The development & evaluation of a set of requirements for a new group-ware
system to improve collaboration within teams

Simon J Smith

Bachelor of Science(Honours) in Computer Information Systems

The University of Bath

2010
The development & evaluation of a set of requirements for a new group-ware system to improve collaboration within teams

submitted by: Simon J Smith

Copyright
Attention is drawn to the fact that copyright of this dissertation rests with its author. The Intellectual Property Rights of the products produced as part of the project belong to the University of Bath (see http://www.bath.ac.uk/ordinances/#intelprop).

Declaration
This copy of the dissertation has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the dissertation and no information derived from it may be published without the prior written consent of the author. Declaration This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of Bachelor of Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

your signature
Abstract

This project investigates collaboration within teams and the group-ware tools used to support them. This project expands on the work by Johnson & Hourizi on large group collaboration. With their framework as a starting point, the project will go onto apply this to a virtual-collaborative team in order to discover which factors within collaboration need to be supported by group-ware solutions. From this analysis a set of requirements were developed for a group-ware system to facilitate shared understanding between distributed group members. A design specification and prototype system based on these requirements was then developed, these were evaluated and showed a reduction in conflict between collaborating distributed team members.
Acknowledgements

I would like to thank my supervisors Professor P. Johnson, Dr Neil Carrigan and Dr Nicholas Forbes for their advice and support throughout this project. I would also like to thank the online .NET community which provided guidance for the development of the prototype.
Contents

1 Introduction 10
  1.1 The problem ............................................. 10
  1.2 The Aim .................................................. 10

2 Literature survey 12
  2.1 Introduction ............................................. 12
  2.2 Collaboration ........................................... 12
  2.3 Collaborative models and frameworks ...................... 13
  2.4 Framework for Large group collaborations ................. 18
    2.4.1 Task Structure ........................................ 18
    2.4.2 Organizational Structure ............................ 19
    2.4.3 Group Structure ....................................... 19
    2.4.4 Resources ............................................. 20
    2.4.5 Conflict .............................................. 20
  2.5 CSCW & Group Ware ........................................ 20
    2.5.1 Video Conferencing ................................... 22
    2.5.2 Document Management Systems (DMS) .................. 22
    2.5.3 Wikis .................................................. 23
    2.5.4 Instant Messengers .................................... 23
    2.5.5 Online Meetings ....................................... 23
  2.6 Virtual Teams ........................................... 24
  2.7 Findings ................................................ 24

3 Analysis 25
  3.1 Scenario ................................................ 25
    3.1.1 Background ........................................... 26
    3.1.2 Goals ................................................ 26
    3.1.3 Resources ............................................. 26
  3.2 The framework and the scenario ............................. 27
    3.2.1 Organizational structure ............................. 28
    3.2.2 Group Structure ....................................... 29
    3.2.3 Task Structure ......................................... 30
    3.2.4 Resources ............................................. 42
    3.2.5 Conflict .............................................. 43
  3.3 Conclusion ............................................. 45

4 Requirements 48
  4.1 Sources ................................................ 48
    4.1.1 Analysis .............................................. 48
    4.1.2 Existing Groupware solutions & literature ............ 48
  4.2 The Concept ............................................. 49
  4.3 Functional requirements ................................... 49
    4.3.1 System Requirement .................................... 51
    4.3.2 User Requirements ..................................... 52
4.4 Non-Functional Requirements .................................................. 53
4.5 Usability ................................................................. 53
4.6 Summary ................................................................. 53

5 Design ................................................................. 54
5.1 System Overview .......................................................... 54
5.2 Design considerations .................................................... 55
  5.2.1 Assumptions and Dependencies ...................................... 55
  5.2.2 Goals and Guidelines ................................................ 56
  5.2.3 Development Methods ................................................. 56
5.3 Architectural Strategies .................................................. 56
  5.3.1 Language .............................................................. 56
  5.3.2 Client Server ........................................................ 57
  5.3.3 Storage ............................................................... 57
  5.3.4 Transfer .............................................................. 58
5.4 System Architecture ..................................................... 58
5.5 Policies and tactics ....................................................... 60
5.6 Detailed system design .................................................... 61
  5.6.1 CustomTPS .......................................................... 61
  5.6.2 SqlWorkSpace ....................................................... 63
  5.6.3 LogInGui ............................................................ 64
  5.6.4 ContainerGui ....................................................... 65
  5.6.5 CustomPanel ........................................................ 65
  5.6.6 UserControlGUI ................................................... 65
  5.6.7 NameDialogGUI .................................................... 65
  5.6.8 WorkspaceViewGUI ................................................ 65
  5.6.9 SqlClass ............................................................. 65
  5.6.10 Server Side ........................................................ 65
5.7 Summary ................................................................. 66

6 Implementation ............................................................. 67
6.1 Client ........................................................................ 67
  6.1.1 Log In Page ........................................................... 67
  6.1.2 Interface .............................................................. 67
  6.1.3 Client GUI ........................................................... 68
  6.1.4 Tab ..................................................................... 69
  6.1.5 The Work-Space ..................................................... 69
  6.1.6 SQLTools Namespace ................................................ 70
  6.1.7 SQL Server query notification ..................................... 71
6.2 Back End .................................................................. 71
  6.2.1 SQL DB .............................................................. 72
6.3 Testing .................................................................... 73
6.4 Summary ................................................................. 73
7 Results & Evaluation

7.1 Evaluation with the Large Group Collaboration framework

7.1.1 Improvements within the previously analysed scenario

7.1.2 Improvements across different scenarios

7.2 Evaluation by other means

7.3 Comparison between the two approaches

7.4 Summary

8 Conclusion

8.1 Limitations

8.1.1 Limitations of the study

8.1.2 Limitations of the system

8.2 Further Work

8.2.1 Dynamic Scenarios

8.2.2 Implementation

8.2.3 Testing

8.3 Summary
List of Figures

1. Weiseth’s Collaboration Framework .............................................. 13
2. The Group Task Circumplex .......................................................... 14
3. Coordination Theory and its Application ......................................... 15
4. Wheel of Collaboration Tools ......................................................... 16
5. Framework for Large group collaborations ...................................... 18
6. CSCW quadrants ........................................................................... 21
7. Large group Collaboration framework ........................................... 28
8. The organizational structure used in the scenario ............................ 29
9. The group Structure used in the scenario ......................................... 30
10. A high level goal structure for the scenario ................................... 32
11. Goal substructures to achieve “develop a set of design criteria
    representative of what the business require” .................................. 33
12. Goal substructure for achieving the goal “Establish a shared under-
    standing of what is required and how to achieve it” ......................... 34
13. Task Analysis for "Determining what is to be achieved" .................... 35
14. Task analysis for "How to reach determined requirements" ............... 36
15. Goal substructures for the team achieving “Decision made as a
    group” ....................................................................................... 37
16. Task structures for Developers to achieve the goal “Implementation
    of code as both individuals, pair programming and as a group” ........ 38
17. Goal substructures for achieving “Work together in dynamic sub-
    groups to overcome problems on the fly” ..................................... 39
18. Task structures for actors to achieve the goal of “Testing and
    debugging code in ad-hoc groups” ............................................... 40
19. Goal substructure for developing a shared understanding within
    a group ...................................................................................... 41
20. Architectural design of the system ...................................................... 59
21. Use Case of options ....................................................................... 60
22. The Client GUI ............................................................................. 68
23. Diagram of the database architecture .............................................. 72
24. Synchronous channels of communication within the group-wares .... 75
25. Olson & Olson’s table depicting the characteristics that contribute to
    negotiating common ground found in various media[10], with the
    addition of the three group-wares analysed earlier in the project as well
    as the “All Collab” system developed ........................................... 77
26. Olson & Olson’s figure showing how technologies can support the
    characteristics of colocated synchronous interactions[10]. This has
    also had the three group-wares analysed earlier in the project as well
    as the “All Collab” system added to it .......................................... 78
27. Olson & Olson’s Key characteristics of colocated synchronous
    interactions[10] .......................................................................... 90
28. Add a new user to workspace ............................................................ 91
29. Closing a Work-Space .................................................................... 91
List of Tables

1. System Requirements ........................................... 51
2. User Requirements ............................................. 52
1 Introduction

1.1 The problem

Collaboration is increasingly important in modern life. It is used across many domains from complex business or military situations to everyday tasks or even video games. People are collaborating more than ever. There are many advantages to using collaborative work in order to achieve a task, these include aspects such as increased diversity and the inevitability of a more comprehensive approach to the problem.

As globalization has increased in the recent past as has the scope of collaboration, especially within its business context. In order to take advantage of this globalization companies must work collaboratively with other entities, be it externally with other companies or internally between business locations to reap the rewards of the global market. With this increased need for collaboration comes the need for tools and processes to facilitate it.

Computer-Supported Cooperative Work (CSCW) was created as the study of aiding collaboration with computer software (SW) and hardware (HW). This study crosses many disciplines (computer science, economics, psychology etc.) and as such is a broad domain. Collaborative tools (Group-ware technologies) are the systems that build on this CSCW work in order to aid people in their collaborative work. Early collaborative tools were based around resource management, while this is an important factor of collaboration it is only one many factors. In order to develop an effective group-ware solution to support collaborations there is a need to understand what it is that makes for effective collaboration. However, due to the complexities of collaboration and CSCW, along with other factors such as the difficulties with evaluation of these systems many of these applications have failed to support collaboration to the extent required [1].

1.2 The Aim

The aim of this project is to analyse a typical business group scenario in which collaboration is used. This will be done in order to determine a set of requirements for a new group-ware system to support the collaboration. From these requirements a design specification and prototype will be produced as a means of implementing the requirements for evaluation.

The analysis will be performed using a collaborative framework developed by Johnson & Hourizi[2]. The framework aims to detail the contributing and limiting factors within the collaboration in terms of conflict mitigation. It has previously been applied to collaborative autonomous systems as a means of determining how to mitigate these conflicts within the collaboration. Within the analysis the merits of group-wares will be assessed as to how they aid and fit within the collaboration detailed.

After the analysis of the collaboration, a set of requirements for the new system will be detailed, these will aim to detail how a new group-ware technology...
might aid collaboration through enabling communication and coordination to mitigate conflicts. The design and prototyping of these requirements is necessary for evaluation purposes. This evaluation will aim to assess the requirements as well as the methods and framework used within the analysis.

Firstly there is a need to establish a firm grounding of collaboration, its contributing factors, CSCW research, group-ware technologies and how businesses use collaboration and collaborative teams. This will be done by reviewing and comparing the relevant literature within these areas and determining their relevance to this report.
2 Literature survey

2.1 Introduction

The aim of this section is to provide an overview of the current literature as well as its importance and relevance within collaboration. Collaboration is a broad domain across many areas and as such, there has been a large amount of work based around understanding, evaluating and improving collaboration, these often take the form of collaborative frameworks. This section will review such frameworks, especially those within a business context as this is the domain in which the report is based. These business collaborations are often some of the most complex collaborations, due to the actors being human and therefore introducing irrational behavior. Business collaborations have become increasingly important due to the complexity of modern day business processes and activities as well as increased globalization within businesses which may require collaboration in order to leverage it's advantages. Many group-ware systems have been developed to support this collaboration, these will also be reviewed within this section.

2.2 Collaboration

There has been a great deal of research into collaboration over the last 50 years, collaboration is a very broad topic with many different applications and as such there have been many different definitions of collaboration proposed. Before exploring how to improve collaboration, a firm grounding of how collaboration is defined and what it is for a group to collaborate needs to be established.

The term cooperation is often mentioned alongside collaboration. Nezamirad et al. claimed that cooperation is a state of being which has a number of components, one of which being cooperation [3]. Therefore it can be concluded that cooperation is a subset of collaboration. This is expanded to explain that a collaborative activity is when participants engage in an activity in order to achieve a common objective, as part of this, or in order to achieve this they cooperate with one another to facilitate the achieving of the common objective. To this end it follows that a team cooperating may not be working towards the same overall goal, whereas a team in collaboration would be working towards the same overall goal or objective.

Collaboration is a prominent aspect of modern work and plays a crucial role within the modern organization. Contemporary work is characterized by constant the exchange of information in distributed environments, whether that be inter department, inter office, across a virtual team or to an outsourcing or off-shoring scheme. These human to human exchanges require the development and maintenance of collaborative relationships which are essential to completing a given task[4][5]. This theme is repeated by Weiseth et al who claim that “Collaboration is increasingly the modus operandi of the modern business world” [5]. This being the case it follows that an organization should look to understand collaboration and facilitate its presence where ever possible.
Tran and Biddle carried out a study for Collaboration within game development, whilst the findings are specific to this area they can be applied to collaboration on a much wider spectrum; “Effective collaboration requires a team that respects each others contributions, communicates frequently and shares a similar conceptual model of the product and goals.” [7]. Whilst this may seem obvious, it is important to understand exactly what leads to effective collaboration so that future collaborative tools can help facilitate these processes. As well as this they proposed that the four main factors affecting collaboration are physical space, social space, the team mental model and organizational goals [7].

Much work has centered around the need for shared understanding or a common ground between the group members for effective collaboration [8, 9, 10]. The term grounding was defined by Clark and Brennan as the process of affirming that what is communicated, is interpreted and correctly understood by the recipient. In the same paper Clark and Brennan go on to state that common-ground or grounding is constantly evolving and building as the team interacts and shares knowledge; “all collective actions are built on common ground and its accumulation” [14].

2.3 Collaborative models and frameworks

Frameworks are popular methods for detailing complex entities or processes. Collaboration is no excuse to this with many frameworks existing [12, 13]. A collaborative framework provides a mechanism that guides users through a set of steps, applications and data conversions via a common interface to the process being followed. They can provide a unified view of the needs and functionality of each user thus allowing a coherent approach to the specification of the collaborative tool as needed in order to realize the implementation of the tool [15].

Generic Collaborative frameworks have been an area of research for the past few decades, resulting in commercial collaborative products such as Microsoft share-point as well as many other systems.

Weiseth et al developed a framework which depicts three elements of collaboration; process, support and environment. These are shown to be implemented through either the business or the technology [6]. This is illustrated in Figure 1. They state that successful collaboration requires management of all three elements through the business and the technologies.

![Weiseth’s Collaboration Framework](image)

Figure 1: Weiseth’s Collaboration Framework
There have been many collaborative frameworks and theories which have been important for building a understanding of collaboration and the contributing factors to its success. These have been used to implement technologies, processes and structures (organizational, group etc.) which aid with collaboration.

Some such research on collaborative frameworks has been performed by McGrath who built on others collaboration work and theories and came up with his Group Task Circumplex. He took previous work on task classification and extracted the main ideas, expanded on them and then put them together into a conceptually related set of distinctions about tasks. This Group Task Circumplex is illustrated in Figure 2.

Co ordination theory is defined as a set of principles about how activities can be coordinated, which is to say that actors work together in harmony [16]. Coordination theory was not developed with a specific domain in mind but instead is meant to contribute to all domains and work alongside domain specific theories and concepts. The theory is designed to be applicable to many areas where questions about how people coordinate their activities are central.

