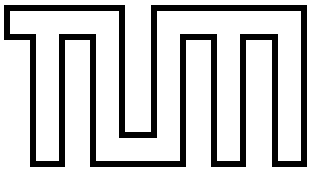


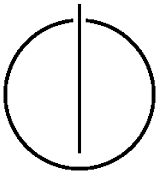
FAKULTÄT FÜR INFORMATIK

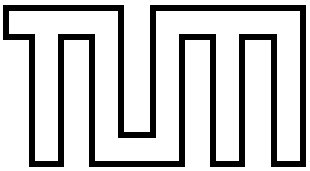
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Vertical social software for remote collaboration over video

Nishant Gupta





FAKULTÄT FÜR INFORMATIK

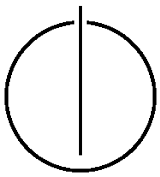
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Vertical social software for remote collaboration over video

Vertikale soziale Software für die verteilte Zusammenarbeit über Video

Author:	Nishant Gupta
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	M. Sc. Adrian Hernandez-Mendez
Date:	February 15, 2016



I assure the single handed composition of this master’s thesis, only supported by declared resources München, February 15th Nishant Gupta

Acknowledgment

I express my sincere gratitude to Prof. Dr. Florian Matthes for providing me this opportunity to write my master thesis in the chair Software Engineering for Business Information Systems (SEBIS).

I humbly acknowledge deep gratefulness towards Mr. Adrian Hernandez-Mendez for his inspiration, expert guidance, continuous encouragement and appreciation, which were very vital for successful completion of my master thesis. I would like to specially thank Mr. Klym Shumaiev for sharing his knowledge about the topic and helping in the evaluation of this project. I also take this opportunity to thank all other faculty and staff members of this chair for helping me by providing the necessary knowledge and resources for project completion.

I also acknowledge my batch-mates who helped in providing calm and congenial atmosphere. I wish to express appreciation to my friends for their valuable ideas and inputs towards this thesis and for all the discussions and communications, which always kept me motivated throughout this phase of time. I thank my parents for the wonderful support they have always provided me in all my endeavors.

Abstract

The work presented in this master thesis presents an efficient solution for remote collaboration over video. The implemented system is based on providing a generic framework for collaboration over video, which is then extended to several vertical social software solutions. Two major vertical social software scenarios covered in the implemented prototype include a *Real-time assistance module* and an *Offline support module*. Real-time assistance module provides a system integrating gesture and live video to support collaboration on physical tasks. This outlines benefits of using gestures and shared visual information for performing tasks, particularly in a highly mobile environment. A component, which does real-time hand segmentation and overlaying segmented hand's video on remote user's problem space, enables the prototype to facilitate communication with remote partner in a natural and intuitive manner. Offline support module ensures the possibility of achieving collaboration over video even if there is no Internet connection available in the task environment. Provision of drawing tools and an effective annotation model makes the offline support module of this prototype an upright system for remote assistance.

Results of several evaluation cycles done on the provided prototype suggest that out gesture over video communication system enhances the performance of tasks over traditional audio-video only systems. These results suggest that addition of gestural communication to remote video collaboration can be a very effective solution for handling interaction-intense activities. Furthermore, inclusion of offline support module makes this system more robust and applicable in highly remote areas with poor or no Internet connectivity.

Contents

List of Figures

List of Tables

1 Introduction

In today's world, people in most working environments tend to collaborate in groups to perform collective tasks. Communication methods like text messaging and voice call have been serving this purpose successfully for so long, and addition of face to face communication to perform collaborative tasks has proven to be a very important asset. Face-to-face communication plays a crucial role in the development and maintenance of these collaborations, and it is also critical for certain classes of workplace communication task such as project definition, initiation and planning, as well as maintenance [6]. However, with the kind of variety of applications requiring collaboration these days, face-to-face interaction alone may not always be either enough for effective communication or possible in certain circumstances. Using collaborative methodologies in application areas like telecommuting for mobile work, for example, from customer sites, or concurrent engineering, with designers, suppliers, and manufacturers increasingly coordinating over widely dispersed geographical locations [15], have become increasingly popular in recent times.

The goal of the research is to improve the quality standard of remote collaboration to match the level of co-located face-to-face interaction. Remote collaboration can be used to achieve physical 3D tasks (e.g. expert mechanic assisting worker to repair car), and solving 2D problems (e.g. assisting in solving puzzles on a paper). When people collaborate to complete physical tasks in face-to-face meetings, their actions, gestures and voice are involved with each other depending on the position of the objects and overall environment. Collaboration required for performing physical tasks remotely would also require gestures to be communicated apart from voice. Several kinds of gestures like pointing gestures or representational gestures may be used by people to increase the precision in their communication [2]. For example, an expert can assist the worker by using pointing gesture to point to particular object during conversation. Similarly, using representational gestures like hand movement or making hand shapes, used along with the speech can convey the instruction much more clearly. According to the findings in some prior works [7], performing tasks remotely is much time consuming and uncomfortable compared to co-located performed task because of the inability of helper to use gestures assistance. Therefore, it becomes very important to add the functionality of using gestures in remote video collaboration. Existing literature provide several methods for remote collaboration, but not much methods are available which use gestures efficiently and to a greater extent.

1.1 Motivation

With the enhancements in multimedia technologies, collaborating remotely for a common interest has served a lot of purposes. Remote collaboration has been enhanced by the use of video conversations which used to be text-only or voice-only conversations traditionally. Collaborative systems can further be improved if the vast interaction skills of users are taken into account. Performing physical tasks using remote collaboration can play vital role in various fields, including industry, education, and medicine. Due to increasing demand of support and expertise in these fields, technologies to support collaborative work remotely are very crucial to bring experts closer and together. Computer tools to support collaborative work should be designed in a manner that remote collaboration should not differ from the co-located users' collaboration. Computer supported collaborative work focuses on enhancing the peer interaction by using technology to share resources and interacting freely. It is important to integrate the interaction mechanics to the user interface so that users may take the advantage of their rich set of existing skills in a natural and intuitive way [13]. System's design should also focus on understanding remote collaborator's actions and the effect that they have on the actual workspace.

1.2 Research Approach

A dedicated and capable research approach is necessary for any organization to improve the effectiveness and efficiency of its research work. It is very important to acquire the knowledge in a manner that it is utilized in the productive application of information technology towards human organizations and their management. So, in order to acquire such knowledge, there is a dire need of using a sophisticated paradigm that seeks to define the effective ideas or practices. This thesis focuses on using such a paradigm as the research approach, which is called *design-science*. *Design Science* creates and evaluates IT artifacts intended to solve identified organizational problems [27].

A research framework should conceptually have two major inputs to it so as to give clear understanding and execution of research work as output. As depicted in figure 1.1, business needs are defined by the goals, tasks, problems, and opportunities as they are perceived by people within the organization. These business needs are then evaluated with respect to the organizational strategies, structure, culture, and existing business processes. Business needs are positioned relative to existing technology infrastructure, applications, communication architectures, and development capabilities [27]. Researcher perceives these business needs as part of the problem statement. Following sections explain the relevance and application of design-science paradigm in

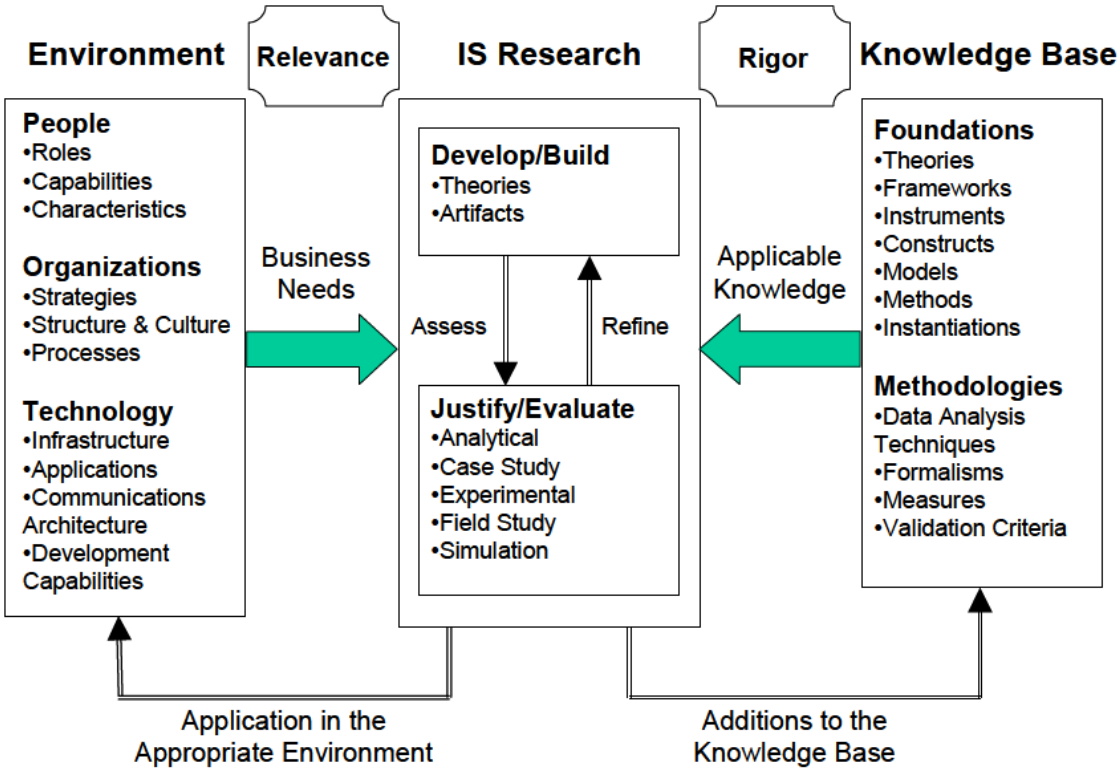


Figure 1.1: Research Framework based on Design-Science paradigm [27]

this thesis. Two major components of the paradigm namely - *Environment* and *Knowledge Base* are discussed below, explaining the approach used in this thesis.

1.2.1 Environment

First element that defines the problem space of the phenomena of interest is *environment*. For an information system research, environment should be composed of the following elements - *People*, *Business Organizations*, *Technology*. We define the environment setup by defining goals, tasks, problems, and opportunities to give rise to business needs to be perceived. For this, stakeholders and strategies are identified which can be affected by the remote collaboration over video.

Business Needs

The vertical solutions for remote collaboration over video can impact the business needs of an organization to a great extent. Business organizations are considering the

expansion of their support network by providing remote assistance to their customers. Same model also applies for providing assistance to on-field employees in order to improve the quality of on-field work delivered, keeping the time and cost to a minimum level. While it is unlikely to replace the complete physical space of the on field environment, but remote video collaboration does allow saving time with the current work and expanding the service area. Following are some of the benefits of having remote collaboration over video within an organization:

- **Cost Effective:** In any business organization, technicians providing support services are paid on the basis of the time they are doing work. Using video collaboration for remote assistance, a technician can easily take help from expert sitting remotely in other area. In this way, expert's travel time and hence the cost for the service providing organization is reduced by great margin.
- **Profit:** Normal business expenses like driving time, fuel consumption, etc. are some of the expenses that an organization factor in but with remote support over video, these can be reduced resulting in more overall profit.
- **Efficiency:** Remote collaboration over video also allows business organizations to work more efficiently. An employee can be involved in multiple tasks at the *loading bar time*. For instance, if he was working on a machine where he had to wait for loading time to get complete, via remote support, he could use that time to work on another person's equipment or do an in-office task making better use of the time.
- **Access To Resources:** On Field workers or service technicians cannot know what every single error message or problem situation means and how to fix it. In those cases , with remote support, they can have access to one of their own expert colleagues who can research the clients problems and have access to a range of applications.
- **Proactive Support to Customers:** If organizations have maintenance contracts with their clients, keeping an eye on their clients' systems can be made much easier and more effective with remote video collaboration. Again, this saves the time driving around all the time to support those maintenance contracts.

Requirements Analysis

This section provides the overview of requirements that should be taken into consideration in order to build prototype for collaboration over video.The requirements are considered based on the application scope of video collaboration in various user modes. The focus of this research has been to provide a prototype which could deliver vertical

solutions for performing physical tasks in highly mobile environment, besides serving the classical remote assistance in stable work environment. Therefore, besides having to provide some common standard features, requirements definition has been extended broadly into two vertical social scenarios:

1. **Online scenario (Real-time assistance):** For providing real time collaboration over video, it is a necessary pre-condition to have a good Internet connection at both ends of the communication. Requirements for this scenario are listed below:
    - Initiate / answer a video call
    - Access the device camera to capture live feed
    - Render and play video feed on browser window
    - Real time hand segmentation
    - Overlaying segmented hand video over original feed
2. **Offline scenario:** Considering the practical situations encountered in any on-field work, it can be noticed that availability of a good Internet connection can not be expected in many cases. This vertical social solution allows collaboration over video without having to connect on Internet instantaneously with remote user. A necessary pre-condition for this solution to be successful is having a mobile device with camera in order to record the working environment or the problem space. Requirements for this scenario are listed below:
    - Get recorded video from local file system
    - Video player feature to play/pause any video
    - Render and play recorded video on browser window
    - Drawing tools for annotating objects of video in browser
    - Create annotated video based on annotations drawn
    - Ability to perform CRUD (Create Read Update Delete) operations on the annotations

Based on the requirements described above, it can be realized that different user roles need to be defined. Since the collaboration over video for remote assistance requires one user to be assisted by an expert user, two major user roles have been defined:

1. **Non-expert Mode:** This user role corresponds to the user who needs assistance. Features like searching an expert, initiating a video call with expert are some of the rights that a non-expert user can use. For the offline scenario, this user records the video of work environment and shares the recorded video with expert

after he gets an access to Internet. A user in non-expert mode can see the annotated video and perform the tasks by taking help from the annotations. He is given only the right to read those annotations.

2. **Expert Mode:** This user role corresponds to a user who provides assistance to the other user in remote location. In this mode, he can only receive a video call, without having privilege to initiate a call. He can provide contextual assistance to other user in online scenario using his hand gestures. For offline scenario, he annotates the video shared by non-expert user to provide contextual help. Every user in expert mode has rights of performing CRUD operations on annotations.

1.2.2 Knowledge Base

The knowledge base provides the raw material using which the research is accomplished. As explained in figure 1.1, knowledge base comprises of two parts - *Foundations* and *Methodologies*. Following is the detailed explanation about how *Foundations* are realized during the course of this research work.

Foundations

The thesis work presented here is based on the prior information science research and their results, from which functional theories or models have been adopted during the build phase of the project. The models that form the foundation of this ongoing research are summarized in this section and is each technology has been described in detailed manner in the chapter 2. Foundations of this thesis correspond to two broad categories defined as follows:

Tools in Market

Most popular tools that lead the market currently in the field of collaboration over video are owned by huge companies like Apple, Microsoft, Google, etc. Tools like Skype, Google Hangout, Facetime is not unknown to masses. These tools are only restricted to voice and video chat, but don't provide more natural ways of collaborating remotely. A person trying to get help in day to day work can't use these tools because of requirement of Internet at all the time. In addition to that, getting assistance in performing physical tasks that require mobility can be very hard to achieve with these tools. So, in order to get more intuitive video collaboration, a lot of research has already been going on. This is explained in the following sub-section.

Research Work

Despite being very successful in market, classical video conferencing tools like ones described above are not sufficient in providing enhance features for remote collaboration over video. Research work being done in video collaboration can be summarized into following categories:

1. **Shared workspace for common material:** This is an extension to classical tools providing video calling facilities. The idea is to allow connected peers to use a common workspace where they can share materials like files or plain text. This workspace helps in sharing the common material in real-time and enhances the traditional video conferencing.

2. **Video annotation using tools:** This category improve the video collaboration by making use to tools like markers or pens to annotate live video feed. *Drawing Over Video Environment* [8] uses such a technique and is quite successful in performing real time video collaboration. It uses a shared workspace in which collaborators can use pen tool to annotate the video objects and provide assistance in real-time by projecting the annotated video at the user's environment.

3. **Video overlaid over another video:** This method of overlaying a video over another is being used very widely for providing remote assistance over video. This strategy can even be used for cases where real-time collaboration is not possible or required. *Collaboard* [19] uses such a technology in which presentation material used for teaching can be projected in the remote ares along with the teacher's gestural indications along with upper body is overlaid over the original presentation material. This is enough to give the intuition that teacher is physically present in the classroom and is standing in front of the projector presentation.

4. **Gestures Projection:** This strategy takes advantage of human gestural communication to make the remote video collaboration more natural. Peers can use their gestures to point at particular video objects, and this method allows these pointing gestures to segment from the unnecessary background and project over original video feed. This can be considered as an application of previously explained method of overlaying one video over the other and projecting them in combined form. *Videoarms* [23] is one of the most successful technologies which makes use of human arms projection over live video feed.

5. **Glass Panes for eye contact:** This method takes into account the necessity of eye contact during any collaboration. Remote collaboration lacks this capability to have eye contact between collaborators which is also disadvantageous compared to face-to-face communication. This is where glass panes technique used

for maintaining eye contact comes into action. Both collaborators have glass panes in front of them, which not only allows them to annotate video objects, but also gives intuition of direct eye contact. This has been successfully achieved in various versions of *ClearBoard* [14].

6. **Wearable Tools:** Wearable tools are gaining a lot of popularity in these days because of their small size. These equipments provide great utility for getting remote assistance while performing physical tasks that require high mobility. All the previously discussed technologies require highly sophisticated apparatus which is bulky and is therefore not at all suitable for the use cases where users need to move a lot. Wearable devices adds this ability to move while giving or receiving assistance using remote collaboration over video. *HandsInAir* [12] is one of the famous tools which falls in this category.

These were some of the categories in which existing technologies can be broadly divided. Some of the tools specific to each of the above mentioned methodologies are explained in chapter 2.

1.3 Research Questions

Based on the research approach discussed in the previous section, which utilizes the design-science paradigm, this section lists the research questions that are intended to be answered by this thesis work. The section will also explain the reasons and requirements that led to the formulation of these research questions. Design science paradigm lets us use well defined structure to present each step of this research work. These steps are described in the following section along with corresponding research question that needs to be addressed in each step.

1.3.1 Awareness of problem

People present in different geographic locations, have been using technologies that provide visual information to collaborate. Simple utilities of collaboration over video, such as casual video chatting with friends and professional video calls for managing official meetings, are being exploited very widely for decades now. No matter which tool available in market is being used by people, some level of inadequacy in communication is always felt. This lack of ability to collaborate in most natural manner is the reason that researchers all around the globe are working on improving the ways of remote collaboration. Performing any kind of physical tasks requires the user to move in the work environment. Getting assistance in such a mobile environment from a remotely located expert brings out the insufficiency in current collaboration supporting tools.

The importance of remote expert assistance for workers performing complex physical tasks like repairing equipment has been studied and very well documented [16] [18] [21].

Research Questions

Some research questions need to be answered while analyzing problem here:

RQ1. What are the different scenarios in which remote collaboration over video needs to be implemented? What are the characteristics of these scenarios?

Output

Based on this phase of problem formulation, a high-level proposal can be prepared as the first step towards implementing its solution. Remote collaboration over video is implemented in two broad categories namely:

- Online Mode for Real-time remote assistance
- Offline Mode

Short description of both these scenarios has been explained in previous section. Implementation details are given in the chapter 4.

1.3.2 Suggestion

This phase marks the beginning of design process of final product. State of the art work proves that developing the tools involving group of people is more challenging than the tools built for single user systems. Collaborative systems, involving multiple users at the same time, must be designed such that besides being useful, they are user friendly and usable enough to attract a critical mass of people to adopt the technology. Since adding multimedia technology to collaborative systems involve design complexities, it becomes challenging to incorporate the natural ways of interaction into the system. Existing research technologies like DOVE [8], ClearBoard [14] use a variety of concepts to improve remote collaboration over video. Suggestions for our problem's solution can be abductively drawn from these existing technologies.

Research Questions

Since this phase deals with taking suggestions and drawing solutions from existing technologies, it becomes important to identify literature that is relevant to the problem statement. Following research questions can be addressed:

RQ2. What are the existing tools/technologies for remote collaboration over video? What are their limitations with respect to the problem statement?

RQ3. How should the design of vertical social software for remote collaboration over video look like?

Output

Based on the suggestions taken from literature survey, high-level proposal from previous step is refined and a tentative tool design is prepared. This tentative design will be used to implement an artifact in next phase. As a part of tentative design, initial prototypes are created to be provided as a proof of concept. Prototype for our problem tries to prove three different concepts:

- Expert's hand segmentation from unwanted background
- Displaying one video overlapped over another video
- Annotating video objects

1.3.3 Development

The tentative design is further developed and implemented in this phase. High-level tools and programming languages are selected which mark the implementation of the artifact. Since the artifact is expected to be an application which could be utilized in highly mobile environments, realising the solution in the form of web application gives the best utility. Decisions to be made in this phase do not involve novelty beyond the state-of-practice for the given artifact. Novelty is majorly done in the design phase, and implementation phase just deals with the construction of the artifact. Artifact in this case is a web application whose development plan can encounter technology selection decisions. For example, decision of using an already existing service for implementing a video calling feature might be a good choice for rapid development, but its less know how can lead to problems in debugging or features extension of application.

Output

This phase results in a developed artifact which provides a solution to the problem statement defined in first step.

1.3.4 Evaluation

In the evaluation phase, the constructed artifact from Development phase is evaluated based on the criteria that are implicitly set in the proposal phase. Certain hypotheses are made about the behavior of the artifact and deviations from desired behavior, both quantitative and qualitative, are explained in this section. Functional specifications are made and implemented artifact is evaluated according to those functional specifications. Certain evaluation scenarios are shortlisted and characteristics of each scenario are defined.

Evaluation can also be done on the basis of purpose for which gestures or annotations are used.

Research Questions

In order to evaluate the provided solution, it is important to figure out the qualitative and quantitative attributes on which the evaluation will be based.

RQ4. What are the metrics for evaluation of implemented artifact for remote collaboration over video?

Output

Evaluation phase results in the performance results of the artifact that has been developed. The current study can be evaluated using specially designed scenarios that would evaluate the effects of communication media on physical task performance. Physical task selected here is joining blocks of *Lego* 1.2 to construct a pre-defined structure. Experts have the knowledge of structure that needs to be constructed, but workers are instructed about the steps to follow during the evaluation session. Workers are not given knowledge about the final expected structure to be built. We evaluate the implemented artifact by comparing performance in audio-video only collaboration, audio-video with gestures enabled collaboration, and side-by-side collaboration in which workers and helpers are physically co-present. We use same experimental setup for each participant and same worker participants take assistance from each helper so that effects of individual differences in skill and conversational style is neutralized.

Hypothesis

Qualitative and quantitative data is used to evaluate the solution and investigate the hypothesis. We predict that task performance will be best in side-by-side scenario because there is no bandwidth limitation and hence latency in communication. Experts can use most natural ways to collaboration. Performance in audio-video only condition would be worst because of restricted gestural communication. Performance in

Figure 1.2: Lego

audio-video with gestures enabled condition would be intermediate because of added advantage of using some gestures while collaborating. Performance of each condition is measured by the average time taken by experts to assist the workers and get the task done. Qualitative measure of performance will be the quality of Lego blocks construction in the end.

1.3.5 Conclusion

This section represents the end of the research effort where results are collectively displayed. Apart from the results being consolidated in this phase, complete knowledge gained throughout the thesis is summarized. All the facts that have been learned are categorized and some deviations of evaluation of results from expected results are tentatively reasoned.

Research Questions

In this section, major question that needs to be answered remains:

RQ5. Does the evaluation indicate improvement of provided solution over existing tools?

Here the word improvement needs to be defined more precisely and can be described based on various tables, graphs or charts. These would be able to indicate detailed comparison of implemented artifact with other existing solutions.

Output

Results accumulated in a distinctive manner along with the possible explanations are presented as the output of this phase.

Although this phase marks the end of current research effort, but it opens the gate for other researchers to capitalize on the loose ends that might be there. Since the knowledge can never be restricted to a particular research effort, it is important to recycle the knowledge gained from this thesis work to a new problem formulation. This work will certainly increase the awareness of the problems related to collaboration over video and a sincere effort done with the findings will definitely lead to a step further of achieving more intuitive and natural ways of collaborating remotely over video.

