Using Agent Technology to Overcome Project Failure in Distributed Organizations

Holly Parsons – Hann[1], Kecheng Liu[1]

[1]School of Systems Engineering, Department of Computer Science, Reading University, Reading, UK

Abstract. As organisations become more global and interest groups more widely distributed reaching a consensus among stakeholders when conducting a global spanning project becomes increasingly harder to achieve. Many software projects have failed because their requirements were poorly negotiated among stakeholders and this problem must be solved if project failure is to decrease. Although many stakeholder negotiation methods have been suggested, validated and employed in projects across the globe, project success rates are still very low, suggesting that there is still work to be done in the distributed organisational domain to increase the probability of project success. This paper will highlight the current project management problems in distributed organisations and suggest a new agent based method of solving them.

1 Introduction

Due to the competitive technological industries prevalent today, it has become common practice for distributed organisations offering usually similar competencies to federate alliances to improve their efficiency and move into a more competitive market position. In order to remain competitive, businesses must use every technological innovation to their advantage, thus not being out performed in efficiency or productivity.

Distributed organisations have the advantage that they are able to changing demands and technological advancement, businesses respond by developing increasingly flexible infrastructures in order to remain competitive in the newly defined market boundaries [13]. However, due to the vast percentage of project failure it is increasingly hard to remain in such an advantageous market position when just over 10% of projects are considered 'successful' e.g. within time, within budget and delivers what the user wants.

One of the main reasons cited for project failure are user requirements (CHAOS, 1995), 13.1% of businesses in 1995 stated that this was the main reason why their projects had failed. This is a common problem in projects and requirements elicited from customers can overlap or conflict in any project. Some requirements may be ambiguous or unrealistic, while others may remain undiscovered [11]. Due to these reasons, requirements need to be specified clearly and concisely without any margin for ambiguity or confusion.

Parsons – Hann H. and Liu K. (2005).
Using Agent Technology to Overcome Project Failure in Distributed Organizations.
In *Proceedings of the 1st International Workshop on Requirements Engineering for Information Systems in Digital Economy*, pages 13-23
DOI: 10.5220/0001416000130023

The problem is further exacerbated when projects are performed in a distributed organisation. This adds a new degree of difficulty to the project before it is even underway as there is a strong likelihood that some of the stakeholders will not be located in one geographic site. This means trying to elicit the requirements from numerous different stakeholders from many different locations which can be incredibly challenging. Project teams cope with distance in a number of different ways by using a variety of media [6].

In this paper, we suggest a new requirements negotiation and prioritisation method, one which takes into account the distributed nature of the stakeholders, as well as the natural ambiguity which is usually present in such projects. By using agent technology, we will demonstrate that requirements negotiation and prioritisation between stakeholders can be more efficient as well as less time consuming and less ambiguous thus, increasing the probability of project success.

This paper will discuss the features of a distributed organisation, note the current problems with requirements negotiation methods and suggest suitable agent-based methods to tackle these problems.

2 Features of a Distributed Organisation

Distributed Organisations have emerged due to new enabling technologies and support copious numbers of e-work, e-business and ecommerce activities all over the world. They are boundary crossing with complementary core competencies. As well as sharing knowledge and usually consisting of a network of independent organisations, there is great geographical dispersion and many changing participants year after year.

There is no hierarchy in a distributed organisation due to a belief of participant equality and nearly all communication between participants is conducted electronically. A distributed organisation is formed by agreement of separate organisations to collaborate, to share knowledge and expertise, in order to achieve a common purpose. Although a group of legally separate entities, the organisation acts as though they were one, thus the customer deals with what appears to be a single organisation. To ensure that the customer sees only one organisational front, the member organisations coordinate their activities [9].

By being part of a distributed organisation, many advantages can be reaped. Resources are less tied up as the organisation is not paying enormous amount of money for building fabrics, therefore has more money to invest in its core values. Changes in the business environment do not bother a distributed organisation as they are flexible enough to embrace change and not hinder their productivity, this greater flexibility ensures a non hierarchical and dynamic business.

However, as well as there being many advantages to conducting business in a distributed fashion, there are also disadvantages to consider. The disadvantages will be identified and analysed in the next section.