Within this coordination theory the normal issues around coordination appear, these are issues such as the issue of dividing overall goals into sub goals and action. The assignment of these actions to groups, sub groups or actors
also needs to be determined along with the allocation of the necessary resources required to complete the task. Models such as this theory have stood the test of time, one reason for this is due to its abstracted, non specific nature. It states a set of principle which would help coordination in the aim of achieving a goal. This coordination theory has been used in conjunction with other frameworks to gain a better idea of a collaboration as a whole within scenarios. Coordination theory is illustrated in Figure 3.

Figure 3: Coordination Theory and its Application

“The Wheel of Collaboration Tools” is a holistic Framework developed by Weiseth et al for analyzing and specifying collaborative solutions. The typology consists of three layers; the inner layer represents the functions for content management and process integration, the middle layer displays a generic decomposition of the functions needed to support coordination, production and decision-making and finally the outer layer represents the interface to the collaboration functions, i.e. the devices, physical work-spaces, portals etc.. This is shown in Figure 4.
The wheel of collaboration tools tackles one of the hardest issues within collaborative tools, the evaluation of their performance. It is hard to define a hard set of metrics for collaborative success and so to determine if a collaborative tool is successful is very difficult.

In their widely cited paper Olson & Olson focused on the “social-technical conditions” they believed to be essential for distributed teams to work effectively. The paper concluded that there are four key concepts in the area: common ground, coupling of work, collaboration readiness, and collaboration technology readiness. These can be summarized as below;

- Common Ground - This refers to the knowledge which participants share or have in common. As well as this, each must be aware of the what knowledge is common. This common ground is not only established from the general, personal knowledge of someone’s background, but can also be determined by factors such as behavior, demeanor and appearance. It then stands that if fewer cues are available it is harder to establish this common ground and misunderstandings or misinterpretations is more likely to occur [10].
• Coupling of work - The coupling of a piece of work refers to the amount and type of communication required to carry out the work. “Tightly coupled” work is said to be strongly dependent on the skills of the workers and the work itself is “non-routine, even ambiguous”. This type of tasks required frequent, complex communication between team members “with short feedback loops and multiple streams of information”.[10] This is in contrast to “loosely coupled” work which is much less dependent on other factors and the work is more routine. This loosely coupled work requires either less complicated or less frequent communications or interactions. The coupling of work is task dependent and the more tightly coupled the work, the greater need for common ground. It is this tightly coupled work which is difficult to achieve within virtually distributed teams.

• Collaboration Readiness - This is the idea of worker motivation, the need and reward for collaborative work, or a “Willingness to share” if you will. The incentive structure needs to be aligned with information sharing.[10]

• Technology Readiness - Not only does the organization have to support collaboration through their structures and hierarchies but also in terms of technological infrastructure and norms.

Olson & Olson state that Common Ground needs to be supported, when working in distributed teams, through technology. This is achieved through recreating the communication cues which are lost through the lack of face-to-face communication. Clark and Brennan [11] described how different types of media allow for the construction of common ground through joint negotiation and expression. The results showed how varying amounts of effort is required to build this common ground[11]. The characteristics of medias which served to build this common ground were concluded to be;

- Copresence—same physical environment
- Visibility—visible to each other
- Audibility—speech
- Contemporality—message received immediately
- Simultaneity—both speakers can send and receive
- Sequentiality—turns cannot get out of sequence
- Reviewability—able to review other’s messages
- Revisability—can revise messages before they are sent

As well as detailing these characteristics of medias allowing for common ground to be established Olson & Olson also detailed aspects of co-located meetings which could possibly be replicated in distributed scenarios through the use of technologies. They detail both “Rapid Feedback” and “Multiple Channels” of information as key areas which can be strongly supported by technology within the future. This list can be found in the Appendix 8.3.
2.4 Framework for Large group collaborations

In 2008 Johnson, Hourizi et al. [2] developed a framework to manage collaborations within complex organizations. The paper specialized on the application of this framework to autonomous systems, however, the framework was developed to work with collaboration of all types of agents, be it among non-human agents, between humans or any mixture of the two. The framework provides a base for understanding a collaborative systems, it does this by detailing its affecting structures and relationships between them, conflicts and the communication and coordination mechanisms used to mitigate them. Johnson argued that while previous collaborative research and frameworks are well developed within the domains they were intended for (small group collaboration), little work had been done with regards the collaborative processes and structures needed for large group collaboration. The framework takes into account the organizational structure, the group structure, the tasks structures and the resources required. It is based upon the concept that collaboration and conflict are inseparable, to this end the framework determines how these can be mitigated. The framework is illustrated in Figure 5.

![Framework for Large group collaborations](image)

**Figure 5: Framework for Large group collaborations**

2.4.1 Task Structure

Task Structures have been modeled in many different ways in order to try to understand the task at hand. Ideas such as Goals, Operators, Methods, and Selection rules (GOMS) and Task Knowledge structure (TKS) have been used in this area in order to gain such an understanding of the tasks. TKS is a theory about how task structures may be reflected in a persons understanding of the task described in terms of the knowledge required to perform the task [17]. However they all aim to achieve the same goal, the decomposition of complex tasks into understandable sub-tasks. This is done so that the sub tasks can be analyzed and the most efficient task structures determined.
Task Analysis

Task Analysis (TA) was developed alongside ideas such as Critical Path Analysis (CPA) & System Analysis, however TA is analysis in terms of human behavior. Several task structure modeling techniques require TA to be performed prior to implementation. TA is typically based around answering three main questions: What information about the task is required? Where to obtain the required information? and how to obtain the required information?

Goals

This concerns the properties of collaboration structures and process required to achieve tasks, rather than the modeling or analysis of tasks themselves [2]. Work may need to be split between various sub-groups and actors within these groups. This distribution of work must be in accordance with the capabilities of the sub-groups and actors to avoid the necessity of external intervention. This includes a full understanding of local sub-goals in order to determine that the group is capable of achieving the goal.

Actions

These sub-groups and actors within the system must gain an understanding of the actions required to achieve their set sub-goals. The level to which this process occurs depends on the actors, sub groups and tasks in hand. If the task is simple and the manager feels the sub group or actor is capable of doing this with minimal interaction then it may simply be an agreement that the sub goal will be achieved, in other circumstances it may be that the sub group require a detailed description of how to achieve a sub goal [2].

2.4.2 Organizational Structure

Organizational structure refers to the structure of relationships within a group or groups as well as in the organization in which the group or groups exist. Organizational structure is very important in collaboration, this is due to the fact that it can very easily have an overbearing affect on other aspects of collaboration very easily. “Organizational structures can both interfere with, and supplement co-operative work practices in a way that current technologies cannot provide support”[18]. As such, organizational structure plays a big part in defining what conflicts are likely to arise.

2.4.3 Group Structure

Roles

All groups must specify and understand the roles both individual actors as well as sub groups fulfill within collaboration. These group collaborative roles will differ depending on organization structure, actors, task etc.. These roles are important as if managed well large groups will be able cope with monitoring
every actor, action and objective, collaborative flow will be maintained and the

group can adapt to dynamic changes such as the loss of an individual actor, loss

of a resource or the alteration of a high level goal by an external authority. As

well as this it will also guarantee that the group goals are achieved as well as

utilizing available resources appropriately [2].

**Actors**

In a group formed to achieve a collaborative task there is a need to know the

resources and capabilities needed to complete the task, these must be identified

by an actor(s). In some more complex tasks this may include the identification

do different combinations of resources and capabilities, any one of which could

be used to achieve the task in hand. In such a case it would be required of any

actor to include the identification of appropriate resource/capability combina-

tions, the mapping of specific combinations into collaborative task completion

outcomes plus the identification of resources and capabilities available by po-

tential group members [2].

**2.4.4 Resources**

Resources which are required to complete the task must be identified and

sourced. Sometimes resources are owned or held by groups or actors. These

owners or holders must be identified and an understanding of the resources

must be established [2].

**2.4.5 Conflict**

Conflicts are common in everyday life. They arise for a number of different

reasons such as “the division of scarce resources, policies, what to consider in

the decision making process, how to approach the task, what humor is funny,

what norms and values are valid and appreciated, and which beliefs are to be

respected.” [19].

Most literature talk of two types of conflict. These being “affective” or in-

terpersonal based conflict and “substantive” or task based conflict [21, 9, 19].

These forms of conflicts have been linked to negative performance within group

situations [19]. There has also been work which has concluded that this negative

performance is worsened when the group is distributed [9].

However, other literature details the importance of conflict within group

situations. Phenomenons such as groupthink, state that a lack of conflict leads

to a negative performance within the group [22, 23].

**2.5 CSCW & Group Ware**

Collaborative tools or groupware refers to software of with the aim of helping

people involved in a common task achieve their goals, or as put by Ellis and

Gibbs “computer-based systems that support groups of people engaged in a com-

mon task (or goal) and that provide an interface to a shared environment.” [24].
The term Computer supported cooperative work (CSCW) is often used in terms of collaborative group work “Computer supported cooperative work (CSCW) is the study of how people use technology, with relation to hardware and software, to work together in shared time and space”[25]. The term CSCW is used to define the research area and groupware defines the technologies within the research space[25]. It is often argued that groupware isn’t a definitively bound area but more of a spectrum. A simple electronic mail system (e-mail) could be considered as groupware, but would be low on a spectrum of group-wares as it doesn’t support collaboration or cooperative work in a vast amount of ways. At the other end the spectrum would be something like an electronic classroom system which uses multiple windows on which information about the subject is posted, emulates a classroom in that the systems allows a teacher to present an on-line lecture while students study remotely from personal workstations, along with other features which represent a traditional classroom[30]. This would be considered a piece of groupware software that was very high on the said spectrum. Within CSCW Johansen stated that two dimensions exist; space and time. Figure 6 shows the four CSCW quadrants as defined by Johansen in 1988[31]. The models shows what type of groupware is needed to support the collaborative work depending on the space and time in which the collaboration exists.

Figure 6: CSCW quadrants

Groupware is often employed by business with the aim of improving co-operation and collaboration both within groups as well as between groups or teams [29]. They find groupware streamline business processes and makes work more efficient. Message systems, multi-user editors, group decision support, electronic meeting rooms, computer conferencing, computer teleconferencing, desktop conferencing, work-flow, groupware tool-kits, intelligent agents and coordination systems are all types of groupware. The aim of these group-wares is to allow group members to collaborate through some type of infrastructure,
whether they be co-located or more importantly distributed.

2.5.1 Video Conferencing

Video Conferencing has evolved from teleconferencing, the idea being that if you can see the person you are communicating with there are more channels (visual cues) and therefore the communication should be of a better quality than simply teleconferencing. The idea became popular during 70’s with the idea that it could reduce travel costs.

The latest incarnation of video conferencing is the idea of electronic meetings and MultiView. Electronic meeting rooms involve video conferencing as well as other tools with the aim of creating a virtual meeting from different geographical locations. MultiView is a new video conferencing system that supports collaboration between remote groups of people. MultiView builds on video conferencing by being spatially faithful. This allows MultiView to support interaction cues that video conferencing has failed to in the past, due to spatial warping and distrust of the perspectives being shown, cues such as gaze and gesture, in a way that should improve communication.

In a paper in 1988 it was estimated that 50% of all business meetings involving travel could have been conducted over videoconferencing with no loss of effectiveness [20].

Video conferencing has not however had the impact which was first predicted when the technology was developed. There are many reasons for this, but three of the main areas where video conferencing still fails are: Lack of, or incorrect eye contact. This affects activities such as conversation turn taking, perceived attention and intent as well as other cues used in day to day communication.

Secondly is the idea of appearance conciseness, the issue of being self aware to a level you would not normally be, due to the presence of cameras. This can restrict how involved certain actors become in the group work. Thirdly is the lack of informal communication, which businesses thrive on [37].

2.5.2 Document Management Systems (DMS)

DMS is software that indexes and profiles documents based on content; controls documents using such functions as check in/check out, version control, audit trails, and security of information; and facilitates searching by profile values or by other hierarchical structure such as folders and files. DMS creates structure and access methods for electronic documents and provides a database of documents that can be searched for and retrieved [42].

Subversion (SVN) is a hugely popular form of DMS. It allows groups to work on the same central set of documents and helps avoid resource conflict by using a check in/out system. This means a user checks out a document from the central repository, works on the document and when it is completed will submit it back to the central store. There are tools for merging and conflict resolution if a document has been altered by two or more different agents.
2.5.3 Wikis

A wiki is a website that has facilities for easy creation and editing of any number of interlinked web pages via a web browser using a simplified markup language. Wikis are typically powered by wiki software and are often used to create collaborative websites, to power community websites, for personal note-taking, on corporate intranets, and in knowledge management systems. The idea of collaborative groups editing the same website and with no outright ownership makes this a very powerful collaborative tool. Wikis rely on their user group to manage the site; it was commented when Wikis were first launched that they would be rendered useless by destructive input, however this has not been the case and successful Wikis now have a strong presence on the web.

2.5.4 Instant Messengers

Instant Messaging (IMs) has been widely adopted both by businesses and individual users. IM is a form real-time text based communication, between two or more users. They use a client at every user station. This form of communication has been shown to be effective for informal communication and is useful for establishing common ground and shared understanding due to its back and forth synchronous nature. Research by Cameron & Webster stated that “IM symbolizes informality” [40].

This informality within communication is crucial and one of the reasons that IMs have been so widely adopted. One failure of IMs is said to be there lack of paper trail. This can lead to compliance issues when used within a business context, while another critique has been the lack of richness within the communication. However IMs are widely used as an effective, quick and informal collaborative tool. This informality is important due to the fact that “Collaborations in organizations thrive on communication that is informal because informal communication is frequent, interactive, and expressive” [37, pp. 37]

2.5.5 Online Meetings

These are some of the most recent developments within collaborative tools. Sometimes refereed to as web-conferencing these tools are used to conduct meetings, presentations or training sessions “Online”. As these are still in the relatively early stages of development they differ greatly from implementation to implementation. Adobe Acrobat Connect Pro is one such solution and is considered to be one of the most complete packages offering desktop sharing, Online meetings & presentations, videoconferencing, IM functionality and others. As well as these it allows drawing or annotations on the slides of a presentation from users. All of this is based around a formal, organized setup. This would not be useful for quick ad-hoc IMs between users but in terms of an organized meeting which users have been made aware of before hand it appears to facilitate many communication channels [41].
2.6 Virtual Teams

There are many definitions of these virtual teams; however a common theme throughout these is the dependence on technology. Virtual teams have been described as “a group of people who interact through independent tasks guided by a common purpose and work across space, time and organizational boundaries through the means of technological communication” [31]. Other definitions have included the idea of global virtual teams as “internationally distributed groups of people with an organizational mandate to make or implement decisions with international components or implications.” [28] pp.473.

Virtual teams have been widely adopted across many industries due to their virtues and now form large parts of organizations structures. They save time and travel expenses, provide easier access to experts and expand labor markets globally [39].

DeSanctis stated that the virtuality of organizations and teams is a granular property, most companies are neither purely traditional nor virtual but a hybrid somewhere in-between the two [27] and while a team may be distributed it is rare there are never any face-to-face meetings. DeSanctis went on to state that how virtual a company is depends on four factors; temporal, spatial, cultural, and organizational dispersion.