2 Related Work

The central idea of this chapter is to summarize different concepts and techniques used for providing remote assistance using video collaboration. In this section, various techniques that have been proposed or developed to contribute towards enhancements in remote video collaboration are reviewed. Concepts like development of shared workspace, gestural communication by projecting hand videos or their shadows, using tools to draw over video, etc. have taken research in collaborative work to a new level. The literature presented here covers a variety of methods used for remote collaboration, ranging from fixed desktop environment, having robust and bulky apparatus to highly mobile environment with better user friendly techniques. Gestural communication has been the focus of most of the technologies being developed. Every method uses a distinct design architecture to achieve gestural communication, covering from the use of marker tool based drawings on shared screen to rich support of hand gestures. First section of this paper presents the need of developing video collaboration techniques. Next section gives the system overview and critical analysis of 9 different methods which have contributed towards the enhancement of remote video collaboration over the years. In the last sections, these techniques are compared to each other, and challenges in development of such systems are discussed.

2.1 State of the art

State of the art work proves that developing the tools involving group of people is more challenging and encompasses specific roadblocks which are beyond the tools build for single user systems. Collaborative systems, involving multiple users at the same time, must be designed such that besides being useful, they are user friendly and usable enough to attract a critical mass of people to adopt the technology. Since adding multimedia technology to collaborative systems involve design complexities, it becomes challenging to incorporate the natural ways of interaction into the system. This section provides the summarized view of some of the developments done so far towards the enhancement of collaborative systems. Existing tools use a variety of concepts which have their own pros and cons. Systems described in this section rely on various techniques like shared workspace, pen/marker tool, glass sheets for eye contact, gestures projection, video overlaid over another video.

2.1.1 Drawing Over Video Environment (DOVE)

DOVE [8] aims at improving remote collaboration to perform physical tasks by proposing a method to use gestures over live video streams. Collaborators share a common virtual workspace by using camera of tablets or computers. The remote expert sees a live video feed of actual workspace on his tablet PC. Expert at his end can use available gesture recording techniques in the live video to provide help to the user. The same video with expert's gestures overlaid on it is fed simultaneously to the worker, which he sees it on his PC's display.

System Overview

DOVE uses dedicated network IP camera which is a server and connected to network independently. Computers on the network (including worker's PC) can get the video data feed from the camera by acting as its client. Remote expert receives this live feed by acting as a client to worker's computer. Expert expresses his gestures in the form of freehand drawing or drawing the predefined gestures using a pen tool over the video feed's display. A dedicated module for gesture recognition is present which recognizes the shape expert is trying to draw. DOVE uses image buffer to show the gestural data overlay over video streams and display them together. This is transmitted to worker's end and he sees this overlaid video on his display. Figure 2.1 depicts the system architecture of DOVE.

Critical Analysis

Expert can use predefined gestures (drawing circle, rectangle etc.), freehand drawing, and a combination of both which makes it user friendly and more realistic. DOVE has a great focus on automatic gesture recognition and is very successful in it. User friendliness such as use of automatic erasure of gestures is given more priority over manual erasure. According to the study conducted by writers, DOVE drawing tool performs a lot better compared to video only collaboration. Collaborators perform best when the gestures are removed automatically which resembles with natural hand gestures getting disappeared after the task is done. Despite all these features, DOVE still doesn't provide the gesture recognition component for handling the physical tasks which require information and interaction in 3D space.

2.1.2 ClearBoard

ClearBoard [14], with a key design metaphor "talking through and drawing on a transparent glass window", is one of the contributions which marked the beginning of highly interactive remote collaboration work. The idea is to emulate two users facing each

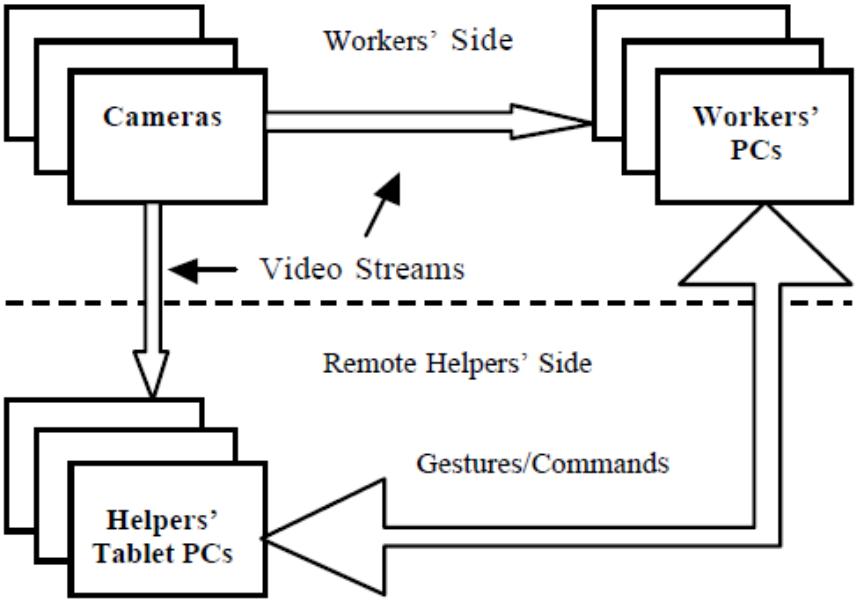


Figure 2.1: DOVE [8] system architecture for remote collaboration

other with a glass pane between them and allowing both of them to simultaneously draw on that glass pane. The basic technique is to use a common drawing surface called ClearBoard between two collaborators. It uses most natural ways of collaborating including eye contact and gestural interpretations which match quite closely to face-to-face communication. Since most of the existing collaborative tools have separate windows for displaying shared drawing/gestures and face of the remote user, ClearBoard tends to improve this by focusing on simultaneous eye contact along with using common whiteboard for sharing gestures.

System Overview

ClearBoard introduced a new concept of "Drafter-Mirror" [14] architecture for designing the system. Main components as shown in figure 2.2 include tilted screen, a video projector and a video camera. Screen is used by collaborators to draw symbols and write something. Video camera located above screen captures live drawings and user's image reflected by half-mirror as a continuous image. This image is continuously sent to other user over the network. Other participant, receiving the video has the option to draw marks over the video projection of received feed. Video camera again keeps on capturing user's and image drawings which is sent simultaneously to other participant.

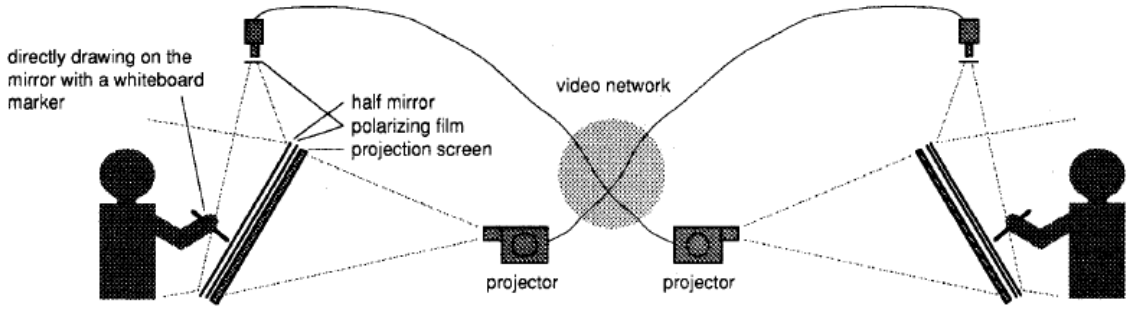


Figure 2.2: ClearBoard [14] system design for two collaborators

Critical Analysis

Design architecture used is very effective and contributes a lot towards the success of ClearBoard. Importance of eye contact along with shared gestures is necessary and the results analysis of ClearBoard proves it. Besides all the success of this architecture, there are various limitations that makes the use of ClearBoard almost impractical. Specialized equipment like projector and screen are very costly and unaffordable to every user. It is hard to setup the whole scenario because both the collaborators should have common orientation of drawing space, which can't be expected in everyday's usage. Also from the performance perspective, it is hard to attain good clarity of images recorded from the screen, because sharp focus can not be attained on all screen marks as well as user's face.

2.1.3 VideoDraw

VideoDraw [25] is a video based collaborative tool which provides a shared workspace named "virtual sketchbook" for collaborators. Participants can use the common workspace to create and see each others' drawings and thereby conveying their hand gestures. Since more than two collaborators can join a session, each collaborator sees a composite image of all the drawing marks made by the group. This means using VideoDraw for remote collaboration would be similar to physically sharing a sketchpad while interacting face-to-face.

System Overview

Collaborators share a common drawing surface in VideoDraw. Video display screen of each participant is used as the drawing surface. Users use dry erase ink markers similar to whiteboard markers to draw marks on their screen. Video cameras capture the display screen of monitors and transmit live video feed over the network to other user.

Collaborators have the option to draw, sense, erase, and gesture over the VideoDraw screens. System design of VideoDraw can be visualized as done in figure 2.3.

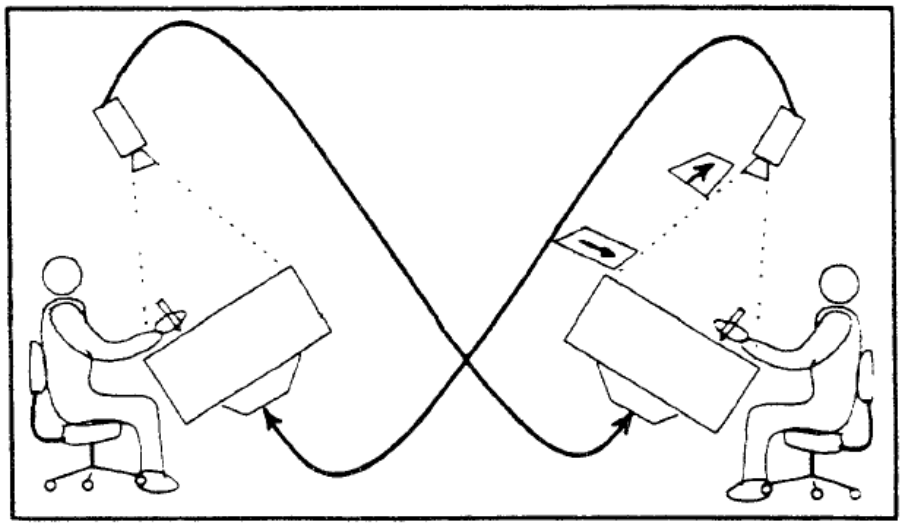


Figure 2.3: VideoDraw [25] system design for two collaborators

Critical Analysis

Basic goals of the VideoDraw system were accomplished successfully. Hand gestures were conveyed properly between collaborators. There were no unexpected and problematic time delays during the conversation. VideoDraw allowed multiple participants to have concurrent access to the shared drawing space. But participant can only erase or edit his own drawing.

2.1.4 VideoWhiteboard

VideoWhiteboard [24] is also a shared workspace based prototype tool which helps in collaboration by sharing the drawing remotely. Main property of VideoWhiteboard which distinguishes it from other shared workspace based tools is the use of shadows of gestures of remotely located collaborator. It allows collaborators to use a virtual whiteboard to create marks interactively and see shadows of each others' gestures in relation to those marks.

System Overview

System architecture design of VideoWhiteboard is inspired from VideoDraw [25] but provides slight differences from it as improvements. A VideoWhiteboard system as

depicted in the figure 2.4 provides a large shared area to be used as drawing space between remote sites. Users draw on the surface using whiteboard markers and video camera on the opposite side of screen captures the image and sends it to the video projector at the other side. The projected image is presented on the screen to the other user. Camera also sends the shadow of collaborator which means a composition of real and video marks, as well as shadows of remote participant's gestures is seen by the other participant. The problem of mirror image formation is corrected by video projector in front projection mode.

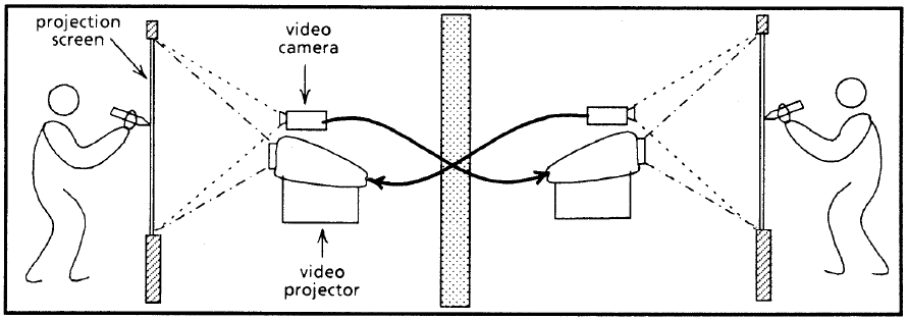


Figure 2.4: VideoWhiteboard [24] system design for two collaborators

Critical Analysis

VideoWhiteboard provides a large shared drawing space to its users where they can use their natural gestures and drawings to convey messages to each other. The gestures can be used over the drawings and hence remote collaborator can see those gestures with respect to corresponding drawing. One major advantage that VideoWhiteboard has over VideoDraw is that the camera and user are placed on opposite sides of screen. By doing this, user's head doesn't block the camera's view of drawing space which was a loophole in VideoDraw. Since the shadows of remote collaborator is superimposed directly over screen image having marks and sketches, this reduces the division of attention that occurs on other collaborative systems, where drawing surface is placed separately from the view of remote collaborator. Although shadows convey a lot of gestural communication between collaborators, they are still less rich than a full color video image. For example, they can not convey eye contact communication between users. Also the projected shadow images can cause interactional difficulties. For instance, gestures performed at some distance away from screen can be hard to interpret at the other end.

2.1.5 CollaBoard

CollaBoard [19] is also one of those techniques which work on shared whiteboard together with audio and video connection. CollaBoard believes in keeping the shared whiteboard editable at both locations. In addition to drawing markers, the system also provides live video which shows full upper bodies of remote collaborators.

System Overview

To achieve this and editable content on both sides, video and acquired content from drawing surface are transmitted separately to the other user. Image segmentation techniques are used to extract person from the background. The resulting segmented image is compressed and transmitted as video stream to other participant. This video stream is then overlaid atop the shared content of common drawing surface. Working principle of CollaBoard is shown in figure 2.5.

Critical Analysis

CollaBoard brings the upper body of remote collaborator over the local content, hence getting clear gesture instructions. But the hardware requirements for CollaBoard include special display screens and expensive commercially available pens and overlay kit. This adds to the limitation of the system as this might be the reason of infeasible requirements in several cases.

2.1.6 VideoArms

VideoArms [23] is an embodiment technique used in Mixed Presence Groupware (MPG) that helps in collaboration of people located in distributed teams. VideoArms focuses on recording arms gestures of remote users and redrawing them on large displays in actual wprkspace. Two groups which aim to collaborate should be running MPG application on a touch sensitive surface present locally with every group. Collaborators can see the arms of remote participants on the surface.

System Overview

VideoArms uses web cameras kept in front of display surface where collaborated video is displayed. Video segmentation techniques are used to extract arms of collaborators from the live video feed by removing unnecessary background. Video frames are transmitted to actual workspace so as to provide arms video overlay atop local video on the surface. So, this method follows a four step process. In first step, video segmentation is done by extracting the skin color matching pixels. Skin color is identified by

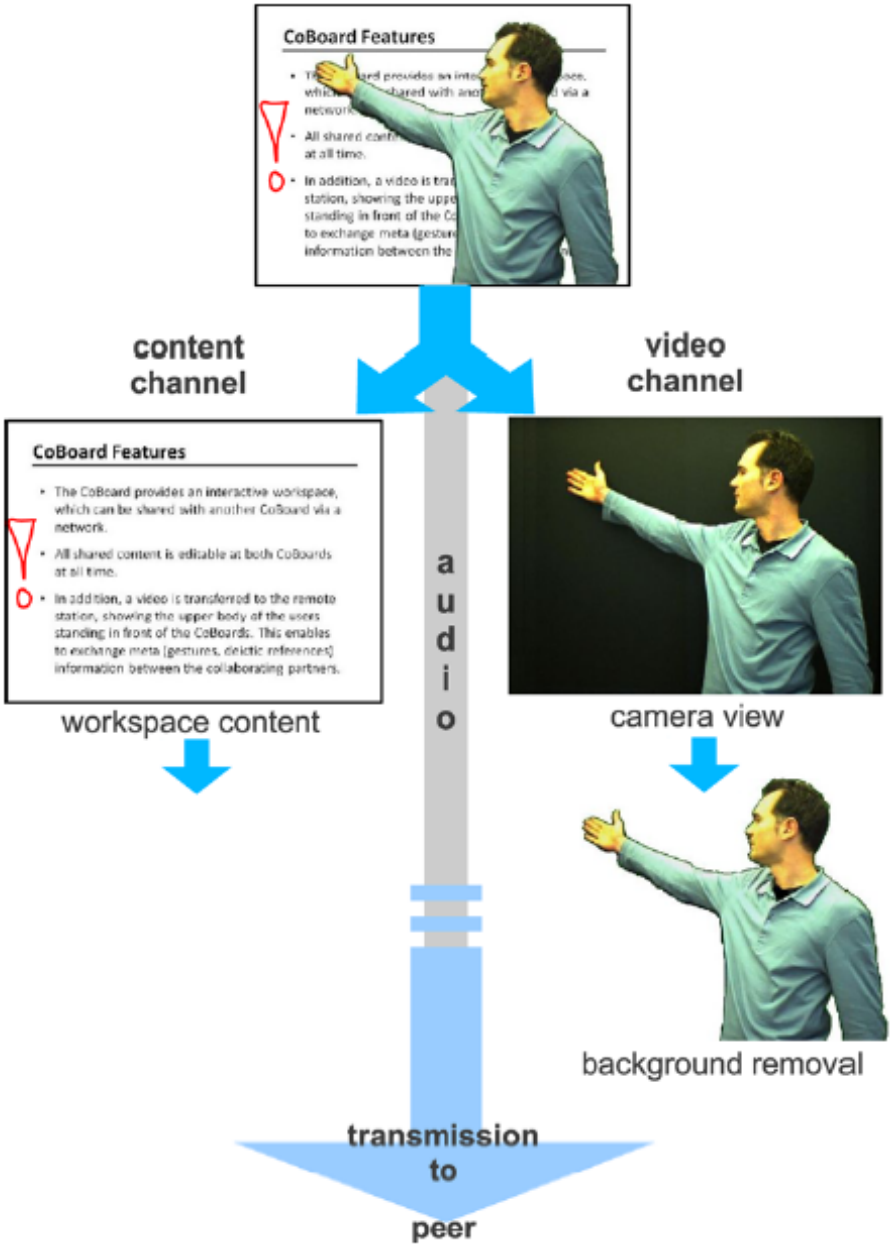


Figure 2.5: CollaBoard [19] working principle

comparing each pixel with 10 different skin sample pixels. This comparison is based on Mahalanobis distance between pixels. In the second step, the extracted mask is combined with original video image. Next, the combined video image is sent on the

network to the worker or client. Finally, the received image is displayed on the surface using standard raster graphics composing techniques. A screenshot of sample MPG session using VideoArms is shown in figure 2.6.

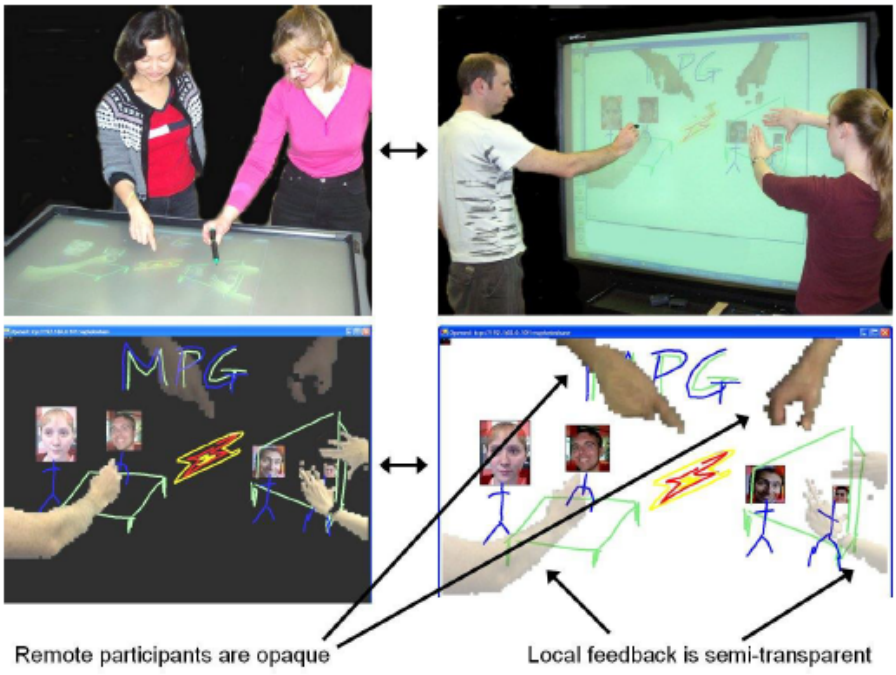


Figure 2.6: A sample VideoArms [23] session

Critical Analysis

Design of VideoArms technique makes it an appropriate choice for remote collaboration. It focuses on promoting more natural communication methods than other few techniques which focus only on using external tools like markers on shared workspace. Using variable transparency for co-located users' arms and remotely located users' arms makes it easier to track the context of the object every user is focusing at. Despite good features VideoArms offer, there are still some limitations which should be taken into account. Poor video segmentation technique results in poor and unclear image quality of arms on screen. Besides this, positioning camera in front of large surface also brings some practical problems. Large screen size used also becomes impractical in today's mobile world. Eye contact and body positioning which were already a part of ClearBoard [14], is not addressed at all in VideoArms.

2.1.7 Facetop

Facetop [9] is a tool designed for carrying out the distributed, synchronous pair work involving remote assistance. Video is integrated into the workspace as a semi-transparent, full screen overlay. There are two main components of Facetop - 1) capturing video, transmitting video over the network, and displaying received video on screen as transparent overlay. 2) Adaptable screen sharing component.

System Overview

The first part is to have a component that emulates face-to-face interaction with a common viewing workspace. Second part which distinguishes Facetop from other tools is the selection tool component that allows users to have screen rectangle sharing. Window sharing allows window to be shared, moved, resized, and occluded with other windows. Users can also share arbitrary portions, instead of just drawing predefined shapes like rectangles. Glass pane concept allows opaque drawing operations to be displayed as drawings on the underlying windows. This way multiple windows can be drawn and shared overlaid over one another, which saves the amount of space used in shared surface. Furthermore, drawing tools like underlining or marking can be used on the glass pane having text. Concept of glass pane can be seen in figure 2.7.

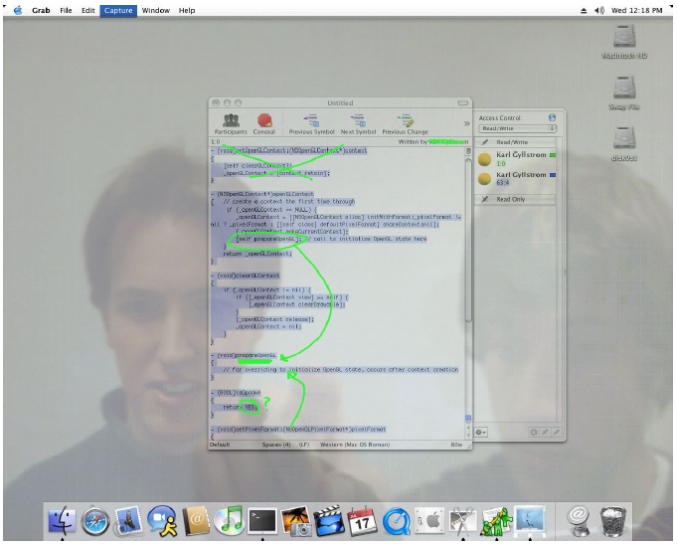


Figure 2.7: Glass Pane

Critical Analysis

Facetop has enhanced the concept of shared whiteboards into desktop, allowing collaborators to easily annotate their shared workspace, and anchor these annotations to movable screen objects. An advantage of Facetop over VideoArms [23] is that it allows full upper body to be shown and integrated in video collaboration whereas VideoArms just involves arms to be shared and projected remotely. It also allows multiple users to be present at one side of the conversation. This way Facetop contributes to more natural way of collaboration.

2.1.8 WACL

Wearable Active Camera with a Laser pointer [22] is a human interface device for supporting telecommunications. This is a setup defined specifically for performing physical tasks and getting the assistance from a remote user. A laser pointer is attached parallel to camera head of a small wearable camera. This setup serves two purposes - 1) gets the video images 2) instructs the wearer via laser pointer.

System Overview

For remote collaboration, the system setup is done differently for helper and worker. At worker's side, a wearable terminal having a PDA and a WACL is mounted on the shoulder of worker. This way worker can use his both hands even while watching displays. Video images captured by WACL are transmitted to wearable PC which controls the pan/tilt angles of WACL and on/off of the laser pointer. Figure 2.8 shows the system design at Worker's side and Instructor's side. Instructor has control over GUI which has functions to turn on/off the laser point, to turn on/off the stabilization, and to pan/tilt WACL. Instructor uses simple clicks to indicate a particular point on video image. WACL automatically adjusts so that indicated point is located at the center of video image. In this way by pointing and indicating, instructor helps the worker to navigate through scenes and achieve a physical task.

