2.Acknowledge-O	4	6.Take Job-O	3
3.Req Satisfactory- O	4	7.Report complete	2
4. Sign-off-I	4	8.Job Stats	
Total	18	Total	7

In the decentralised scenario, local autonomy, coupling between the customer and the engineer, and between the engineer and the supporting bulletin board is low. Coupling between the service engineer and the controller, who now has a management supervisory role would also be low as this is restricted to discretionary reporting of job statistics. The dramatic differences between the two scenarios demonstrate the work organisations that are possible for this system. To maintain service levels in scenario 2, an incentive system for the engineers to complete jobs as quickly as possible could be implemented and would this increases coupling, but not to the extent of the first centralised control scenario. The coupling analysis shows that a decentralised approach would be more flexible and imposes fewer restrictions on the service engineer's job. However, coupling analysis is no panacea for work design on its own. The advantages of the lower coupling in the second scenario would need to be assessed in light of engineer performance, resource costs and customer satisfaction.

The two scenarios have different requirements for technical system support. The first, centralised control scenario implies functional requirements for an automated call allocation system, a matching algorithm and an accurate database of engineers' location, work activity and training. This in turn necessitates a call reporting and progress tracking system. In contrast the second scenario requires a simpler system composed of an electronic bulletin board to record calls that engineers can not deal with and a limited reporting system to capture details of completed calls for management reports and the incentive scheme. In our ongoing work we are investigating how different patterns of coupling and organisation design can be linked to appropriate requirements templates of co-ordination and workflow systems. These templates are composed of 'generic' requirements which are high level definition of system functionality, or design options that fit a particular type of inter-organisation relationship. These can be used either to involve procurement of appropriate products or to guide further requirements analysis. Generic requirements have been proposed for classes of application domains in other papers (e.g. functionalities for sensing and monitoring applications [15], [16]); however, this paper only deals with generic requirements for workflow and group-co-ordination systems. Some examples of generic requirements for the two scenarios analysed above are:

- Organisation type: High coupling, high automation (scenario 1)

Generic requirements: reliable communications, secure protocols, message logs, message sequence control, system monitors, system status displays, fall-back and recovery procedures, user involvement and responsibility.

- Organisation type: Low coupling, shared information (scenario 2)

Generic requirements: bulletin boards, shared databases, intranets, email logging systems, email broadcast, user motivation for co-operation.

Note that these templates are based on the outcome of the coupling analysis and properties of the organisation design such as information exchange and the degree of automation present in the system. The generic requirements then recommend properties of the technical system e.g. reliable communication), possible system components (e.g. intranets) as well as human factors issues that may need attention (e.g. user motivation). The schema of these templates is still a matter of our current search, as is the range of templates that can be proposed for workflow/groupware applications. Space precludes further treatment of these developing ideas. In the final section of this paper we turn to a variant of coupling analysis motivated by business theory.

4. Coupling analysis and Organisational design

While coupling analysis can investigate problems within an organisation, a theoretical framework is required to guide re-design. The theory of transaction costs [7] models the relationships between firms according to how their transactions are organised by cost, frequency and contractual formality. For instance, companies may either compete in a decentralised marketplace, or form synergetic supply chain networks, or bureaucratic hierarchies with more formal control.

Although Williamson’s theory has had its critics it has stood the test of time [8], and provides a basis for economic assessment of the relationship between processes and organisation units. This approach is particularly useful when reengineering inter-organisation relationships and outsourcing functions. The transaction cost model asserts that client-supplier relationships depend on the transaction frequency, unit value and specificity of goods, and stability of association between suppliers and customers. Where the product has high value and requires considerable research and development expenditure, more stable process relationships are advisable. When low cost, high volume products are being produced less stability in relationships can be tolerated. Looser process relationships are often desirable as this enables more flexible responses to market changes. More closely coupled relationships imply a hierarchical structure whereas looser coupling suggests a network. The following heuristics are proposed to guide the design of inter-organisation and process relationships according to their control structure.

- If the unit value of goods passed between processes is low then assume low coupling; if high assume high coupling.
- If the volume of goods passed between processes is high assume low coupling and vice versa.
- If the goods are specific to the related processes (i.e. not horizontal, general market products) then assume high coupling, for the opposite assume low coupling.

- If the production time of the goods during the supply chain process is high, assume high coupling.
- If the goods require considerable research and development investment in the supply chain processes, assume high coupling.