Temporal refers to the dimension of time and is based on the extent at which the employees act asynchronously. Spatial refers to the extent of geographical spacial diversion of the employees. Cultural refers to the extent to which an organization consists of employees from different countries or cultures and organizational dispersion refers to the degree to which organizational processes extend the boundary of the focal organization [27].

2.7 Findings

This section has established a knowledge base in a few key areas, those of: what is collaboration, some of the frameworks used to understand and define collaboration, an explanation of the framework which is intended to be used within the analysis, some group-ware technologies and the idea of virtual teams. From this point the report can now move forward to apply this knowledge, especially with regards to the large group collaboration framework, in order to perform the analysis.
3 Analysis

This section details how the analysis was carried out in addition to the summary of the findings. The objective of this analysis is to understand collaboration within a typical business context. This entails a distributed team working together to achieve a goal, i.e. a collaborative distributed (or virtual) team. These teams, as detailed within the literature survey are common within the modern workplace. This study will be achieved through a thought experiment applying scenario based analysis as well as the use of relevant literature within the domain. Scenario based analysis entails devising the scenario first, a synthetic description of an event or series of events or actions. This scenario will be of a business team, acting collaboratively to achieve certain goals. After this has been detailed an analysis of the contributing and limiting factors towards collaboration within the scenario can be carried out, using the framework in order to determine where collaboration can be aided by a new group-ware system.

3.1 Scenario

As detailed above scenario based analysis will be used to critique the effectiveness of the collaboration within the business. This analysis gives a good context on which to apply the framework in order to do determine the limiting factors of collaboration within the business group and where new group-ware could help with facilitating collaboration.

The problem domain to which the framework is applied needs to be defined. The framework was originally designed to work in dynamic collaboration environments, these include characteristics such as highly diverse and dynamic problem domains and scenarios which require highly emergent solutions. This area of collaboration is the final target for the framework, however as this is the first iteration applying the framework to a human group collaboration within a business context this scenario will be modeled as a more static collaboration. Further work will introduce the idea of dynamic collaboration within human collaboration.

The scenario was developed in order to apply, evaluate and test the framework against a human collaborative group. The analysis will then assess how the collaborative group-ware tools outlined within the scenario help to reduce conflict and improve collaboration through communication and coordination. The analysis should also outline where the collaboration is limited and by what factor(s). These bottlenecks are then the areas which a new group-ware technology should improve the collaborative flow. In previous papers the framework has been applied to non-manned autonomous systems (AS’s) with a scenario based approach in order to improve the framework and the understanding of AS collaboration [32], the same approach has been adopted in this report. The scenario has the following structures:
3.1.1 Background

Human Collaboration is a huge part of modern business, the idea of virtual teams and groups is common place in many work environments. This means that a team may not be geographically located in the same area and so a traditional face to face meeting is not possible without the high cost of travel. For this scenario the team is a software development team split across two different geographic locations. There are three developers in a New York office and two developers and a business analyst based in London. Each developer’s specialist skills are within different areas. The team are a subset of a large accountancy firm and are responsible for the development of a new application which is to be designed, developed and tested by them before being released to the business to help with accounting. The main business need is being driven from the New York side.

The team has been selected such that each developer is a specialist in a different area and so they need to work effectively as a team to produce an effective piece of software. They have been set a time-line of six months to have the software released and so they will need to be as efficient as possible to avoid wasting time.

3.1.2 Goals

The team’s overall goal of “designing, developing and testing a new application to help with accounting” can be split into several high level sub-goals.

1. Develop a set of design criteria representative of what the business requires
2. Make decisions as a group on several areas such as; Final design specification & Development work allocation
3. Implementation of code as both individuals, pair programming and as a group
4. Work together in dynamic subgroups to overcome problems on the fly
5. Test and debug code in Ad-hoc groups

3.1.3 Resources

Each member of the team has their own desktop computer with all the required software to work on the project. For the scenario three groupware technologies are available to help overcome the geographic differences between the team and to aid collaboration. These are Wiki’s, Instant messaging and Videoconferencing. The company has invested heavily in these technologies in order to help with collaborative group work.
3.2 The framework and the scenario

According to the large group collaboration framework, conflict is an inevitable part of collaborative work. This framework sets out independent components each important in its own right to group collaborative work and each having an effect on the conflicts experienced within the collaboration. However, they also overlap each other and are interrelated and in some cases one component can dictate another, i.e. group structure and organizational structure can have large overlaps.

With all the components of the framework there are clearly a huge number of combinations between the layers. This is all assuming that each component only has one definition. If the collaboration was between two companies in a scenario such as outsourced product development, two organizational structures would have to be considered. (e.g. even within the scenario described there could be a possibility of two group structures split between London and New York if the groups were treated as two separate groups collaborating to achieve a goal rather than internal group collaboration of one multi-based group.)

Not all combinations of these components can be expected to produce effective collaborative flow between the group members. The idea of the framework is to start to build a map of which combinations produce an effective environment for collaboration. In the scenario, each component of framework will assume only one static definition. The purpose of the paper is to analyse this form of collaboration, between group members in a business context in the hope of finding the bottlenecks which occur within communication and coordination. These bottlenecks themselves do not lead directly to conflict, but they can impose limitations on the level of conflict mitigation which can be achieved dependent on the given circumstances.

The analysis will also include an evaluation of the group-wares outlined in the resource structure to determine how these resources currently perform to facilitate coordination and communication in order to avoid, identify and resolve conflict as shown in figure 7.
3.2.1 Organizational structure

Organizational structures are determined by many influential factors, and are set from the top down. Although Collaboration is important to companies it is often unlikely that they would be willing to change such a factor for the sake of collaboration. Modern companies which embrace innovation are much more likely to structure their organization in such a way as to suit business needs, one of which may be collaboration. However, many companies have organizational structures that were set far back in time and are observed for heritage reasons and not for the benefit of collaboration. This being said it stands to reason that this is a rigid inflexible variable and is simply inherited from the company.

In the scenario a simple hierarchical structure will be assumed. This is due to the fact that many companies follow such an organizational structure at least to some extent. The team both sides of the Atlantic is under the control of the Business Analyst in London, who is the team’s manager. After that the team is part of the hierarchical organizational structure with has many tiers.
Figure 8 shows the a section of the organizational structure along with the lines of communication. This shows that not only can information flow from the business analyst to the developers but also back from developers to the analyst as well as between developers. The communication paths depict normal behavior within organizational structures in companies, a team being able to communicate between themselves and with their manager is commonplace. The figure does not depict the whole organization just a sample section of it, in reality the company is likely to have many more tiers.

### 3.2.2 Group Structure

This area is one of obvious overlap, as the group structure has been bounded by the organizational structure. However within the group there are general roles to be allocated. From the organizational structure it is clear that there would be no point in appointing a developer as a supervisor due to the potential conflicts that would arise between the organizational structure and the group structure. The Business Analyst is immediately responsible for the managing of the project, however it is likely in reality (as in this scenario) that overall responsibility belongs at least one tier above. For some of the goals the Business Analyst must work with the Developers to achieve the goals; design criteria and decision making must involve all members of the group. Developers alone are responsible for carrying out some of the tasks in order to achieve other goals, such as; coding, debugging, testing and problem solving. This is shown in Figure 9 below.

Business Analyst - This is a link role, a business analyst liaises between the business and in this case the build team. They have a combination of business and technology skills making them suitable for this role.

Developer - These members are responsible for the implementation of the coding, having an input on decision making as well as problem solving. They have specialist skills in technical areas, not necessarily the same for each.
### 3.2.3 Task Structure

To understand collaboration within human groups we need to gain an understanding of what tasks are required to achieve the goals. To establish the required task structure, task analysis needs to be applied. This is used to understand tasks performed by the operators, in this scenario that is the Business Analyst and the Developers. The output of such a task is a formal definition of goals, tasks and activities to be performed in order to achieve them. There are a multitude of different ways to perform task analysis, methods such as GOMs and Hierarchical task analysis have been popular among the research community. Most have their own advantages and strengths which make them more suitable for certain domains and not for others. Whilst they all differ in their approach, they are all designed to achieve the same purpose, defining goals and activities required to achieve them.

In this scenario task knowledge structures have been chosen for the purpose of task analysis, this has been proven, through other research, to be an effective approach for such a scenario. Task Knowledge structures (TKS) involves the storing of the knowledge required to perform a task that has been learned. TKS theory proposes that when a person is confronted with a new task, which they do not possess the knowledge to complete the task they can apply the TKS from other similar tasks in order to complete the task.
Task Knowledge Structures

Tasks knowledge structures are traditionally gained from empirical research methods such as interviews and observations of the operator. The aim of this is to establish:

1. A goal-oriented substructure – goals, sub-goals, enabling/conditional states
2. Task procedures – executable procedures for achieving particular goals/sub-goals and associated conditional/contextual knowledge
3. A Taxonomic substructure – properties/attributes, representativeness, importance of objects required in task completion.

The goals and substructures will be defined for the scenario, the task procedures in the scenario will be set more rigidly than is likely to be encountered in a real world case. The taxonomic substructure is not presented due to the nature of the analysis, being a conceptual scenario with no expert task performers or specialist within this area (collaborative software development within an accountancy firm) this cannot be detailed.

The framework also supports dynamic task procedures, however due to the static nature of the scenario the use of these will be limited.

Task structures in the scenario

As the project’s overall goal is very high level it has been broken down into several sub-goals as in section 3.1.2.

1. Develop a set of design criteria representative of what the business requires
2. Make decisions as a group on several areas such as; final design specification and development work allocation
3. Implementation of code both as individuals, pair programming and as groups
4. Work together in dynamic subgroups to overcome problems on the fly
5. Test and debug code in Ad-hoc groups

These five sub-goals can be further broken down into subsequent goals. It is obvious however that these sub goals depend on the states of other sub-goals. A high level goal structure has been developed for the scenario. It starts with the overall goal of “designing, developing and testing the desired software”, from here the initial goal is set which must be achieved before attempting any other goals. From here downwards all goals are conditionally dependent on the previous goal having been achieved. This is illustrated in Figure 10 below.
High Level Goals

From this illustration of the high level goals, the can be divided into their subgoals and conditional states. From these the sub goals and activities of actors required to achieve these sub-goals can be detailed. This high level analysis will also include bottlenecks experienced within the tasks between actors while they are working to achieve the set sub goals. Throughout the analysis areas where the groupware systems are utilized will be highlighted both as means of avoiding bottlenecks as well as for information sharing. As shown in figure 10 the initial goal is to “develop a set of design criteria representative of what the business requires”. The sub goal structure for this is illustrated below in figure 11.
Design Criteria

The above substructure shows the goals and procedures required for the team to achieve the initial goal of developing a set of design criteria upon which all subsequent goals are dependent. It shows that the team must first communicate with other actors outside of the group in order to gain an understanding of what is required from the system. After this step the rest of the procedures are bounded within the group structure. Subsequent to this, the group must determine what is feasible with the available resources (developers, time, hardware, software etc.) and go on to determine which, if any of these factors are the limiting factors or bottlenecks. Dependent on these restrictions the group must then achieve a shared understanding of what is required of the system and how they will achieve it. With the team being geographically dispersed this level of shared understanding and knowledge can be hard to achieve, according to Hinds and Weisband, for a team to achieve a common goal they must have a shared understanding of the goal and the process required to achieve it [8]. “Members of a virtual team can be expected to have more difficulty developing a shared group identity and attending to the information flows among team members. Such a lack of attention is likely to reduce shared understanding in virtual teams.” [8, p.27]. This sub goal can be further decomposed. Figure 12 shows

Figure 11: Goal substructures to achieve “develop a set of design criteria representative of what the business require”
this decomposition into further sub-goals required for achieving the overall goal.

Establishing a shared understanding of what is required and how to achieve it

From these sub-goals an in depth task analysis can be performed in order to determine how group members establish this shared understanding. Below is the task analysis for step two of figure 12 “Determine what is to be achieved for a specific task as a group”. This details the tasks required for the group to build a shared understanding of what it is they need to achieve.

Figure 12: Goal substructure for achieving the goal “Establish a shared understanding of what is required and how to achieve it”
Determine what is to be achieved

This task analysis highlights the processes required in order to build a shared understanding of what is to be achieved within a task. The analysis shows that the methods for establishing the shared understanding are different dependent on if the team is distributed or co-located. If teams are co-located then the increased communication cues and channels of face-to-face communication are used to establish this common ground and shared understanding. When not colocated the team need to use group-wares to establish this common ground. This is inherently more difficult due to the loss of a number of the communication cues and channels of information which were present in face-to-face communication.

As well as a task analysis for sub goal two (what is to be achieved), an in depth task analysis of sub goal 4 (how to reach the desired achievements) has been determined. Figure 14 illustrates this.
How to reach the determined achievements as a group

This is another area group-ware technologies are required when the team are distributed, aiding information flow to help build this shared understanding. Of the three group-ware used in the scenario instant messaging & Wiki’s especially can help build shared understanding among virtual teams by allowing synchronous communication and rich information sharing respectively, however there are still bottlenecks that these technologies don’t address. Areas such as design and other creative tasks often require more than text to communicate. It is often harder to communicate an idea than a fact or problem and even more so to put an idea down on paper, therefore trying to share a creative idea or problem electronically with a colleague or a team is incredibly hard, especially when synchronous communications, in this scenario instant messaging, is restricted to text only. This difficulty was detailed in work by Nemiro who found within her research that “idea generation could not be effectively accomplished without face-to-face interaction.” [36, pp. 75].

After establishing this shared understanding of the problem domain the team can determine what it is that is required and how they can achieve it. From here
they can then go on to produce a set of design criteria from which the system can be implemented from.

**Group Decision**

Figure 15: Goal substructures for the team achieving “Decision made as a group”

Once initial design criteria have been established the team need to make decisions on several matters as a group. Issues such as final design specification, work allocation, milestones, deliverables etc. need input from all of the team members. Figure 15 shows the processes required to achieve the goal of making these decisions as a team and achieving an optimum output. In order to achieve this goal and come to a group decision the team must again build a shared understanding of the problem domain and possible solutions. To facilitate this, all group members must feel comfortable to contribute possible solutions, critique others solutions and question where necessary. This could be by means of videoconferencing in a virtual team, allowing all group member to synchronously brainstorm solutions as well as gaining the shared understanding through the flow of ideas and information. However, this does not necessarily lead to optimum input from team members due to factors such as the group-think phenomenon [22] [23]. Wikis can alleviate this to an extent due to the non-synchronous nature of communication.

No matter whether the solutions presented are optimum or not the best solution from the ideas presented still has to be selected. This process can be
mediated by decision making software, however this is not available within this scenario. The outcome of this process must be a decision that all team members agree is a suitable way forward and as such should be an optimum decision.