Critical Analysis

WACL, being a wearable system with stabilization added to it, is very beneficial tool for carrying out collaborative physical tasks remotely. This system can support various types of interactive remote instructions, like repair work, education, shopping, emergency life guard, etc. Despite the provision of such features, WACL can become difficult to use because of complex setup and expensive tools. Quality of stabilization done while recording video is also an issue because in rough and highly unstable work environment, stabilization can be tough to achieve.

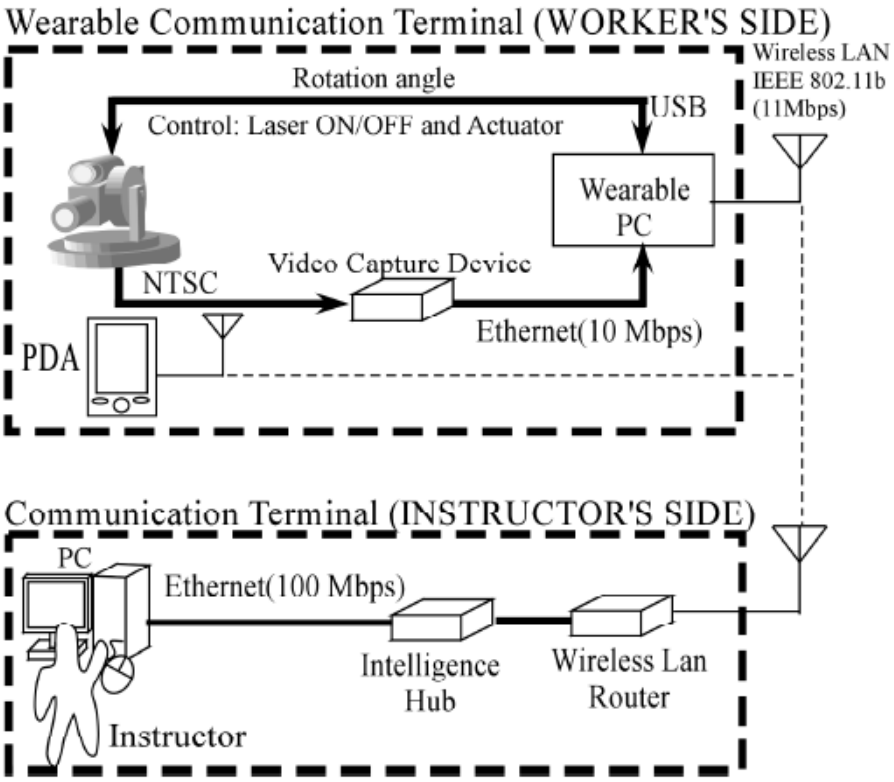


Figure 2.8: System Design of WACL

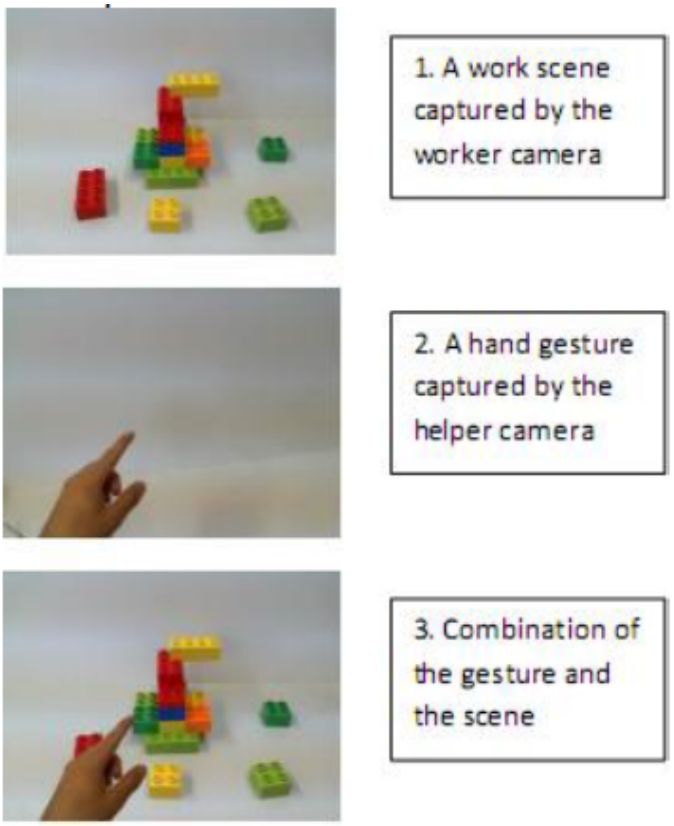
2.1.9 HandsInAir

HandsInAir [12] is a tool built for real-time collaboration to support real world scenarios like assisting a remote mobile user to perform a physical task. As in other collaboration methods, HandsInAir also works on the concept of shared visual space between collaborators. But since this tool aims to collaborate users in a mobile world, it does not use fixed desktop settings to perform collaboration.

System Overview

As shown in figure 2.9a, the helper wears a helmet mounted with a camera and a near-eye display. The helper performs the required hand gestures in the air, which is captured by the helmet's camera. The near-eye display is used to display actual scene videos for helper. Both the worker and helper nodes acts as video server and video client. An adaptive skin detector algorithm is implemented to extract helper's hands from video feed of helper. The extracted hand video is combined with worker's scene

(a) Helper recording hand gestures



(b) Combining hand gesture and workspace scene

video and transmitted to worker to achieve assistance. This is depicted in the figure 2.9b.

Critical Analysis

Technical implementation requirements in terms of hardware is minimum compared to other remote collaboration approaches which require large display screens or fixed video cameras for recording. This method allows helper to perform hand gestures in air without having to touch any tangible objects. One of the constraints in this approach is the use of plain background like a wall for recording hand gesture video, so that adaptive skin detector algorithm gives the best result of background removal.

2.2 Summary

VideoDraw, VideoWhiteboard and ClearBoard all provide a shared workspace, where each participant of the collaborating pair can draw. These systems use analog cameras to transfer the video feed having images of participants as well as other content of the workspace. Participants' images refer to their arms and bodies in VideoDraw and VideoWhiteboard, while their faces in ClearBoard. All these systems proved to be very effective, yet they suffer from some major limitations. Users could not manipulate each other's drawing marks which makes the shared space to be overloaded with information, thereby creating confusion. If the number of sites is increased, composite video image quality (containing the drawing marks and video feeds of all participants) encounters strong degradation. VideoWhiteboard although has been extended in Large

Tool	Shared Drawing	Gestural Communication	Facial Expression	Eye Contact	Supports Mobility
DOVE	✓				
ClearBoard	✓		✓	✓	
VideoDraw	✓		✓		
VideoWhiteboard	✓	Shadow Image	Shadow Image	Shadow Image	
CollaBoard	✓	Upper Body	✓		
VideoArms	✓	Arms			
Facetop	✓	Upper Body			
WACL		Laser Pointer			✓
HandsInAir	✓	Hands			✓

Table 2.1: Comparative analysis of existing technologies for collaboration over video

Interactive Display Surfaces (LIDS) [1] to support fully digital system for distributed PowerPoint presentations, the limitation of having shadows still poses a challenge of having considerably less detail than full fidelity images. VideoArms has some advantages over these systems, one of which is rich gestures support using arms display. In VideoArms, local participant also has option to see what remote participant is seeing because of his own embodiments being shown as semi-transparent feedback. Apart from using shared workspace, some techniques like WACL and HandsInAir proved very successful in performing physical tasks because of their wearable apparatus. These methods can be used in highly mobile environment, where users need to move to complete tasks. Tools like Facetop and CollaBoard focus on bringing upper body into communication which can improve the interaction quality compared to arms only methods. DOVE provides features like automatic gesture recognition in drawing and automatic erasure of markers, all of which enhance its capability and user acceptance.

2.3 Challenges

It becomes very challenging to add more features and multimedia technologies to collaborative systems. Making remote collaborative systems successful is complicated because users don't share same real workspace and can't get every gesture details. When collaborating face-to-face on physical tasks, people can readily combine speech and gesture because they share the same environment. Combining speech and gesture is more complicated in remote collaboration because of the need to reference external objects [20]. That's why, designing the system architecture for such tools is the biggest challenge.

Design should not only focus on adding useful features, but also on usable features, so that the system is accepted by users in market. Building tools with fixed, bulky and expensive components can contribute to an impractical design leading to failure in market. Similarly, usability of a tool depends on the application and the environment. So, environmental setup required for executing the tool should be minimally constrained. There are some techniques which force users to use Chroma Key environment. Such constraints can be a big question mark for users working in highly mobile environment. Limited bandwidth availability also adds to the constraints in incorporating multimedia into collaborative tools. Being able to have an eye contact takes a system closer to most natural way of interaction, but is not implemented in most of the systems because of the design complexities. Therefore, it can be concluded that a lot of efforts have been made to improve the quality of remote collaboration over video. Most of the tools developed have been proved successful in their respective application areas. But there is always a scope of improvement and researchers around the world are considering every opportunity to develop most natural ways to collaborating remotely over video.

3 Conceptual Design

3.1 System Overview

System is broadly divided into two components: backend of the server and the frontend. Backend, which is labeled as *Core Server* contains the core logic of the application. It is responsible for user management and database interaction. On the other hand, frontend represents the user interface and the logic to glue user interface with the backend. A high level overview of the system [3] is shown in figure 3.1

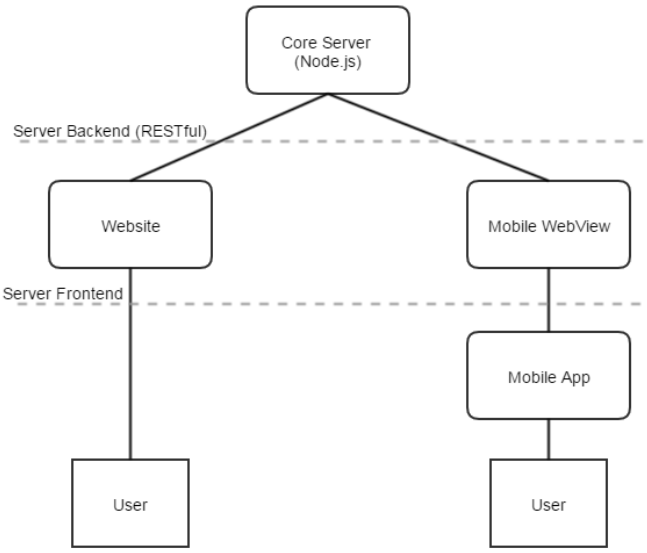


Figure 3.1: High Level System Overview

Following sections describe the back-end and front-end components in more detail.

3.1.1 Backend Overview

Backend is mainly exposed through a Representational State Transfer (REST) interface. This is directly connected to the express.js application router, which is used to manage all the incoming requests and serving them by sending them where they need to go. As depicted in the overall design of backend in the figure 3.2, MongoDB is being

used to store the data as raw data objects for users and other call related properties. All the technologies used in the backend have been discussed in the following section.

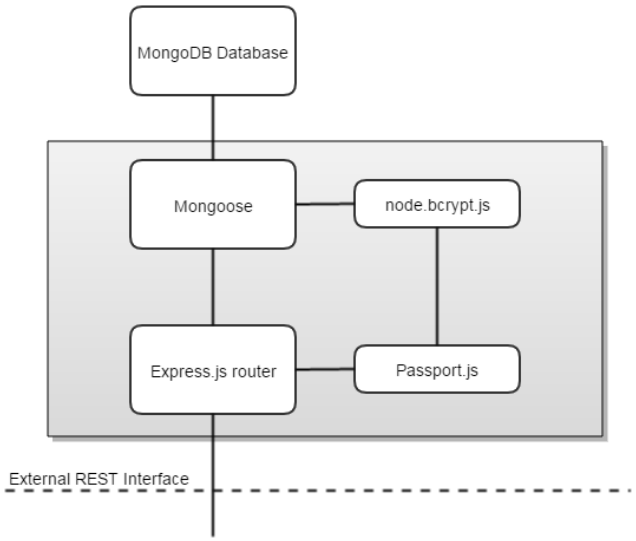


Figure 3.2: Backend Overview

Node.js

Node.js is a very powerful framework which uses an efficient event-driven, non-blocking I/O model. It is considered as a perfect choice for developing data-insensitive real-time applications that can run across distributed services. Following are some of the important features that make Node.js a perfect choice for web application development [26].

- **Asynchronous and Event Driven:** Since, all APIs of Node.js library are asynchronous, this essentially means a Node.js based server never waits for an Application Program Interface (API) to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

- **Single Threaded but Highly Scalable:** Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests.

- **No Buffering:** Node.js applications never buffer any data. These applications simply output the data in chunks.
- **Very Fast:** Node.js library is very fast in code execution.

Express.js

Express is a flexible Node.js framework that facilitates a rapid development of Node based web applications. Express is Node Package Manager (NPM) package for building web servers. Being able to provide following core features, it has become one of the most common packages used for building web applications with Node.js.

- Allows to set up middlewares to respond to Hypertext Transfer Protocol (HTTP) Requests.
- Defines a routing table which is used to perform different action based on HTTP Method and URL.
- Allows to dynamically render HTML Pages based on passing arguments to templates.

MongoDB

MongoDB is a document-oriented non-Structured Query Language (NoSQL) database that stores structured data in a from close to JavaScript Object Notation (JSON). MongoDB is by far the most common choice for Node.js applications. The reason behind MongoDB being suitable for Node.js applications are explained in the following sections. MongoDB is a good fit since it uses a JSON superset as its own storage engine and hence represents a document naturally. Unlike classic relational databases, it uses dynamic schema which makes integration of data easier. This is because schema can be changed on the go without affecting already existing objects. One of the biggest advantages is that MongoDB is schema-free, which means there is no schema migrations. The only schema that's going to be defined is in the code [28].

- **Databases for Node.js:** Node.js being written in JavaScript, supports JSON in a very efficient manner. Manipulating JSON data is convenient and since clients typically use JSON data, it enhances the advantages of using a database that works closely with JSON data. On the back end, Structured Query Language (SQL) databases are not a great fit for Node.js. Object-Relational Mappers (ORMs) (which are the tools that connect your code to a relational database) and other tools out there are really young, and not a lot of work is currently being done to develop them [5].

- **Why MongoDB?** MongoDB has a number of advantages over a traditional SQL database. Instead of storing data in a series of rows and tables like in a relational database, MongoDB uses *collections* and *documents*. One of the main advantages of documents over traditional SQL rows is that documents contain way more information. In this application, data corresponding to a user is stored in database. If this user data was to be represented in SQL, three different tables would have to be used, as shown here:

Table 3.1: SQL Tables for users' data

USERS			
first_name	last_name	username	email_id
Andrew	Clark	a.clark	a.clark@zzz.com
Tim	Smith	t.smith	t.smith@yyy.com
EXPERTISE			
expertise_id		name	
101		Architecture	
102		Construction	
103		Hardware	
USER_EXPERTISE			
expertise_id		username	
102		a.clark	
103		t.smith	
101		a.clark	

In order to find out the expertise of a user, tables of USERS and EXPERTISE are joined to give USER_EXPERTISE table. In MongoDB, this is much more simpler. A document for user details would look like the following:

```
{
"_id" : ObjectId("12584dfg15vd8"),
"first_name" : "Andrew",
"last_name" : "Clark",
"username" : "a.clark",
"email" : "a.clark@zzz.com",
"expertise" : ["construction", "architecture"]
}
```

In MongoDB this would be the only item in the database to represent the same data. The goal in a MongoDB database is to make the data as easy for the appli-

cation to use as possible. That's why, finding the expertise of a user is much more easy in MongoDB. Querying for a user would just return the expertise along with other information. Advantage not only lies in the way it makes querying easier, but also much faster under the hood. The format in which MongoDB stores data is called *Binary JSON (BSON)*. BSON is essentially just JSON stored as binary with few differences. It creates an ObjectId which is a datatype used to identify documents uniquely in Mongo. Using BSON results in following advantages over tables/rows [5]

- Documents map naturally to objects in programming languages.
- Embedded documents and arrays prevent the need for joins.
- Polymorphism is much easier since there is no schema.

There are three parts to the hierarchy of data in MongoDB:

1. **Database:** There is usually one database per app, but there can be several per server.
2. **Collection:** This is a group of similar pieces of data. It is usually a noun and is analogous to a table in SQL.
3. **Document:** This is a single item in the database, such as an individual user record. If a collection were a class, a document would be an object.

Models with Mongoose

Mongoose [17] is a Node.js library that provides MongoDB object mapping similar to *Object Relational Mapping* (ORM) with a familiar interface within Node.js. As the complexity of data structure being used increases, it becomes difficult to use MongoDB queries directly. This is when Mongoose comes into action. With Mongoose, objects can be created to represent the data and call object methods instead of manually querying database. This acts as a glue that allows to interface with a MongoDB instance from the node application. Mongoose provides an abstraction from the details of database organization by creating Schema and instantiating objects based on defined Schema. This is similar to creating classes in an object oriented programming language and creating instances of these classes. Furthermore, it provides simple predefined functions for doing operations on Schema created in MongoDB. Hence, it adds a structured, object-oriented way to interface with the underlying database.

Passport.js

Passport [10] is authentication middleware for Node.js. Passport can fit into middle of any application stack and handle all authentication issues. This ensures that a user

is logged in only if he tries to access a route that has been defined for successful authentication. If user accesses a route that he is not permitted to use, he is redirected to login page again. Furthermore, Passport.js also takes care of session handling by working in conjunction with the session support from express.

bcrypt

This is an encryption module that is used to hash the passwords based on bcrypt algorithm, in order to have secure password storage in the database. Hashing is a good technique of encrypting confidential data and saving it securely in the database. In this project, bcrypt is used for hashing whose output is then saved into MongoDB using Mongoose. Apart from using this module to encrypt password during creation of user account, it is also used to verify that a username/password combination was correct.

Web Real-Time Communication (WebRTC)

RTC stands for Real-Time Communication and WebRTC provides peer-to-peer connection between browsers. WebRTC offers web application developers the ability to write rich, real-time multimedia applications on the web, without requiring plugins, downloads or installs. It's purpose is to help build a strong Real-Time Communication (RTC) platform that works across multiple web browsers, across multiple platforms. Following is the high level overview of the architecture followed in a WebRTC application. As

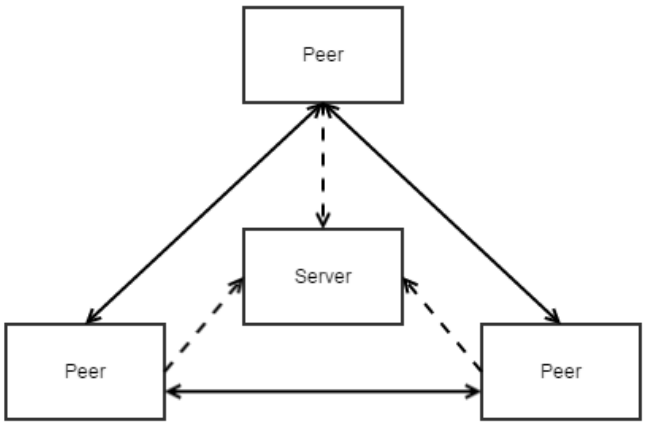


Figure 3.3: WebRTC High Level Architecture

shown in figure 3.3, a server component is required for initial connections and signal-

ing WebSocket connections. Sequence diagram shown in figure 3.4 depicts the manner in which signaling server plays its role in establishing a peer-to-peer connection. In the

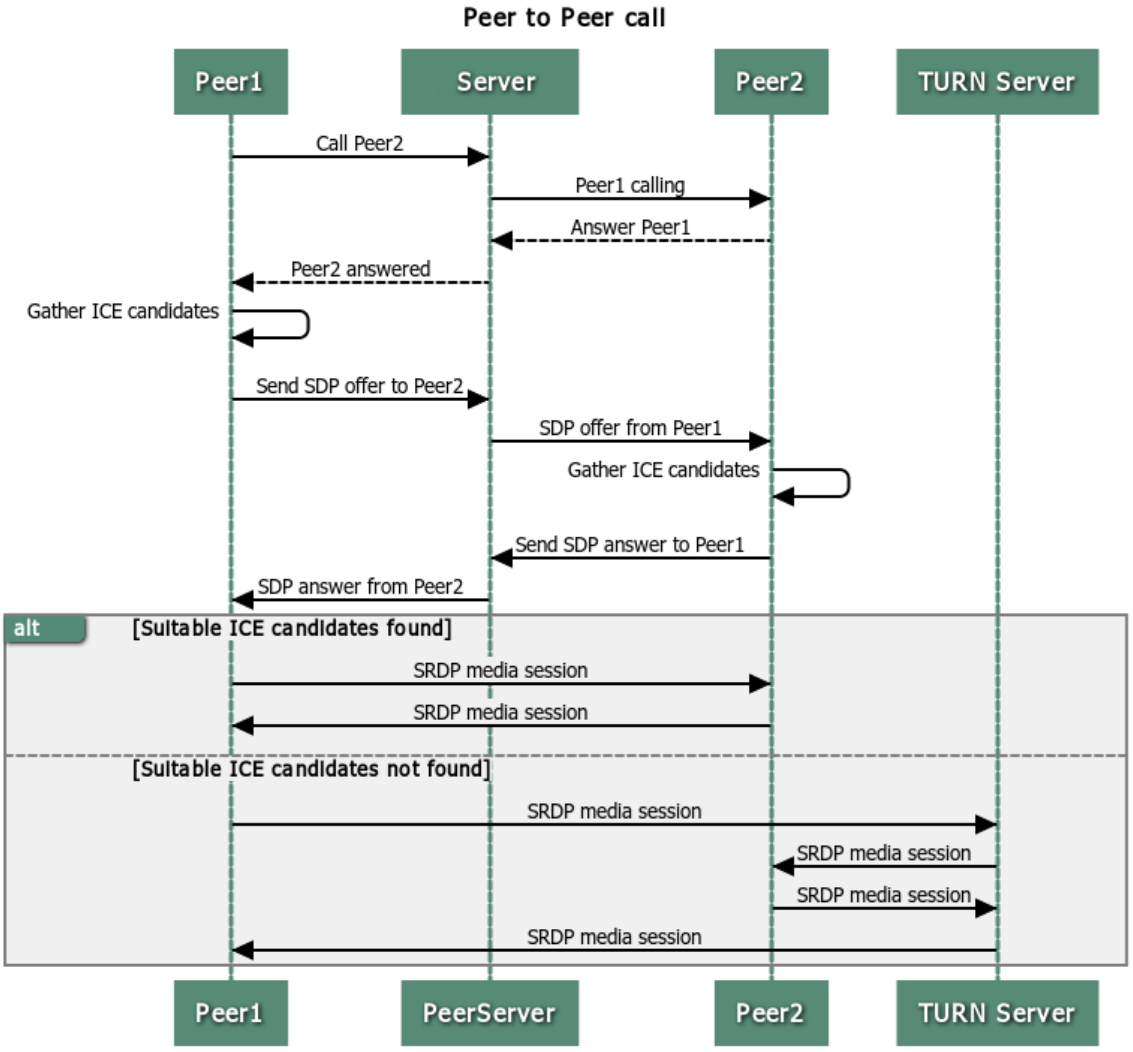


Figure 3.4: P2P Sequence Diagram

sequence diagram, it can be easily understood how *Peer1* establishes connection with *Peer2*, through the application server in the middle.

1. *Peer1* initiates call with *Peer2* through the application server by invoking a RESTful method POST.
2. Application server notifies Peer2 through a push notification that Peer1 is calling

him. The server may use WebSockets to send a notification.

3. Peer2 responds to the push notification by accepting it.

4. The server redirects Peer2's response to Peer1.

5. After receiving response from Peer2, Peer1 starts the Interactive Connectivity Establishment (ICE) candidates gathering process.

6. After gathering a set of ICE (Interactive Connectivity Establishment) candidates, Peer1 prepares an Session Description Packet (SDP) offer, which includes the ICE candidates and some additional information (like supported video/audio codecs, etc.). Peer1 then sends this offer to Pee2, via the signaling server.