3 Requirements Negotiation

Requirements negotiation is concerned with the high level statement of requirements elicited from stakeholders. Negotiation has become an essential part of system speci-

fication where stakeholders negotiate among themselves and with system engineers[14] with tradeoffs happening in order to resolve conflicts.

Many software projects have failed because their requirements were poorly negotiated among stakeholders. Negotiating requirements is one of the first steps in most Software systems life cycle, yet its results have a significant impact on the system's value [2]. Getting project requirements right, is crucial for project planning and ultimately, customer satisfaction as without customer satisfaction the project is not considered a success. Should the stakeholders not agree on the requirements of the project, there is little chance of a successful project as one of the three criteria for project success will not be present e.g. 'giving the user what they want'.

Getting a general consensus between stakeholders is a crucial, yet hard task to fulfill in a traditional hierarchical company, without having the additional complication of distributed stakeholders who employ different working cultures and social norms.

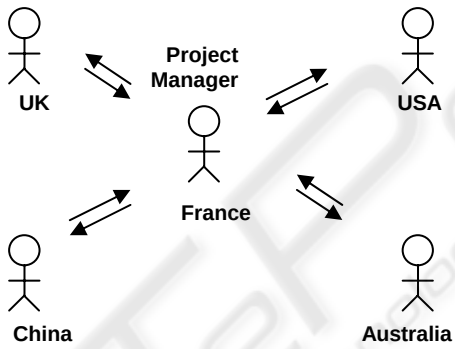

Fig. 1. Distributed Stakeholders around the world

In spite of years of experience, many organisations still do not allow enough time to resolve requirements conflicts. Due to the rapid technological development, many companies now rely on other mediums of communication which are faster and cheaper in some ways, yet at a cost. Using E-mail for negotiation takes significantly longer than face to face meetings yet has the advantage of sending long documents to various stakeholders in a very short time.

Figure 1 illustrates the problems that are present when trying to negotiate stakeholder requirements in a distributed organisation. Research has been done into a number of 'capability barriers' which prevent effective communication in geographically dispersed groups [16] The three identified problems included not sharing a common first language, being separated by sixteen time zones and the difference in typing ability when communicating via a messaging program. There are many different working cultures in the world, and many stakeholders will experience and be part of these work defining characteristics.

Although a minor difference in interpretation, these small misunderstandings can grow and cause problems later in the project lifecycle if a common understanding of all the project terms are not reached.

4 Current Requirement Negotiation Methods

Groupware is technology designed to facilitate the work of groups. This technology may be used to communicate, cooperate, coordinate, solve problems, compete, or negotiate. While traditional technologies like the telephone qualify as groupware, the term is ordinarily used to refer to a specific class of technologies relying on modern computer networks, such as email, newsgroups, videophones, or chat.

CSCW (Computer-Supported Cooperative Work) refers to the field of study which examines the design, adoption, and use of groupware. It has also emerged as an identified research field during the last dozen years. It focuses on the role of the computer to assist people working together. The development of suitable software for small and large groups is a major focus of CSCW.

Traditional collaboration is necessarily sequential: one individual can only add something after another one has finished. This even applies to real-time meetings or conversations: only one person can talk at a time. In a CSCW environment, on the other hand, people can add information in parallel. Moreover, the collaboration is not restricted to real-time or other *physical constraints* depending on time or space. The different people collaborating need not be present in the same location or even at the same time.

In meetings assisted by various groupware technology, people who are assertive, fluent or in a position of authority will tend to dominate the discussion, while those who are shy or of a lower rank will find it very difficult to have their ideas accepted or even paid attention to, however good those ideas may be. This can happen due to both personality and the main mother tongue of the stakeholder.

The EasyWinWin process helps success-critical stakeholders to jointly discover, elaborate and negotiate their requirements [1]. The process has been built upon the The WinWin model [3], which provides a general framework for identifying and resolving requirement conflicts. Easy WinWin enables and facilitates heterogeneous stakeholder participation and collaboration.