Coding

Figure 16: Task structures for Developers to achieve the goal “Implementation of code as both individuals, pair programming and as a group”

Figure 16 shows the processes required to achieve the goal of implementing the code. The initial process leads on from the previous goal of group decision making as this will be required to split the coding of the project into modules and assigning them to sub-groups. The code is implemented either individually or as pairs dependent on how complex & critical the code is within the system. This pair programming is restrained by the group structure, pair programming is inherently difficult if the pair are not co-located. There has been work suggesting it is possible to perform pair programming in distributed teams [38], however with the resource structure in the scenario it is not likely to be feasible. This is very much a bottleneck, the New York office has three developers, if two modules which are determined to be complex enough to be pair programmed, are assigned to New York they can not both be carried out at the same time as there are three developers and one developer is an idle, wasted resource.

After the coding has been finished the developer(s) must report to the team that they have completed their allocated modules as well as merge their code into the central code base.
Overcoming problems ad-hoc

The above illustration shows the processes for overcoming problems with ad-hoc, dynamic subgroups. It shows that for the goal to be achieved first the problem must be identified. This goal substructure is for overcoming problems which occur while trying to achieve the overall goal. It is based around the idea that while one person may not be able to spot or solve an issue, if there are more people working to solve it then the knowledge base will be broader, because the team members have different strengths and expert areas. So once the issue is spotted the next task is to form a sub-group, this may only exist for the task at hand or may last longer, and ensure there is a shared understanding of the issue. This can be easily done in co-located subgroups but across distributed subgroups the information flow isn’t so easy, group-wares are designed to support this flow within the distributed teams. In this scenario the only group-ware resource which offers synchronous communication paired with the informality which helps build this shared understanding is instant messaging. However as with figure 15 this flow is limited by the lack of richness within instant messaging, this is a bottleneck for communication. Once the shared understanding has been achieved the subgroup must come up with solutions to the problem and choose a suitable solution.
Testing and Debugging

Testing and debugging is the final high level sub goal of designing, developing and testing the software. Initially a test plan has to be developed, this is a team task and will require input from all group members. A Wiki would be used for this development. This would help the team contribute from geographically dispersed locations. This is then available for all members to refer to during testing as a central shared resource. The testing will then be split between the group, this would utilize the group decision making process shown in figure 15. From here the developer would carry out there own parts of the test plans if all tests from the plan are passed they will mark them as passed and report this to the team via the Wiki, else they will have to change the code in order for the tests to be passed and re-run the tests, this continues until the tests all pass, at which point, they carry on to mark the tests as passed and report via the Wiki. There is however a bottleneck within this situation, in that one developer could be testing an area which has numerous failures and so takes a longer period than the other developers who, once they have finished there sections, are idle resources.

Figure 18: Task structures for actors to achieve the goal of “Testing and debugging code in ad-hoc groups”
Figure 19: Goal substructure for developing a shared understanding within a group

Task structure findings

Within the goal substructures a certain sub-goal appears several times. This is the goal of achieving a shared understanding between group members and has been mentioned previously in section 3.2.3. The section explained that this shared understanding is often a bottleneck as it is inherently hard to achieve when a team is not co-located as the richness and cues of face-to-face communication are lost. Group-ware technologies aim to introduce richness into the communication in order to manage the bottleneck this brings. This goal can be abstracted out to be purely the goal of achieving a shared understanding between group members. Figure 19 shows this abstracted goal decomposed further into more sub-goals.

Shared understanding

This shared understanding, or common ground is referred to by Olson & Olson [10] is critical and is required in many of the task structures shown above.
3.2.4 Resources

The main resources which are used within the scenario are the group-wares outlined at the beginning. These are instant messengers (IM), Wikis and Video-conferencing. Group-ware technologies are designed to improve information flow and support collaborative work. These should act to reduce the number of bottlenecks found within the scenario with this improvement in communication and information flow. These are typically used when the communication needs to flow between group members who are not co-located, these members cannot communicate using face-to-face methods and so must try to communicate in another way. Each of the three chosen group-wares aid collaboration in different ways but all aim to increase the flow of communication and information required for collaboration to occur. Each one lends itself to different situations due to its strengths and limitation. The method of communication (or which group-ware) to use is dependent on factors such as; richness required, formality, severity, time span, criticality etc.

The media richness of Wikis which include items such as embedded videos, graphics, text, hyperlinks, animated diagrams aids with the communication of creative and design based information. The downside of Wiki’s is their asynchronous nature, the communication is not real time, the user creates and the Wiki resource but has no idea, when or if other users will refer to the page. There is also no feedback as there is no guarantee group members have seen the Wiki. Wikis communication richness can help in the avoidance and identification of conflicts. However the lack of synchronicity mentioned means that for conflict resolution it is not as useful.

Instant Messaging (IM) addresses the synchronicity issue that Wikis possess. IM however, has its own downfall in that it lacks the richness of Wikis. IM also suffers in that it is not a lasting resource to be referred back to in the future. The synchronous nature of IM means it is particularly useful for conflict resolution. IM seems to be a good way of building “rapport” which helps with the shared understanding due to its back and forth synchronous nature and informality. Research by Cameron & Webster stated that “IM symbolizes informality, and that IM is perceived to be much less rich than face-to-face communication” [40]. This informal communication within businesses is essential and makes IM a valuable tool [37], however although the disadvantage of richness is widespread within group-ware platforms it is especially a limiting factor of IM. This limitation is particularly evident in tasks such as creative design sessions, design sharing & problem solving where a richer media is required.

Videoconferencing as mentioned within the literature survey is a powerful collaboration tools. It replicates the visual and audible aspects of face-to-face communication which no other group-ware can achieve. However it does not serve to replicate some of the more subtle cues which face-to-face posses [35], nor is videoconferencing an informal, ad-hoc means of communication due to the set up time required for typical videoconferencing systems [37].
3.2.5 Conflict

The literature has highlighted two main types of conflict within collaborating teams. These are interpersonal conflict and task conflict. Interpersonal conflicts are disputes that arise from personal incompatibilities as well as clashing personalities. Task conflicts are disputes which arise from differing opinions and viewpoints about the work being performed [9]. This work has proved that the two forms of conflict are associated with lower performance in both co-located and distributed teams and that these associations become stronger when the team is distributed. This concluded that “conflict occurs between distant members as they struggle to come to terms with different perspectives, unshared information and tensions between the distant sub-groups” [9, pp. 290]. Task conflicts can also arise when there are two contradicting factors governing a task. For example within this scenario the need for both security and usability within the design criteria. There are many other examples of possible conflicts within this scenario. For example in tasks such as “design criteria representative of what the business require” one such task conflict could arise due to unshared information as to what the “business require” and so different sub-group’s ideas of how to meet the requirements will be different. Another example of a possible conflict could occur during the pair programming task. The two team members could conflict due to either interpersonal or task disputes; a task dispute could arise due as each developer believe a certain way is best to implement a certain functionality, or an interpersonal conflict could arise due to clashing personalities. Both of these causes are plausible for co-located and distributed team members.

Identify conflict

The framework states that an external change or dynamic changes to the collaboration dynamics will lead to new conflicts arising within the scenario. This is to say that a change to the group, task, resource or organizational structure leads to conflict. While this is true, the scenario analysed within this context is based around a static set of dynamics and so this identification of conflict is likely to be of less importance compared to a highly dynamic scenario. The use of dynamic sub-groups within the collaboration does change the group structure, these are used in problem solving on the fly within sub-groups, as seen in section [35]. This specific task structure is also likely to be lesser known to the participants, while the task structure of problem solving can be abstracted to a fairly generic structure the specifics are likely to be different for each occasion. In the circumstances it is important that the group maintain awareness of and communicate their resource, group, tasks and organizational structures within the group.

The framework identifies the need for the group to maintain awareness of the collaboration structures which govern the collaboration. This is facilitated through the mechanisms of collaboration and communication. Within this scenario of sub groups the group members must be aware of who is in the newly formed group, the tasks structures required to achieve the goals as well as any
resource changes. These matters need to be communicated between the group for any possible conflict to be identified. However, interpersonal conflicts are especially hard to identify within a task due to their personal nature.

As well as this need for maintained awareness there is also a need to understand the interactions between the structures within the collaboration. These are all interconnected and so a change in one can lead to the need for change within another. The group need to identify this change and adapt the other structures accordingly. Within the sub-groups example, the change of the group structure may mean that extra resources are required to achieve the goal, the group must identify this and adapt accordingly.

**Conflict avoidance and resolution**

This leads onto the fact that once identified actions must be taken to resolve the conflict or avoid it. This could be through a change in any of the structures which would lead to diffuse the conflict. This could mean increased or different resources, or the revision of the task in which conflict was identified. The mechanisms of communication and coordination must be used to ensure group members are aware of the situation as well as serving to reduce the potential for future conflict.

Examples of how these mechanisms can be used to resolve risk will build on the example already defined within the conflict section. Within the design criteria example, improved communication between the sub-groups would resolve the conflict as both subgroups would be basing their design ideas on the same underlying information and this will have been distributed using the communication mechanism.

For the pair programming conflict the resolution is much less straightforward. The mechanism of communication may help to resolve the conflict if the pair were geographically distributed and the conflict was a task conflict. This communication would allow each programmer to detail their viewpoint and come to a decision as to which is this best way to perform the task.
3.3 Conclusion

This analysis has worked through the scenario and examined in greater detail the structures within the collaboration well as the implications of each structure on others. The framework details the structures of the collaboration and how they may aid or restrict collaboration as well as providing a mechanism through communication and coordination to monitor, resolve and reduce conflict. The analysis has highlighted areas where communication and coordination are bottlenecked in their attempt to mediate conflict due to the group, task, organizational and resource structures;

**Bottlenecks within communication:**

1. The tasks structures produce some bottlenecks within communication
   
   (a) Some of the tasks require very high levels of communication which is hard to achieve given the group structure of the scenario. These are typically the tasks around tightly coupled work

2. The resource structures, i.e. the group-wares, limit this communication also with the:
   
   (a) Wikis not offering synchronous, tight feedback for effective communication within tightly coupled tasks;
   
   (b) IM systems not offering rich Media which also limits the effectiveness of the communication within tightly coupled tasks;
   
   (c) Videoconferencing being such a formal, cumbersome activity and as such not supporting informal, spontaneous communication between subgroups.

3. The group structure also introduces some bottlenecks within communication:
   
   (a) Groups working in different time zones makes communication difficult;
   
   (b) Communication across the distributed team, is inherently harder. This is especially true when dealing with tightly coupled tasks

**Bottlenecks within coordination**

4. The group structure imposes bottlenecks within the coordination of the group
   
   (a) This is due to time zones between the groups

5. Resource structure

45
Many of the bottlenecks and issues with communication can be attributed to the group structure with the team not all being co-located. The main challenge seems to be the establishing of a shared understanding between these distributed members. This appears to be in tasks which are undertaken as a group rather than split and worked on separately. The group-wares analyzed each have their strengths and downfalls which make each of them suitable for different forms of communication. However no one group-ware is completely suited to supporting the flow of non-text based creative information. Tasks such as interface design are examples which are failed by these solutions. Wikis don’t possess the synchronicity or tight feedback required for collaboration in this form of task, more for design and redesign, while IM doesn’t possess the richness or streams of information needed to represent the designs. Videoconferencing, whilst possessing the synchronicity and richness required, is not designed for such a task and as such is cumbersome and inefficient.

Tasks which require high levels of communication across the distributed group structure seem to be the main bottleneck within the analysis. This builds on work by Olson & Olson who concluded that “tightly coupled” work (tasks which require frequent, complex communication between team members) is the hardest to achieve across distributed teams [10]. They state that “Tightly coupled work” has a strong dependence on the skills the group members in ad-hoc, non-routine tasks.

Examples of such tasks within the scenario are;

- “Developing a set of design criteria” - This is a tightly coupled tasks with the need for multiple streams of information and short feedback loops in order to develop a set of design criteria representative of the business’s needs. Olson & Olson state that “collaborative design tasks are tightly coupled”[10] pp. 162 and this is the case for this task
- “Overcoming problems on the fly in subgroups” - This is tightly coupled as it depends on the specialist skills of group members within the ad-hoc sub groups for non-routine tasks. The analysis showed how this is difficult across distributed members due to the requirement for a shared understanding of the problem.
- “Decision making as a group” - These complex decisions that need to be made as a group are another example of tightly coupled tasks. This is especially true when the criteria or required outcome of the decision is ambiguous.”[10] pp. 162]
- The task structures for “How to reach determined requirements” as well as “Determining what is to be achieved” also contain highly coupled tasks.

These tasks which require frequent, complex communication between team members “with short feedback loops and multiple streams of information.” in
order to be successful [10]. The combination of these short feedback loops along with multiple streams of information appears to be limited in either one way or the other within the group-ware technologies analysed.

From this analysis a set of requirements for a tool which acts to aid collaboration within the scenario can be detailed.
4 Requirements

4.1 Sources

With the background investigation and analysis completed, the requirements for a new group-ware system can now be constructed. These come from two main sources.

4.1.1 Analysis

From analysis carried out several bottlenecks were identified. These bottlenecks were found when the group was working together as a distributed team, however within this analysis the group structure is non-changeable, as is the case in many business situations. As such the framework states that other structures must be altered to facilitate improved communication and coordination as mechanisms to manage conflict. The areas where the bottlenecks and communication issues were prominent are shown in section 3.3.

The requirements developed are for a new group-ware system which aims to improve communication & coordination within distributed team collaboration. The requirements will aim to resolve, or at least reduce the bottlenecks found by the analysis in order to reduce, identify and resolve conflict and therefore improve collaboration.

The reason for using the analysis for the requirements capture process is due to the need for group-ware technologies based on research on collaboration. There have been far too many group-ware technologies developed for the sake of the technology rather than due to a systematic understanding of collaboration[1]. This has lead to products which while being well implemented, do not aid the collaboration process. The analysis carried out in this report and the requirements built from it is based on a framework of collaboration developed by Johnson & Hourizi[2] and as such the requirements should be a good representation of how reduce the bottlenecks within the scenario presented and aid collaboration with a group-ware technology.

As well as these requirements gathered from the analysis, other requirements will be formed based on the findings within the literature review on collaboration.

The combination of these two sources of requirements will outline what the system should achieve and from these a design criteria should be developed to meet these requirements.

4.1.2 Existing Groupware solutions & literature

There are many current group-ware technologies mainly used to improve communication & coordination within groups. Technologies such as telephones and emails sit at one end of the spectrum of collaborative technologies while systems such as google wave, along with virtual meeting software such as adobe connect at the other. These systems aim to facilitate communication in geographically distributed teams.
The strengths and weaknesses of these technologies have been noted in both the literature review as well as in the analysis with regards to facilitating collaboration within these virtual teams. These will be taken into account when developing the requirements.

4.2 The Concept

The solution intends to improve communication and coordination within a collaborative group, these, as shown in the framework [2] are the medium used to identify, resolve and reduce conflict within a collaborative situation. This improved communication is specifically aimed at creating a shared understanding and common ground between team members which should intern lead to the improved collaboration desired [8, 10]. This will be achieved by allowing users to communicate synchronously with a high level of media richness, or as stated by Olson & Olson, communicate with short feedback loops and multiple streams of information [10]. These increased channels of communication and richness will be achieved by features such as free hand drawing or annotations and drag arrows, as other means of illustrating, alongside graphics and text. As well these increased channels increased synchronicity and tight feedback loops should also facilitate this shared understanding and common ground. This improvement should be more prominent in areas where the highlighted bottlenecks (3.3) occur, i.e. tightly coupled tasks such as shared design tasks across distributed team members.