7. The server redirects Peer1's offer to Peer2.

8. Peer2 gathers its own ICE candidates, prepares an SDP answer and sends it back to Peer1, via server.

9. Server redirects Peer2's response to Peer1.

10. Both peers try to establish p2p connection through the ICE candidates they already have.

If peers are not able to establish p2p connection using the ICE candidates, it is very likely that they both are present behind symmetric NATs. In this case, if we have provided a *Traversal Using Relay NAT (TURN)* server, the video/audio connection will be relayed through it, otherwise they won't be able to initiate connection between each other. Interactive Connectivity Establishment (ICE) is a framework that allows WebRTC to overcome the complexities of real-world networking. ICE helps in finding the best path to connect peers. In the case of asymmetric Network Address Translation (NAT), ICE will use a *Session Traversal Utilities for NAT (STUN)* (Session Traversal Utilities for NAT) server. A STUN server allows clients to discover their public IP address and the type of NAT they are behind. This information is used to establish the media connection. As shown in figure 3.5, a STUN server is only used during the connection setup and once that session has been established, media will flow directly between clients. In case of Symmetric NAT, STUN is not enough to establish the connection. In this case *TURN* Traversal Using Relay NAT come into action. TURN is an extension to STUN that allows media traversal over a NAT that does not do the consistent hole punch required by STUN traffic. Unlike STUN, a TURN server remains in the media path after the connection has been established. A TURN server literally relays the media between the WebRTC peers.

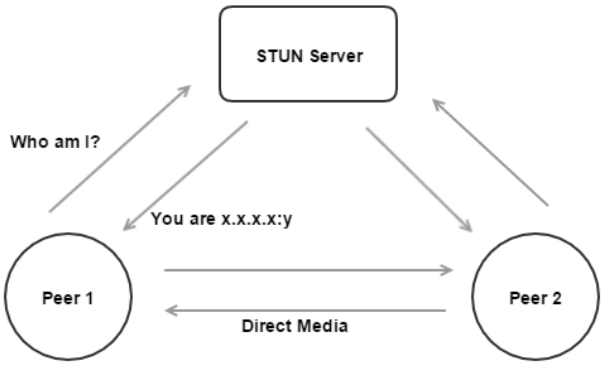


Figure 3.5: STUN Server during connection setup

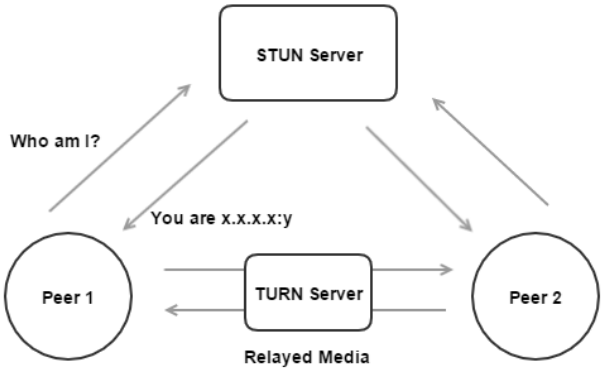


Figure 3.6: TURN Server relaying media between peers behind symmetric NATs

Socket.IO

In order to implement the whole functionality of WebRTC application with JavaScript, Node.js has been used for the backend. Node.JS is an asynchronous, server-side JavaScript engine powered by Chrome's V8 JS engine. Asynchronous is key for the nature of WebRTC because everything is an asynchronous event i.e. signaling, incoming/outgoing calls, presence, chat, etc. In addition to asynchronous, these events need to be handled in real-time.

Socket.IO is an event-based bi-directional communication layer for real-time web applications. It abstracts many transports, including Asynchronous JavaScript and XML (AJAX) long-polling and WebSockets, into a single API, and allows developers to send and receive data without worrying about cross-browser compatibility. Socket.IO provides both server-side and client-side browser components with similar APIs. Socket.IO primarily uses WebSockets to provide connectivity, but in addition to that, it automati-

cally switches to another transportation method in case of non-availability of WebSockets connection. Other transportation methods include Ajax long pooling, Ajax multipart streaming, Adobe Flash sockets, etc. Socket.IO also provides an option to make it forcefully use specific protocol, which is very useful where WebSockets are not supported. It can be forcefully made to use long pooling, e.g. xhr-pooling, and later on use WebSocket support when it is added.

To summarize, a flow diagram for nodejs architecture used along with authentication is shown in figure 3.7

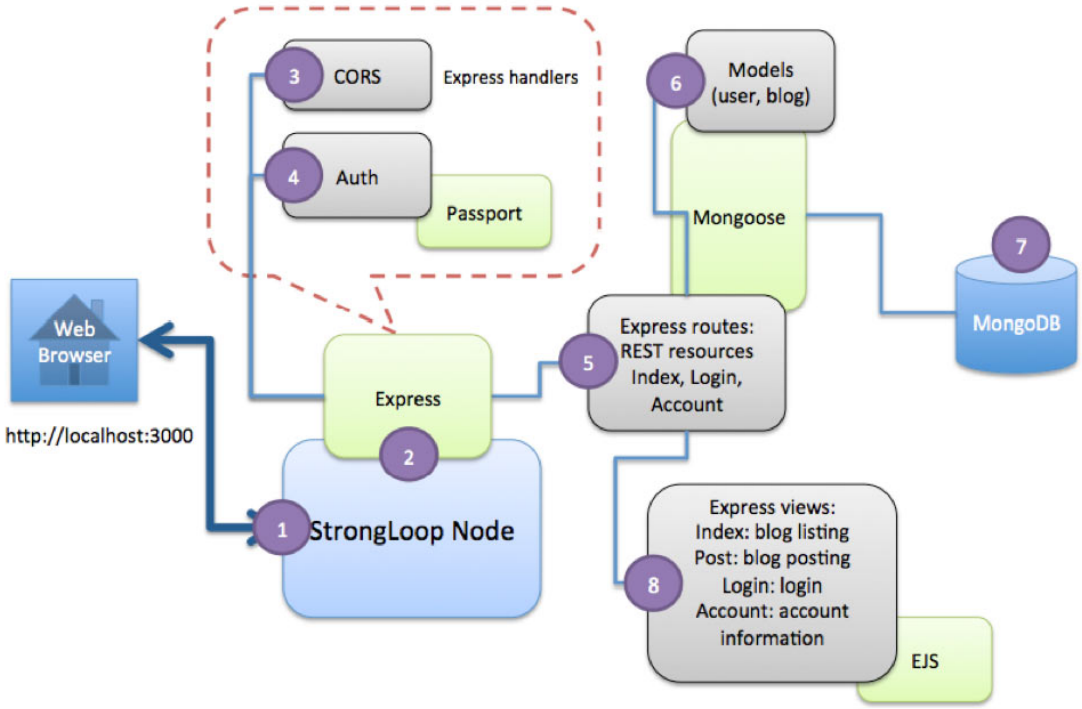


Figure 3.7: Backend Overview [29]

3.1.2 Frontend Overview

This section gives a broad overview about the frontend of the application developed in this thesis. Overall design of frontend is shown in figure 3.8

AngularJS

AngularJS is a very powerful framework for building Single Page Applications (SPA). It is targeted toward highly interactive applications and comes with many components

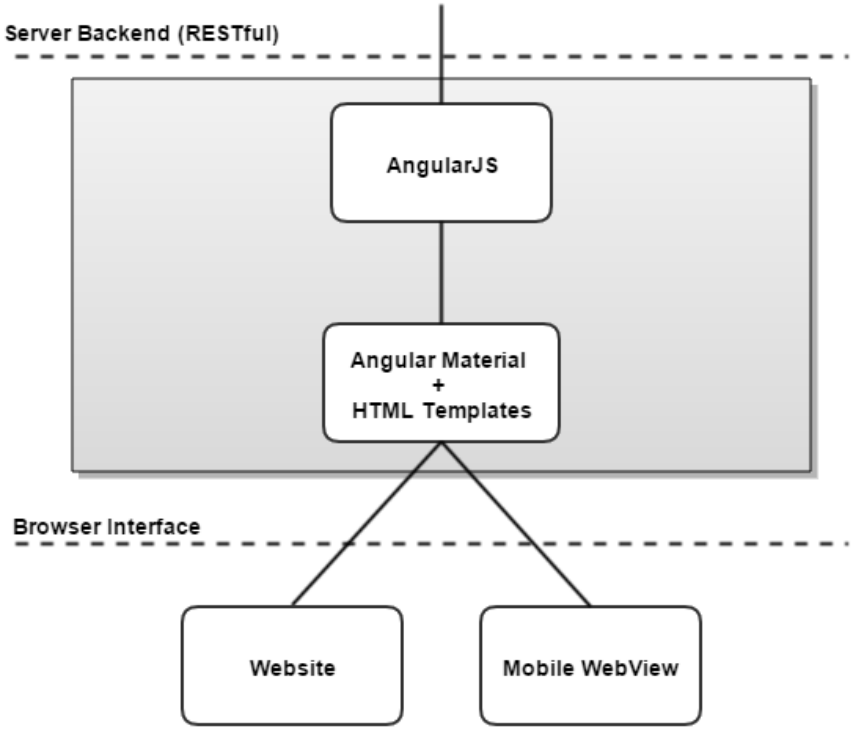


Figure 3.8: Frontend Overview

useful for building applications. AngularJS provides developers option to write client side applications in a clean, structured Model View Controller (MVC) way. This makes applications highly modular and easily testable. Some of the tools it includes are two-way data binding, a router, DOM manipulation, animations, dependency injection, testing utilities, Ajax helpers, and many more. In order to understand the architecture, it is very important to know about the working of MVC design pattern. A MVC pattern comprises of following three parts:

- **Model:** It is responsible for managing application data. Its major task is to respond to the requests from view and instructions from controller to update itself.
- **View:** It is the data presented in a form that a user wishes to see. The presentation of data in a particular format is triggered the controller's decision to present data.
- **Controller:** It responds to the input given by a user of the application, and performs interactions on the data model objects. The controller receives input, vali-

dates it, and then performs business operations that modify the state of the data model.

The MVC abstraction can be graphically represented as follows:

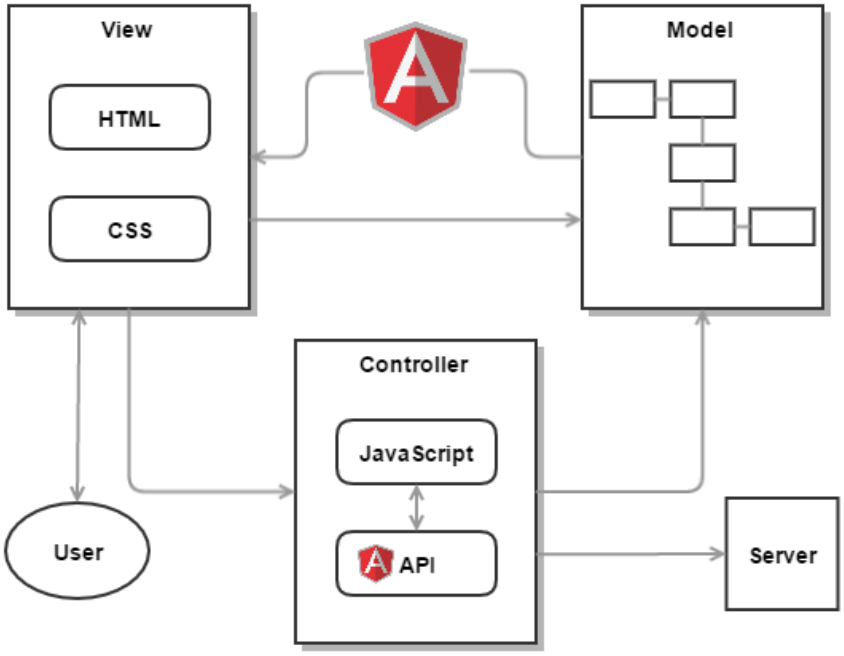


Figure 3.9: AngularJS Architecture

Angular Material

The Angular Material is an implementation of Material Design in Angular.js. It provides a set of reusable, well-tested, and accessible UI components based on the Material Design system. It is being developed and supported internally at Google. Material Design is a specification for a unified system of visual, motion, and interaction design that adapts across different devices and different screen sizes. Some of the fascinating objectives of creating Angular Material were:

- Create such visuals that synthesizes classic principles of good design with the innovation and possibility of technology and science.
- Develop a single underlying system that allows for a unified experience across platforms and device sizes.

Angular Material depends on modules like angular, angular-animate and angular-aria. *ngAria* provides accessibility support for assistive technologies, such as screen readers. It automatically sets aria attributes using the corresponding angular directives.

3.2 Core Services

This section discusses some of the core services that almost every web application should implement. Focus will be drawn on using above mentioned technologies to build these core services.

3.2.1 Serving Requests

The first step to doing anything with node.js is creating a server to accept incoming HTTP requests. Node.js provides tools for creating and serving HTTP and HTTPS requests. With the use of more advanced and sophisticated modules like Passport.js, it is possible to use secured cookies for essential session data. The server pushes the request's received URLs to the router, which makes the decision on how to server that request.

3.2.2 Routing

Express.js provides a straightforward mechanism to support routing. Routing can be required first for rendering the views so as to present them to the web browser, or it is also used for serving JSON requests coming from browsers or any other clients. Routes can be added in order to achieve one or more tasks. These can be briefly explained with examples as follows:

- **List:** GET to /contacts should respond with a list of contacts
- **Create:** POST to /contacts should create a new contact
- **Show:** GET to /contacts/:id should respond with a specific contact
- **Update:** PUT to /contacts/:id should update a specific contact
- **Destroy:** DELETE to /contacts/:id should delete a specific contact

3.2.3 User Authentication

User authentication is a must for an application which requires login/register feature to give a user right to access the content. Mongoose, as already discussed provides great

API to build a User Model. Now to achieve a good quality and secure authentication feature in application, following requirements should be met by the authentication service used:

1. The User model should fully encapsulate the password encryption and verification logic
2. The User model should ensure that the password is always encrypted before saving
3. The User model should be resistant to program logic errors, like double-encrypting the password on user updates
4. bcrypt interactions should be performed asynchronously to avoid blocking the event loop (bcrypt also exposes a synchronous API)

Passport.js provides most of the known authentication methods, including OAuth 2.0, Facebook, Twitter, Google+ etc.

3.2.4 Static Files

Express.js provides a feature to define a folder in the project to be static. Within this folder, complete client-side code an be stored and can be made available via HTTP. This way, by storing scripts, templates, images, etc. in a static folder, a browser client can be built and complete web application can be tested as a part of single project. This makes project development a lot easier.

3.3 Architecture Summary

The application is developed using JavaScript and is running on MongoDB Express.js Angular.js Node.js (MEAN) stack. MongoDB stores the data in binary JSON, which through Mongoose is exposed as JSON. The Express framework sits on the top of Node.js, where the code is all written in JavaScript. In the frontend AngularJS has been used to provide amazing data binding. Overall MEAN stack architecture with the flow and connection is illustrated in the figure 3.10. This architecture gives the ability to create all aspects of the web application using just one language and its related frameworks and libraries. Each of the components has distinct role, but they all function in cooperation to provide a very compelling end-to-end solution.

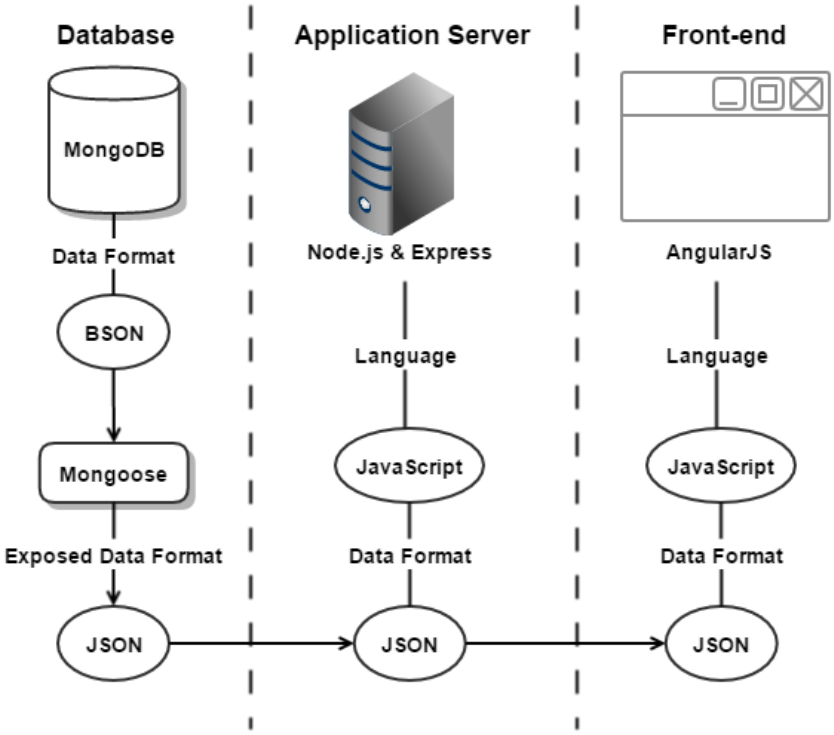


Figure 3.10: MEAN stack Architecture

4 Software Design & Implementation

In this chapter, we take a look at our application's implementation. Most of the technical implementation details pertaining to the prototype developed are explained in this chapter. Starting from the right project structure followed for best coding practices to implementation decisions, everything is presented in various sections of this chapter. Broadly, this application can be divided into two main components:

- **Frontend:** The AngularJS application, which provides actual user interface for online and offline support scenarios to the users. Everything that can be seen when a user navigates around the application, from fonts and colors to drop-down menus and sliders, is all included in Frontend and is controlled by the web browser.

- **Backend:** The server, which is responsible for the communication between different peers. This is the component which makes the frontend of the application work. This is responsible for storing data, fetching data and doing communication between components of the server. The backend consists of a server, an application, and a database. When a user opens this application, server component sends information to the user's device to turn it into the view that he sees.

4.1 Project Structure

Implementation phase consists of writing a many lines of code and as project progresses with time, it starts becoming more error prone and hard to manage. Therefore, it is necessary to follow a project structure so that developer can code rapidly and at the same time maintain good code quality. The project structure followed in this thesis represents many good practices and this section describes the overall structure of the project. Model View Controller (MVC) is the software design pattern that has been used in the implementation. Reason behind choosing this pattern is it's architecture which isolates the application logic from the user interface layer and makes the application easier to maintain and extend. AngularJS utilizes the MVC pattern and provides a powerful framework to build single page applications.

My implementation provides a structure that is modularized into very specific functions. The practices followed in the project pertain to a big application structure with

focus on commercialization. Following directory structure has been followed, and is shown in figure 4.1:

- Root directory contains some application configuration files such as package.json and bower.json
- *"client"* directory in *"public"* folder is the root of front-end structure and contains all the application logic, which is again structured very systematically. Similarly *"server"* directory in *"public"* folder is the root of back-end structure of the application.
- *"index.html"* file in *public* directory will primarily handle loading in all the libraries and Angular elements.
- *"assets"* folder contains all the assets needed by app that are not related to AngularJS code.
- *"app"* folder is where the meat of AngularJS app lives. *"app.module.js"* handles setup of AngularJS app by loading dependencies. *"app.route.js"* handles all the routes and route configurations.
- *"components"* directory contains actual section of the application. Static views, custom directives and services for a specific section of application are all located as a component in this directory. Each component here resembles a mini-MVC application by having a view, controller and services file.
- By this approach, it is very way to add new components based on the views and every model-view remains loosely coupled from the other.
- *"shared"* directory present in *app* directory contains individual features that this application should have. Examples can be a slider which can be made a separate component which is shared by multiple views as it's design and features will not change in different views.
- Modularization of code has been the focus of my implementation. Providing separate folders for *core* components and *other* components like header, footer enhances the modularity. It can also be seen that the routes have been separated into separate files in each component. This is also done to modularize the code and improve the development structure.
- To make this application ready for production, I have included a *Gulpfile.js* in the root of the application. Running specific *gulp* commands will make this application production ready.

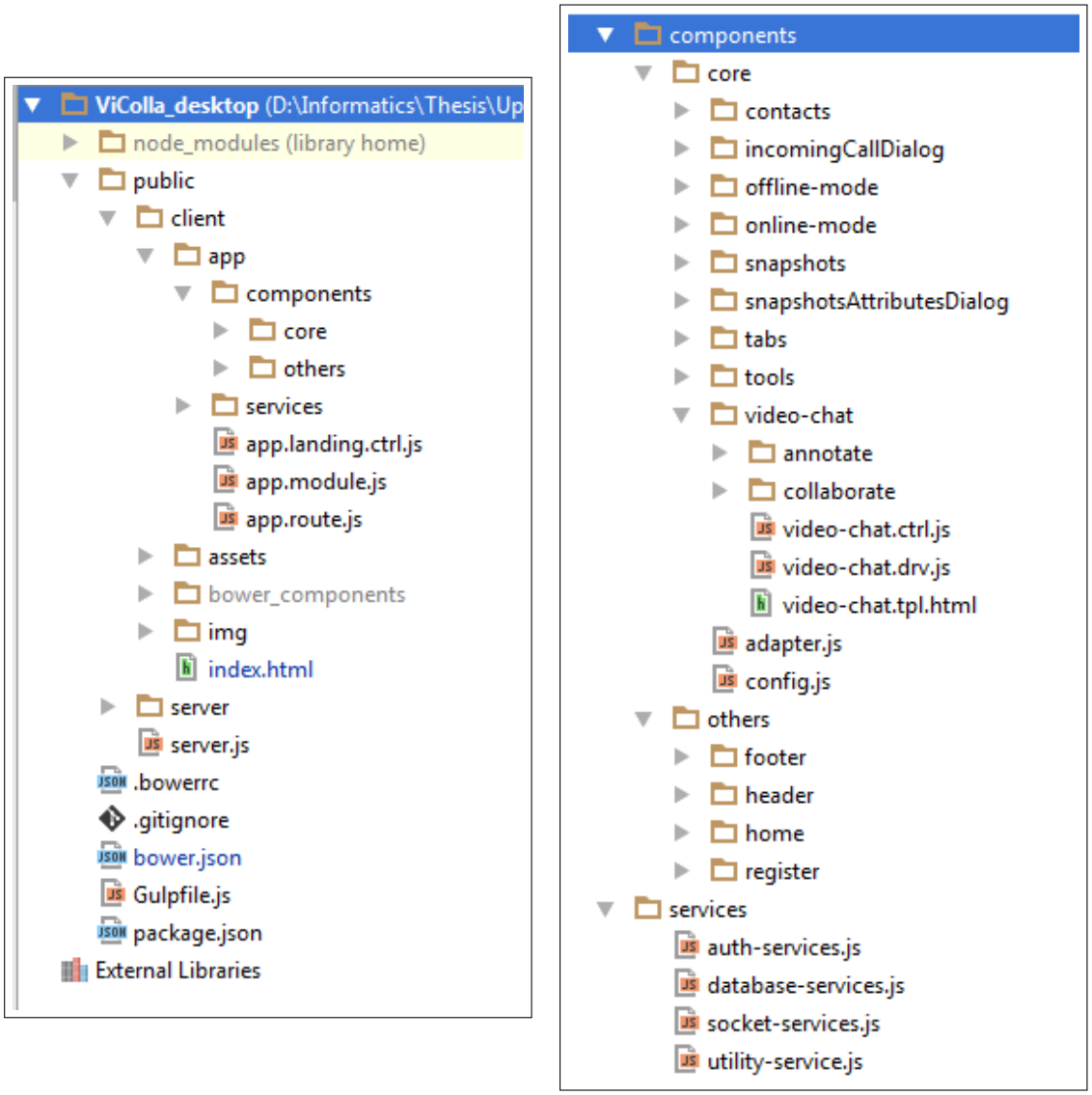


Figure 4.1: Project Structure

4.2 Application Design

User interface should be designed in a self explanatory and intuitive manner, so that users can get used to it at the first sight. Also it should be extendable so that later improvements could be easily integrated. Since one part of the application is supposed to be used by ordinary workers/users in real time, so it is necessary that it should take care of the mobility of users. Hence I decided to implement a browser client focusing on mobile devices. The other part is supposed to support experts who would provide assistance to other users. These experts would need a stable environment to work so that gestural communication and video annotations occur steadily. So, this part is focused on desktop devices with user interface implemented to be run on web browser.

Views Flow Diagram

Mobile web applications are quite different from the desktop web applications in the manner they are designed. Mobile devices have smaller screens and use touch screen as an input, so user interface design is a bit different. So, it is always beneficial to use the approach of layered applications. Home screen is the top layer [], from where the user will be able to log off and get to the login page, or access other features. Core functionality then goes on the layer below, which provides a great opportunity for extending the application in future.

Figures 4.3 and 4.2 illustrate the state diagram in UML which represent the flow among different views of the desktop version and mobile version of this application respectively. Desktop version would serve for experts and mobile version for workers/users.