As Boehm *et al*. point out, the Win Win models' primary distinguishing characteristic is the use of stakeholder win-win relationship as the success criterion and organising principle for the software and system definition. Our research incorporates various specific techniques from the Win Win model, however the use of intelligent agent technology will help make requirements negotiation a faster and less time consuming process. Briggs and Gruenbacher [4] acknowledge the potential problem of checking every requirement by hand by stating 'Intelligent agents may be able to conduct exhaustive pair-wise comparisons amongst thousands of win conditions, a task that would be impossible for individual humans'

5 Agent Technology

The basic dictionary definition for an agent is *one who acts.* However, when developing Information Technology systems this definition is far too vague and therefore needs to be defined in terms of attributes and properties that an IT agent possesses. The term 'agent' has become increasingly common within the IT industry and is usually used to describe computational entities such as a 'multi agent system'. When referring to agents within the computing industry, agents are defined as

‘A computer system capable of flexible autonomous action in a dynamic, unpredictable and open environment.’ [10]

It is imperative to point out than an agent can be a person, a piece of software or a variety of other things. During the 1980’s, research into the Artificial Intelligence paradigm was prevalent and various new concepts arose. One of the new branches was designated as Distributed Artificial Intelligence (DAI) This branch of AI, strongly linked with social sciences is mainly concerned with the study of multi agent systems [7].

Sample applications of agent technology to date include data filtering and analysis, brokering, process monitoring and alarm generation, business process and workflow control, data/document retrieval and storage management, personal digital assistants [8], Computer Supported Co-operative working (CSCW) simulation modelling and gaming also contain various agent technology. These mechanisms are now being realised by *agent technologies*, which are already providing copious and sustained benefits in several business and industry domains.

There are many properties which agents can possess in various combinations. Although there is no industry standard for the definition ‘Agent’ (Object Management, 2000) most experts agree that agents which are bound for IT systems are not useful unless they exhibit at least three specific attributes. The first attribute is autonomy, therefore the agent is capable of acting without direct external intervention. Components should be enabled so that they can respond dynamically to changing circumstances and act based on the agents own experiences. Agents should be able to perform the majority of their problem solving tasks without the direct intervention and they should have a degree of control over their own actions and their own internal state [5].

The development of intelligent adaptive agents has been rapidly evolving in many fields of science. Such systems should have the capability of dynamically adapting their parameters, improve their knowledge-base or method of operation in order to accomplish a set of tasks.

An information agent, on the other hand, is a computational software entity (an intelligent agent) that can access one or multiple, distributed, and heterogeneous information sources available. It can pro-actively acquire, mediate, and maintain relevant information on behalf of its user(s) or other agents preferably just-in-time. The benefit of employing information agents is that they are able to cope with the difficulties associated with the information overload of the user and output the information in a detailed and orderly way.

An information agent can communicate with its environment, other agents or human users depending on who has the information that is needed. If two agents are communicating with each other, it is vital they can ‘speak’ the same language, therefore the use of a commonly agreed agent communication language (ACL) such as FIPA ACL and KQML has to be considered.

6 Our Research

Assuming the stakeholders have been identified already, the first action to take is to ‘rank’ the stakeholders from rank 1 to rank 4. In previous literature Boehm et al. state that ‘optimistically expecting a result that is mutually satisfactory to all stake-

holders is nearly absurd.' [12] Therefore by identifying all stakeholders and classifying which are success critical and which are not, various levels of involvement can be obtained based on which rank a stakeholder has been given.

Therefore the question that must be asked is *How can we judge which stakeholders are the most important and what can we base this judgement on?* Sadly, there is no easy answer and comparatively little research has been done into prioritising the stakeholders themselves. One method which is widely used however, has been developed by business strategy theorists Gerry Johnson and Kevan Scholes. It was initially developed as a tool to help organisations implement strategic developments, but is equally applicable to project management [17].

Once all the stakeholders have been identified, a matrix can then be plotted which classify the stakeholders according to interest and influence. Interest and Influence are defined in the Oxford English Dictionary as:

> **Interest**: *The state of wanting to know about something or someone*
>
> **Influence**: *The capacity to have an effect on the character or behaviour of someone or something*

Figure 2. shows the Matrix , along with an example of the type of stakeholders who will be in each category. It is now possible to assign each stakeholder a 'rank' which will help the project team to assess which requirements should be given more consideration than others.