4.3 Functional requirements

There are two types of user to consider when producing these requirements; the Developers and the Business analysts as highlighted in the scenario. While both have different roles they share certain characteristics and skills. As both communicate through the same means, the requirements of the system do not differ for each user.

The functional requirements have been drawn up in tabular form as shown in the sections below. The tables detail the requirement, the origin or reason of the requirement and what tasks within the collaborative scenario it should aid, i.e. where will it alleviate the bottlenecks detailed in the analysis.

As mentioned in the introduction to the requirements section the requirements have been derived from the analysis performed as well as the literature survey conducted. For example the requirement 7. “Provide tight instant feedback” comes from the analysis which concluded that parts of the resource structures did not offer “synchronous, tight feedback for effective communication within tightly coupled tasks” which lead to a bottleneck in communication, this requirement then aims to solve or reduce this bottleneck. As well as these analysis based requirements there are also the requirements based on the literature survey. For example, the literature survey stated that collaboration could be improved between distributed teams through supporting multiple channels of
communication. Requirement 6. “Facilitate communication through multiple channels” implements these improvements from the survey.
4.3.1 System Requirement

The system must provide the following:

<table>
<thead>
<tr>
<th>Affected task structure(s) **</th>
<th>Reasoning for Requirement *</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>As collaboration between group members entails more than one users of the system</td>
<td>1. Be capable of having multiple user</td>
</tr>
<tr>
<td>Debug &amp; Testing against design criteria (Figure 11)</td>
<td>Analysis showed a bottleneck within coordination as decisions made using IM aren’t saved for future reference (5b)</td>
<td>2. Provide a means for keeping (saving) the data generated</td>
</tr>
<tr>
<td>Debugging &amp; Testing, Problem solving (Figure 10)</td>
<td>The analysis showed a lack of informal, spontaneous communication between members (used to build common ground (2c)) especially within videoconferencing</td>
<td>3. System should support ad-hoc groups as well as including all members</td>
</tr>
<tr>
<td>Establishing design criteria as a group (Figure 16)</td>
<td>The analysis &amp; the literature determined that it was the collaboration of creative tasks which are currently not well supported by group-ware technologies(36).</td>
<td>4. The system must allow members to be creative and share their creative ideas</td>
</tr>
<tr>
<td>Many</td>
<td>Analysis along with literature review concluded that it was this tightly coupled work which is hardest to perform within distributed teams (3b)</td>
<td>5. Must support “tightly coupled” work</td>
</tr>
<tr>
<td>Many of the tasks in analysis were shown to be highly coupled</td>
<td>The literature review stated that the communication required to support collaborative work needs to be as rich as possible, especially in tightly coupled tasks (1)</td>
<td>6. Facilitate communication through multiple channels</td>
</tr>
<tr>
<td>design criteria as a group, shared understanding (Figure 17)</td>
<td>This aids tightly coupled task and builds a shared understanding/common ground which is not supported through the current resource structures as shown in the analysis (2b)</td>
<td>7. Provide tight instant feedback</td>
</tr>
<tr>
<td>All</td>
<td>Numerous studies have defined the need for this informal communication to build bonds between virtual team members</td>
<td>8. Must support informal communication</td>
</tr>
</tbody>
</table>

* () correspond to bottleneck items in Analysis conclusion
** () correspond to Task structures in Analysis

Table 1: System Requirements
### 4.3.2 User Requirements

All users of the system should be able to:

<table>
<thead>
<tr>
<th>Affected Task structure(s) **</th>
<th>Reasoning for Requirement *</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem solving, Testing &amp; debugging (Figure 10)</td>
<td>Analysis showed lack of informal, spontaneous communication between members used to build common ground within video conferencing (2c)</td>
<td>9. Communicate with the rest of the team, a subset of the team or an individual synchronously</td>
</tr>
<tr>
<td>Establishing a shared understanding (19)</td>
<td>This synchronicity has been detailed within the literature review as “rapid feedback” and the richness as “multiple channels” (Appendix 27)</td>
<td>10. Communicate synchronously with high levels of media richness</td>
</tr>
<tr>
<td>All</td>
<td>Analysis showed a bottleneck as IM don’t make topic of conversation clear (5a)</td>
<td>11. Create a new work-space for each topic</td>
</tr>
<tr>
<td>-</td>
<td>From previous successful systems</td>
<td>12. Drop files into the work-space</td>
</tr>
<tr>
<td>All</td>
<td>This should help facilitate effective communication in tightly coupled, distributed tasks as in (3b)</td>
<td>13. Edit the state of a work-space</td>
</tr>
<tr>
<td>All</td>
<td>The analysis stated time zone was an issue. This will allow all users to be aware of activities and tasks being performed even if these occurred while they were not at their desk due to time zone differences (4a)</td>
<td>14. Join a work-space at any time and view all past activity as well as the current state of the work-space</td>
</tr>
<tr>
<td>All</td>
<td>Aid coordination as the analysis showed a lack of permanent resource for reference (5b)</td>
<td>15. “Publish” the “Shared workspace” once finished so they are retained as a permanent resource &amp; record</td>
</tr>
</tbody>
</table>

* () correspond to bottleneck items in Analysis conclusion
** () correspond to Task structures in Analysis

Table 2: User Requirements
4.4 Non-Functional Requirements

There are also factors which should be considered in the design and implementation of the system but are not functional.

1. Have access control, prevent unauthorized access
2. Have user limitations, dependent on invitations
3. Retain all data shared and communicated for Audit trail, Record keeping and Monitoring user activity
4. Must work across countries
5. Allow users to Name “work-spaces”
6. Provide security against malicious, server side changes
7. Keep the informality and ease of use of IM
8. Server side be hostable on various platforms
9. System should be quick and efficient to use

4.5 Usability

Requirement 22 mentions an ease of use, this is to say we should consider usability. This leads to the following usability requirements

1. Use a consistent theme and structure throughout
2. Increase intuitiveness by familiarity of design

4.6 Summary

These requirements are created to facilitate improved collaboration within the scenario highlighted in the analysis. They do this by using the analysis as a prime source and as such are based around improving the communication and coordination within the collaboration. From these, a design specification can be detailed in order to create a prototype of the system for evaluation.
5 Design

After gathering the requirements in section 4 based on the analysis carried out in section 3, design criteria have been developed in order to meet the requirements. These design criteria form the design specification on which the system is based. Many groupware solutions have failed in the design section, due to design for designs sake or design for technologies’ sake rather than design based on the understanding of collaboration and collaborative work.

The design of the system must not only fulfill the requirements but must do so while maintaining intuitive and simple to use. If the system fails to achieve this it will the uptake of users will be limited. The design must simplify and aid group collaboration, not hinder due to unnecessary complexity. The design must fulfill the essence of the requirements in that it must allow users to communicate with increased levels of synchronicity and increased streams of information.

5.1 System Overview

The solution intends to improve shared understanding between team members in order to improve collaboration. This will be achieved by allowing users to communicate synchronously with multiple streams of information. This increased richness and synchronicity should lead to the desired improvement in collaboration through an increased level of shared understanding. These improvements will be enabled through features such as free hand drawing, annotations and drag arrows, as other means of illustrating, alongside graphics and text in order to achieve multiple stream. Along with the synchronicity and back and forward nature of an IM this should improve the shared understanding between group members. This may be more prominent in areas which have been highlighted as bottlenecks in the analysis, such as design tasks and lead to the better group performance.

The project is aimed at virtual, geographically dispersed teams. Collaboration becomes inherently more difficult when teams are not co-located and as such many groupware technologies are aimed at this type of team.

Along with the synchronicity and richness the project implements other features which aim to aid collaboration. The project implements the idea of “shared workspaces” these are used for each new “Thread” or topic. Each workspace has its own user privileges, allowing the creator of the workspace to determine which members have access to it. As well as this, each “shared-workspace” is hosted centrally, this is for three purposes: When a group member makes a contribution to the workspace it will be shared among all other users with access to the workspace. Users joining at any point can see the whole workspace from time “0” and thirdly this means all workspaces can be backed up as well as being stored for future reference and audit.

Once all contributions have been made to a workspace it can then “published”, this saves the workspace content as a web page which can then be published for future reference. In this respect, once published the workspace becomes similar to a Wiki in that it is a multi-user generated web page.
The system will be called “All Collab” and will be referred to as such from here on in.

5.2 Design considerations

This section and subsequent subsections describes the issues which were addressed before continuing to produce a complete design solution.

5.2.1 Assumptions and Dependencies

This project makes certain assumptions and dependencies about the Hardware and Software of the users machines as well as the characteristics of the End-User. The system is dependent on each user having a machine and that machine being connected to the Internet via a TCP/IP connection. This connection is used for the transfer of data between the SQL database and the user client. As well as this the client is dependent on a pointing device which must has a click function, commonly this is a mouse or touch-pad, however the project should work with devices such as touch screen inputs. This is used for the drawing and annotation aspects of the system.

As well as hardware dependencies there are also software dependencies. The main dependencies is that of the .NET framework (3.0). The project is to be coded in C# .NET as explained in section 5.3.1 and as such requires the .NET run-time being installed. Unfortunately at present the .NET run-time is only available for windows platforms. While the MONO project aims to break this dependency, without code manipulation there is no .NET run-time which can simply be installed on Linux or MAC platforms and there is an Operating System dependency. This requirement also introduces a hardware requirement in that to install the .NET 3.0 run-time a minimum of 96kb of RAM is required as well as a CPU equivalent to Pentium 400 MHz or quicker.

On top of these Hardware & Software dependencies, the project makes certain assumptions of the end-users. It is assumed that the user’s of the project are familiar with group-ware technologies and have a certain level of experience and expertise with such tools. With the project being aimed at business teams, it is a fair assumption that most user’s will have knowledge of business productivity applications.
5.2.2 Goals and Guidelines

This section details the goals and guidelines which run throughout the design of the system;

- Allow multiple, synchronous channels of communication
  - All communication should be synchronous
  - Allow multiple forms of communication

- Appear, Act and Work in a similar way to an IM
  - IMs act synchronously in a way which adds much value to virtual teams. The idea of this goal is to use the IM model as a guideline but to obtain a level of richness in the media that IMs do not.
  - This will also help the usability of the system as users will be familiar with the layout and the basic functionality of the project.
  - The requirements state the need for informal communication, IM allows this form of informal communication and so keeping the design similar should aid with this objective.

5.2.3 Development Methods

As the development is only of a prototype Rapid Application Development (RAD) will be adopted as the software development methodology. It will not be strictly RAD due to the lack of feedback from a business owner or user due to the requirements being gathered from the analysis rather than a requester. This methodology was adopted due to the fact that such an approach enables prototypes to be developed in minimal time. These prototypes can then be critiqued against the requirements and any amendments or changes made.

5.3 Architectural Strategies

This section details the design decisions made which affect the system at a high-level.

5.3.1 Language

Microsoft’s C# .net was chosen for the implementation of the project. This was chosen for several reasons, one of which was that C# is the most familiar language to the author of this report. This prevented wasted time learning a new language and meant more time was spent implementing the prototype. Although this was the main contributing factor to choosing C# there were other factors which make C# a suitable language for this project.

C# is a high level language with some very useful high level features. This allows the developer to spend more time on code which matter to the system rather than having to spend time garbage collecting and managing memory. C#
and the .NET platform provide a framework alongside the Visual Studio IDE for quick application development. With the .NET framework also providing the Common language Run-time (CLR) there is no need to consider the specific CPU which will run the system and also provides functionality for tasks such as memory management and exception handling.

The .NET framework also provides a very neat and seamless data link to a SQL server (Microsoft SQL Server). This is very useful as the project will utilize SQL servers for data storage. As well as this C# has a great support community. This is both inside and outside of Microsoft, not only with the MSDN help pages but numerous forums, blogs, walk-throughs etc. are produced by and for the community.

The main disadvantage of C# development is the platform dependence. In order for a .NET application to run on a machine the .NET framework must be installed, which is only compatible on Windows operating systems. However this can be overcome by the use of Novell’s free, open source MONO project. This provides the common language run-time which can be installed on many platforms (GNU/Linux, BSD, UNIX, Mac OS X, Solaris and Windows operating systems). The MONO project is still an ongoing process and is in no way perfected yet.

5.3.2 Client Server

The project could have been implemented in several different architecture models, the solution could have been implemented in peer-to-peer, client-queue-client or client-server architecture. However, the distributed nature of the project lends itself more towards a client-server architecture. This architecture is ideal for several of the requirements also. The idea of a hosted “shared-workspace” as detailed in the requirements is based around one central resource accessed by all rather than keeping copies on a client machine. Other requirements such as central retention of data for audit as well as allowing any user joining at any point can access the whole workspace and keeping the workspace as a permanent resource also make the client-server architecture the obvious choice.

5.3.3 Storage

The problem of how to store the rich media in a synchronous way is a major topic within the design. Most IM is implemented as peer-pier systems or for centrally hosted IM the architecture is designed to utilize SQL in order to maintain a central copy of everything. However the need to transfer, quickly and synchronously; text, graphics, sketches, files etc. between users whilst keeping the central version up to date is tricky. With traditional IM any additions are simply added to the top of the “stack”, where as in a shared workspace the whole “stack” can change with one user action meaning large data transfers. For this reason data shown in the workspace is split so that actions that affect the whole workspace such as sketches on the workspace are stored and loaded separately. As Rich Text, Images, files etc. are added in a traditional “push”
method these will simply be stored into a several tables. These tables will be split into a “Rich-Text” tables containing all the rich text and any embedded items while objects such as graphics added to the workspace or files transferred will be stored in an “Objects” table. The contents of these two tables will be pulled together and added into a dialog box. As well as this there will be another object, the “sketching pad”, which sits on top of the display box order for drawing to occur and is stored separately in its own “Sketch” table. This means that change to the sketch doesn’t require the whole workspace to be reloaded, only the sketch pad and vice versa.

Storage of “published” resources

Once all activity has finished on a topic or workspace the page can be published. This action then merges the contents of the several SQL Tables (Sketch, Text, Images etc.) into one page, which can then be saved in HTML format. This allows the workspace to be exported as a format which can then hosted and accessed, either on an internal intranet or over the Internet for future reference.

5.3.4 Transfer

In order to transfer the data ‘synchronously’ from the database (DB) to other clients and vice-versa the project will use “SQL Server Query Notifications”. Writes to DB tables will be triggered when a user changes the state of the workspace by either entering text, graphics, files or sketching on it etc.

For transferring these writes to other clients the Query Notifications will be used, these notifications are triggered when the result of a query changes. This can be implemented so that a client is alerted every time something new is added to a table. This logic is used for notifying the client application as to when it needs to retrieve new data from the DB, i.e. when a user has changed the state of the workspace. The client will then pull the latest data and apply it to the workspace.

For new users joining a workspace a snapshot of the tables within the DB will be pulled in order to fill the workspace’s open with the past communications.