Figure 4.4 shows an example of JSON data for new user's registration, which is sent as POST request to the server. If the data is marked correct by *Passport* service, it is stored in the *MongoDB* database as a user's record. After successful storage of data in database, response object is sent back to the client to mark the confirmation of user's registration. Example of this request-response objects pair is depicted in the figure 4.4

4.3 Online Mode

As illustrated in the views flow diagrams, online mode is provided for a user to get instantaneous assistance from an expert. Only requirement for a user to use online mode is the Internet connection so that he can contact an expert for the instantaneous help. There are three different ways in which a user can get online assistance from an expert. These three scenarios are explained in the following sections:

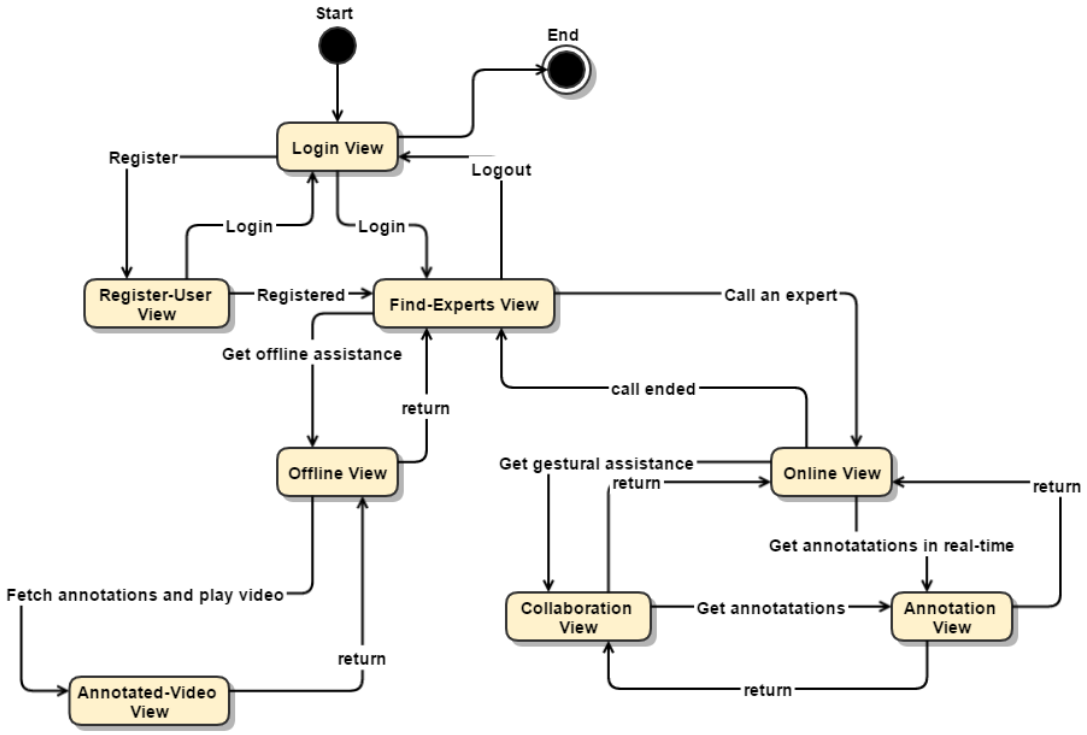


Figure 4.2: Mobile Version - State diagram in UML Views Flow

4.3.1 Video Chat

Video chat module, as the name suggests, is utilized by a user to get assistance from an expert in real time. A user can search for an expert by going to the *"Find Expert"* tab of the application. A complete list of experts irrespective of the designation, place or expertise areas is listed there. A user can also search for an expert corresponding to a particular expertise area. Expertise tags are used to define the expertise areas of an expert which can be seen or searched in the list. As already mentioned in the Views Flow diagrams in figure 4.3 and 4.2, the view for searching and initiating a call is only restricted to the mobile view. The reason being, only workers should be able to find and search an expert. Desktop view being used by an expert only has the possibility to answer or decline the incoming call, but an expert can't call any other user or an expert.

Implementation of this module can divided into many smaller parts, which are as follows:

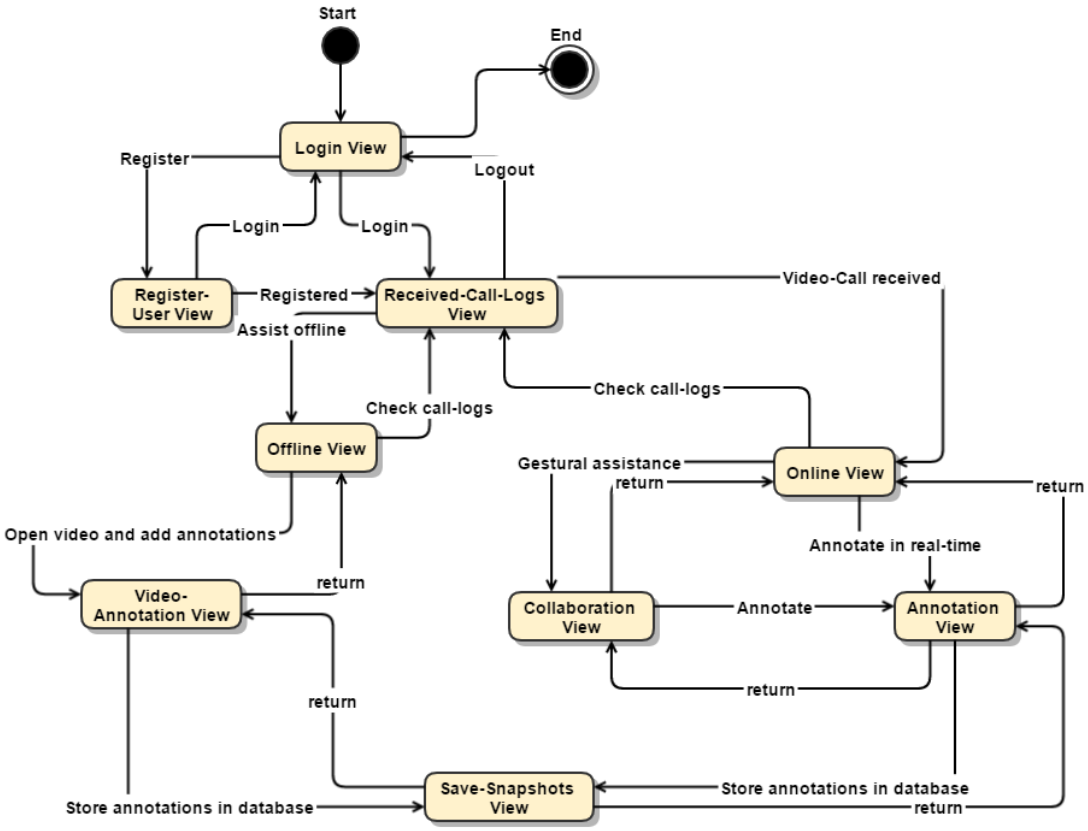


Figure 4.3: Desktop Version - State diagram in UML Views Flow

1. Get streaming audio and video

2. Get network information such as IP addresses and ports, and exchange this with other WebRTC clients (known as peers) to enable connection, even through NATs and firewalls.

3. Coordinate signaling communication to report errors and initiate or close sessions.

4. Exchange information about media and client capability, such as resolution and codecs.

5. Communicate streaming audio and video.

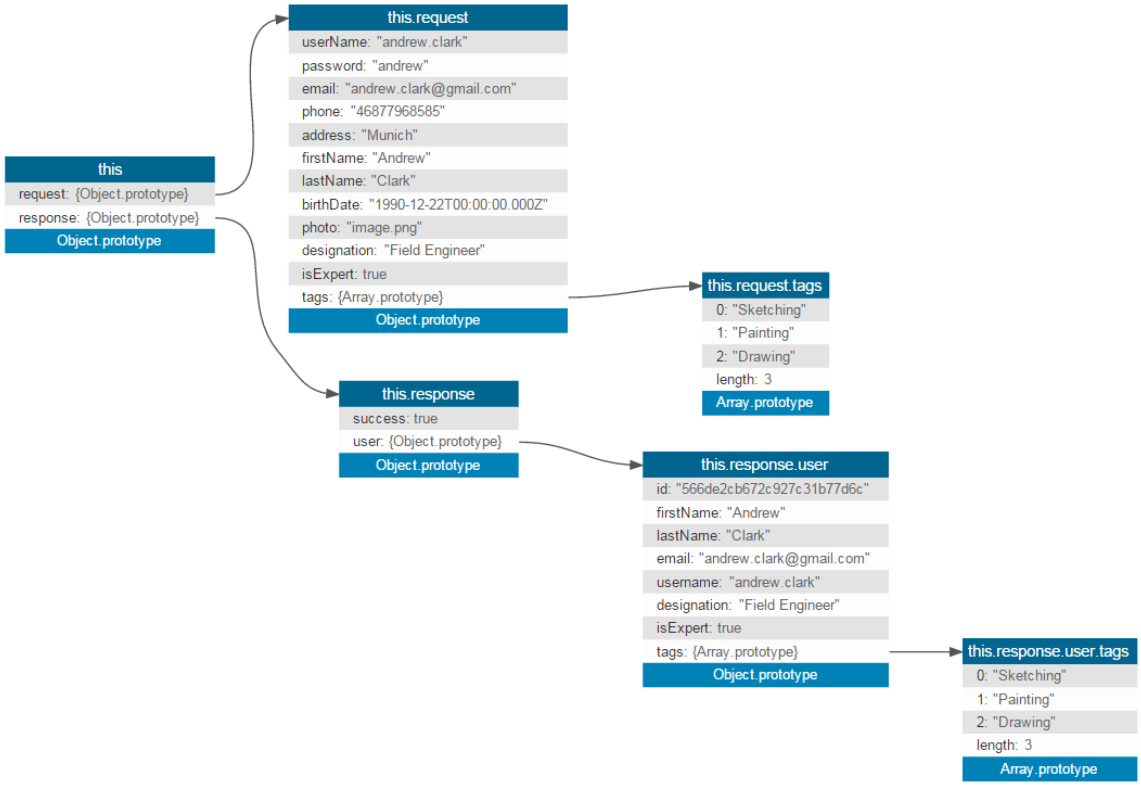


Figure 4.4: Register-Response objects during User Registration

Video chat module has been implemented in this application using the concepts of WebRTC. 2 APIs of WebRTC have been used in the implementation namely:

- **MediaStream** (aka **getUserMedia:**) This API is used to get access to data streams, such as from the user's camera and microphone. Each *MediaStream* has an input, which might be a MediaStream generated by **navigator.getUserMedia()**, and an output, which might be passed to an HTML5 video element or an RTCPeerConnection.

- **RTCPeerConnection:** This API is used for audio or video calling, with facilities for encryption and bandwidth management. *RTCPeerConnection* is used to communicate streaming data between peers. Apart from this, there is a need of a mechanism to coordinate communication and to send control messages. This application is built using *Socket.IO* running on a *Node* server for signaling purpose and hence passing the control messages between two peers. Signaling to exchange media configuration information proceeds by exchanging an offer and an answer using the Session Description Protocol (SDP):

1. Peer-1 runs the RTCPeerConnection *createOffer()* method. The callback argument of this is passed an *RTCSessionDescription*: Peer-1's local session description.
2. In the callback, Peer-1 sets the local description using *setLocalDescription()* and then sends this session description to Peer-2 via their signaling channel.
3. Peer-2 sets the description Peer-1 sent him as the remote description using *setRemoteDescription()*.
4. Peer-2 runs the RTCPeerConnection *createAnswer()* method, passing it the remote description he got from Peer-1, so a local session can be generated that is compatible with Peer-1's. The *createAnswer()* callback is passed an RTCSessionDescription: Peer-2 sets that as the local description and sends it to Peer-1.
5. When Peer-1 gets Peer-2's session description, he sets that as the remote description with *setRemoteDescription*.

This is how signaling mechanism is working in this module with the help of SDPs. Rest of the work is taken care by MediaStream API which fetches the audio and video stream and publishes it on browser using HTML5 video element. The *create-offer* object sent by caller to signaling server, which has SDP as its property is shown in figure 4.5. This offer is forwarded by signaling server to call receiver, who then creates an answer. Object constituting the *answer* also looks similar to the *offer* object with an SDP as its property. Only values in SDP are changed based on the IP address and port combinations of the other peer.

Successful video call in the application is seen as depicted in figure 4.6. User in non-expert mode using the mobile view of the application checks the online status of respective expert user. If the expert (using the desktop view of this application) is online, then non-expert initiates the video call, which is received by the expert user at the other end. Both the views , mobile and desktop, for video call are shown in the figure 4.6.

4.3.2 Contextual Assistance using Gestural Communication

This module extends the video-chat module to improve the real time collaboration by making the use of gestural communication. In this mode expert opts to collaborate more intuitively with remote user by using his hand gestures to provide assistance. This is achieved by allowing expert to use his hand gestures in front of a commonly found plain white/gray wall. The hand's movement is segmented in real time from the background wall. This segmented hand is then overlapped on the live video stream and transmitted back to the remote user. So, this gives the intuition that expert is using his hand gestures to assist the user as is done in the case of face-to-face collaboration.

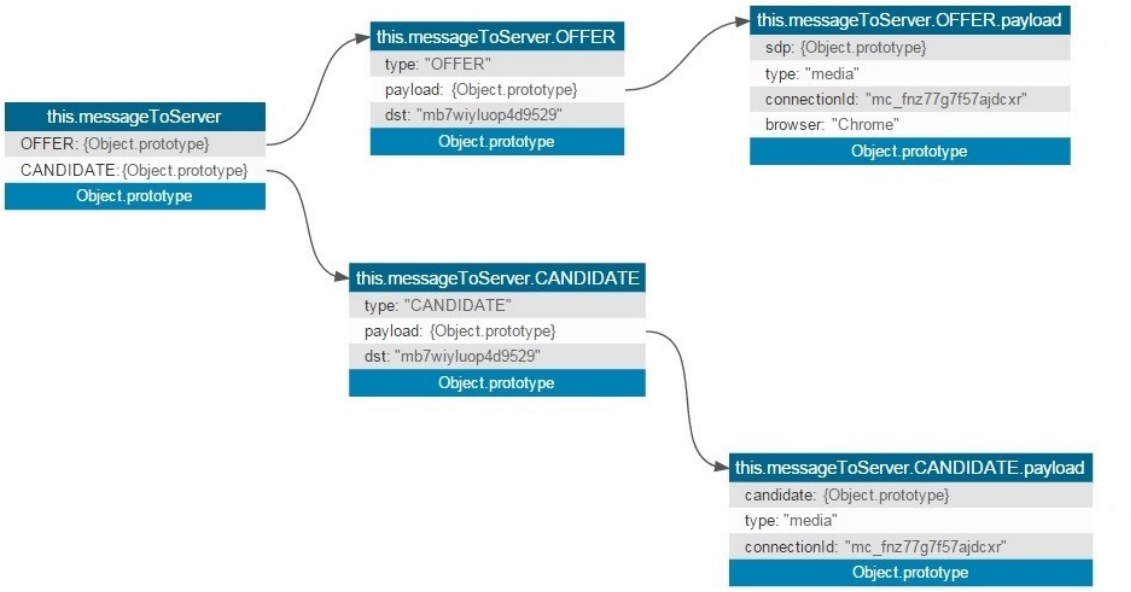


Figure 4.5: Create Offer Message To Signaling Server From Caller

Implementation of this module is divided into following parts:

- Identifying the hand and its gestural movements in the live video feed of expert
- Hand segmentation and hence removal of the unnecessary background
- Overlapping of segmented hand video over live video feed coming from remote user who needs assistance.

All the steps are achieved by using the pixel manipulation techniques on HTML5 canvas. It is possible to get the access to the individual pixels of an HTML5 canvas. This is done using an *ImageData* object. The *ImageData* object contains an array of pixels. By accessing this array the pixels of canvas can be manipulated. After the pixel manipulation has been done, these pixels are copied onto canvas to display them.

Creating an *ImageData* object is done using 2D context function *createImageData()*. The *width* and *height* properties of *createImageData()* function sets the width and height in pixels, of the pixel area represented by the *ImageData* object created.

```
1 var canvas  = document.getElementById("inputCanvas");
2 var context = canvas.getContext("2d");
3
4 var width  = 100;
5 var height = 100;
```

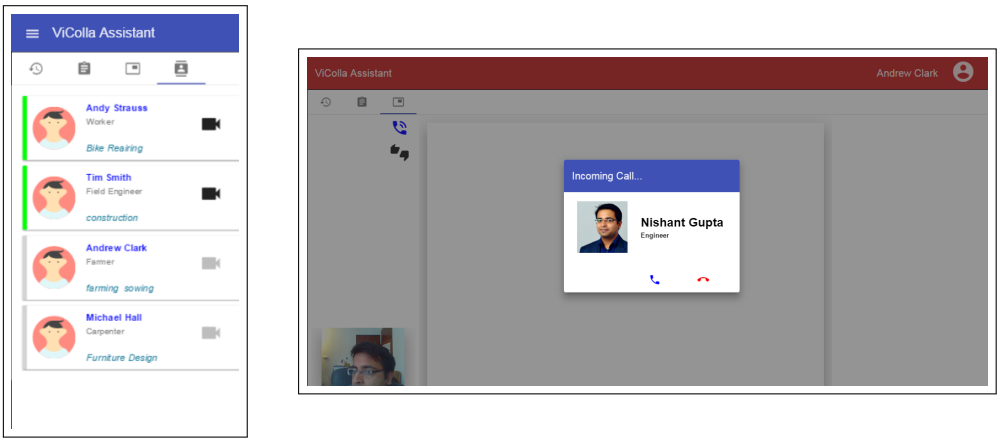


Figure 4.6: Expert user providing contextual assistance

```
6  var imageData = context.createImageData(width, height);
```

Listing 4.1: Creating *ImageData* object

Each pixel in the *ImageData* array consists of 4 bytes values. One value for the red color, green color and blue color, and a value for the alpha channel. The color of a pixel is determined by mixing red, green and blue together to make up the final color. The alpha channel tells how transparent the pixel is. Each of the red, green, blue and alpha values can take values between 0 and 255. In order to set the color and alpha values of a pixel, following code can be used:

```
1  imageData.data[pixelIndex    ] = 255;  // red   color
2  imageData.data[pixelIndex + 1] =   0;  // green color
3  imageData.data[pixelIndex + 2] =   0;  // blue  color
4  imageData.data[pixelIndex + 3] = 255;
```

Listing 4.2: Setting color and alpha values of a pixel present at index *pixelIndex* in ImageData array

End result of the implemented Contextual Assistance in this application is depicted in figure 4.7. It can be seen clearly that a user in expert mode is using his hand gestures in front of the web-cam attached to his desktop. The other user, who is seeking the assistance sees the remote user's hand overlapped with his live video of environment. In the figure 4.7, user can be seen seeking help in operating the elevator's panel.

4.4 Offline Mode

This mode is an alternative for users who can't connect instantaneously to an expert for assistance because of lack of good Internet connection in the work environment.

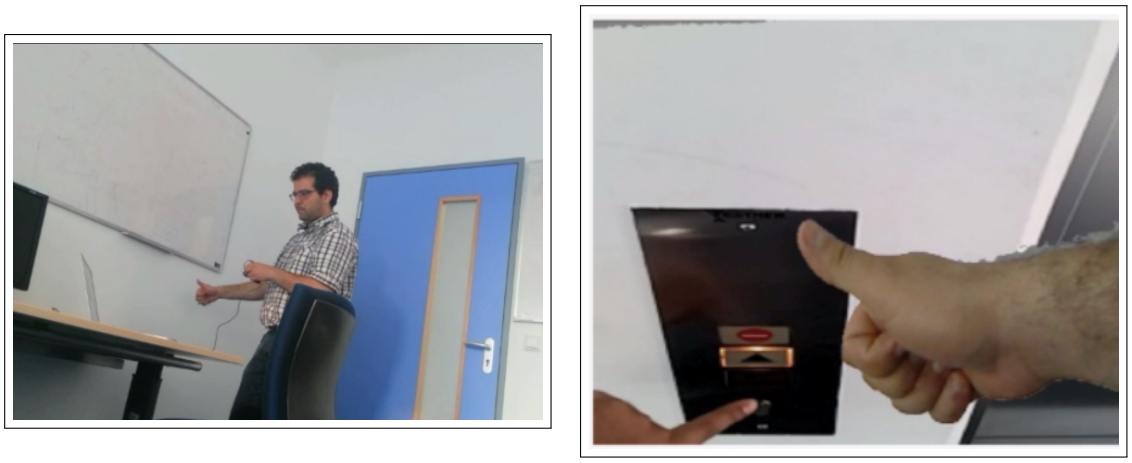


Figure 4.7: Expert user providing contextual assistance

Following phases summarize the way in which offline mode can be used for remote collaboration over video:

1. **Capture Video:** User captures the video of the work environment using his device's camera. He explains the problem and matter of assistance required.

2. **Send Video:** After he moves to a location with sufficient Internet connectivity, he sends that video to an expert of his choice. Expert can be found by logging-in to the application and searching for an expert with relevant expertise tag.

3. **Annotating Video:** Expert, on receiving the video, analyzes and annotates the video using annotation tools available in the offline assistance mode of the application. Application allows the expert to save the snapshots of annotations drawn on video. The set of snapshots for a particular video are saved with a unique identifier in the central database.

4. **Fetch Annotations:** After the expert has annotated the video and saved annotated snapshots in database, user/worker can fetch those annotations simply by opening the original video and clicking the fetch annotations button.

5. **Play Annotated Video:** In the last step, User can now construct annotated video from the fetched annotations by using the utility provided in the application. This annotated video would contain all the necessary annotations defined by the expert in with appropriate information.

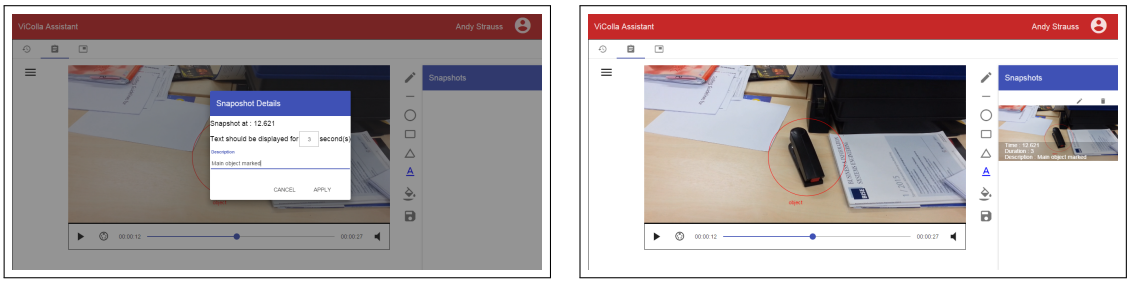


Figure 4.8: Add annotations and save snapshot

4.4.1 Annotation Model

Annotation model is the most crucial aspect of the offline mode of this thesis. It is very important to follow an annotation model which not only solves the desired purpose of annotating video objects, but also focuses on doing it in an efficient manner. Sharing a video over Internet or transferring a video to remote location requires time and consume resources, which is why the annotation model provided in this work aims at providing minimal data to be transfered after annotating the video. Following are the characteristics of the annotation model followed in this thesis:

- Only a user in expert mode has the privileges to annotate the video. User in non-expert mode can only view/read the annotations corresponding to a particular video.
- After the annotations have been drawn over a paused video frame, user in expert mode needs to save that snapshot.
- While saving the annotated video frame snapshot, it is mandatory to provide a time duration. That time duration indicates for how long that video frame will appear with its annotations, when it is played by the user.
- This annotation model allows a user in expert mode to apply CRUD operations on the annotations for a video.
    1. **Create:** Create refers to the process of adding new annotations to a video frame. The annotations are created using drawing tools available in ViColla offline view. Drawing tools include a freehand pen tool, text tool, or generic shapes like line, rectangle, triangle, circle, etc. The annotations drawn for each frame are required to be saved into database, so that other user willing to see those annotations is able to fetch them from database. This addition of annotated frame into database occurs by clicking *save* icon given in the tools set of ViColla offline mode. As shown in the figure **??**, details for the

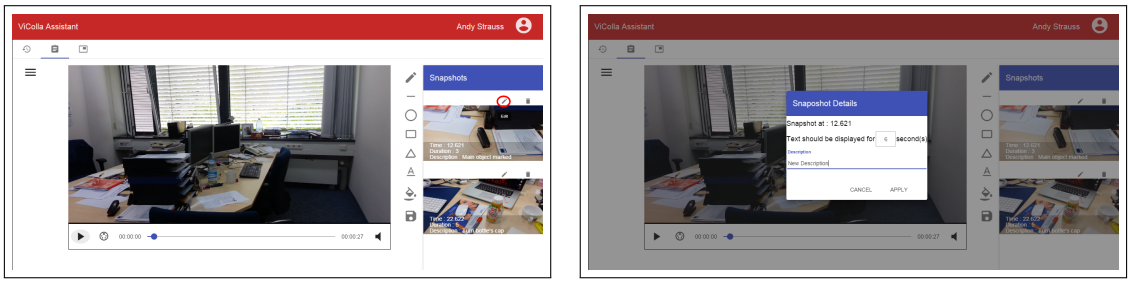


Figure 4.9: Edit existing annotations

frame like duration, description can be set by the expert while saving it. This annotation model makes use of **MD5 Hash** of a video as a unique identifier for saving corresponding annotated frames in the database.