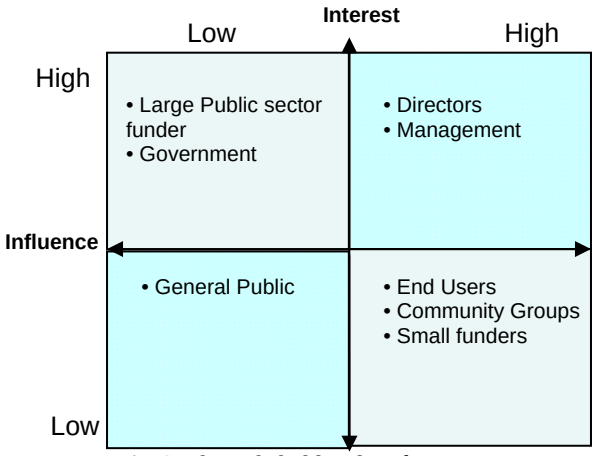

Fig. 2. The stakeholder classification Matrix

Unlike the WinWin method which identifies and focuses on just the success critical stakeholders, we believe that even non-critical stakeholders are important. The stakeholders who are rank 2, e.g. the high influence and low interest stakeholders may not be success critical, however they must still be given time to discuss any issues they have with the project. If the project changes direction or the individuals involved in the project change frequently, the group may have an increased interest in the project.

Therefore it is very important that rank 2 stakeholders are kept as informed as they wish. Rank 3 stakeholders (low influence and high interest) and Rank 4 stakeholders (low influence and interest) should also be treated as part of the project, albeit in a smaller role. Useful information can be learned from these stakeholders therefore they must not be ignored.

Due to its open methodology DOORS (Dynamic Object Oriented Requirements System) can be flexibly used in any industry for virtually any product. It is based on

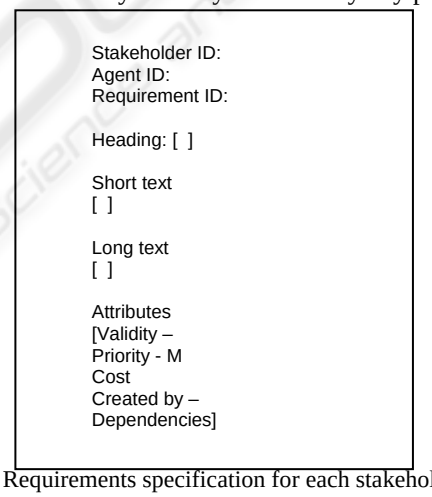

Fig. 3. Requirements specification for each stakeholder

an integrated OO (Object Orientated) data store and can be applied to either a one main or whole company project. In this method, each requirement is known as a 'Requirement Object' hence the Object Orientated perspective. It provides good visualisation of such documents as hierarchies, and its extension language enables a wide range of supporting tools to be built, and many are provided as menu commands and examples.

Figure 3. illustrates how, using the DOORS specification framework, we have adapted it to suit the stakeholders needs when asking them to enter their requirements. The stakeholder ID and agent ID are assigned by the project team whereas the requirement ID is assigned by the agent and is created by combing various parts of the stakeholder ID and Agent ID into a unique requirements reference.

Throughout the process of entering each requirement, the agent will be able to monitor each word to check for spelling errors and also be able to detect 'keywords' for example, if the stakeholder entered 'The system must be able to operate 24 hours a day' The agent might ask the stakeholder to define the word 'operate'. All identified keywords will then be saved in an online glossary which the project team can access and check for ambiguities between stakeholders, e.g. if 'operate' is a culturally dependant word and different stakeholders interpret the word in a different way.

Once all requirements have been entered, spelling checked and keywords highlighted. The stakeholder is then asked to prioritise their requirements – 1 being the most important, 2 the next important and so on, until they all have a priority associated with them. When all requirements have been entered, the agents will therefore enter the negotiation process without any assistance from the project team.