5.4 System Architecture

The system, as outlined in the architectural strategies section (5.3.2), is based around the client server methodology. The client is designed to be installed on the users workstations with a central SQL Server database (server). The computation is all performed on the client side with the server side performing no computation, simply acting as a data store. This is shown if figure 20 below. This design means no server side component is required, simply a SQL Server is required.
Figure 20: Architectural design of the system

Due to this architecture the client is responsible for all features of the system. Due to this the client will be split into several sections; the MainGUI, the SQLTools section and the Logic section. This aims to decouple the client as much as possible, so that a change to the GUI needn’t change the logic section and vice-versa. This takes its inspiration from the Model View Controller MVC model widely adopted by the .NET community.

The intended behavior of users and their interactions with the client can be modeled using use cases. Figure 21 shows a use case of actions a user can choose to do when using the system. It shows that “Log-in” precedes all tasks. Also it shows that selecting a specific Work-Space precedes most of the actions. This is due to the fact that all actions bar creating a new Work-Space are Work-Space specific.
5.5 Policies and tactics

This section describes the design policies used which do not affect the system at a high level but do never the less affect the project.

The project will be developed using Microsoft Visual Studio 2008. This Integrated Development Environment (IDE) was selected for the same reasons that the language (C#) was chosen, the IDE allows quick development of programs with a very useful visual designer for the creation of graphical user interfaces (GUI's). Visual Studio also integrates very well with SQL Server 2005.

C# coding conventions, or best practices are to be implemented within the project. This includes features such as but not limited to; Naming conventions, File organization, Indentation, Comments & indentations/white space.
5.6 Detailed system design

This section details the system component’s design. These only go into specifics in a few classes in the application as much of the information is repeated within the implementation section of this report. These class descriptions and designs should show how the design was carried out for the system.

The client is made up of several sub components, these are;

5.6.1 CustomTPS

This class signifies a tab and therefore a workspace. Each instance of the class has its own instance of SqlWorkSpace as well. These are created when a user initiates a new workspace. They are docked within the a TabControl on the ContainerGUI. This class holds all the GUI functionality for the work-spaces. This includes the CustomPanel used for the annotations.

### User controls:

<table>
<thead>
<tr>
<th>Control Name</th>
<th>Control Type</th>
<th>Description/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>rchDisp</td>
<td>RichTextBox</td>
<td>The main display box that contains all of the work-space contents except the annotation</td>
</tr>
<tr>
<td>rchEnt</td>
<td>RichTextBox</td>
<td>The box used to enter text to be submitted to the work-space</td>
</tr>
<tr>
<td>pnlDisp</td>
<td>CustomPanel</td>
<td>The sketch panel which is used for the annotations on the work-space</td>
</tr>
<tr>
<td>btnSubmit</td>
<td>Button</td>
<td>The button used to “send” text entered in the entry box into the work-space</td>
</tr>
<tr>
<td>btnDraw</td>
<td>Button</td>
<td>Used to toggle on and off the drawing functionality of the workspace</td>
</tr>
<tr>
<td>btnArrow</td>
<td>Button</td>
<td>Used to toggle on and off the arrow drawing functionality</td>
</tr>
<tr>
<td>btnBox</td>
<td>Button</td>
<td>Used to toggle on and off the box drawing functionality</td>
</tr>
<tr>
<td>btnPaste</td>
<td>Button</td>
<td>For pasting objects into the work-space</td>
</tr>
<tr>
<td>btnUsers</td>
<td>Button</td>
<td>Launches the usercontrol dialog and allows users to be added to the workspace</td>
</tr>
</tbody>
</table>

### Class Variables: bool “draw”, int “workSpaceId”
### Methods:

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Parameters</th>
<th>Return</th>
<th>Description/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>btnSubmit_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Takes the text from the entry rich text box and passes it to the SqlWorkSpace class to be written to the work-space</td>
</tr>
<tr>
<td>btnDraw_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Brings panel to the front to allow drawing to occur</td>
</tr>
<tr>
<td>btnArrow_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Brings panel to front and sets to arrow mode</td>
</tr>
<tr>
<td>btnBox_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Brings panel to front and sets to box mode</td>
</tr>
<tr>
<td>btnPaste_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Takes item from clipboard and passes to SqlWorkSpace class to save to objects Table</td>
</tr>
<tr>
<td>btnUsers_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Opens the user selection dialog box and passes control to it</td>
</tr>
<tr>
<td>textToWorkSpace</td>
<td>string textToApply</td>
<td>-</td>
<td>Takes the text as a parameter and displays it in the richDisp box. Used when SqlWorkSpace passes text from DB to class</td>
</tr>
<tr>
<td>graphicToWorkSpace</td>
<td>image imageToApply</td>
<td>-</td>
<td>Takes the graphic from the parameter and displays in the work-space. Used when SqlWorkSpace passes graphics from DB to class</td>
</tr>
<tr>
<td>annotationToWorkSpace</td>
<td>image annotationToApply</td>
<td>-</td>
<td>Takes the annotation from the parameter and displays in the work-space. Used when SqlWorkSpace passes sketch-pad from DB to class</td>
</tr>
<tr>
<td>onDrawFinished</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Used when a user has annotated on the workspace to save the annotation to the DB to be transferred to others work-spaces</td>
</tr>
<tr>
<td>informationRecieved</td>
<td></td>
<td>-</td>
<td>Used to change colour of tab to indicate that a change to the work-space had been made</td>
</tr>
</tbody>
</table>

This class uses the SqlWorkSpace class to transfer data to and from the DB as needed. The work-space tab will change color if the state of the work-space has been changed. All user interactions are through this class and as such it is a very important part of the system.
5.6.2 SqlWorkSpace

The SqlWorkSpace class implements the SQL functions which are work-space specific. An instance is created per workspace, not only do they handle the transfer from database to client but they also implement the query notifications which act to tell the client new data has been written to the DB. As there is no GUI for this class there are no user controls, simply the methods used.

Class Variables: SqlConnection “sqlconn”, int “workSpaceId”

Methods:

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Parameters</th>
<th>Returns</th>
<th>Description/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>setListenerText</td>
<td></td>
<td></td>
<td>Sets the query notification for anything added to within the RICHTEXT, SKETCH or OBJECTS table for the workspace id</td>
</tr>
<tr>
<td>setListenerSketch</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>setListenerObject</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onChangeText</td>
<td></td>
<td></td>
<td>This is triggered when new text, a change to annotation or a new object is saved which applies to this workspace, it fetches the new text and passes to the CustomTPS instance with the same workspace id</td>
</tr>
<tr>
<td>onChangeSketch</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onChangeObject</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>writeText</td>
<td>string rtfent</td>
<td>bool</td>
<td>Takes the rich text as a string format and save it to the RICHTEXT table</td>
</tr>
<tr>
<td>writeObject</td>
<td>string table, object obj, char type</td>
<td></td>
<td>Takes an object and type, converts the object to its binary byte stream and save to either the sketch or objects table.</td>
</tr>
<tr>
<td>readFromTable</td>
<td>string commandText, List &lt;string[]&gt;</td>
<td></td>
<td>Reads from a table using a command text passed to it, returns data as a list of string array</td>
</tr>
<tr>
<td>readFromTable</td>
<td>string table, string columns, List &lt;string[]&gt;</td>
<td></td>
<td>Reads from a table using a given table name and column names to select, returns data as a list of string array</td>
</tr>
<tr>
<td>readFromTable</td>
<td>strings table, columns, conditionCol, condition, List &lt;string[]&gt;</td>
<td></td>
<td>Performs a select using the table, columns, cognitional column and condition it is passed returns data as a list of string array</td>
</tr>
</tbody>
</table>

This class is tied to the CustomTPS class and implements all the SQL functions required for it.
5.6.3 LogInGui

This is a class made to allow users to log into the system. It will enable the system to be password protected as well as supplying the username to the system to use within the work-space communications. For the functionality of this form user must already have registered a username and password with the system. The class will be made up of only a few methods and components.

### Usercontrols:

<table>
<thead>
<tr>
<th>Control Name</th>
<th>Control Type</th>
<th>Description/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>txtUName</td>
<td>Text entry box</td>
<td>A text box to enter the username in</td>
</tr>
<tr>
<td>txtPWord</td>
<td>Stared text entry box</td>
<td>A Text box to enter the password which stars(*) characters</td>
</tr>
<tr>
<td>btnOk</td>
<td>Button</td>
<td>Send username and password (hashed) to check authentication</td>
</tr>
<tr>
<td>btnCancel</td>
<td>Button</td>
<td>Close the program</td>
</tr>
</tbody>
</table>

### Class Variables: “uName” and “pWord”

### Methods:

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Parameters</th>
<th>Returns</th>
<th>Description/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>BtnCancel_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Closes the program</td>
</tr>
<tr>
<td>BtnOk_click</td>
<td>Object sender, EventArgs e</td>
<td>-</td>
<td>Set variables uName &amp; pWord as contents of text boxes, calls SendDetails</td>
</tr>
<tr>
<td>PasswordHasher</td>
<td>String pWord</td>
<td>String</td>
<td>Hashes password to send across network</td>
</tr>
<tr>
<td>SendDetails</td>
<td>String pWord, String uName</td>
<td>Bool</td>
<td>Calls PasswordHasher before sending details to DB to be checked, returns true if username &amp; password are valid</td>
</tr>
</tbody>
</table>

This class uses the SqlTools class to submit the details to the DB and retrieve the result. As well as this if the user details are correct it then opens the ContainerGui class. This class needs the SQL Server DB to be Online in order to function.
5.6.4 ContainerGui

This Class is the main GUI of the project. It will be the parent GUI on which all other controls are mounted. It contains all user controls and the work-spaces and as such all use of the system is through this GUI.

5.6.5 CustomPanel

This class will be the panel used to create the sketch pad. It will be contained within the CustomTPS. It captures the mouse down action to allow users to draw. It also provides a canvas for boxes and arrows to be drawn onto. It then passes this information to the CustomTPS which uses SqlWorkSpace to save the sketches to the Sketch table.

5.6.6 UserControlGUI

This will be a dialog box used to select which users can access a workspace. This will be used when first creating a new workspace as well as when editing users within a work-space. The data will be generated through the SqlClass class.

5.6.7 NameDialogGUI

This GUI will be a dialog box used to name work-spaces, this will be used when a new work-space is created. This is a very simple GUI.

5.6.8 WorkSpaceViewGUI

This form will provide a list of all work-spaces a user has access to and allow the user to select which work-spaces they wish to view. This will get the list of work-spaces user has access to using the SqlClass.

5.6.9 SqlClass

This is the for non work-space specific SQL functions. These include password checking, gathering lists of users in a workspace, gathering a list of work-spaces a user has access to, getting the next available workspace ID, and getting the Sqlconnection details from the config file.

5.6.10 Server Side

The details of the Database are shown in the implementation section. This is purely the SQL database as no server side components are required.
5.7 Summary

This design specification should detail a system which will fulfill the requirements set in the previous section. From these the system can be implemented, it follows that if the design specification fulfills the requirements so should the system. From this specification the system will be implemented.
6 Implementation

This chapter details the implementation of a prototype based on the design specification from the previous section. The prototype was developed in order to test and critique both the findings of the analysis carried out in section using the framework as well as the methods used to obtain the requirements. The Implementation, as designed was built as two separate sections, the Client and the Back End data storage.

6.1 Client

The Client was implemented in C# and the .NET framework. The client side was created to the design specifications developed. With the project being somewhat RAD there were changes and additions along the way. The workspace was implemented in the two separate sections of the sketch pane and the media display as detailed in the design. The client is where all the human interaction takes part, this is all the user sees and so the implementation of this part is very important.

6.1.1 Log In Page

The log-in page is launched when a user opens the program. This consists of two fields to enter the user-name and the password, as well as these there are two buttons, one to close the program, the other to attempt to log in.

The password field is starred (*) in order to protect passwords when being entered. To Log in the user enters their user-name and password before pressing “Log in”. This calls the SendDetails method which takes the user name and password as parameters. This method in turn hashes the password and sends the details to be checked against the database. There are catches around blank user names and blank password to limit the amount of rejections as well as catches on network or SQL errors.

If the user name and password are valid the main container GUI is launched allowing access to the system, the user-name is also passed to the . This is not a very complicated or unique part of the system, it is purely used to restrict access to none-users and for user credentials to be loaded, as such it will not be covered in any more detail. A screen-shot of this can be seen in Appendix C - Screen-shots.

6.1.2 Interface

In order to meet the requirement specifying that the solution should keep the ease of use of IM the layout of the client was similar in design to a traditional IM. This should also help with the adoption of the technology by the critical mass. The “shared Work-Spaces” are implemented as tabs within a controller with each tab being a new work-space as specified in the design. This was implemented as such in order for users to have several topics of “Work-Spaces” open at any one time.
6.1.3 Client GUI

The client GUI is the class which contains all components of the client. This is shown in figure 22, the class itself has minimal functionality. The menu bar is the only user controls implemented directly within the class. It also provides a dock for the tab container, this is the frame which contains the collection of work-spaces.

![Image of Client GUI](image)

**Figure 22: The Client GUI**

**Menu bar** Controls which are none work-space centric are implemented in the menu bar across the top of the GUI. The aim of this is to make the project as intuitive as possible. The bar was implemented with two drop downs; the “File” drop down and the “Help” drop down. The only work-space specific operation from the menu bar is the commands to close the current workspace system in use as part of a design task.

The “File” drop-down offers several functions. The first of which is the new tab function. This launches a name dialog box which contains a text entry field for the name of the new work-space as well as a button for cancel and a button to confirm the name, this is shown in Appendix C figure 38. If a valid name is
entered and confirmed, the UserControl GUI is launched, as shown in Appendix C figure 37. This is for the selection of which users the creator of the new work-space would like to have access to it. The users names appear in a check list box, if a user is checked this means they will have access to the workspace. There is an option for the user to select all users. Selecting users and pressing the “OK” button which calls the CreateNewTab method. This method creates the work-space locally as well as writing the new workspace and user access properties to the database.

Second on the “File” drop-down menu is the close current workspace option. This method closes the work-spaces which is currently open. This is implemented in such a way that a user closing a workspace doesn’t affect other users access to the work-space.

The third drop-down is named “Recover closed Work-Spaces”. This function loads a dialog box with all work-spaces which the logged in user has access to and adds them into a checked list box. This is shown in Appendix C figure 40. This way the user can select which work-spaces they want to view in the client and open any sessions they have closed. This function does this by selecting from the DB all work-spaces which the user is registered with.

The final option within the “File” menu is “Publish”. This function calls the method MergeAndSave, this merges takes the workspace state, and any content from the sketch pad. This then creates a single file which can be published as a permanent resource for future use.

6.1.4 Tabs

Each tab is representative of a work-space. These are self contained with all the controls for editing the work-space being located on the tab, these can be seen in figure 23 above as the grey area. Creation of a new tab or work-space is done through the menu bar. These tabs are implemented as custom tab pages, each as an instance of the CustomTPS class which inherit from System.Windows.Forms.TabPage. The custom tab allows for run-time creation of new tabs with the same layout as new work-spaces are created for new topics. These tabs sit within a “TabControl”, this is a container which simply holds the collection of tabs.