2. **Read:** Read operation refers to the process of viewing the already created annotations for a particular video. This read operation can be performed by any user, whether in expert or non-expert mode. For viewing the annotations created for a video, it is only required to open the original recorded video in *ViColla video player*. When a video is opened in *ViColla video player*, it checks for are annotations corresponding to that video already present in database. This annotation model checks annotations by calculating **MD5 Hash** of the video and looks for snapshots saved in database with this hash value as the identifier. All the annotations created for that video are fetched from database and displayed in the *Snapshots Palette*.

3. **Update:** Any user in expert mode is allowed to edit the annotations corresponding to a video available with him. This ability to update annotations is extremely useful when multiple users annotate one video. A user can view all the annotations added by other users and can edit the information like description or time duration for which annotation frame should be displayed. As depicted in figure 4.9, this editing can be done by clicking on *edit* icon present in *tools* section of ViColla offline mode.

4. **Delete:** Besides creating new snapshots for a video frame, or editing already existing snapshots, it is an important feature to add the possibility to delete snapshots which are no longer required or have been added by mistake. In ViColla offline mode, the annotation model allows a user in expert mode to delete existing snapshots. User in non-expert mode does not have the right to delete or edit the existing snapshots. Clicking the *delete* icon in *tools* section, allows a user to delete snapshots, not just from the view for one session, but also from the database permanently.

4.5 Other Features

This section represents some of the features which provide extension to the main application purpose to make it a complete and self contained product. These features basically connect the missing dots between different client views so as to present complete application in more logical and user friendly manner.

4.5.1 New User Registration

User registration is required for a new user trying to access the application for the first time. It is very important to provide authenticated access to the user and manage the session data at the same time. Figure 4.10 illustrates how new user registration occurs using Passport.js authentication and Mongoose module for storing user data in MongoDB.

User registration begins with a user entering his desired credentials in the *registration* view. This user model is handled by *RegisterController* which uses *authService* to validate the user inputs and proceed with registration process. *authService* in turns sends *POST* request with all the credentials entered by user using *$http* core service. *$http* service passes the task of user authentication and validation to *Passport* module which communicates with database *MongoDB* via *Mongoose* module offered by *Node.js*. *Mongoose* performs the check if user with same credentials already exists in the database. If not, then it stores the new user's credentials and proceed by sending success response back to the *Passport*. *RegisterController* receives the notification about user registration, which is then reflected in the *Registration View*.

4.5.2 Find experts

The mobile client provided to the user who needs assistance allows him to find experts pertaining to his problem criterion. By default, all the experts contacts are loaded into the *contacts* tab of the application. Sequence diagram shown in figure 4.11 depicts the process how application loads all expert contacts from the database. Please note that details of only those users are fetched from the database who are registered as experts.

As illustrated, process begins when an expert uses the mobile client of the application to log in to the whole environment. **authService** validates the user credentials and allows the user to log in. If validation succeeds, a request is sent to **databaseService** which handles all the database related requests. In this case, *databaseService* finds all the users whose *isExpert* field is set to *true* in the **users** schema of the database. Relevant records are returned to the client with properties like id, first-name, last-name, designation, expertise tags.

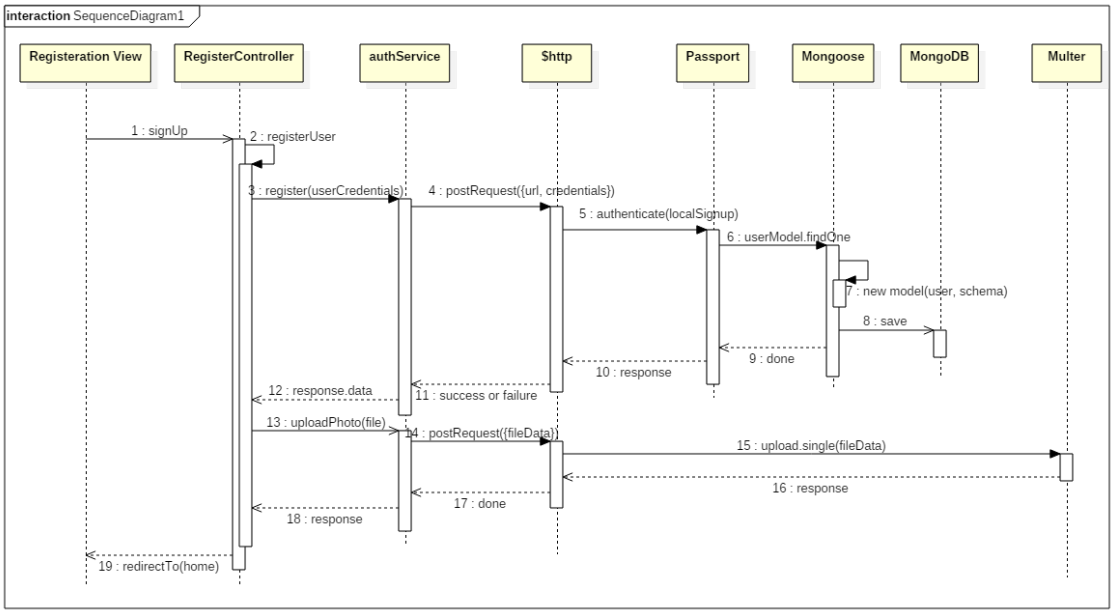


Figure 4.10: Sequence Diagram depicting User Registration process

There is further enhancement in this utility for assisting mobile user to look for particular expert. Search mechanism provided in *contacts* view enables a user to search for an expert by his name, field of expertise (as depicted by his expertise tags), or his designation.

4.5.3 Call Logs

An expert, running this application in desktop environment, can see the details of previously initiated, received or rejected calls. Figure 4.12 shows the process in which the call logs are created and fetched from the database and rendered in the view.

As illustrated, process in this case again begins with an expert using the desktop client to log in to the environment. **authService** validates the user credentials and allows the expert to log in. Then *databaseService* finds all the received or initiated call details in database where these logs are defined by the schema called *callHistory* schema. Call logs with necessary properties including caller-name, call-date and duration of call is returned as response to the client. This response is then rendered and presented in the form of a list in the *receivedCallLogs* view.

This is helpful in setting up a record of the calls that have been received or initiated so far. At the same time, it also helps in managing the call priorities for future and might be very useful for an organization where expert gets paid based on the amount

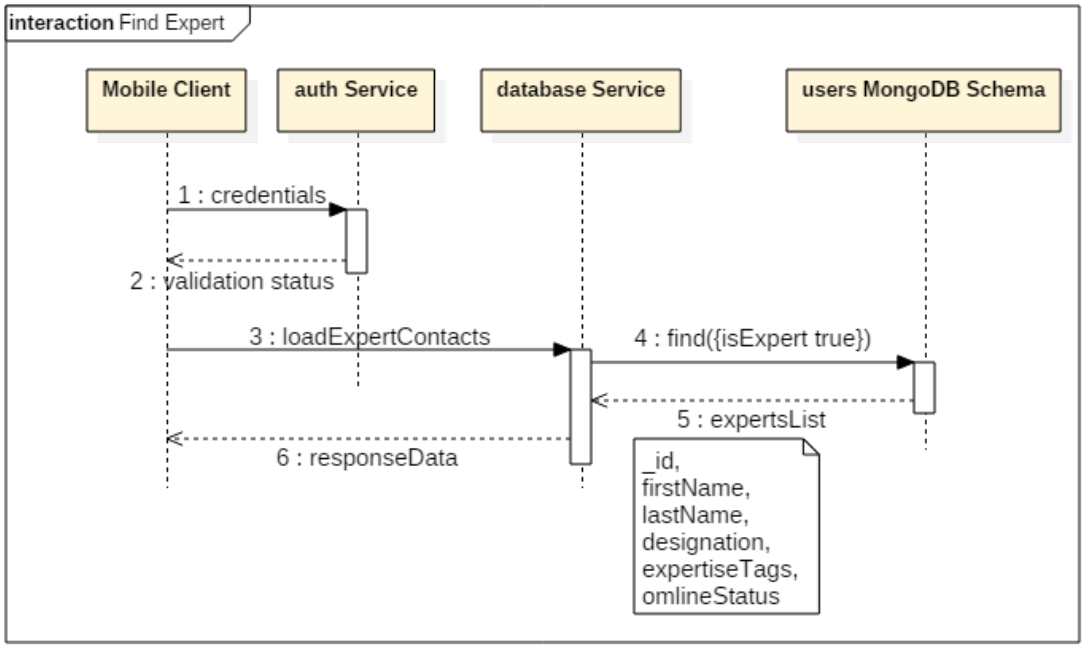


Figure 4.11: Sequence Diagram for fetching list of experts from database

of calls he receives and hence provides his assistance.

This feature is implemented with standard color coding scheme used for initiated calls, receiving calls, or rejected calls. As can be noticed in the figure 4.13, rejected calls are marked with red colored icon, while others care marked with blue colored icons. The views of calls logs in mobile as well as desktop are shown in the figure 4.13.

4.6 Implementation Decisions

This section describes some of the major implementation decisions taken during the development process of this prototype. These decisions were made based on the performance factors so as to improve the prototype quality. Following sub-sections illustrate details of some of those decisions which include *server configurations, user authentication*, and some of the *core services* used in the implementation.

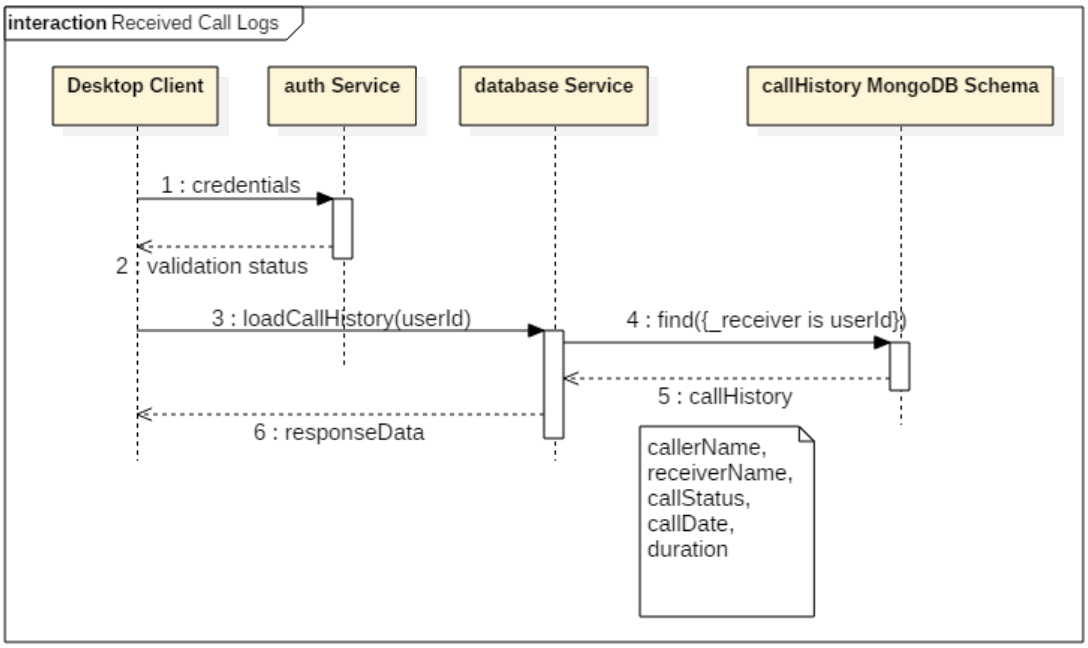


Figure 4.12: Sequence Diagram for fetching call logs from database

4.6.1 Configurations

This section discusses the configuration decisions taken during the implementation. This is further divided into three parts. First is the configuration of *Express.js* middleware. Here all the middleware components used and reason behind using those components have been described. Second comes the STUN server configuration, which is essential for establishing video-call, especially when peers are located behind NATs. Apart from these configurations, the kind of database schema used for different purposes like *received-call logs*, *user-credentials*, *annotated snapshot's details* are also described in this section.

Configuring Express middleware

In *server.js* file, there are a bunch of lines that start with *app.use*. These are called the **middleware**. When a request comes in to the application, it passes through each piece of middleware. Each piece of middleware does something with the request and then passes onto the next one until it reaches the application logic itself which returns a response. Figure 4.14 illustrates the flow in which Express middleware functions and provides the first view of the application.

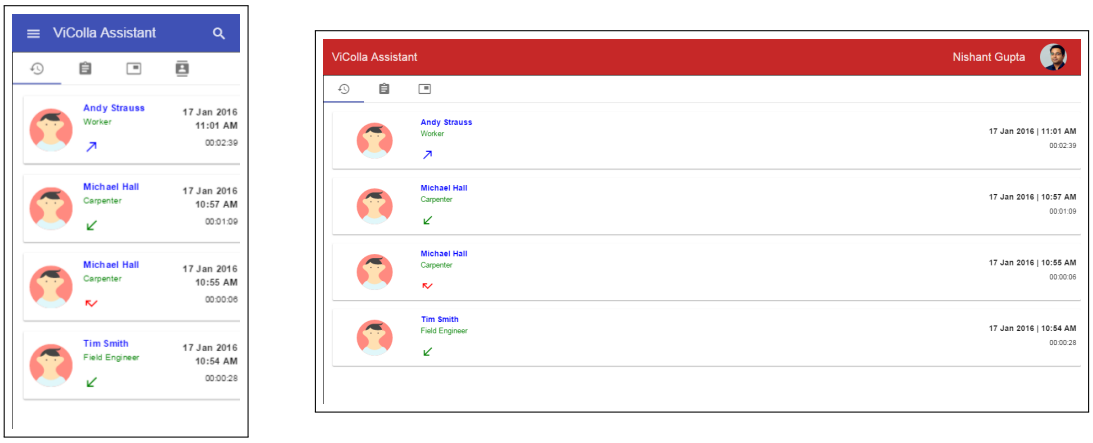


Figure 4.13: Call Logs View

STUN server configuration

Since, STUN server plays a major role in initial connection establishment between peers, it is important to tell the WebRTC application where to find the STUN server. In the implementation, this is done while *RTCPeerConnection* object is created for WebRTC signaling. STUN server that has been designated in the configuration is shown below:

```
var config = {stunurl: 'stun:stun.l.google.com:19302',
      stunip:'stun:23.21.150.121'}
var pc_config = webrtcDetectedBrowser === 'firefox' ?
      {'iceServers':[{'url':config.stunip}]} :
      {'iceServers': [{'url': config.stunurl}]};
pc = new RTCPeerConnection(pc_config, pc_constraints);
```

Listing 4.3: STUN server configuration

Database Schema

Three collections have been used to store the necessary information in the application. Definition and purpose of there collections can be discussed as follows:

- **User:** This schema stores all the details of a user when he registers for the first time. Login information of each user is stored in this schema.
- **CallsLog:** This schema stores the details of the calls received by each user. Information related to status of the call being answered or missed is also a part of this schema.

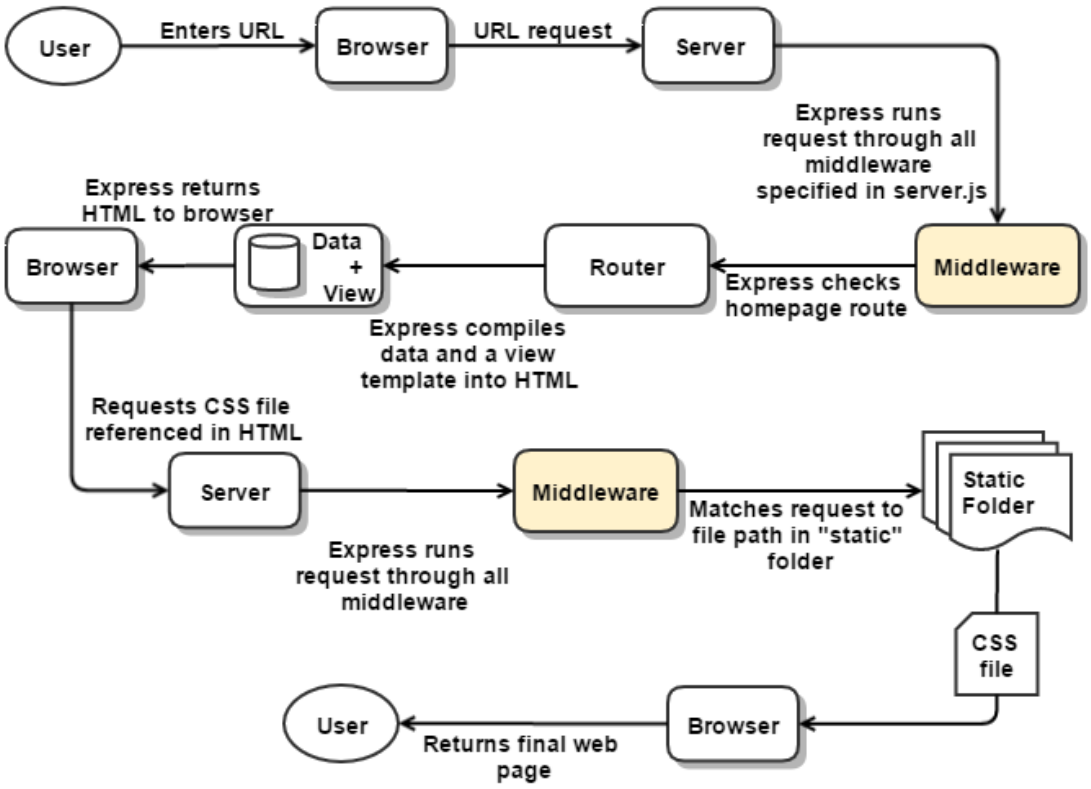


Figure 4.14: The key interactions and processes that Express goes through when responding to the request for landing page of user-login

- **Snapshots:** Since an expert can save snapshots of the annotations drawn over video, it becomes necessary to store them so that other intended user can see and use them to construct annotated video. This schema stores those snapshots and related information such as time duration, description and name of the video which was annotated.

4.6.2 Authentication

Rather than adding the same logic to authenticate to each individual route, a sophisticated middleware is used to intercept the request before it gets to any of the routes, and perform some action. As depicted in figure 4.17 middleware consisting of *cookie-parser*, *body-parser*, *express-session* solves the objective.

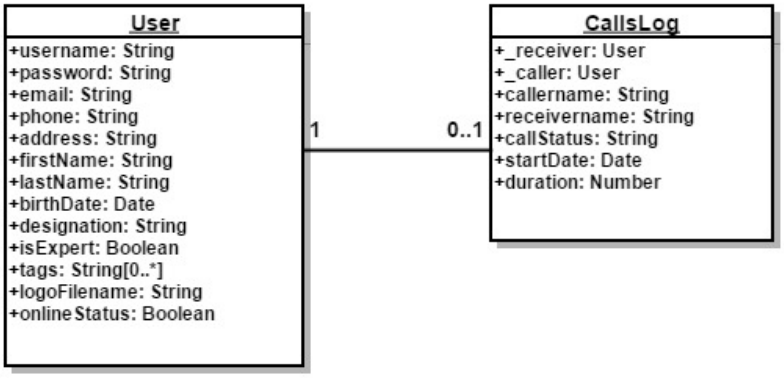


Figure 4.15: Schema representing a User and Call Logs

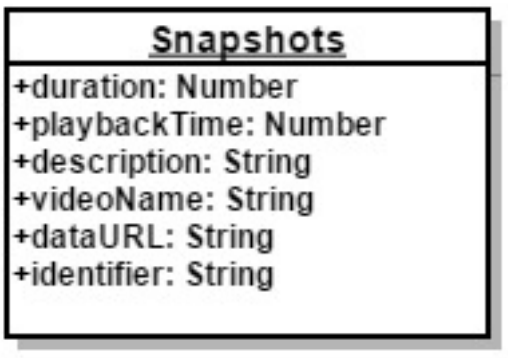


Figure 4.16: Snapshots Schema

4.6.3 Identifier for Video used in Offline Annotation Model

As explained in the annotation model of *Offline Mode*, a user can create, view, edit or delete annotations created for a video. These annotations are stored in the database, and a user fetches those annotations from database while opening video in the application's video player. This means that there has to be a identifier that would uniquely identify a video. This unique identifier would be used for storing in database the snapshots corresponding to a video. This annotation model presents identifier in the form of a MD5 Hash which is always unique for a video.

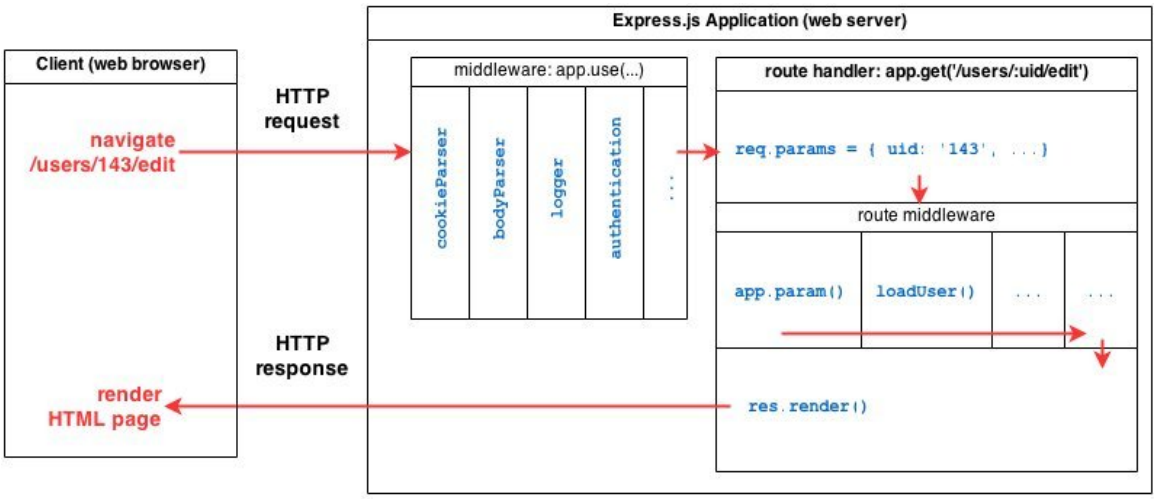


Figure 4.17: Middleware intercepting request before routing during Passport.js authentication [11]

Message Digest algorithm 5 (MD5)

MD5 message digest algorithm is a cryptographic hash function which produces a 128-bit hash value. This hash value is usually represented in text format as a 32 digit hexadecimal number. MD5 processes a variable-length message into a fixed-length output of 128 bits. To calculate MD5 hash of a video file, **SparkMD5** [4] has been used. **SparkMD5** is an implementation of the MD5 algorithm in JavaScript. It is based on the *JKM* MD5 library which is the fastest algorithm to calculate MD5 hash. Being a suitable option for browser usage, because of nodejs implementation, makes it perfect for use. Features like processing the video file incrementally to calculate its MD5 hash make SparkMD5 highly performant.

4.6.4 Core Services Used

$q

$q is an AngularJS core service that allows developers to work with asynchronous functions and user their return values when the execution has been completed. With $q, it is possible to write custom promises which gives the advantage of resolving or rejecting a promise when appropriate.

Promise

A promise in the JavaScript and AngularJS world is an assurance of getting a result from an action at some point in the future. There are two possible results of a promise:

- A promise is said to be fulfilled when a result from that action is received, no matter the response is good or bad.
- A promise is said to be rejected when a response is not obtained. For instance if some data was to be retrieved from an API and for some reason a response was never received because the API endpoint was down.

Figure 4.18 illustrates how an asynchronous invocation occurs in AngularJS using promises.

