Video conferencing

Jonathan Kasongo

OCR A-level Computer Science NEA

Full name: Jonathan Kasongo
Candidate number: N/A
Centre number: N/A
Centre name: Harris Academy Purley
Qualification code: H446

Contents

Chapter 1

Analysis

1.1 Problem identification

1.1.1 Context

My client Axel Alabi has asked me to create an interactive video conferencing application to allow others to view talks in realtime. The current solution is to use the Zoom video conferencing application. While it is true that the application is technically sound and can work fine, there is a large number of elderly users that also try to connect to the conferences. These users often don't fully understand how to correctly use the application and then end up accidentally disturbing the conference/talk [1], by leaving their microphone's on, accidentally raising their hands and so on. This makes my client's job difficult since he is in charge of managing the Zoom call. To combat this situation he would like a simple and user friendly video conferencing application that provides the features needed for people to view and interact with the conferences in real time. This includes features like (but not limited to) audience participation, the ability to speak to others via one's microphone and the ability to vote on polls. The application should be created specifically to help elderly people have a better experience whilst watching any conferences, so may also include extra accessibility features to ensure comfortable viewing for all, irrespective of one's age and/or disabilities.

1.1.2 Stakeholders

Stakeholder:	Axel Alabi
Category:	Client

Description:
Axel Alabi is a 22 year old male, and is currently in charge of managing the video broadcasts for conferences. He also works as a data analyst for a company specialising in analysing geographical data, and has experience working in computer science. Unfortunately, managing the broadcasts has become quite challenging because there is often a number of elderly people who join the broadcast and find difficulty in interacting with the broadcast. Axel would use the proposed solution as a replacement for the Zoom videoconferences he uses currently. When asked to manage/set up a videoconference by his company he would go onto his web browser onto my website, and begin a conference. He would be given a unique entrance code and then he would give this code to the people who are to be invited to the videoconference. The solution would provide a simpler, more intuitive and more accessible alternative to Zoom. The proposed solution would be appropiate to Axel and his needs because the new system will be easier to use for his audience and consequently he will have less trouble in managing the call. Instead of him having to worry about muting the audience's microphone's and responding to any technical difficulties/issues, he would be able to run the conference smoothly. This would improve his life greatly as he would be able to now focus solely on ensuring that video and audio is clear during the broadcast. In removing some of Axel's issues that he faces in his job, he should become more content and calmer with his work life, adequately showing how this solution will be appropiate to his needs.

[1] From this point forward we will avoid using "conference/talk", and simply replace it with "conference".

Stakeholder:	People aged ∼50 and over, with limited experience working with technology
Category:	Target users/audience

Description:
This group of users typically have limited experience working with technology. This will be my application's target audience, so it is important that the final solution is appropiate to their needs. These people will have spent most of their life without technology so figuring out how to connect to and properly interact in an online video conference may prove to be difficult for them. These users will use the final application to join video conferences they were invited to, and interact with them in a manner that is convienient to them. Furthermore they may use the application to create macros enabling them to perform common tasks easily and efficiently, like muting/unmuting their microphone. These configurations will be saved to their account and will be applied whenever they log onto their account even if they're on a different device. The solution will benefit them greatly as they will now no longer have to ask for help from others to resolve issues with their video conferencing application. Rather they will be able to experience a simple, easy to work with and intuitive system for their video conferencing. Moreover the users will now gain the oppurtunity to expand their social lifes as they will be able to chat with their loved ones and friends over their web browsers easily. Not only will the final solution make their lives more convinient but it will also provide the user with the oppurtunity to improve their mental health through socialising and engaging in conversation with others. [3]

Stakeholder:	IT Staff
Category:	Support/Maintainers

Description:
The IT Staff would be experienced in working with technology because of their qualifications in this field. These people would be expected to have a degree in computer science, mathematics or another closely related field. They should also be expected to have significant experience in working in various programming languages. This group of users would be expected to be able to update and maintain the system as required. The staff will use the documentation provided along with the application in order to understand what each function and class does along with it's purpose. Furthermore the clear and readable code will enable them to perform any necessary changes with ease, something they could not have done previously with the off the shelf software they had before. The solution will be appropiate to their needs as they will be able to access the source code of the final application and change the application to be tailored to work well for their needs. Additionally users who enquire about the security of their data when using the application will be able to check how the application handles their data themselves. This means that the IT staff will not have to be unsure about answering user's queries, but rather they will be able to read the source code and give a correct response to the user every time.

