**Google Hangout Translate Project**

**Design Specification Version 2**

Google Project Manager: Keith Stevens

Huan-Sony Nguyen | Developer and Project Point Person | [sonya8@ucla.edu](mailto:sonya8@ucla.edu)

Luna Li | Developer | [lunaxli@ucla.edu](mailto:lunaxli@ucla.edu)

Amy Lin | Developer | [amy\_y\_lin@outlook.com](mailto:amy_y_lin@outlook.com)

Zhuo Li | Developer | [a-tion@hotmail.com](mailto:a-tion@hotmail.com)

Xinyou Yu | Developer | [yuyuxinyou@gmail.com](mailto:yuyuxinyou@gmail.com)

**Table of Contents**

* Introduction………………………………………………………………………………..2
* System Overview………………………………………………………………………….3
* Design Considerations…………………………………………………………………….4
  + Assumptions and Dependencies…………………………………………………..4
  + General Constraints………………………………………………………………..5
  + Goals and Guidelines……………………………………………………………...5
  + Development Methods…………………………………………………………….6
* Architectural Strategies……………………………………………………………………6
* System Architecture……………………………………………………………………….7
* Policies and Tactics………………………………………………………………………11
* Glossary………………………………………………………………………………….12
* Bibliography……………………………………………………………………………..12

**Introduction**

The purpose of this document is to report and record the implementation of the Google Hangouts Translate Application Version 1.0. This document will specify the system overview and functionality, design considerations, architecture and design strategies, and the system architecture of the product. It will include a detailed architecture diagram and details on the components of the system, including how they collaborate with each other. This document is intended to provide users with a thorough understanding of how the application was implemented.

This document will be read by the developer team, specifically those designing and implementing the application. It will also be read by the product manager, who has a strong developer background and is assumed to have at least the same technical knowledge as the developer team. The project manager should be able to gain an adequate understanding of the project, such that they are able to grasp the major design decisions of the developer team and advise them on their choice. The developer team should be able to gain sufficient understanding of the project so they are able to understand the overall structure of the source code and components associated with the project and jump in on the work with little assistance. The document will be sufficiently detailed to gain its audience these levels of understanding without becoming too difficult to maintain as changes are made to the application.

This document presents the overall system and functionalities first before delving deeper into the individual components and exploring details of the system. For the product manager, the first three sections (“System Overview,” “Design Considerations,” and “Architectural Strategies”) will be the most relevant. For the developer team, all three of these above would be important to read first before reading the last section on “System Architecture,” which contains detailed descriptions of the system on a component level. For all audiences, it is recommended to read this document from the top down in order to gain an understanding of the overall system before exploring its components.

References and related documents will be listed in the “Bibliography” section.

Important Terms will be defined in the “Glossary” section.

**System Overview**

This project will be added as an additional feature to the existing Google Hangouts framework and will therefore utilize the Google Hangouts and Google Translate APIs to implement. More specifically, the feature will be a part of the Google chat box/messaging and video chat service.

Below are the main functionalities of the system:

* *Chat Box Translation*: User A will have the ability to type in their input and set their language (language A) from a dropdown list. User B will be able to set their language (language B) and the input from User A will be automatically translated from language A to language B and vice versa.
* *2-5 Participants per Session*: The product will have the ability to handle more than 2 users at once and all inputs will be taken and translated into each users’ specified language.
* *Video Chat Translation*: The product will also have a voice translation feature that performs the same functionality as text but translates vocal input to text output and prints it on the video windows of the speakers.

The basic design approach will be to divide this system into independent sections where we can present a functional product at the end of it. By ensuring that we aim towards this goal, we will be able to demonstrate this intermediate version of our product to our client and obtain feedback on whether or not to proceed, or if there are any other aspects we should fix or implement. This allows us to change aspects of the application and satisfy our client’s needs more closely.