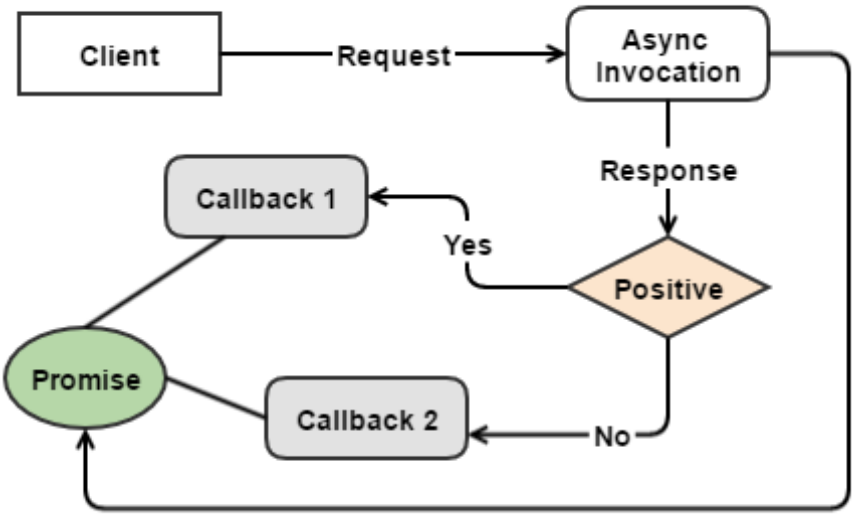


Figure 4.18: Asynchronous Invocation with promise

$http

The $http service is a core Angular service that facilitates communication with the remote HTTP servers via the browser's XMLHttpRequest object or via JavaScript Object Notation with Padding (JSONP). The $http API is based on the deferred/promise APIs exposed by the $q service. The $http service is a function which takes a configuration object as argument, that is used to generate an HTTP request and returns a promise.

$http is being used in AngularJS framework for performing CRUD operations and generating the AJAX calls in much more simpler manner.

Routing

AngularJS provides core service called $route which can be used to organize routing around URL routes. But considering advantages of **AngularUI Router** over Angular's core $route service, AngularUI Router has been used in this project to organize the routing at client side. AngularUI Router is a routing framework for AngularJS, which allows to organize the parts of an interface into a state machine. Unlike the $route service in Angular core, which is organized around URL routes, UI-Router is organized around states, which may optionally have routes, as well as other behavior, attached. States are bound to named, nested and parallel views, allowing you to powerfully manage your application's interface.

$stateProvider is the service that is used by AngularUI Router to manage routing using the concept of *routes*. A state corresponds to a *place* in the application in terms of the overall UI and navigation. A state describes what the UI looks like and does at that place. The $stateProvider provides interfaces to declare these states for the application. $stateProvider has dependency on $urlRouterProvider, which has the responsibility of watching $location. When $location changes it runs through a list of rules one by one until a match is found. $urlRouterProvider is used behind the scenes anytime a url is specified in a state configuration.

5 Evaluation

In this chapter, the implemented prototype is evaluated based on the some pre-determined criteria. Certain hypotheses have been proposed about the behavior of prototype and deviations from desired behavior, both quantitative and qualitative, are explained in this section. Some evaluation scenarios are shortlisted and characteristics of each scenario are defined.

Since the aim of the problem statement was to provide more natural ways of communication, ability to use gestures becomes very important. Gestures can be characterized into three categories based on the way they can be used for performing physical tasks [2]:

- **Kinetic Gestures**: Kinetic gestures are used to mimic the interaction of users with the product. Certain gestures like circular motion can be used to describe any ongoing process. Figure 5.2 shows the example of using a kinetic gesture. As depicted clearly, here a group member is trying to illustrate something using his hand's motion.

- **Spatial Gestures**: Spatial gestures can be used to show spatial quantities like size of an object or distance between two objects. Spatial gestures could signify more abstract qualities like similarity or other distance based metaphor. An application of using spatial gestures in real world can be seen in figure **??**, where a user is using his hands to indicate the size and orientation of an object.

- **Pointing Gestures**: Pointing gestures are be used to refer to objects, persons, places or ideas. These can be used to point at some objects while performing physical tasks. Figure 5.3 depicts the use of pointing gesture in real world. Here a user is pointing to something on the paper to put more stress on a particular section of the paper.

5.1 Evaluation Strategies

In order to do the evaluation of the implemented prototype, several methods were used. It is very important to evaluate the performance of the prototype for a variety of tasks. On the other hand, it also becomes necessary to evaluate the usability or

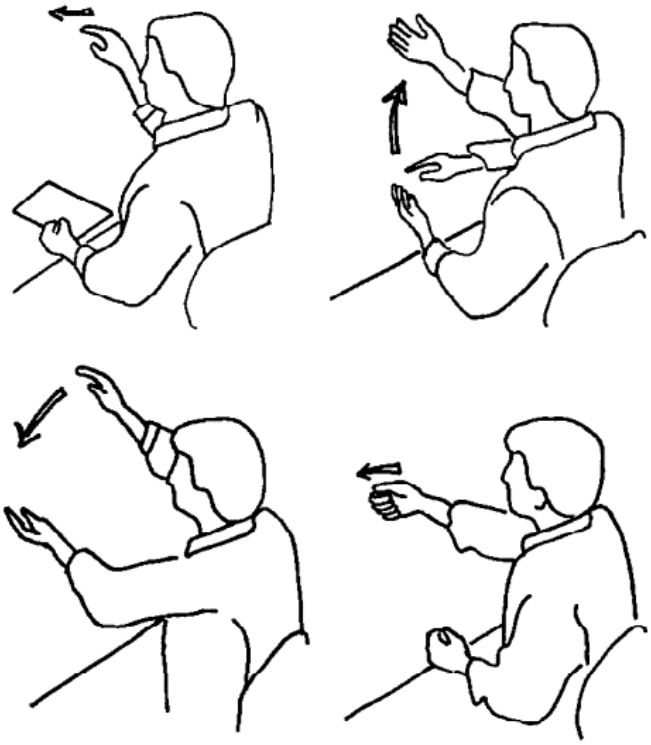


Figure 5.1: Kinetic Gesture illustrating hand's motion to give instruction [2]

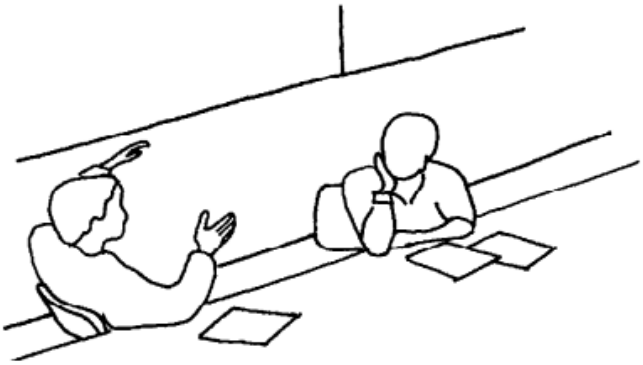


Figure 5.2: Spatial gestures, indicating the size and orientation of an object [2]

user friendliness of the tool. A tool with great features can not always do good in market until it is deemed usable by the customers. Therefore, for the evaluation phase of this thesis, certain evaluation scenarios were made in order to evaluate not just

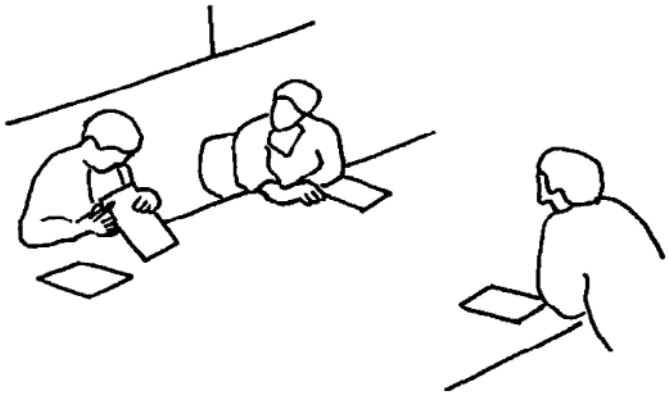


Figure 5.3: Pointing gesture being used to indicate a section of paper [2]

the performance of prototype, but also the usability and look & feel of it. Following sections describe those evaluation strategies followed during the evaluation phase of this thesis. All the processes followed for each strategy along with the experimental setup is defined in the following sections.

5.1.1 Experimental Observations

Based on the application areas of this application, it becomes essential to choose such scenarios for evaluation that have similarity in principles of usage. Since this work is focused on making it more convenient to use collaboration over video in performing physical tasks, it is important to define scenarios with physical tasks to do the evaluation.

For evaluating *Online Mode* of the application, a famous game known as **LEGO** was chosen. LEGO is a game in which several colored interlocking plastic bricks are provided. There are other various parts including plastic gears, figurines etc. These Lego pieces can be assembled and connected in many ways, to construct objects of various shapes and sizes. Anything constructed can then be taken apart again, and the pieces used to make other objects.

In the evaluation phase of *Online Mode*, assembling multiple Lego plastic blocks to form a certain structure was considered as one physical task. For each evaluation cycle, two users were chosen with one acting as an expert user and other as non-expert user. Several evaluation cycles were followed with a different user acting as non-expert each time. Expert user remained the same every time in order to ensure the same quality of instructions given in every evaluation cycle. Following sections describe multiple evaluation cycles and corresponding details along with observations.

Evaluation Cycle

In each evaluation cycle, three Lego structures were used for the evaluating the performance of non-expert user in performing the physical task. Evaluation cycle was divided into three parts, where in each part, a user in expert mode assists other non-expert user to assemble one Lego structure.

Figure 5.4: First Lego Structure for performance evaluation

After defining the scenarios, the implemented artifact is evaluated by comparing its performance in several modes which are already existing in market. Following comparison modes have been selected to test the performance of the implemented solution:

1. **Audio-Video only:** This part of evaluation represents the classical audio-video calls between two users. In the evaluation cycles, prototype's video chat module was used for establishing audio-video calls. The call duration, after the expert user started providing assistance, was recorded. Total time taken for assembling the right structure from scratch was noted and considered as the observed value for this phase.

2. **Audio-Video with Gestures enabled:** This part of evaluation focuses on the online module of the application developed in this thesis. Expert user provides assistance to the other user by using contextual assistance mode. He uses his hand gestures to collaborate more naturally with the remote user to build final

Figure 5.5: Second Lego Structure for performance evaluation

Lego structure. Since all three structures are of approximately same complexity, in terms of number of blocks and types of blocks, second structure is assembled in this phase.

3. **Physically co-present:** This is the third category of observations in the evaluation cycle. It was very straight-forward to implement. Both the users, expert and non-expert, were physically co-present in the same area. Expert user dictated the complete process of building the Lego structure with his voice and gestures. Since both the users were co-located and voice, therefore gestural communication was much more simple to realize. Time was recorded for which complete communication went on between both users for constructing the Lego structure. Besides using previously mentioned structures, a new random structure was also created to test this scenario. This structure as shown in figure 5.6 was chosen to be more complex so as to compensate the mutual understanding developed between peers during previous evaluation scenarios.

Same experimental setup with same set of Lego structures were used for each participant and same worker participants take assistance from each helper so that effects of individual differences in skill and conversational style is neutralized.

Hypothesis

Qualitative and quantitative data was used to evaluate the solution and investigate the hypothesis. Performance of each condition was measured by the average time taken

Figure 5.6: Second Lego Structure for performance evaluation

by experts to assist the workers and get the task done. Qualitative measure of performance was the quality of Lego blocks constructed in the end. Following predictions constituted the hypothesis during the evaluation of prototype:

1. It was predicted that task performance will be best in side-by-side (co-located) scenario because there is no bandwidth limitation and hence latency in communication is negligible. Experts can use most natural ways to collaboration.

2. Performance in audio-video only condition would be worst because of restricted gestural communication.

3. Performance in audio-video with gestures enabled condition would be intermediate because of added advantage of using some gestures while collaborating. In this way, the idea of implementing this prototype would prove beneficial because we would get much closer to using the natural ways of communication, rather than using audio-video only.

Procedure

Two rooms are reserved for carrying out the evaluation scenarios where participants are provided the instructions to be followed. One room is meant for workers to perform the task of constructing pre-determined Lego structure. Second room is for experts

from where an expert provides assistance to the worker working in the other room. Worker and expert are given a tablet which is connected to a wireless network and has the implemented application running on it. A worker's assistant is also present in the worker's room for capturing the start and finish of task and record the time taken to complete the task. Each participant labeled as worker does the Lego structure construction task with the help of expert in all three evaluation conditions- audio-video only, audio-video with gestures enabled, side-by-side presence. This procedure is repeated with other participants having the role of expert and average time taken for task completion is calculated for each evaluation condition. Quality of task completion is also taken into account as a qualitative measure of evaluation. During the evaluation procedure, following strategy was used to ensure fair and legitimate feedback process:

- *Survey Data*: A pre-session survey was done to collect the general information about participants like their age, gender, prior experience with using smart-phones/tablets, experience of being involved in performing physical task (Lego in this case). A survey was also conducted after the collaboration session in which feedback from participants was taken based on a specific questionnaire. Table 5.1 shows the pre-session survey results, which contain general information about the participants in evaluation. Participants were asked to rate themselves out of 5 for their proficiency/experience in building Lego structures. Similar ratings were asked from them regarding their proficiency with using smart-phones. It can be seen clearly in table 5.1 that most of the users were not very much familiar with building Lego structures. Although everyone rated them pretty good in using smart-phones, but it can be seen that most if the younger people rated themselves higher in smart-phones experience, as compared to users with more age.

Table 5.1: Pre-evaluation session survey

Users' Details				Users' Proficiency (out of 5)	
Sr. No.	User	Age	Gender	Lego	Smart-phones
1	Barbora	19	Female	4	5
2	Emilio	27	Male	1	4
3	Raman	31	Male	1	4
4	Aneesha	25	Female	2	5
5	Dikshit	25	Male	0	3

- *Performance Measures:* Performance measures such as task completion time was recorded ensuring the quality of block construction achieved.
- *Real-time Observations:* Overall qualitative observations of collaboration were

made while the task was being performed. Besides this, communication between expert and worker was also rated quantitatively on a scale of 5.

- *Video Recordings:* Every session of an expert and worker was recorded and analyzed later with respect to the collaboration rules, measures and quality. Some pre-defined rules for collaboration were checked and validated (e.g., finger pointing to indicate to a particular object, hand waving to indicate the step completion, etc.)

With multiple users asked to use the implemented prototype for constructing Lego, predefined set of steps were explained to them before start. Certain steps followed during this phase of evaluation are explained in the following section.

- **User login:** Both the users logged in with two different modes. One user with desktop logged in with *Expert mode* and the other user, who acted as non-expert used his/her mobile phone to log in to the application. See figure 5.7

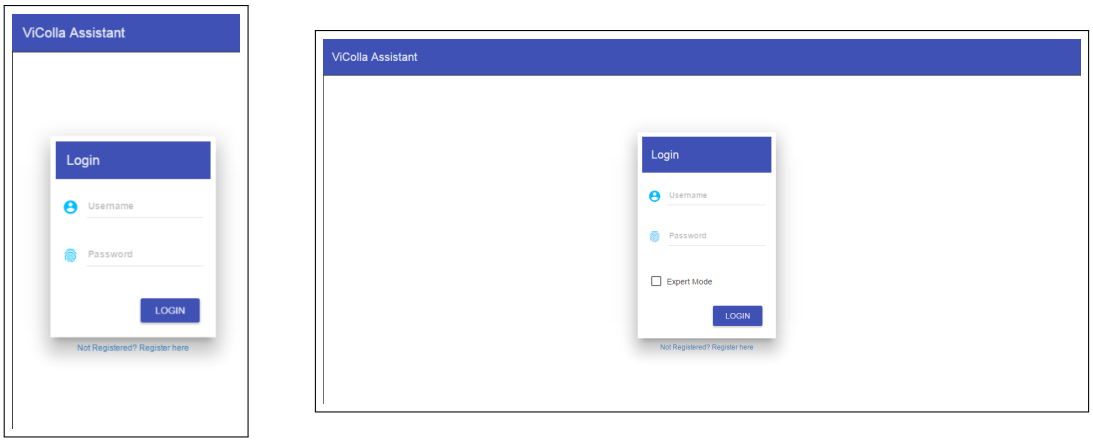


Figure 5.7: Login View

- **Find Expert:** After successfully logging in, the user in non-expert mode looked for a user who was expert in non-expert's desired field of interest. By switching to the *contacts* tab, he could see experts who were online at that moment. Since the user in expert mode had also logged in during our evaluation cycle, non-expert user could see him online. Figure 5.8 shows this step. Here, non-expert user Tim could see the expert user Andy online in his contacts tab.

- **Initiate Video Call:** User in non-expert mode initiated video call with expert user in order to get the contextual assistance for assembling the Lego structure. Expert user at the other end picked up the video call an started video conversation. One of the strategies of providing assistance was using audio-video call only.

Figure 5.8: Tim finds expert user (Andy) online at the moment

So with this video chat module, the evaluation for *audio-video only* was done. Table 5.2 illustrates some of the pictures taken during video call sessions between different users. The observations regarding time duration taken for building Lego structure is listed in the table 5.4.

- **Start Contextual Assistance:** Now, for this evaluation, we assume that expert user has all the steps to assemble the Lego blocks to construct the final structure. Since user in expert mode has the privilege to start the contextual assistance, he initiated it to start the performance evaluation of this application. Then, the time duration for which this contextual assistance went in progress was recorded as the observation for an evaluation cycle. Figure 5.3 represents some of the pictures taken during different evaluation rounds held between different users.

Table 5.2: Expert using video call for assisting remote user to build Lego structure

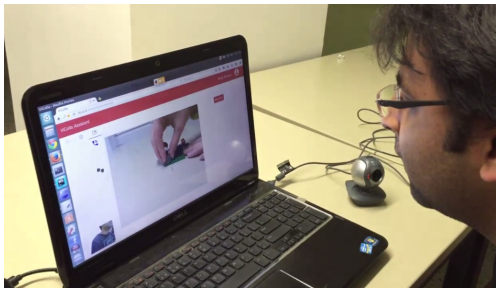
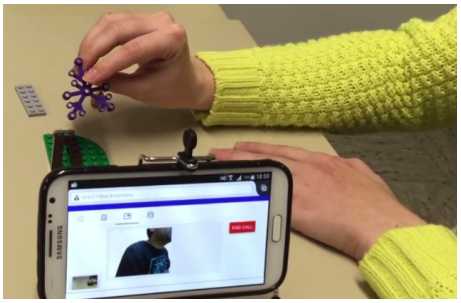
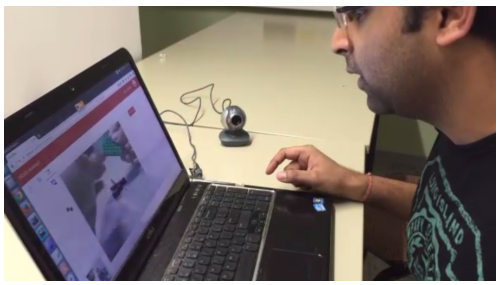
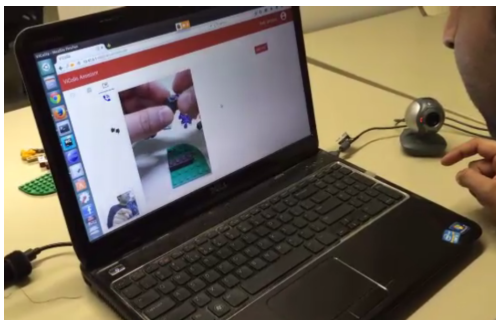
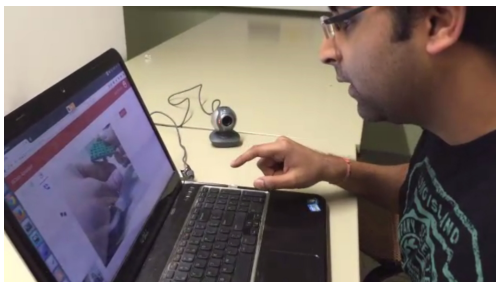
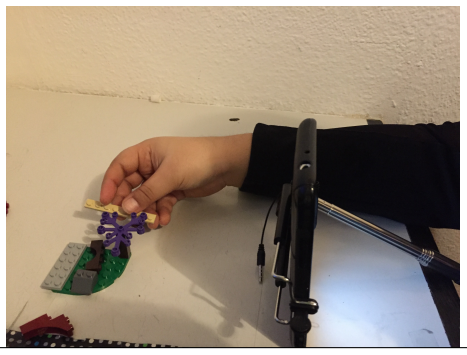
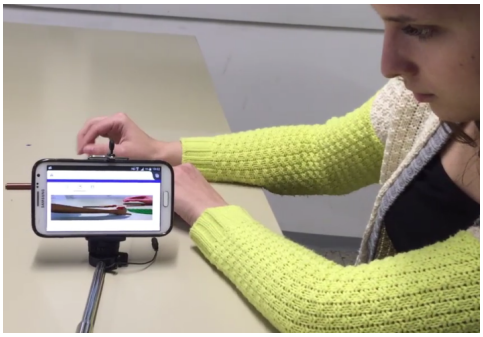
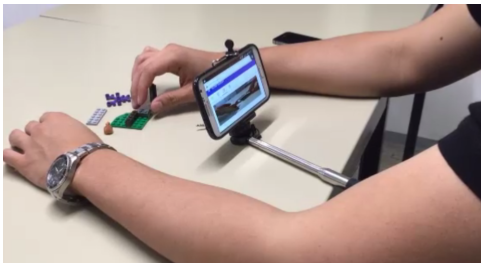
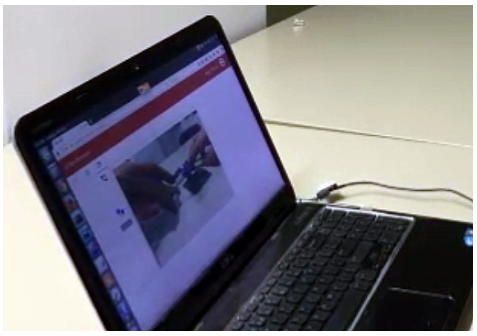
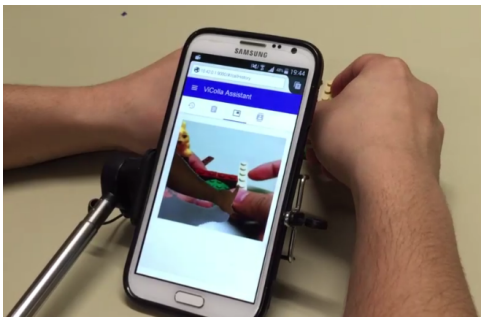
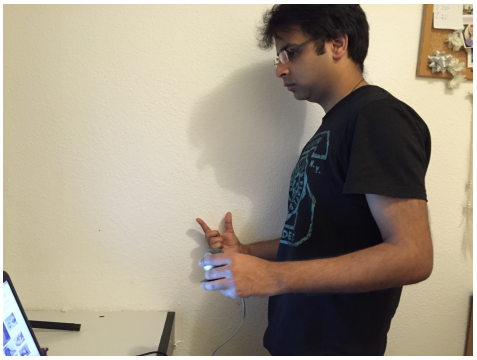
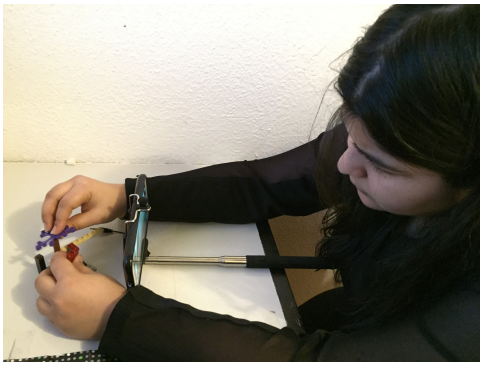
Expert	Worker
	
	
	
	

Table 5.3: Expert using contextual assistance with hand gestures to help other user in building Lego structure

Following table 5.4 represents the outcome of experimental phase done to evaluate the prototype. The table represents recorded observations for all three scenarios in which users were asked to build the Lego structures with the help of remote expert user.

Recorded Time				
Round No.	User	Audio-Video Only	With Gestures	Face-to-face
1	Barbora	04:18	02:02	01:15
2	Emilio	02.18	01:50	00:51
3	Raman	02.52	01:22	00:57
4	Aneesha	03.13	01:54	00:49
5	Dikshit	03.29	02:10	00:56

Table 5.4: Observations from Experimental Evaluation Session

5.1.2 Feedback from stakeholders

Taking feedback from various stakeholders was a vital source of evaluating the prototype throughout the course of this thesis, which led to consistent improvement of the project developed. Several steps were involved in achieving this feedback. The approach followed to accomplish this has been explained in the following points:

1. **Proof of concept:** During the phase, when it is not clear if the final product would prove to be useful enough, its is always better to start with implementing a proof of concept. Therefore, a small prototype was implemented in the beginning which was sufficient enough to showcase the final aim of the thesis. That initial prototype had two distinct parts, one depicting the real time contextual assistance, and the other illustrating the use of annotation tools on live video.