1.1.3 Research

We begin our research by observing solutions to similar problems. Then justifications of suitable approaches are given based on the existing solutions.

Zoom

Accessibility features

Zoom is a popular closed-source video chat application, developed by Zoom video communications. From their website [6] they implement 5 major features to ensure that *"Zoom is for everyone"*.

- Live transcriptions
- Automatic closed captioning
- Customisation of font sizes
- Keyboard shortcuts
- Screen reader support

Some advantages of this software include the fact that there is a good amount of accessibility features able to help a wide variety of people. For instance for those who have limited mobility keyboard shortcuts can be set up to conveniently perform common tasks, whilst those who have trouble hearing well can enable closed captioning during the meeting.

However there are still a number of limitations with the way these features are implemented. Automatically generated closed captioning is unfortunately only available in english, and may have varying levels of accuracy depending on external factors such as background noise, speaker's clarity and profiency in spoken english.

Zoom proposes a solution to the problem of exculsivity when in the context of video conferencing applications. Instead of only designing the application to be usable for one group of people they aim to instead tailor it to cater towards *everyone*. I believe that the wide range of accessibility features is a big advantage of the system, and this could motivate the descision to implement a similar set of features in my application. However while a good number of features is appreciable, it is also necessary to ensure that these features are simple to find and to use for an optimal user experience. This is justified by the fact that my client is specifically requesting an application for those who aren't comfortable with modern technology. To solve this problem I could perhaps implement a built-in tutorial that demonstrates how to use the accessibility features to teach the user how they can use the application to it's fullest potential.

Skype

Accessibility features

Skype is a proprietary messaging and video chat application developed by Skype technologies a subsidiary of Microsoft [5]. From Microsoft's support webpage Skype for Windows 8 and above has the 3 following key accessibility features.

- Narrator screen reader
- High contrast colour settings
- Magnifier

These features have the advantage of being especially accessible for blind people or for those with low vision. The combination of high contrast colours along with a magnifier and/or screen reader ensures that low vision users can still make use of Skype, independently. Skype also doubles up to be an instant messaging platform, permitting users to have all their conversations and other communications in 1 place.

Whilst the features mentioned are very beneficial for those with low vision, there is no true support for people who are hard of hearing or have low mobility. This set of features is one-dimensional only catering to 1 group of less abled people.

Skype proposes a solution to *"help people with disabilities navigate and control their device as well as get better access to online content."* [2] The solution from Skype allows for disabled ones especially those with reduced vision to benefit from Skype the most, as previously discussed. Whilst Skype offers a good set of features for the disabled, the features that they mention on their website are all already either implemented on most popular operating systems, or can be installed as browser extensions with a few clicks. Therefore I believe that there is insufficient justificaiton to take the time to implement any of the features that Skype implements, and will not be implementing any of those features.

To obtain a better understanding of the nature of the problem and what features should be implemented in the final solution, I decided to collect some qualititative data through an interview with my client. This data should allow me to have an insight into what the final solution should look like based on my client's requirements and desires respectively.

[2] Quote from https://support.microsoft.com/en-gb/skype/what-accessibility-features-are-available-for-skype-89c34c52-f463-437a-b3be-2ea114c5de13

Interview with Axel Alabi

Date: `29/06/24` **Time:** `3.50pm`

Q: What are some of the limitations of the current system used for video conferencing?
A: It tends to be difficult for people who aren't experienced with technology to properly interact in the conferences. Often times participants will accidently turn their microphone on or are unable to turn their microphone on when the speaker invites them to.

Q: What are some essential features that should be required in the final application?
A: Well to start the app should allow users to see and hear one another in real-time, there should be a focus on simplicity and users should be able to raise their virtual "hand" to interact with the talk.

Q: What are some non-essential features that would be desirable in the final application?
A: The app could perhaps provide a suite of accessibility features to allow disabled ones to have a comfortable viewing experience. This may include closed captioning, volume control and a screen reader.