One of the problems in operationalising transaction costs is setting the values of high and low value and volume. This depends on domain knowledge and analysis of particular markets. In spite of these limitations, this approach can lead to sector-specific guidance on how to structure process networks for different types of business. We describe a preliminary case study; however, in depth treatment is part of our ongoing research.

Case study: In this section we focus on two scenarios for the customer company relationship in the Service Engineering system. Scenario one is a loosely coupled relationship, in which the customers have either no service contract and maintenance calls are service on demand, or a low priority contract which just assigns a customer to a service engineer with no particular guarantee of service response time. Note that the latter scenario fits with the patchworking scenario used earlier. The second scenario is a closely coupled service contract relationship in which the customer is given a guaranteed level of service. The business analysis question is which relationship might be better suited to the market conditions of the company and then the requirements analysis question is what type of system support is necessary to support the client- service engineer relationship.

The photocopier market has many competitors and the value of machines is relatively high and volumes medium to low, depending on the size of the client company. Applying transaction cost heuristics the following relationship profile is generated:

- Relationship: Photocopier Supplier - customer
- Volume- medium to low, although a major account customer may purchase many machines, most customers only buy 1 or 2.
- Value- moderate to high in terms of the office equipment market.
- Specificity- moderate, there is no substitute for the product, but there are several rival suppliers.

According to transaction cost theory, the low to moderate volume with moderate value and specificity indicates a closely coupled and hierarchically controlled relationship between the client and supplier. This relationship suggests that scenario two is the more appropriate organisational design for the service engineer client relationship as it delivers added value to the customer and hopefully added loyalty to the supplier. The central control model in turn makes the relationship more specific by making the product and service package more specific to the company. The requirements implications are for high transaction support costs as the company will have to develop a call allocation system, customer care monitor and progress tracking for service engineer performance.

5. Discussion

The analytic techniques described in this paper provide the basis for systematically investigating socio-technical system requirements. These build on the i* framework

[9] that analyses requirements by reasoning about relationships between agents, tasks and goals. Models of dependencies between people and systems in the i* framework of enterprise models [9] enables the impact of different technical solutions to be assessed, but it does not provide an analytic method based on any theory of business organisational design. The metric based approach we have adopted complements the i* style analyses. One advantage of the metrics is that they can be applied to high level scenarios of prospective system designs to establish the strengths and weaknesses of different options. Coupling analysis has many applications in organisation design that we are only beginning to explore, such as span of command, and different command structures in organisations [11]. In our future work we will incorporate this analysis into design rationale representations so trade-offs can be assessed in quantitative as well as qualitative terms.

Few theories of organisations give firm design recommendations, apart from transaction cost theory [8] which does have the merit of link properties of the organisation's environment (the market) to recommendation for the organisation's structure. However, such analysis poses several difficulties. Assigning values to the variables of volume, value and specificity is subjective and furthermore depends on the analysts interpretation of the scope of the organisation environment. In spite of these difficulties, we found transactions cost theory to be an instructive tool for thought in organisation design. The coupling analysis complements this analysis as suggests measures for control in organisation hence a business level theory can be integrated with a socio-technical systems analysis. This in turn can provide recommendations for information system requirements. The framework we have proposed is tentative, however, we believe it is a novel attempt to integrate computer systems requirement analysis with business modelling. Some information systems methods have been extended for enterprise modelling and limited business process analysis (e.g. [17]), but no method has incorporated a business theory into the analytic process, nor have other methods integrated requirements analysis with business modelling.

The other contribution of this paper is to suggest how generic requirements could be linked to metric based analyses. Generic requirements (GRs) by their nature are not detailed, hence the utility of such advice needs further evaluation; however, we believe that GRs add value by raising requirements issues, even if they do not always provide solutions. The method spans a wide range of issues which GRs can not deal with in depth, so we see the method as a framework that points the requirements analyst towards other sources for more detailed advice including generic models of requirements for classes of application which we have partially explored for information retrieval [18].

The coupling analysis draws on theories of autonomy and work organisation from management science (e.g.[19]). High level requirements for groupware and workflow systems can be proposed as a result of this analysis, however, we have to improve the connection between generic requirements and organisational design. Furthermore, the recommendations of coupling analysis need to be interpreted in a domain context. While increasing autonomy might help many business organisations the converse may be true for military command and control systems. In producing a method for socio technical organisation analysis we have drawn together several literatures. We have pointed out where business process engineering may profitably benefit from

Williamson's [7] transaction cost model for designing inter-organisational relationships according to a market context and how organisation design can be linked to requirements analysis of technical systems. Design of business relationships will be increasingly vital as more companies outsource functions and concepts of the virtual firm and symbiotic productive networks become a reality (see [20]).