2. **Verified applicability of proof of concept:** Before approaching the stakeholders, multiple series of tests were conducted in order to visualize the usability and applicability of the initial prototype. In these tests, a test scenario for operating a elevator at *Department of Informatics and Mathematics* was chosen. Two users with defined user role as an expert and non-expert used the initial prototype to accomplish the task.

3. **Initial test session recorded:** The complete test session of both users trying to solve the problem statement of operating *elevator* was recorded. Figure 5.9 shows the snapshot of the video recorded while the evaluation task was in progress. As seen in the snapshot, here the expert user (in the right) was assisting the non-expert user (in the left) in operating the elevator using the initial

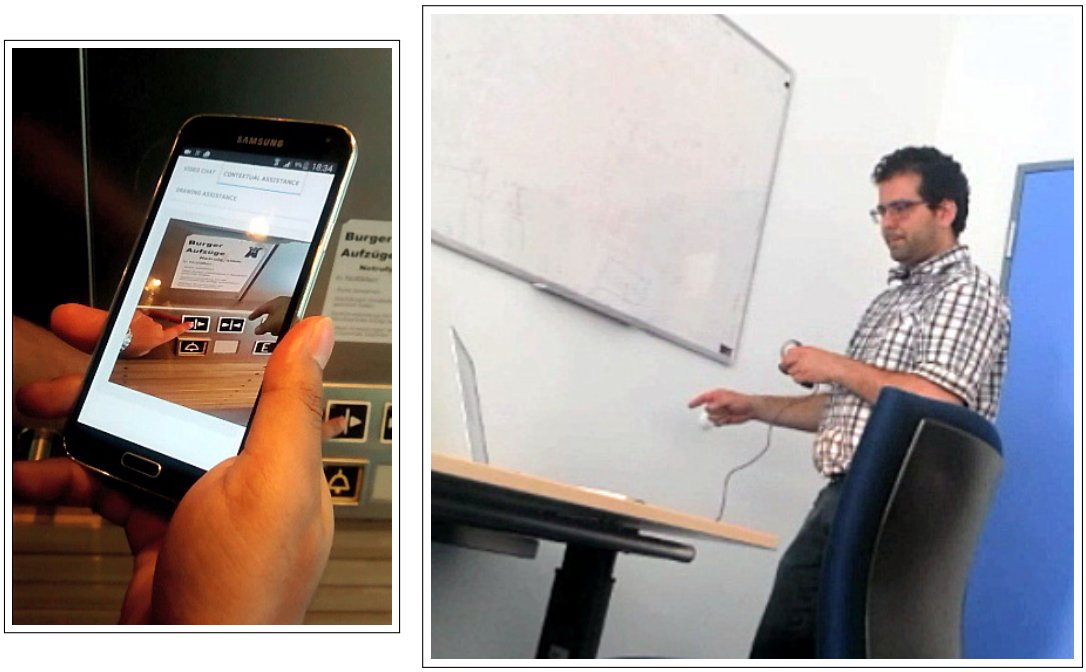


Figure 5.9: Evaluating proof of concept

prototype built to verify the proof of concept. Expert user used his hand gestures to point at various button present in the elevator panel. Non-expert user followed the instructions and the test was successfully accomplished.

4. **Initial Feedback:** The test session mentioned above was recorded from both ends - expert as well as non-expert. Both the recorded videos were combined and a movie with a storyline was created to make different stakeholders understand the concept and its applications. The movie depicting the usage model was presented to stakeholders at *Siemens* and their feedback regarding the prototype was collected. Feedback came out to be very positive for this idea and based on the improvements suggested during the evaluation of prototype, it was decided to extend it further to make it a full fledged application.

5. **Final test session recorded:** Based on the initial feedback it was realized that a user should be able to act in two different user roles. After few months of further prototype development, two application views/modes were made. A desktop view, representing the stable environment, was released for the users in expert role. Another view for mobile devices was released to be used by non-expert users, who work in dynamic and mobile field environment. After this implementation was complete, feedback cycle from stakeholders was repeated. A test session was

recorded again with newly implemented prototype. This time, a separate module for *offline support* was also added to the prototype. Both *online module* as well as *offline module* were tested and recorded during the test session. Table 5.5 illustrates the pictures of newly added *offline support module* under test session.

Table 5.5: Offline support module under test

Here a scenario for testing the *offline supprt module* was created. Non-expert user was supposed to assemble the components required to setup the router connection at work. Since the user did not have knowledge about setting up the router, he recorded the video of the work environment (as depicted in first picture of table 5.5). This video was then shared with the expert user in router set-up over Internet. Expert user annotated the received video of other user's work environment. He provided several drawings to indicate how the router can be made ready to user in the work environment. Certain annotations can be seen marked in the third figure of table 5.5. In the last step of test session, non-expert user again opened the same recorded video, but this time he saw all the annotations provided by expert to assist him in setting up the router. It can be seen clearly in the last picture of table 5.5 that non-expert user assembled the whole setup by looking at the annotations provided by expert user. Non-expert user used his smart-phone to play the video in *offline support module* to see the annotations. Expert user also used *offline support module* to annotate the video provided by

non-expert user.

6. **Final Feedback:** The final test session described above was recorded from both ends - expert as well as non-expert. Again a movie with a storyline was created to demonstrate the working and usage of this application to different stakeholders. For getting the feedback from stakeholders, a presentation was given describing the new additions made to the initial prototype. A live demonstration of the application was shown in which both *online support module* as well as *offline support module* were shown working on particular user cases. After the live demonstration, the movie depicting the usage model was presented to stakeholders at *Siemens* and their feedback regarding the prototype was collected.

5.1.3 Feedback from users

Besides doing the experimental observations and taking the feedback from stakeholders, it is always essential to take the feedback from users who would might be the ultimate user of this application. These users should not be directly related to the initiation, implementation or verification of the project. Therefore, in this thesis multiple users were approached for the evaluation. A demonstration as well as the movie prepared for stakeholders was shown to them. They were asked to use the prototype for few minutes so that they could form a general perception about the application and its possible use cases. Different measures, as mentioned in the following section, were considered while collecting their feedback about the prototype:

- **Feasibility:** It is very important to do the feasibility study of an application. Besides providing useful features, it should also provide usable features. For feasibility study, users were asked if they think the features provided by the application seem usable and feasible to them.

- **Features Coverage:** Users were asked about the kind of features they expect from such an application. They were asked to rate the application and provide their feedback based on the range of features it provides. It was analyzed later that, whether or not the number of features are enough to establish a productive session of collaboration over video.

- **User Friendliness:** Last parameter of judgment was the overall look and feel of the application. Users were asked to provide their feedback based on the their experience with complexity of the user interface of the application. Several critical points pertaining to improving the user experience were noted.

Users were then asked to compare the implemented prototype with other famous tools already available in market. *Online support module's* features were compared with the

features provided by tools such as Skype or Google Hangout. *Offline annotation model* was compared with *YouTube* annotation model.

6 Conclusion & Future Work

This chapter illustrates the collective results and corresponding inferences from end of the research effort of this thesis. All the learned facts are summarized and some deviations of evaluation of results from expected results are reasoned. Conclusion is drawn by considering different measures as mentioned below.

6.1 Conclusion drawn from Experimental Observations

For inferring the experimental observations, results are presented in three parts here: First the impact of different collaboration conditions (audio-video only, audio-video along with gestures, side-by-side presence) on task performance is discussed. Then Real-time observations and video recordings interpretations are explained. Finally the conclusion of relationship between collaboration condition and corresponding effort applied for communication is analyzed.

- **Impact of different collaboration conditions:** The observations made from 5 rounds of experimental sessions of evaluation with 5 different users are concluded here. According to the time duration for each evaluation session taken by each user, it gives a clear indication that performance of collaboration over video was improved drastically when gestures enabled communication came into action. As stated clearly in observations table 5.4, video collaboration with *Audio-Video only* performed worst among all three experimental conditions. For all the users, enabling the gestural communication using the *Contextual Assistance module* provided by our application, proved to be a very good improvement. Every user was able to build the Lego structure much faster compared to the mode with *audio-video only*.

  Average time taken by a user to build Lego structure using *gestural communication* was 1 minute 56 seconds. This is a huge improvement compared to the average time taken by a user in *Audio-Video only* mode, which was 3 minutes 14 seconds. Therefore, results predicted in the hypothesis for the evaluation phase were correct. Direct face-to-face communication, as expected, took the least amount of time for collaboration. The reason, as explained in the hypothesis, was the fact that users could use all natural ways of communication without

any latency. In addition they were in same real environment instead of sharing a virtual environment for collaboration. So the conclusion drawn from these observations is very straight and clear that addition of gestures to collaboration makes it much easier to perform a complicated physical task, and the prototype provided as a result of this thesis is very much capable of achieving this enhanced performance.

A bar graph 6.1 is plotted from the data readings gathered from evaluation phase. The graph project the observations in number of seconds each user took to build Lego structure in three different collaboration conditions. The results in this graph are directly inferred from table 5.4.

- **Real-time observations and video recordings interpretations:** Throughout the course of video call for achieving the physical task of constructing Lego structure, several observations were made regarding details of procedure followed and behavior of peers.
  - It was noticed that besides gestural communication, audio had a significant impact on the performance of structure construction. In order to verify this, few tests were made without audio, with just gestures enabled. And it was seen that there was a significant drop in the average time required for accomplishing task.
  - Using a stable holder for smart-phone also increased the performance because non-expert user could use both his hands. In addition this provided a temporary stability to the non-expert's working environment.
  - Using the smart-phone in portrait or landscape orientation also resulted in difference in performance. This brought out a small design issue in the application. Video of expert and his gestures were seen little stretched in landscape orientation of phone which made it little confusing for non-expert user to determine the exact point of focus suggested by expert user. But despite this, the overall performance was still much better compared to the mode where no gesture enabled communication was used.
  - After going through the video recordings of all the evaluation sessions, it could be inferred that same expert user adapts to the work environment of non-expert user and becomes more comfortable with the way assistance should be provided.
- **Relationship between collaboration condition and effort required for communication:** This is a very crucial conclusion drawn by expert user from his observations during multiple evaluation rounds. For the collaboration condition where *only Audio-Video mode* was used, it was observed that expert user faced

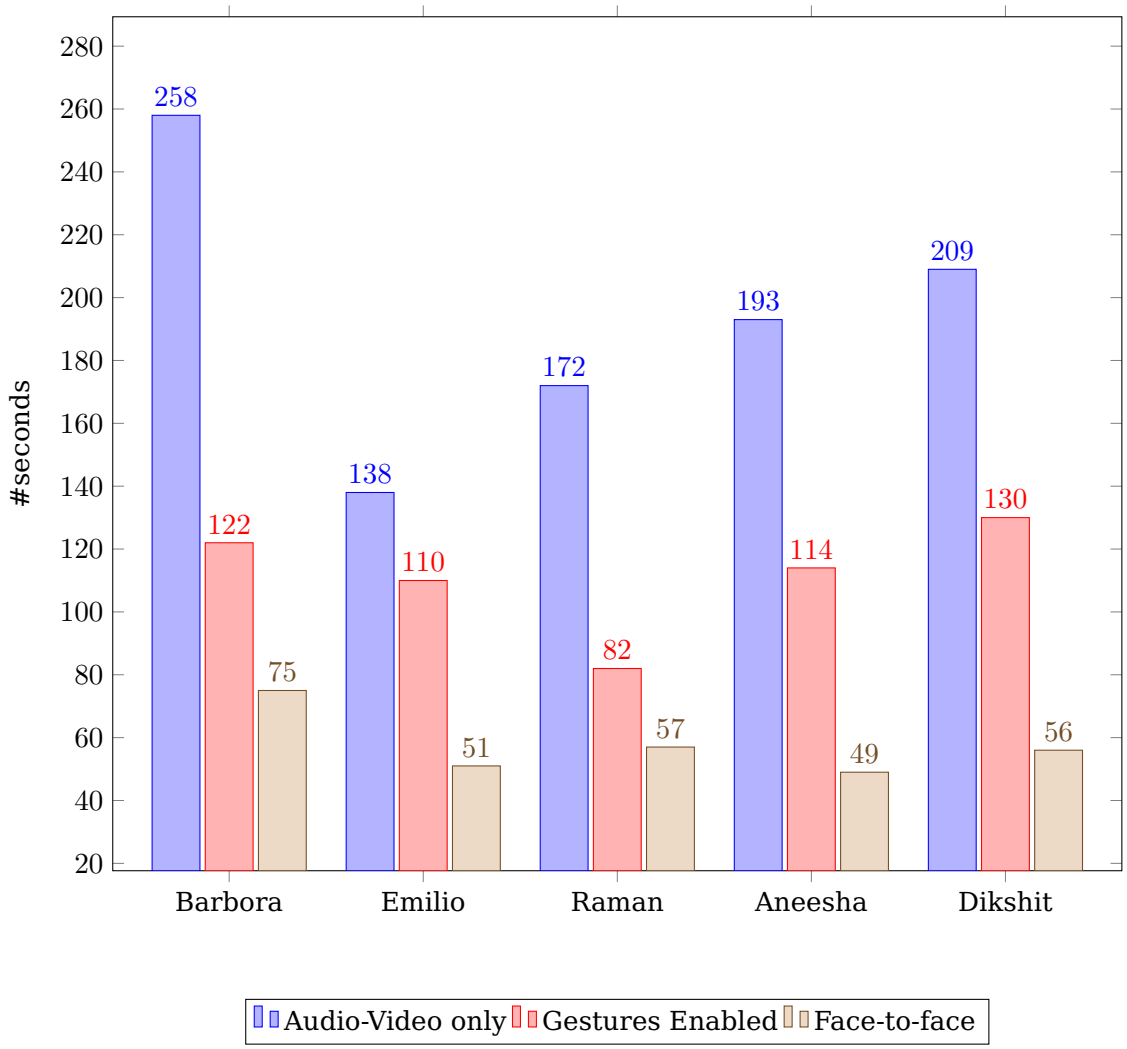


Figure 6.1: Comparison of three different collaboration conditions based on physical task performance achieved by users

great difficulties for explaining specific details about an object in non-expert's work environment. Parts of symmetrical objects were very hard to explain. Moreover, elaborating non-expert user about directions, angles or distance was also very tedious for expert user. This was the main reason behind the worst performance results of *Audio-Video only* mode. This drawback was very nicely overcome by enabling gestures within audio-video communication. In this case, it was very simple for expert user to elaborate directions and distance by utilizing kinetic gestures or spatial gestures.

6.2 Conclusion drawn from feedback taken from users

As stated in chapter 5, several users were asked to use the prototype developed and provide corresponding feedback. They were asked to compare *Online support module* of this application with tools like *Skype* or *Google Hangout*, and *Offline support module* with *YouTube* annotation model. Based on the feedback taken from users, following conclusions were drawn:

- Based on **feasibility** as criteria for judgment, all the users felt that there is a need for such a tool in market. They agreed to the fact that performing physical tasks have always been tedious and remote collaboration over video would definitely help in solving those purposes. On comparing feasibility of *Online Support module* of the prototype with existing tools, following comments were made. Some users felt that Although the prototype provides unique feature of using gestures, but some issues related to latency were observed. After trying the tool for longer call durations, effect of drawing video frames over canvas could be seen. This deteriorated the performance to a little extent. Use of hand gestures also added complexity of using a compulsory USB web camera because hand segmentation from the live video is a major part of the algorithm. And built-in camera of a laptop does not enough flexibility to adjust the position of the expert user.

- Most of the users were very satisfied with the **feature coverage** of the prototype. According to them, the features in *Online support module* can not be matched with any existing tools in market because those tools do not provide such features at all. Furthermore, no such tools are available in market which provide offline annotation model. *YouTube* can not be used without the use of Internet at all, and it does not give the facility to record or save the complete video. In this way, *Offline Support Module* was a great addition to the success of this prototype among users. Only feasible feature addition that they could suggest was allowing users to edit, undo and clear the annotations created. The users felt that this feature would have been implemented and marked it as a crucial missing feature for the prototype.

- Considering the **User friendliness** of the prototype, users were very amazed with the user interface. It looked pretty simple and straightforward to use. They liked the way responsive layout was working for different sizes of devices. One of the improvements they could suggest was handling video canvas properly with change in orientation of mobile device screen. Since all of them experienced that video appears stretched in landscape orientation, they made a note to provide it as feedback. Additional difficulties that a lot of users faced were related to shadow of expert's arms or hands coming in the video, and prototype's inability

to segment shadows in a better manner.

6.3 Future Work

Providing remote collaboration over video is a progressive field and has been center of research for more than two decades now. The research strategy followed in this thesis and the prototype implemented as a result is a step ahead towards the achievement of easy and natural ways of collaboration. Although the results inferred from evaluation of the prototype have been satisfactory but there is always a scope of improvement in such as progressive area. This section provides some details about how the prototype can be improved and extended so as to make it more efficient and effective. Following are some of the points which can be considered as great extensions to this provided prototype for remote collaboration over video.

- **Download Annotations:** The *Offline support module* of this prototype allows a non-expert user to fetch all the annotations for a video from a central database whenever he gets connected to Internet. But at the present version, application does not allow a user to store those annotations permanently. In order to make the annotations available for use any time, it is recommended to provide a feature to download the annotations corresponding to a video.

- **Distinguish annotations based on name of expert users:** Current version of the prototype provided does not distinguish annotations created by multiple expert users for a single video. All the annotations are seen collectively irrespective of the user who created them. Extension to this feature would be to visually mark annotations provided by different expert users so as to distinguish each expert's annotations. Users would then be able to see which user has created corresponding annotations.

- **Non-experts to draw annotations:** Annotations defined so far are the marks or drawings which are created by a user in expert mode, in order to assist a remote user in performing a task. The user in the non-expert mode does not have the privilege to draw or mark anything on video. Thus, the right to annotate a video is limited only to users in expert mode. But there might be situations where non-expert users would like to make expert user to emphasize on small section of the video recorded by them. Providing drawing annotation rights to users in non-expert mode can be a solution to this. By enabling feature of drawing annotations for non-expert users would enable them to ask for assistance in a more specific manner.

- **Improve Hand Segmentation:** As observed in some of the evaluation cycles, the hand segmentation algorithm does not perform good for different types of backgrounds. In many real world scenarios, it is not possible every time to restrict an expert user to use a white/gray wall as the background while providing contextual assistance. Several other strategies were tried, such as Chroma Key configurations, but that also could not perform splendidly well. Addition of several other constraints like improper lighting, angle at which light falls on the hand, etc. also enhances complexity of the real world situations. Therefore, it would be a great improvement to this prototype if a more sophisticated algorithm is implemented for segmentation of hands from unnecessary background.

- **Network Configurations:** During the evaluation phase of the prototype, it was observed that functioning of the *Online support module* is vulnerable to different network configurations. Since configuration objects used in WebRTC connection are subject to change for different browsers in future, the video call feature might face issues while getting connected in different network conditions. For example, the current implemented prototype does not work in the network which allocates Internet Protocol version 6 (IPv6) addresses to the device running this application. This difficulty was faced during the implementation of this prototype as well. Although the application works fine for network providing IPv4 addresses only, but it should be made compatible with network allocating IPv6 addresses also.

- **Improve Robustness:** It is always important to make an application more robust to different environment changes. It should also be able to handle all kinds of user interactions resulting in a stable state. Since the project was done in a limited time frame, it was not possible to test every functionality in a detailed manner. Therefore, the application can be made more robust by adding automated test cases for features at different levels.

List of Abbreviations

AJAX	Asynchronous JavaScript and XML.
API	Application Program Interface.
BSON	Binary JSON.
HTTP	Hypertext Transfer Protocol.
ICE	Interactive Connectivity Establishment.
IPv6	Internet Protocol version 6.
JSON	JavaScript Object Notation.
JSONP	JavaScript Object Notation with Padding.
MD5	Message Digest algorithm 5.
MEAN	MongoDB Express.js Angular.js Node.js.
MVC	Model View Controller.
NAT	Network Address Translation.
NoSQL	non-Structured Query Language.
NPM	Node Package Manager.
ORMs	Object-Relational Mappers.
REST	Representational State Transfer.
RTC	Real-Time Communication.
SDP	Session Description Packet.
SPA	Single Page Applications.
SQL	Structured Query Language.
STUN	Session Traversal Utilities for NAT.
TURN	Traversal Using Relay NAT.

WACL	Wearable Active Camera with a Laser pointer.
WebRTC	Web Real-Time Communication.

Bibliography

[1] Mark Apperley and Masood Masoodian. Supporting collaboration and engagement using a whiteboard-like display. 2000.

[2] Mathilde M Bekker, Judith S Olson, and Gary M Olson. Analysis of gestures in face-to-face design teams provides guidance for how to use groupware in design. In *Proceedings of the 1st conference on Designing interactive systems: processes, practices, methods, & techniques*, pages 157–166. ACM, 1995.

[3] David Cosgrove. Smartpark. 2013.

[4] André Cruz. Js-spark-md5. 2015.

[5] Jeff Dickey. *Write Modern Web Apps with the MEAN Stack: Mongo, Express, AngularJS, and Node. js*. Peachpit Press, 2014.

[6] Tom Finholt, Lee Sproull, and Sara Kiesler. Communication and performance in ad hoc task groups. *Intellectual teamwork: Social and technological foundations of cooperative work*, pages 291–325, 1990.

[7] Susan R Fussell, Robert E Kraut, and Jane Siegel. Coordination of communication: Effects of shared visual context on collaborative work. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 21–30. ACM, 2000.

[8] Susan R Fussell, Leslie D Setlock, Jie Yang, Jiazhi Ou, Elizabeth Mauer, and Adam DI Kramer. Gestures over video streams to support remote collaboration on physical tasks. *Human-Computer Interaction*, 19(3):273–309, 2004.

[9] Karl Gyllstrom and David Stotts. Facetop: Integrated semi-transparent video for enhanced natural pointing in shared screen collaboration. *May*, 15:1–10, 2005.

[10] Jared Hanson. Passport: Simple, unobtrusive authentication for node.js.

[11] Michael Herman. User authentication with passport and express 4. 2015.

[12] Weidong Huang, Leila Alem, and Jalal Albasri. Handsinair: a wearable system for remote collaboration. *arXiv preprint arXiv:1112.1742*, 2011.

[13] Ellen A Isaacs and John C Tang. What video can and cannot do for collaboration: a case study. *Multimedia Systems*, 2(2):63–73, 1994.

[14] Hiroshi Ishii and Minoru Kobayashi. Clearboard: a seamless medium for shared drawing and conversation with eye contact. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 525–532. ACM, 1992.

[15] Robert Johanson. Teleconferencing and beyond, 1984.

[16] Robert E Kraut, Mark D Miller, and Jane Siegel. Collaboration in performance of physical tasks: Effects on outcomes and communication. In *Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 57–66. ACM, 1996.

[17] LearnBoost. Mongoose: elegant mongodb object modeling for node.js. 2010.

[18] Bonnie A Nardi, Heinrich Schwarz, Allan Kuchinsky, Robert Leichner, Steve Whittaker, and Robert Sclabassi. Turning away from talking heads: The use of video-as-data in neurosurgery. In *Proceedings of the INTERACT'93 and CHI'93 Conference on Human Factors in Computing Systems*, pages 327–334. ACM, 1993.

[19] Thomas Nescher and Andreas Kunz. An interactive whiteboard for immersive telecollaboration. *The visual computer*, 27(4):311–320, 2011.

[20] Jiazhi Ou, Susan R Fussell, Xilin Chen, Leslie D Setlock, and Jie Yang. Gestural communication over video stream: supporting multimodal interaction for remote collaborative physical tasks. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 242–249. ACM, 2003.

[21] Patricia Sachs. Transforming work: collaboration, learning, and design. *Communications of the ACM*, 38(9):36–44, 1995.

[22] Nobuchika Sakata, Takeshi Kurata, Takekazu Kato, Masakatsu Kourogi, and Hideaki Kuzuoka. Wacl: Supporting telecommunications using wearable active camera with laser pointer. In *null*, page 53. IEEE, 2003.

[23] Anthony Tang, Carman Neustaedter, and Saul Greenberg. Videoarms: embodiments for mixed presence groupware. In *People and Computers*, pages 85–102. Springer, 2007.

[24] John C Tang and Scott Minneman. Videowhiteboard: video shadows to support remote collaboration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 315–322. ACM, 1991.

[25] John C Tang and Scott L Minneman. Videodraw: a video interface for collaborative drawing. *ACM Transactions on Information Systems (TOIS)*, 9(2):170–184, 1991.

[26] tutorialsPoint. tutorialspoint: Simply easy learning.

[27] R Hevner von Alan, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, 28(1):75–105, 2004.

[28] Vassiliy Zhovner. Cloud services design. 2013.

[29] Zqdevres. Nodejs architecture map. 2013-06-14.