Q: What operating system should the application be designed for?
A: There is no preference for operating systems.

Q: What are the software requirements?
A: It should be a web-based application. Any suitable mainstream programming language is fine as long as the code is clear enough for me and the other IT staff to understand.

Q: What are the security requirements?
A: There should be some form of end to end encryption to ensure that hackers or others cannot access the video feeds. There should also be some kind of username and password system in order to enter a call. Passwords should also be of a good strength e.g. at least 1 symbol, capital and lowercase letters.

Q: How will the new system benefit you?
A: This new system will ensure that all video conferences I am in charge of managing will run much smoother, not only giving me more time to work on other essential tasks but also providing a better viewing experience for all.

From this interview, I can see that my client has some features that should definitely be included in the solution: real-time audio and video as well as the ability to raise and lower one's virtual hand. Furthermore I believe that security should also be very important when designing the final solution. This should be a requirement because we don't want any of our users to have their important data stolen during their video conference.

Bearing this in mind we should also think about suitable approaches to implementing a system with these features. In order to implement the real-time video/audio I could make use of the WebRTC API from Google, which would enable me to establish secure peer to peer connections from the browser. Not only does this approach allow us to establish audio and video, but it also ensures that these connections are secure through the use of signalling servers. Signalling servers are servers that manage connections between peers. To understand signalling servers we will go through an example. Consider 2 users, Alice and Bob that want to have a video call. Alice creates an offer for Bob to connect. Consequently WebRTC creates a session description protocol (SDP) object. The SDP object holds information like media types, name of the session and the video codec being used. This data is then saved to a *signalling* server. Bob then reads this SDP offer from the server and WebRTC creates a SDP answer and writes this to the server. Alice and Bob have now established a peer to peer connection. In essence, signalling allows for users to exchange the metadata of their connection through the WebRTC API. To justify usage of the WebRTC API, I believe it is first necessary to explain why I rejected the idea of implementing an API from scratch. Unfortunately implementations of fully functioning and secure peer to peer API's aren't trivial at all. The time it would take to fully understand how to implement peer to peer connections using VP9 packetizers [4] and SHA-256 cryptography for data security would simply be too time-consuming and cannot be justified when fully functioning, tested and perfor-

mant implementations already exist. The next portion of the justification covers why did I choose the WebRTC API over other API's? WebRTC is from Google and it is well known that Google is a credible technology company. Therefore it is sensible to assume that their API is of highest quality publicly available right now. However, there are alternatives to WebRTC like: VideoSDK, Twilio and MirrorFly. Whilst it can be acknowledged that these API's could potentially be a better fit for my project in terms of performance, features and simplicity, all of these alternatives are paid for. I don't think it would be sensible to pay for commercial API's when there are free one's like WebRTC available that will work just fine.

When discussing security it is also important to discuss the username and password management system that may be implemented since it is one of my client's requests. In order to design a system that has secure password management it will be beneficial to study how other industry applications have chosen to solve the problem of managing user passwords. I chose to study Bitwarden, a freemium open source password management service. The following information was found on their architecture webpage [2]. As per their website they use the *"Command and Query Responsibility Segregation (CQRS) pattern"*. The CQRS model developed by Microsoft separates reads and writes into different models. *Commands* are used to write to the database, whilst *queries* are used to read from the database. Each command and query has one *single* responsibility and should be based on actions rather than operations on data. For instance Microsoft give the example rather than use the data-based command: `SET ReservationStatus to Reserved` we should prefer to use the action-based command: `Book_Hotel_Room()` instead. Some of the benefits of this model include:

- Security. It's easier to ensure that only people of authority are performing reads and writes to the database if they're separated.
- Separation of concerns. Splitting the read and write components makes the system more modular and more maintainable.
- Simplified commands. Instead of directly manipulating data commands should be based on the task they try to achieve.

Bitwarden's servers use the MSSQL database and makes automated nightly backups to ensure data is protected. Furthermore the company uses *zero-knowledge encryption* so that the company cannot see it's users data. Through this kind of encryption the company is able to have it's users complete trust as they are sure that their data is safe and cannot be read by anyone other than them. Our application could definitely implement the CQRS model. The zero-knowledge encryption could perhaps be implemented using an existing API, this is because the algorithms needed to implement zero-knowledge encryption include SHA-256 as well as AES-CBC 256 bit, both algorithms which require large amounts of higher level mathematics.