An email does not fall neatly into the domain of text classification. There is much extraneous information that is redundant when classifying the 'type' of email it is (e.g. needed or spam mail) All the methods serve to simplify the problem and reduce the amount of noise inherent in the problem Applying these methods to each email message transforms the message into an acceptable input document to a text classifier. Each message can then be viewed as a collection of words which can then be examined for frequency of occurrence.

The classification problem can be simplified by removing the common pronouns, verbs and adverbs such as "the", "it", "here". These words can be put into a list called a 'stoplist' and filtered out when trying to classify emails. Since a majority of these words appear in almost all documents, they can be ignored without losing any information. A stoplist will be implemented in the agents in order to reduce the 'noise' and allow focus to be on the less frequent text.

The Porter Stemming Algorithm was developed by Martin Porter in 1980 and is another methods for refining the text even further. It has been very widely implemented and coded in various programming languages since its publication with research still being conducted into making the algorithm more efficient [18]. 'Stemming' has the purpose of treating words the same root as being equivalent. For example, see, seen, sees and saw would all be transformed into their root 'see', thus eliminating the irrelevant differences between 'saw' and 'seen'.

By using both methods and comparing the similarity of the text left behind it should be easy for the agents to judge whether the two requirements are the same or different. If the agents decide they are the same requirement, the agent with the lowest ranking stakeholder will 'freeze' that requirement and not use in the negotiation process. This means that the requirement will still be prioritised, but no more than once.

Once all duplicate requirements have been identified, agents will then attempt to negotiate their stakeholder requirements with other agents. Before entering into negotiation with another agent, the agent identifies the stakeholders rank and identifies the stakeholder multiplication factor.

Table 1. The stakeholder multiplication

Stakeholder rank	Multiple factor
1	x 1
2	x 3
3	x 5
4	x 7

Stakeholder Rank 2: Multiple factor = x 3

Requirements

1. [<Requirement>] x1 = 1. [ ]
2. [<Requirement>] x1 = 2. [ ]
3. [<Requirement>] x1 = 3. [ ]
4. [<Requirement>] x1 = 4. [ ]

Fig. 4. Applying the stakeholder multiple factor to a rank 1 stakeholders requirements.

Table 1. shows the stakeholder ranks 1-4 and also shows a 'multiplication factor' This is used to multiply each requirement priority with the corresponding multiple factor for the correct stakeholder rank. For example, a rank 1 stakeholder would have their requirement priorities all multiplied by '1',thus the priorities 1,2,3,4 would remain the same.

Stakeholder Rank 1: Multiple factor = x 1

Requirements

1. [ ] x3 = 3. [ ]
2. [ ] x3 = 6. [ ]
3. [ ] x3 = 9. [ ]
4. [ ] x3 = 12. [ ]

Fig. 5. Applying the stakeholder multiple factor to a rank 2 stakeholders requirements.

On the other hand, a rank 2 stakeholder would have their requirement priorities multiplied by '3', thus 1,2,3,4 would in fact be turned into 3,6,9,12 and so forth. Figure 5 shows what will happen to a rank 2 stakeholders' prioritised requirements.

The stakeholders will not know the rank they are assigned as this could be seen as unprofessional and risks embarrassing the stakeholders should they find out they are

rank 4. Therefore the information is undisclosed to all but the project team and the agents.

It is worth noting that although this method should prove to be quicker and more efficient than other negotiation methods presented, it is not supposed to be used as a standalone project technique. As soon as the agents have finished the negotiation and prioritisation process, we cannot assume that the requirements will remain static and at the same priority throughout the entire project. Once the project progresses and stakeholders gain a better understanding of what is needed their requirements will, in turn mature with the project.

Therefore other methods should be used to compliment the agent negotiation technique rather than using it in a standalone fashion. When the agents produce a requirements negotiated, prioritised list, the project team will have a clearer idea of what is needed to fulfil the requirements and a project plan can be created. Stakeholders can then be contacted and their requirements discussed and refined. This helps lessen any ambiguity that may still be present and strengthen the project teams' understanding of each stakeholders' needs and wants.