**Design Considerations**

**Assumptions and Dependencies**

*Assumptions:*

* We assume that the hardware and software that this project relies on (listed in the “General Constraints” section) is generally available to the majority of the users who utilize Google Hangouts.
* We assume that the operating systems (listed in the “General Constraints” section) that support our APIs and Google Hangouts will continue to do so in the future and that there will be a very low failure rate that results from running our application on these operating systems.
* We assume that the basic purpose of this project, which is to improve the international user experience with Google Hangouts, will not change. From that we assume that a key component of this will include a method for translating between the varying languages that users utilize. From this we assume that our choice of Google Hangouts as our platform will not change in the near future, and that we need to incorporate the Google Translate API to provide translation services.
* At this time, we also assume that the voice to text requirement is not likely to become obsolete in the near future, which gives us the confidence to plan to incorporate this API into our project. However, we have scheduled this aspect as our last requirement to be completed as it is not the central-most requirement for our application to provide translation services.
* We assume that our application will be changed in the future to fulfill expanded and new requirements.

*Dependencies:*

* Our application depends upon the current version of the user interface of Google Hangouts. If it changes in the near future, our interface might not conform to its appearance.
* Our application depends on the current Google Translate API, which supports 80 languages. If the Google Translate API is changed (e.g. to include more languages) the specification regarding the number of supported languages will also change.

**General Constraints**

*End-user Environment:*

* A user is required to have the ability to log in using a google account and get general user access. The product will thus be contained solely within a Google Hangouts application.
* A developer, for testing and debugging purposes will have a full access account and ability to access both ends of the chat box. We will need to give developers the ability to do this.

*Operating Environment:*

* The application will be available on a Computer / Tablet as an add-on to an already existing Google Hangouts system so all operating environments that support Google Hangouts must also work with this product. Thus, we must test that our application works with these operating environments and not include any aspects that fail on these operating systems.
* Our application will rely on the internet connection that our users use. Traffic sent over the internet should be succinct enough not to slow down performance (described later in this section).
* The application is restricted to run on Android / iOS devices that have at least 13 mb of memory for Android and 12.6 mb for iOS. This memory restriction limits the devices our application can run on.

*Authentication:*

* The app is going to be accessed via a Google account (Google Plus) so any authentication checking would have to be checked with Google Plus. Our implementation must accommodate this security system.

**Goals and Guidelines**

*Integration with Existing Google Hangouts Interface*

* To decrease the time it takes for users to figure out how to use our interface, we would like it to blend smoothly in with the existing Google Hangouts interface.
* Additionally, we would eventually wish for our add-on to be completely incorporated by the Google Hangouts interface and such strong integration is necessary for this step.

*Smooth, Fast Communication*

* The main purpose of this application is to increase the communication quality and ability between international users. In order to provide users with a product that can maintain a “conversation-like” quality, we must ensure that users do not wait for a long period of time to receive translations of their conversation.

*Memory Usage Only Considered for Mobile Use*

* Speed is the upmost consideration in most cases of usage. However, mobile devices generally have less memory available than tables and computers. In the case of mobile devices, we must ensure that we do not overuse memory in order to ensure that our add-on can function.

**Development Methods**

* We utilized an incremental method of software development. We initially spoke with our client and obtained the overall idea for the project. We separated our application into three main sections which we plan to implement sequentially one after the other.
* As a team, we will be building our application by adding onto each portion incrementally. After each portion or demonstrable sub-portion is finished, we will re-connect with our client to obtain feedback for our work.
* Some changes can be accommodated for this application, as each section is chosen to be completed while leaving the client with a functional product.
* Time was an issue when choosing this approach. We are able to deliver functional stages of our application to our client for feedback, but have little room for changes as we are on a 10-week quarter system.

**Architectural Strategies**

*Compatibility:*

* The app is going to be accessed and run from Google Hangouts so it has to be compatible with the Google Hangouts API.

*Interface:*

* The app will overlay the Google Hangouts API, so not only will its internal code have to be fully integrated with Google Hangouts, but also its UI will be restricted by the existing Google Hangouts UI.

*Language:*

* All of Google Hangouts applications and Google Hangouts itself are written in Javascript, so the Translate app might have to conform with Javascript for maintainability.

*Communication:*

* The application will be using an existing Speech to Text API from Google, which has its performance limitation of only being able to process 60 seconds of speech to text at one time. We will need to accommodate this such that it doesn’t cause a lag in user communication.