Acknowledgements

This research is funded by the European Commission ESPRIT 21903 'CREWS' (Co-operative Requirements Engineering With Scenarios) long-term research project. The authors wish to thank other members of the CREWS project for their comments and advice.

References

1. Macaulay, L., Requirements Engineering, Springer Verlag, Berlin, (1996).

2. Lubars, M., Potts, C., and Ritcher, C., 'A review of the state of the practice in Requirements Modelling', IEEE Int. Symposium on Requirements Engineering (RE'93), 2-14, (1993).

3. Sommerville, I., and Sawyer, P., 'Requirements Engineering: A Good Practice Guide', John Wiley & Sons, (1997).

4. Harker, S.D.P., Eason, K. D., and Dobson, J. E., 'The Change and Evolution of Requirements as a challenge to the practice of Software Engineering', IEEE Int. Symposium on Requirements Engineering (RE'93), 266-272, (1993).

5. Potts, C., Takahashi, K., and Anton, A. I., 'Inquiry-Based Requirements Analysis', IEEE Software, vol. 11, no. 2, pp. 21-32, (1994).

6. Hsi, I., and Potts, C., 'Towards Integrating Rationalistic and Ecological Design Methods for Interactive Systems', Georgia Institute of Technology, Graphics, Visualisation and Usability Centre *Technical Report*, 1-15, (1995).

7. Williamson, O.E., 'The economics of organisations: The transaction cost approach', American Journal of Sociology, Vol. 87, 548-577, (1981).

8. Williamson, O. E., 'Markets, hierarchies and the modern corporation: an unfolding perspective', Journal of Economic Behaviour and Organisation , Vol. 17, No. 3, 335-352, (1992).

9. Yu, E., 'Modelling Strategic Relationships for Process Reengineering', Technical Report DKBS-TR-94-6, University of Toronto, (1994).

10. DeMarco, T., Structured Analysis and Systems Specification, Englewood Cliffs, New Jersey, Prentice Hall, (1978).

11. Robbins, S. P., 'Organisation theory', Prentice Hall, Englewood Cliffs, NJ, (1990)

12. Jacobson, I., Christerson, M., Jonsson, P., and Overgaard, G., 'Object-Oriented Software Engineering: A Use-Case Driven Approach', Addison-Wesley, (1992).

13. UML, 'Unified Modelling Language: Method', Rational Corporation, (1999). Rational's web site: http://www.rational.com

14. Mintzberg, H.,'The Structuring of Organisations', Prentice-Hall Inc., (1979).

15. Sutcliffe A.G., Maiden N.A.M., Minocha S. and Manuel D., Supporting Scenario based requirements engineering. IEEE Transactions on Software Engineering, Vol.

24, No.12, 1072-1088. (1998)

16. Sutcliffe A.G. and Carroll J.M., Generalising claims and reuse of HCI knowledge. People and Computers XIII Proceedings of the BCS-HCI Conference Sheffield. Editors Johnson H., Nigay L., and Roast C., 159-176, Springer Verlag, (1998)

17.Veryard R. and MacDonald I.G., EMM/ODP: A methodology for federated and distributed systems. In Proceedings of IFIP WG 8.1. Conference, Methods and Associated Tools for the Information System Life cycle, Eds Verrijn-Stuart A. A. and Olle T.W., 241-273, North Holland, (1994).

18.Ryan, M., and Sutcliffe, A. G., 'Analysing Requirements to Inform Design', People and Computers XIII - Proceedings of the BCS-HCI Conference, Sheffield, H. Johnson, L. Nigay and C. Roast (Eds.), Springer Verlag, 139-158, (1998).

19. Clegg, C., Axtell, C., Damodran, L., Farbey, B., Hull, R., Lloyd, R., Nicholls, J., Sell, R., and Tomlinson, C., 'Information Technology: A study of performance and the role of human and organisational factors', Ergonomics, Vol. 40, 851-871, (1997).

20. Holland, C. P., 'Co-operative supply chain management: The impact of interorganisation information systems', Journal of Strategic Information Systems, Vol. 4, No. 2, 117-133, 1995.