Conclusion

By using intelligent agents to perform much of the requirements negotiation and prioritisation, less effort is needed by either the project team or the stakeholders. The method addresses the problems commonly found in distributed organisations such as culture and time zone differences, as well as identifying and hopefully decreasing a lot of the initial ambiguity of the requirements. Taking into account all stakeholders, rather than just the success critical ones should enable a richer and more solid foundation for a project to unfold and thus decrease the chance of project failure.

Future work will include refining the agent negotiation protocols as well as starting to build a prototype and actually implement the methods described in this paper.

References

1. Boehm, B., Grünbacher, P., Briggs, B. 2001. Developing Groupware for Requirements Negotiation: Lessons Learned, *IEEE Software*, pp. 46-55
2. Boehm, Barry., Egyed, Alexander., 1998. Software Requirements Negotiation: Some Lessons Learned, *Proceedings of the 20th International Conference on Software Engineering*, Kyoto, Japan
3. Boehm, B., Bose, P., Horowtiz, E., and Lee, M., 1995. Software Requirements Negotiation and Renegotiation Aids: A Theory- W Based Spiral Approach, In *17th International Conference on Software Engineering*, Seattle: IEEE Computer Society Press.
4. Briggs, Robert., O.; Paul Gruenbacher, 2002. Easy WinWin: Managing Complexity in Requirements Negotiation with GSS, In *Proceedings of the 35th Hawii International Conference on systems sciences*
5. Castelfranchi, C., 1995. Guarantees for autonomy in cognitive agent architecture. In Intelligent Agents - Theories, Architectures, and Languages, (eds. M. Wooldridge and N. R. Jennings). Springer-Verlag Lecture Notes in AI Volume 890. pp 56-70.

6. Damian, Daneila., E., Didar, Zowghi,., 2003. The impact of stakeholders' geographical distribution on managing requirements in a multi-site organization, In *IEEE Joint International Conference on Requirements Engineering.*
7. Filipe, Joaquim, 2000. Normative Organisational modelling using Intelligent Multi-Agent Systems, Ph.D thesis
8. Graham, Ian., 1998. Requirements Engineering and Rapaid Development – An object Orientated Approach, Addison- Wesley, ACM press, ISBN 0-201-36047-0
9. Lethbridge, N., 2001. An I-Based Taxonomy of Virtual Organisations and the Implications for Effective Management, http://inform.nu/Articles/Vol4/v4n1p017-024.pdf (Accessed 23/02/2005)
10. Luck, Michael., McBurney, Peter., Preist, Chris., 2003. Agent Technology: Enabling Next Generation Computing, A Roadmap for Agent-Based Computing, AgentLink II report, ISBN 0854 327886
11. Maciaszek, Leszek., 2001. Requirements Analysis and System design – developing information systems with UML, Addison-Wesley, IBSN 0-201-70944-9
12. Park, Jun-wong.; Daniel Port.; Barry Boehm., 1999. Supporting Distributed Collaborative Prioritization forWinWin Requirements Capture and Negotiations, In *Proceedings of the International 3rd World Multiconference on Systemics, Cybernetics and Informatics* , Vol. 2, pp.578-584, IIIS,
13. Parsons-Hann, Holly.; Kecheng Liu., 2005. Measuring Requirements Complexity to Increase the Probability of Project Success, In *IEEE 7th International Conference on Enterprise Information Systems.*
14. Robinson, W., N., 1990. Negotiation Behaviour During Requirement Specification, In *Proceedings of the . 12th International Conference of. Software Engineering* IEEE Computer Soc. Press, Los Alamitos, Calif, pp. 268–276.
15. Standish Group, 1995. CHAOS report.
16. Toomey, L., Smoliar, S., Adams, L., 1998. Trans-Pacific Meetings in a Virtual Space, FX Palo Alto Labs Technical Reports
17. Vogwell, Deborah.; (2002) Game Theory, Building Magazine http://www.davislangdon.com/europe_middleeast/uk/images/Game%20Theory.pdf (Accessed 24/02/2005)
18. Yamout, Fadi.; Rana Demachkieh; Ghalia Hamdam; Reem Sabra; 2004. Further Enhancement to the Porter's Stemming Algorithm, In *Machine Learning and Interaction for Text-based Information Retrieval*