*Future Plans:*

* While this current developer team currently doesn’t have any set plans to expand the functionality of the application in the near future, the client is open to the idea. Thus, we constructed our application to be completed in different layers of functionality so that each of our completed versions could potentially be used in the future

**System Architecture**

*Major Responsibilities*

* The overall system is decomposed into three main subsystems: Hangouts API Subsystem, Speech-to-Text API Subsystem, and Translate API Subsystem. Though our implementation focuses primarily on the Translation subsystem, all three subsystems have to be integrated and synced in order for the application to offer a complete experience for the users.
* The Hangouts API Subsystem acts as the basis for the entire application. As the application is built to function as an add-on to Google Hangouts, this forms the base for all other integrations.
* The Speech-to-Text API Subsystem is responsible for converting voice to text. This is essential for the requirement of allowing users to speak naturally instead of using text-to-text conversions.
* The Translate API Subsystem forms most of the body of the application. That is, the main purpose of this application is to provide translations, which this subsystem handles.

*Components*

1. Hangouts API Subsystem:

* This provides an application for people to hold group hangouts online with participants who speak different languages as well the authentication system that allows users to login to their accounts.
* This provides an audio and graphics interface for participants to connect and communicate on the web.
* This provides chatting through both texts and audio/video.
* This supports a minimum of two and a maximum of five participants.

1. Speech-to-Text API Subsystem:

* It provides conversion from participants’ speech into text.
* It is supported by Chrome only.
* It has 60 seconds limit for each session.
* It requires users’ consent to start translating.

1. Translate API Subsystem:

* It translates messages/conversations between participants in a group chat.
* It supports 80 languages.
* It translates live as the conversation is taking place.

*Rationale behind Decomposition*

The decomposition was formulated so the application could be implemented in three steps:

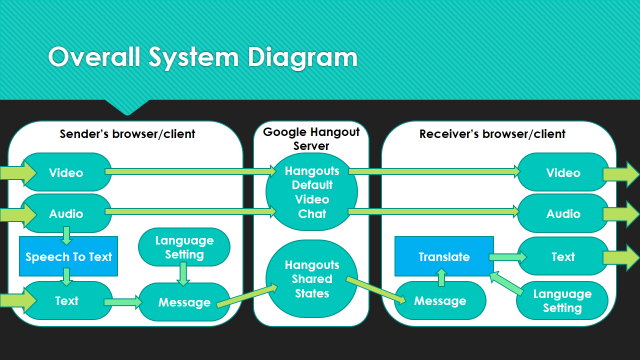
1. A Google Hangouts application with the UI in place for the translation service will be created. By creating a basic Hangouts application, we will have the ability to create at least two UI versions of the application without worrying about how it functions. This will also allow us to test if our base for the application (i.e. Google Hangouts) has any underlying issues with our development environment (so we can estimate the likeliness of this out if any issues occur later in development).
2. The Translate API Subsystem will be integrated. With the translation aspect implemented, we will be able to demonstrate our application’s ability to translate messages from the users’ chat boxes to each other. With this implemented, we will have a product that fulfills one aspect of translation between users.
3. The Speech to Text API Subsystem will then be integrated. There is far more uncertainty working with this as we will need to control more variables with this (including the recording ability of our devices).

This decomposition allowed our development team to produce functional versions of the application on a weekly or biweekly basis to demonstrate to our client in order to gain feedback. From this feedback, we would have the ability to fix a certain portion of our changes to fit this feedback.