1.1.4 Essential features and their justifications

Real-time audio/video feeds

This will enable multiple users to connect to each other via their browsers and view each other's webcams, as well as hear each other in real-time. The justification for this feature is that it is explicitly requested for by my client in our interview, furthermore we don't have a *video* conferencing application if we cannot actually see other people's video feeds on our application.

Raising one's virtual hand

This feature will allow people to be able to interact with the person giving the talk as if they were present in real life. The virtual hand will be made visible to all participants in the conference so that the speaker/host will be able to ask the audience members to speak when appropiate. This feature is sufficiently justified because of my client's specific request to implement it as a feature in the interview. Furthermore the implementation of this feature will help the migration from Zoom to our new platform be familiar as this feature is also found on Zoom.

Designation of a host

This feature will allow the creator of the conference to assign 1 or multiple people to be the *host* of the conference. That means that these people will be in charge of managing the conference and will control who to admit into the conference call, who to unmute and who to remove from the call, and

will be able to lower other's virtual hand. The justification of this feature is that I believe there should always be someone of some sort of authority to coordinate and manage these conferences. This is very common in real life also because without management and coordination people would be clueless, and anarchy would run rampant. This phenomenon has been seen in other video chat applications more specifically, on Zoom [3], and in order to provide the best experience for our users it is vital that events like this aren't allowed to happen.

Usernames and passwords

Each user will be able to choose their unique username so that users can easily identify one another upon joining a call. Furthermore each user can set their own password so that their account is protected and others cannot pretend to be them. Assigned to each user account will be their saved settings and options, so that if a user has a specific configuration of settings that are adapted for their needs they don't have to set these up each time they log onto a conference. Passwords will be made to fit some set of requirements to ensure that the password is of sufficient strength. When a user creates a call they will be given a unique passcode that they will be able to share with anyone else whom they would like to invite to the video conference. The justification of this feature is clearly sufficient enough thanks to my research, for it to be considered essential to the final solution, and moreover the feature was also requested during my interview with the client.

WebRTC for peer to peer connections

We will make use of the WebRTC API, in order to establish peer to peer connections between all major web browsers, like Google, Firefox and Safari. This decision was fully justified in 1.1.3. The API will enable users to transfer their video and audio data between one another securely and efficiently, over the internet.

1.1.5 How is the problem solvable via a computational method?

The population of elderly ones in the UK has seen a 52% increase in the last 40 years. This group of people includes a large number of those who are isolated and feel a sense of loneliness in their lives. However through video conferencing these ones gain the ability to socialise and interact with others from the comfort of their homes. This is especially useful as many elderly ones have limited mobility or are bed bound. Without regular opputunities to socialise and interact humans become depressed and our mental health will begin to decline. By developing software to enable those disabled people to talk to others can improve the quality of their mental health significantly. When people are limited by their disabilities or by their illnesses the oppurtunities to go and talk to new people in real life are far and few between. For these people real physical interactions may not be possible, meaning that a computational solution to their problem is not just preferred but necessary.

Accessible video conferencing software is useful to young people aswell. People like my client also have problems with current systems. In order for him to carry out his job effectively he requires a simple and reliable computational solution. Current systems pose a challenge for elderly people to use, which in turn means that my client has to spend a good majority of his time helping people set up their webcams, microphones or other settings. If my client instead had access to video conferencing software that was easy for elderly people to understand how to use comfortably his job would be simplified significantly, and in order to create that new software it is evident that a computational method must be used. Moreover we have evidence from the previous systems like Zoom that were in use, that creating such an application is amenable to a computational approach. Applications such as Zoom are able to transfer 1080p quality video and audio data at 30 FPS with millions of users worldwide. These results demonstrate the validity of a computational approach towards solving this problem, because we are able to see that the desired solution is computationally feasible.