6.1.5 The Work-Space

The work-space was implemented in two parts, the sketch pad and the rich display. The two are tethered together for display purposes so the sketches stay in the same position relative to the display.

Sketch-Pad

The sketch pad is implemented as a custom frame, each an instance of the class MyPanel which inherits from System.Windows.Forms.Frame it builds on the traditional functionality of a Frame to support the scrolling nature of the
workspace along with the transparency required for the Rich Display to come through. The custom frame captures the mouse down event in order to draw the sketches the user requires. This is then transferred to the SQL DB and stored in the Sketch table. Other clients subsequently get this in their workspace as detailed later in section 6.2.

Rich Display

The display which contains all other media is simply implemented as a Rich Text Box with the addition of the capability of drag & drop for files and the ability to paste graphics onto it. This is simply done using standard event handlers.

As well as the Work-Space which is shared there are also controls to edit the workspace on each tab. These implement several key features within the system. First to note is the text entry box. This is implemented as a Rich Text Box which user enter text into, before pressing the “Submit” button. This button's click action method saves the text in the text box to the SQL database into the RichText table as well as adding it to the user's Rich Display.

Secondly, the button draw (shown in Figure 22, the pencil button) allows users to draw on the Sketch-Pad, it is a two state button and so the workspace is either in draw state or text state. While in draw state, drawing on the pad is implemented as in section 6.1.5. The arrow and box buttons are implemented in much the same way.

There is also an add user function, this is for if users who were excluded when the workspace was set up now require access. This launches the UserControl GUI which is detailed in section 6.1.3.

The last function on each workspace is a paste button. This can be used instead of “ctrl” + “V” command. Its simply pastes what is on the clipboard into the rich display.

6.1.6 SQLTools Namespace

There are two main SQL classes within the project. Each client session creates an instance of the class SQLTools. Each Work-Space open creates a instance of SQLWorkSpace.
6.1.7 SQL Server query notification

Both these processes implement SQL Server Query Notification. This process can inform a .NET application of when a query’s result has changed. This is done by using the “SqlDependency” object which creates an event handler which is triggered whenever the results change. The code below shows how this can be implemented to trigger an event when the Sketch table is modified, i.e. when a user modifies the sketch on the workspace. From here the “ChangeToSketch” method can pull the change to the sketch pad from the Sketch table.

```csharp
string cmd = "SELECT ImageData FROM Sketch WHERE idWorkSpace = @WorkSpaceId;"

SqlDependency depend = new SqlDependency(cmd);
SqlDependency.Start(connstring);
depend.OnChange += new OnChangeEventHandler(ChangeToSketch);
```

The same methodology is used to fire an event when an object has been added to the Object table and when the display box has content added.

**SQLClass**

This class provides SQL functionality for none workspace specific tasks. It is used for to build the connection string, submit and respond for the password authentication, convert user names to IDs, convert workspace name to ID and sets a listener for new work-spaces the user is invited to.

**SQLWorkspace**

This implements SQL functionality for workspace dependent tasks. An instance is created for each workspace and takes the workspace id and SQLConnect string from the SQLClass. The provides the functionality for listening for changes to the sketch pad from the Sketch table, listening changes to the workspace from the RichText table, writing content to the RichText table, writing to the Sketch table and saving object to the Object table.

6.2 Back End

Due to SQL Server query notification along with .NET’s SQL data handler there is no need for a separate server side process. These two aspects of .NET development along with SQL Server were two of the reasons they were chosen. Using the System.Data.SqlClient library .NET very smoothly integrates with SQL Server and the SqlDependency object within is great for handling refreshes.
6.2.1 SQL DB

As first mentioned in part 5.3.3 the data will be stored in a SQL database across several tables. The architecture and relationships of these tables are shown below in figure 23.

![Diagram of the database architecture](image)

Figure 23: Diagram of the database architecture

There are six tables in total, the WorkUser table being the mapping between Work-Spaces and Users. The mapping consists of the workspace IDs and the User IDs which belong to the work-spaces. This mapping is dependent on which users are invited to the workspace when it is formed or when modified. These IDs are then mapped to Work-Space & User names, the client uses these mappings to convert names to IDs. Each workspace contains one sketch pad. This is stored as a binary byte array which is generated from the image file which the sketch is saved as, it is replaced each time there is a change to the sketch. The Rich Text display is saved as continuous data and as such has a many to one relationship with the workspace. That it to say that the workspace contains many entries in RichText table. Rich Text is also saved as a binary byte array. The same relationship applies between the objects and workspace tables with
many objects to work-space.

6.3 Testing

Testing of the implementation was applied throughout the development in line with the RAD approach. As the implementation is purely of a prototype the testing was not exhaustive, but was used to check that the system was true to the design criteria and bug free. This testing was required in order to determine that the prototype fulfilled the design and so any evaluation of the prototype is evaluating the design built from the requirements as well as the system.

6.4 Summary

The prototype was implemented as detailed within this section. This implementation of a system to meet the requirements can now be evaluated to determine if it fulfills the requirements it was designed to. The evaluation can also determine if the requirements the system was designed upon lead to improved collaboration within the scenario proposed in the analysis.
7 Results & Evaluation

This project is an analysis and research project based around using a new collaborative framework to analyse group collaboration within a business context and gather requirements based on the findings. As such it is these findings, the methods used (the framework) to achieve them and there application to future business collaborations which will be evaluated within this section.

7.1 Evaluation with the Large Group Collaboration framework

The holistic framework used within analysis to gather the requirements will be used as the main form of evaluation. Within the framework “All Collab” fits within the resource structure. For the system to be judged a success, the first evaluation criteria is that it should, improve the communication and collaboration within the scenario detailed in the analysis. This will done by using the same scenario as within the analysis \([3,2,3]\) but with the use of “All Collab” within the resource structure to determine if there is any improvement. Secondly the evaluation should determine if any improvements in collaboration remain true for different scenarios and structures. That is to say that if there were changes to or different organizational, task or group structures the “All Collab” system still work to aid the flow of communication and cooperation.

7.1.1 Improvements within the previously analysed scenario

The scenario used to evaluate the system is almost exactly the same as the scenario used within the analysis. The only difference will come within the resource structure. The “All collab” system will be used a group-ware system. These are detailed under this resource section.

A couple of the tasks which encountered the most bottlenecks as detailed in the analysis were “Developing a set of design criteria” and “Decision making as a group”. The bottleneck in both of these task structures was found to be the task structure for achieving the goal “achieving a shared understanding between group members”.

This goal, was shown to be the underlying bottleneck in many tasks. This is highlighted in the task structures in section \([11]\) and \([17]\) where this shared understanding is required in order to progress with the task.

The requirements detailed from this analysis aimed to address this bottleneck identified. There were two criteria which helped achieve this shared understanding. The first was the synchronicity of a communication medium, which was shown to have an effect on the shared understanding achieved both in the analysis \([2]\) and in the literature \([10]\). The more synchronous the communication the easier it is to establish the shared understanding. This was addressed in requirement 7. The other factor shown to be important in building a shared understanding was channels of communication. The more channels the easier
it is to build the shared understanding as detailed in the literature [10]. These multiple channels were addressed in requirement 6.

Within the scenario the task of developing a set of design criteria, as with most collaborative design task was an example of “tightly coupled” work. This type of collaboration was found to be the hardest to support across distributed teams. The tasks structure in section [11] shows how this is broken down into sub tasks as shown also in the analysis. It was shown in this analysis that step five of the structure was an area where the communication and coordination was restricted. This step is task to “Enable a shared understanding of what is required & how to achieve it” and within this it is the task of “establishing a shared understanding” which, as identified as a task which requires high levels of communication. This task was further detailed within the analysis in figure [19] which details the task structure required to achieve this shared understanding. This was a task where communication was shown to be bottlenecked for given circumstances. The media of the communication is dependent on the group structure, if the group is distributed (as in analysed scenario) then the resources available must be used in order to facilitate the multiple channels of communication, and synchronicity or rapid feedback loops that face-to-face communication possesses in order to aid the communication. It is the communication across group-ware within the task structure (“Use group-ware technologies to increase levels of richness and synchronicity within the communication”) which is the underlying bottleneck that the “All collab” system aims to improve. The resource structure within the originally analysed scenario depicted three group-ware technologies to perform this task. They were videoconferencing, Instant Messaging and Wikis. To maintain a fairness between the comparisons rather than adding a resource the system will replace Instant messaging within the resource structure making it videoconferencing, Wikis and All Collab.

Figure 24: Synchronous channels of communication within the group-wares
Figure 24 shows the synchronous methods of communication which groupwares implement. The more channels of synchronous communication available the easier for shared understanding to be created and tightly coupled task to be performed by distributed teams. Figure 24a shows the synchronous channels within the technologies analysed in the original scenario. This shows that Wikis posses no synchronous communication channels, this is due to the nature of the technology as eluded to in section 3.2.4. Videoconferencing possesses four channels of synchronous communication, these are; Vision or Visibility, Speech or Audibility, Physical gestures & Referential. Instant messaging possesses only two channels; Text & Pictorial. Compare this to figure 24b which shows the synchronous channels of the resources within the evaluation scenario. This shows the channels implemented by “All Collab”, these include the Text and Pictorial as with Instant Messaging but also implements two more. These are graphical and referential. Graphical is implemented through the use of the freehand drawing & annotations as described in the implementation section 6. This can be used to share information synchronously which couldn’t be done with the previous resource structure. Referential is implemented by the workspaces and arrows. Referential communication is the understanding that an “item is present in an individual’s proximal life space and may be the topic of conversation” [33]. This communication is fulfilled through the visual aspect of videoconferencing where by a user can point at an object in the room [43]. The same applies in All collab, the user adds an arrow or annotation pointing to the object on the workspace in order to create a referential link. This is shown in the screen-shots in the appendix C, figure 43.

These new synchronous channels of communication established with the system as part of the resource structure will help to relieve the bottlenecks experienced within the tasks according to the framework as well as other readings [10, 33, 8]. The reduction of these bottlenecks should in turn lead to more effective communication and coordination as a mechanism to mitigate conflict within the collaboration.

7.1.2 Improvements across different scenarios

These improvements hold for the specific collaboration dynamics within the scenario analysed, however would these improvements in the resource structure hold if there were changes to the group or task structures? This is unfortunately outside the scope of this report and as such is a limitation of the findings depicted. Ideas of how this could be taken forward are presented within the further work section.

7.2 Evaluation by other means

As alluded to in the Literature review (section 2) there are many collaborative frameworks which have been developed. In order to evaluate the requirements generated from the analysis one of these other frameworks along with the Large group Collaboration framework [2] will be used. The second framework chosen
is Olson & Olson’s conclusions of sociotechnical issues [10]. As reviewed in the literature review this sets out four sociotechnical issues. This evaluation will concentrate on the “Common Ground” aspect of their work, and how to facilitate it across distributed teams is to be used.

The papers results include the table shown in Figure 25. This details the characteristics they concluded are important in medias for building common ground.

Figure 25: Olson & Olson’s table depicting the characteristics that contribute to negotiating common ground found in various media[10], with the addition of the three group-wares analysed earlier in the project as well as the “All Collab” system developed

The evaluation is performed by adding the group-wares used in the analysis to the table and detailing there characteristics along with the “All Collab” system. While the group-wares selected all failed to achieve the high levels of face-to-face communication this can be expected, this is purely used as a benchmark as it is well known that this is the best medium of communicate, with the most richness and cues. However, the results of the evaluation show no improvement on instant messaging systems. This, although disappointing, is understandable within the context of the evaluation when considering the only characteristics IM could improve on. These include; Copresence, which is not obtainable within distributed teams, Visibility, an area which the project was never aimed at and audibility. From these results Clark & Brennan would suggest no improvement in common ground would be gained from the system as opposed to using an IM.
Figure 26: Olson & Olson’s figure showing how technologies can support the characteristics of colocated synchronous interactions [10]. This has also had the three group-wares analysed earlier in the project as well as the “All Collab” system added to it.

The other area Olson & Olson’s paper can be used to evaluate group-ware technologies is with their ability to mimic the characteristics of co-located, traditional meetings.

The paper first list characteristics of co-located teams. Explanations of these characteristics can be found in Appendix A figure [27]. The tables shows how group-ware technologies help to recreate these in distributed teams at the time of writing the report (2000) (Column “Today”). The next column depicts Olson & Olson’s predictions of what technology can be expected to replicate well and poorly in the future within distributed teams.

The figure shows they believe that technology could be used to implement rapid feedback, establish multiple channels of communication, allow the identity of the contributor to be known as well as facilitating the flow of nuanced information. These factors, Olson & Olson believed could be supported to a good level by technology. Factors they feel would not be well supported by technology were characteristics such as informal ad-hoc chats, establishing joint references to objects, allowing users to control their participation and the focus, implement extra implicit cues as well as increasing spacial cohesion.

On top of this Video conferencing, Wikis, IM and “All Collab” have all been evaluated to determine which characteristics they support. It appears that although Video conferencing was an established technology when the paper was written it seems to fulfill the predictions made for the future. Video conferencing, as the tables shows, offers rapid feedback, multiple channels of information flow, indicates the identity of the contributor and includes nuanced information such as body language and tone of voice.

Wikis with there non-synchronous nature do not provide rapid feedback, however their rich content allows for multiple channels of information. However they only poorly support the rest of the characteristics in the table.

IMs do recreate this rapid feedback characteristic well, however this is all it supports in a strong manner, it does not have the richness to support multiple information channels. Support for the rest of the characteristics is either not available or poor.
“All Collab” brings the rapid feedback of IM along with the multiple information flows of Wikis as well as supporting more characteristics to replicate colocated meetings. Personal information is implemented with the user name identifying each user's contributions, while this is supported for text input it is rated as “poorly supported” as for graphics and sketches this is not supported. Nuanced information, is implemented to an extent in the text. Rich text can convey some of these nuances with bold, capitals and quotations. However this is nowhere near the flow of nuances found in co-located meetings. The informal “hall” time that Olson & Olson mention is found in the “handshaking” between users in a newly created work-space. This initial flow of information is more informal & personal and replicates the idea of “hall chat”. co-reference is established through the implementation of arrows and sketching. This allows users to share a reference to an object. If there is an object (e.g. a design picture) within the work-space which a user wants to establish a reference to part of, they can achieve this by drawing an arrow to the object. This is one of the criteria Olson & Olson predicted technology would not be able to replicate with a high degree of success, while the replication is not as good as co-located meetings this functionality performs the task of co-reference adequately. The idea of individual control is also implemented with a higher degree of success than predicted. In Appendix 27 individual control is described as “each individual can freely chose what to attend and change the focus of attention easily”. This is implemented through user Work-Space selection. If a user wishes not to participate in a topic then they simply close the work-space, this allows the selection of attendance as mentioned above, if they wish to then attend they can “resource” to be involved with the topic. Opening a new work-space can change the focus of attention and subsequently the topic. Additional Work-Spaces are inherent implicit cues. They provide information on other topics and around different areas while not being the main point of information flow. However this is by no means anywhere near as strong as in co-located teams. Spatially reference is another idea that is implemented through the Work-Space concept. The Work-Spaces can be referred to as if they were physical whiteboards within a co-located meeting.