*Collaboration between Components*

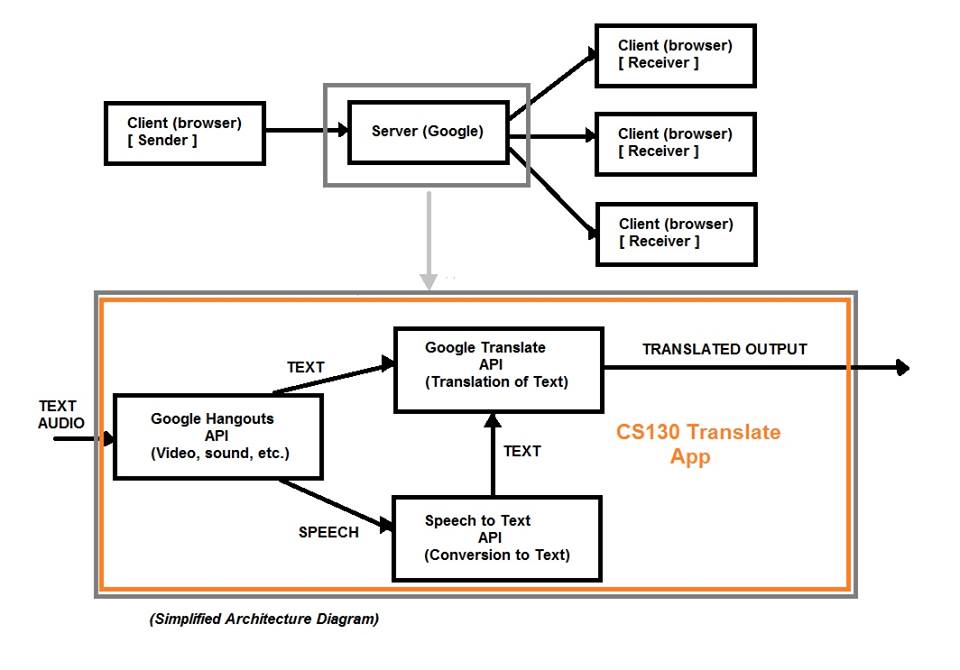
* The Hangouts API will be used to implement a user interface. The other subsystems will be integrated into the UI.
* The Translate API will utilize the chat box implemented with the Hangouts API to convert text between languages.
* The Speech-to-text API will transcribe the audio conversations into text. It will then pass this text to the Translate API to translate into the appropriate language. The Translate API will then utilize the Hangouts API to display the translated text.

*Detailed Architecture Diagrams*



*Figure I: Overall System Design*

Figure I illustrates the overall system design of our application. The sender and receiver communicate via video and audio, which operate via the Hangouts default video chat. In our application, we linked the sender’s audio with the Speech-to-Text API, which transcribes the audio to text. The message is then passed through the Hangouts shared states, which then passes the message through the Translate API on the receiver’s side. The message is translated and then displayed as text for the receiver. A similar process is applied for multiple receivers.



*Figure II: Google Hangouts Server Design*

Figure II provides an overview of the relationship between the Google Hangouts API, Translate API, and Speech-to-Text API within our application (i.e. the Google Hangouts Server). The text and audio are processed via the Hangouts API, which sends the audio to the Speech-to-Text API to be transcribed to text. The transcribed text is then sent to the Translate API, which translates the text along with any text directly passed through from the Hangouts API. The translated output is sent to the receivers.

**Policies and Tactics**

*User Interface*

* Two versions of possible end-user interfaces were created and shown to the client.

*Maintenance*

* As our developer team will only be working on this application for fewer than ten weeks, it is essential that we take measures to extend its lifetime as long as we can for future usability.
* We chose to utilize the Google Translate and Speech-to-text APIs, as well as the Google Hangouts platform with the assumption that these were built to be functional for quite some time.

*Testing*

* The developer team will each individually test our application on their own machines.
* We will develop a list of test cases and scenarios in which to run our tests.
* We will test in a variety of environments (e.g. with varying noise levels).
* We will allocate at least one week in which to conduct tests on our final product.

**Glossary**

* *Google Hangouts*: an instant messaging and video chat platform developed by Google, which launched on May 15, 2013 during the keynote of its I/O development conference.
* *Incremental (Build Model):* a method of software development where the model is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping.

**Bibliography**

Here are some references for pertinent documents:

* Google Hangouts Hardware Requirements: <https://support.google.com/plus/answer/1216376?hl=en>
* Sound Level Requirement: <http://www.cisco.com/en/US/docs/telepresence/endpoint/misc/user_guide/video_conferencing_room_primer_ver02.pdf>

Here are the links to the source code for the project. Please note that these are temporary links; we will include a link to the official published source in our final submission. That is, after we publish our application with Google, it will be publically available.

* <https://xycs130test.appspot.com/static/app.xml>
* <https://xycs130test.appspot.com/static/app.js>