Decomposition is a computational method that involves breaking down complex problems into multiple smaller and more manageable problems. The problem of sharing real-time audio/video feeds between users can be decomposed into numerous sub-problems that are easier to accomplish. For example we could break this problem into 4 sub-problems:

1. Establish a connection to the other user's computers
2. Ensure user has connected a suitable webcam/microphone

[3] See https://en.wikipedia.org/wiki/Zoombombing for more information

3. Access the webcam/microphone using the relevant API's
4. Send the video/audio data to the other users in the conference

Breaking larger problems into multiple simpler problems reduces the complexity of a system and allows for easier debugging. Rather than trying to debug a large and complex system, it is much easier to debug a single function or class that accomplishes only 1 task. This is because the large and complex system may be throwing errors for a number of reasons, perhaps it could also be throwing errors because of one mistake that was written in the code some several hundred lines ago. With smaller and more concise code organised into functions and similar structures, we can tell exactly where the code is throwing an error and start working on resolving the issue immediately. This problem is amenable to a computational approach as we are able to improve the maintainability of our codebase by applying the technique of decomposition. Moreover decomposition allows for a much simpler approach to problem solving in programming. When we face a large and complicated task we can first decompose the problem as we did in the example above, and then piece together those smaller solutions to the sub-problems, into 1 solution that achieves our intended goal.

Often once we have decomposed a problem, patterns emerge from the smaller decomposed problems. In recognising these patterns we can reduce the amount of code needed to solve each sub-problem by placing the common operations of each function/class into a function/class of it's own. In order to ensure these repetitive actions aren't found in my code I will employ the use of the DRY software development principle. That is the "**D**on't **r**epeat **y**ourself" principle. The purpose of the DRY principle is to avoid writing redundant code by replacing it with abstractions. This principle guarentees clear and concise code. One feature of the problem that could benefit from pattern recognition is the username and passcode system. There are 2 cases in which we would need to check usernames and passwords, those are when the user initially logs into their account and when the user enters a passcode to enter a video conference. Instead of writing 2 seperate pieces of code to check for the correct username and password I could apply pattern recognition and instead implement 1 function called `Check_User_and_passcode()` to be used in both cases. We have seen that this problem is definitely amenable to a computational approach, through the application of pattern recognition to ensure clarity and conciseness in our codebases.

As mentioned in the above paragraph abstraction is another computational technique that is suited to being used in my solution. Abstraction is the process of removing unnecessary details, and only keeping in the parts of the solution that are important. Abstraction could be applied to the design of the UI of my application. The main focus of this application should be for it to be simple and easy to grasp as highlighted by my client in 1.1.3. So in the design of my UI I will not give the user every single piece of information available about the call because the vast majority of that information will be useless to them. Furthermore it can be argued that the presentation of less pieces of information is better for the user's mental health than the presentation of large amounts of information [1]. When humans have too many options we become unable to make descisions, so by limiting the amount of information available to the user we allow them to focus on the few important configurations that they should be in control of, in turn permitting them to give the majority of their attention to whomever they are having a call with. This appropiately demonstrates the amenability of applying abstraction in our final application. This choice will enhance the user's experience while on our webpage and make the design less cluttered and more aesthetically pleasing, making the application more desirable to use.

The last computational technique is algorithmic thinking.

1.1.6 Problems, issues and limitations

Bibliography

[1] Chernev Alexander, Böckenholt Ulf, and Goodman Joseph. "Choice overload: A conceptual review and meta-analysis". In: *Journal of Consumer Psychology* 25 (2 2015). ISSN: 1057-7408. URL: https://myscp.onlinelibrary.wiley.com/doi/10.1016/j.jcps.2014.08.002.

[2] *Bitwarden*. Version 2024.6.3. URL: https://bitwarden.com/.

[3] VanKim Nicole and Nelson Toben. "Vigorous Physical Activity, Mental Health, Perceived Stress, and Socializing Among College Students". In: *American Journal of Health Promotion* (2013).

[4] *RTP Payload Format for VP9*. IETF, 2021. URL: https://datatracker.ietf.org/doc/draft-ietf-payload-vp9/.

[5] *Skype*. Version 8.119.0.201. URL: https://www.skype.com/en/.

[6] *Zoom*. Version 6.1.1. URL: https://zoom.us/.