These characteristics of the new system are very interesting. They replicate areas of co-located meetings which were not previously supported, or not supported as to such an extent by the other group-wares used in the comparison. These increased characteristics & replication of co-located meetings should serve to improve collaboration in tightly coupled work between distributed teams. This will have to be tested in further work.

7.3 Comparison between the two approaches

It was the Large group Collaboration framework [2] which was used to understand the factors within collaboration and therefore analyse bottlenecks within the scenario. Further work when the framework was developed suggested the application of the framework to virtual organizations but until now this has been purely used for collaboration between autonomous systems. While being a
good oversight of collaboration, due to its abstraction from a specific domain or context as well as its approach it does not offer specific suggestions for improvements nor metrics but does offer a good oversight of the contributing factors of collaboration to then base an analysis around.

The framework is designed to work in highly dynamic situations, however in a more static, business context it is arguable how flexible the dynamics of collaboration can be in order to facilitate conflict management. The solution for many of the problems faced within the scenario analysed would have been to simply change the group structure to co-locate the team in one space, however in business domains structures such as organizational and group are unfortunately often fixed.

Olson & Olson’s paper “Distance Matters” [10] is a very highly cited paper which looks at collaboration within a purely business domain and from a sociotechnical point of view. Their work, however is based on communication within collaboration and not collaboration as a whole. Figures 26 & 25 which are depicted during their work show this and are purely based on the communication channels within the group. This framework provides a tool for in depth analysis of the methods of communication within a collaboration but not the collaboration as a whole.

There is however an overlap between the two models. The Olson & Olson communication criteria is applied to group-ware technologies, which are a resource within the original framework. By using Olson & Olson’s criteria as an evaluation method for the group-ware alongside the large group collaboration framework the structures within the collaboration may be “bound”. These limitations may then mean that other structures need to be changed, for example the group structure may need to be changed to colocated teams for effective collaboration if the resources (according to Olson & Olson) don’t have suitable collaboration characteristics.

7.4 Summary

Overall this section shows that the proposed system has the potential to improve collaboration within a group situation. The results & evaluation allude to the success of the requirements and their impact on collaboration. This improvement was found to be especially prominent when the collaboration was between distributed group members and entailed tightly coupled work.

Two separate evaluation techniques were used to determine the effectiveness of the “All Collab” tool for the task of supporting collaboration. The evaluation using the framework developed by Johnson & Hourizi considered the tool in a specific static scenario. Within the scenario the tool was evaluated to determine if it had improved the communication and or coordination in order to mitigate conflict. The findings of this first evaluation suggest that the tool facilitates improved communication between group members as shown in sections 7.1 & 7.2. Within the framework this communication is a mechanism of reducing conflict. Conflicts have been shown to be detrimental to collaboration [9] therefore it makes sense that improved communication leads to improved collaboration.
The second evaluation was by means of Olson & Olson’s work on the characteristics of which facilitate common ground, which they argue is key in collaboration. The findings in this evaluation suggested that the characteristics of the tool were similar to Instant Messaging and as such their is no improvement on the original scenario when the tool was used. However, Olson & Olson’s work also depicted the characteristics of colocated interactions and their replication by technologies. The results from this measure show the tool supporting characteristics of colocated interactions which are not supported by the other technologies evaluated. This improvement in replicating colocated interactions leads to the reduction of the limiting factor introduce by the group being distributed.
8 Conclusion

The aim of this project was to determine a set of requirements for a group-ware tool to aid collaboration within a human group. These requirements were to be based on the previous work of Johnson & Hourizi [2] and the large group collaboration framework they developed. To do this it was decided that a scenario of a collaborative group would be developed in order to apply the framework to and analyse collaboration within human groups.

The analysis of the scenario was carried out and found several barriers to collaboration within human groups. Firstly it was observed that it is inherently harder to collaborate when not co-located with the other members of the group. Secondly it was identified that certain tasks were harder to collaborate on than others, these tasks were tasks which had to be worked on and completed as a whole by all group members. These have been described as “tightly coupled”. Finally it was noted that the bottlenecks within these tasks were due to one task structure, that of creating a “shared understanding” between group members.

The requirements were then created to overcome these barriers and improve collaboration. These were in turn used as a base for the design specification. The design aimed to meet the requirements set out and therefore overcome the barriers.

The prototype was then implemented from this design and evaluated. The evaluation was not based around the implementation of the prototype but around the requirements detailed and the design generated from them. There have however been limitations throughout the project.

8.1 Limitations

While the evaluation section detailed the findings and concluded that the requirements are likely to lead an improvement in collaboration there are some limitations to this. These limitations can be split into two broad categories; limitations of the study and limitations of the system.

8.1.1 Limitations of the study

The analysis carried out was based solely on one framework. This means the analysis and any products derived will only ever be as good as this framework and if there is any bias within this framework then the system will inherit this bias. The framework used for the analysis has not previously been used with actors as complex as humans. The use of the framework is based around identifying, avoiding and resolving conflict. The framework suggests that this conflict adds no value to the collaboration and therefore must be removed or mitigated as much as possible. This seems sound logic, especially when dealing with autonomous systems which simply follow rules, however it has been suggested that this conflict is an important part of group work and collaboration especially within a business context. “Growing evidence suggests that conflict may be beneficial to performance in groups and organizations” [21, pp. 9].
This removal of conflict can also be self affected within groups. This is due to the fact that humans do not like conflict and shy away from it within group situations, this can leave group members view points unvoiced as they do not wish to leave there comfort zones. This is the phenomenon known as Groupthink [23, 22], within groupthink the output from the group can be irrational, hasty and un-critiqued. Forcibly removing conflict from collaborations could have the same effects on group output as this phenomenon. In order to determine if this renders the framework unsuitable for use in business groups much more extensive work and testing would be required.

The analysis was carried out on a static scenario, which while thorough for that scenario, it is purely that, an analysis for that scenario. The findings may not fit other collaboration scenarios, meaning that while the system will be useful for this scenario it may not be for any other combinations of task, resource, organizational and group structures.

The results and evaluation of the system have been purely based on collaboration theory and frameworks. Without empirical testing it is hard to draw any solid conclusions about the usefulness of the system.

The fact that only three group-wares were analysed limit the extent to which these findings can be credited. For these findings to carry more weighting more group-wares would have to be analysed.

8.1.2 Limitations of the system

The system will always be limited in terms of Olson & Olson’s criteria as it doesn’t enable audibility or visibility. These are factors which they state of high important within group-wares.

There were a few limitations within the physical implementation of the prototype however the main point was the SQL Query notifications. This is due to the fact that when a notification is sent out to the users every client running with that listener connects and reads from the database at that time. This can lead to timeouts when there are many users within a workspace, dependent on the time out set and the size of the read requirements.

The .NET inferred platform dependency is also a limitation of the system. MONO was not utilised as anticipated within the design and so the client is tied to a windows operating system.

8.2 Further Work

The initial evaluation of the requirements are encouraging, however this section details some areas where the project can be furthered both in terms of the creation and evaluation of the requirements and the implementation of the tool.

8.2.1 Dynamic Scenarios

Results from the static prototype suggest that such a system would improve collaboration within the scenario laid out in the analysis section. In order to
test of these improvements hold across different collaborative situations this
evaluation should be broadened. First of all to static scenarios which contain
different organizational, group task and resource structures. If the improvements
hold through these static scenarios then the next step would be to implement a
computer modeling tool which allows the “All Collab” systems to be applied to
dynamic scenarios. This would also allow for the the application of collaboration
metrics to be applied against the outcomes of the model.

8.2.2 Implementation
The results from the evaluation of the system against the scenario used within
the analysis suggest a reduction in conflict and and thus an improvement in
collaboration. If these findings hold across the dynamic scenarios then a full
implementation of the system based on the prototype and the requirements
gathered should be fashioned. This full implementation could then be used to
perform a thorougher testing of the system and its application to collaboration.
This should involve real world testing with real world tasks to determine the
effectiveness of the implemented system within the business domain.

Physical Implementation
In terms of the taking the prototype further to a full production implementation
there are a few areas which would be of high priority

Infrastructure
The use of SQL query notifications has been some what a limiting factor. These
are not designed to be implemented as done within this project. Instead going
forward a web cache layer between the Database and the client would handle
data transfer much better. This is due to the issues of mass simultaneous
reads as detailed in section 8.1.2. The web cache could have been updated on
notification and then subsequently data pulled from there. This would improve
performance and reliability within the system.

8.2.3 Testing
As mentioned within the implementation section above the testing should be
carried out within real world, live business collaborations. To gather data from
this form of testing would require the use of collaboration metrics...

Metrics of collaboration One of the issues with evaluations of collabora-
tions has always been the metrics chosen to determine the success. Should these
simply consider the output of a group collaboration or should other factors such
as individual understandings, sustainability of the team, team understanding,
interactions between members group members etc. This choice is often deter-
mined by the length of the collaboration with a more short term collaboration
focusing more on the outputs of the group. Once the choice of metric has been decided they will need to be applied to the collaboration process.

8.3 Summary

The aim of the project was to develop a set of requirements for an effective groupware tool. These requirements have been created and have been accredited with the improvement of collaboration within the scenario they were developed for. However, the requirements generated have only been generated and tested within the scenario first developed and so the success of these requirements on a broader base is still to be determined and require more work to do so. In order to establish a definitive set of requirements this extra work would have to be carried out, this has been detailed in section 8.2. Other limitations of the findings have also been detailed in section 8.1. With these extra investigations on top of the initial work a definitive set of requirements which aid collaboration across many situations and domains could be achieved.
References


[34] Lipnack, J. and Stamps, J. 1997. Virtual Teams- Reaching across space, time and organizations with technology, New York: John Wiley & Son


[38] Scotts, D., Gehringer, E., Baheti, P. 2002, Exploring the Efficacy of Distributed Pair Programming , Extreme Programming and Agile Methods — XP/Agile Universe, Chicago, USA, August 4-7


88


## Appendix

### Appendix A: Olson & Olson

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
<th>Implications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rapid feedback</td>
<td>As interactions flow, feedback is as rapid as it can be</td>
<td>Quick corrections possible when there are noticed misunderstandings or disagreements</td>
</tr>
<tr>
<td>Multiple channels</td>
<td>Information among participants flows in many channels—voice, facial</td>
<td>There are many ways to convey a subtle or complex message; also provides</td>
</tr>
<tr>
<td></td>
<td>expression, gesture, body posture, and so on</td>
<td>redundancy</td>
</tr>
<tr>
<td>Personal information</td>
<td>The identity of contributor to conversation is usually known</td>
<td>The characteristics of the source can be taken into account</td>
</tr>
<tr>
<td>Unarmed information</td>
<td>The kind of information that flows is often analog or continuous, with many</td>
<td>Very small differences in meaning can be conveyed; information can easily be moderated</td>
</tr>
<tr>
<td></td>
<td>subtle dimensions (e.g., gestures)</td>
<td></td>
</tr>
<tr>
<td>Shared local context</td>
<td>Participants have a similar situation (time of day, local events)</td>
<td>A shared frame on the activities; allows for easy socializing as well as</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mutual understanding about what is on each other’s minds</td>
</tr>
<tr>
<td>Informal “hall” time</td>
<td>Impromptu interactions take place among subsets of participants on arrival</td>
<td>Opportunistic information exchanges take place, and important social bonding</td>
</tr>
<tr>
<td></td>
<td>and departure</td>
<td>occurs</td>
</tr>
<tr>
<td>Coreference</td>
<td>Ease of establishing joint reference to objects</td>
<td>Gaze and gesture can easily identify the referent of decicis terms</td>
</tr>
<tr>
<td>Individual control</td>
<td>Each participant can freely choose what to attend to and change the focus</td>
<td>Rich, flexible monitoring of how all of the participants are reacting to what is going on</td>
</tr>
<tr>
<td></td>
<td>of attention easily</td>
<td></td>
</tr>
<tr>
<td>Implicit cues</td>
<td>A variety of cues as to what is going on are available in the periphery</td>
<td>Natural operations of human attention provide access to important contextual information</td>
</tr>
<tr>
<td>Spatiality of reference</td>
<td>People and work objects are located in space</td>
<td>Both people and ideas can be referred to spatially; “air bands”</td>
</tr>
</tbody>
</table>

Figure 27: Olson & Olsons Key characteristics of collocated synchronous interactions.[10]

The above figure shows Olson & Olsons characteristics which are/may be/are expected to be supported by group-ware. They are all currently present in face-to-face meetings and are characteristics of this form of communication which can be supported through technology within distributed groups.
Appendix B: - Design use cases

Below are the use cases developed as part of the design section. These were created in order to step through what steps the user should have to take to perform an activity. These show what a user can do. In all the below cases the “user” can be either a developer or business analyst.

Figure 28: Add a new user to workspace

The diagram in figure 28 shows the steps taken to add a user into a workspace. Another user must take the action of finding selecting the work-space, followed by the add-user function, after this the user must select which user(s) to add before confirming.

Figure 29: Closing a Work-Space

Figure 29 shows the steps required to be taken for a user to close a Work-Space. The user selects the Work-Space followed by the File drop down from the menu bar and the close Work-Space command. This will only close the workspace for the user performing the act.
Figure 30: Draw on Work-Space

Users will be able to draw freehand on the workspace, the steps for a user to achieve this are detailed in Figure 30. The user selects the appropriate workspace, select the pencil from the workspace icons and then draws using the right hand mouse action button. When finished the user releases the mouse button to act as to take the pencil away from the workspace. This drawing is then displayed on all users workspaces.

Figure 31: Copying a graphic into the work-space

Figure 31 shows the stages required to paste a graphic into a workspace. After selecting the appropriate workspace the user must then paste the graphic in using the paste button, this will then appear on other users workspace.

Figure 32: Create a new work-space
In order to create a new work-space the user must follow the steps detailed in figure 32. The user selects the file drop down menu, followed by the create new work-space command. After this the user will be prompted to enter a name for the work-space. Once this has been done the user must select which other users they want to have access to the work-space. Once finished all users selected are alerted to this new work-space.

Figure 33: Steps to save a workspace in order to be published later

For a user to save the work-space ready to be published they will have to follow the steps shown in figure 33. The user must first select the work-space which is to be published, after selecting this the next step is to use the file drop drop from the menu and then publish. This will prompt the user for a file to output too using a save file dialog.

Figure 34: Adding text to the workspace

The last use case is for a user to add text to the workspace as shown in figure 34. The user once again selects the relevant work-space, after doing this the user enters the desired text into the text entry box, the next step will be to submit the text. After doing this the text will be in the work-space shared by everyone.
Appendix C - Screenshots

Figure 35: The LogInGUI used to enter the system

Figure 36: The ContainerGUI used as the parent and main form of the client
Figure 37: The dialog box used for users to control which other users can access the work-space

Figure 38: The dialog box which is used to name a work-space
Figure 39: The file drop-down menu

Figure 40: The dialog box used to select work-spaces to display
Figure 41: Early iteration of the system
Figure 42: The system with arrows implemented
Figure 43: The system in use as part of a design task
Figure 44: The system with both arrows and sketching