Planning.Pdf – Planning for the Next Problems to Solve

[Requirements] Elicitation

Find 2-3 people to interview as target users. Target users are people who currently use a tool like Streams or intend to. Collect their name and email address.

Develop a series of questions to ask these target users to understand what problems they might have with teamwork-driven communication tools that are currently unsolved by Streams. Give these questions to your target users and record their answers.

The following questions were asked to three target users who using Streams-like communication tool:

1. What do you primarily use teamwork-driven communication tools for?
2. What features of teamwork-driven tools do you use the most and why?
3. Were there any issues of benefits you found when using online teamwork-driven tools in comparison to standard in person communication?
4. What are some of the problems you are facing that in your day-to-day use of current teamwork-driven tools?

Here the target user's responses to the above questions:

Name: Bob He Email: bobbiehe101@gmail.com

1. I use it for university work specifically when communicating with group/class members for projects, assignments and homework.
2. Easier to communicate information to all members at the same time. Also, it allows us to work together without all having to travel to the university.
3. One problem is that it is difficult to collaboratively work on a single document without the need of additional tools such as Google docs. It would be good if there was inbuilt.

Name: Clarissa Wang Email: cmwyi@gmail.com

1. Uni work involving group projects and society work
2. The features I use the most are group messaging and group calls
3. Easier to organize times and ease of access, e.g. most people are able to send a message rather than going somewhere. However technical difficulties can also arise as people are reliant on having a good connection to be able to properly use group call features.
4. Want to be able to collaboratively draw or write on something such as a whiteboard as right now there isn't a simple and easy way on most teamwork-driven communication tools for teams to collaborate in a whiteboard style where people are free to draw or add images. This would be useful as although written communication is already implemented sometimes collaborative diagrams make more sense for the task at hand. For example, drawing a flowchart together with my team.

Name: Tom McIntyre Email: tom.mcintyre100@gmail.com

1. Used for connecting with other uni students in my group to complete assignments as well as in my job to communicate with other coworkers
2. I primarily use chat and uploading of files like documents
3. Easily accessible and can go online whenever it suits me. I can have regular updates with other uni students without having to be present in person. However, online teamwork makes me feel disconnected as there is no face-to-face interaction.
4. I want to be able to access certain messages based on different topics. There is only one main stream of messages which becomes cluttered as I can't find the messages that I need. For example, I want to access a conversation but it is hard to find under random updates and files that are sent in the main stream of messages.

Once you have done this, think about how you would solve the following problem and write down a brief description of a proposed solution.

Following the interviews with Bob, Clarissa and Tom, there were a number of similarities between their responses and issues they faced and hence they were merged …

Problem 1: Lack of being able to collaboratively write or draw with team:

Proposed solution: Implement a button within each channel to create and open a document/whiteboard where members are able to collaboratively write/draw together on a single document.

Problem 2: The list of messages in a channel is messy and it is hard to sort through and find important information sometimes:

Proposed solution: Implement the tagging of messages from a list of different tags that users can change as they like. Additionally, implement a 'filter' button that will only show messages tagged with specific categories.

[Requirements] Analysis & Specification - Use Cases

Once you've elicited this information, it's time to consolidate it.

Take the responses from the elicitation and express these requirements as user stories. Document these user stories. For each user story, add user acceptance criteria as notes so that you have a clear definition of when a story has been completed.

User Story for Collaborative Document (Problem 1):

As a student in a group project, I want to be able to simultaneously draw or write with my team on a single document in Streams, so that I can easily collaborate with my team.

Acceptance Criteria:

- A 'Create new document' button is present in the channel
- A 'List existing documents' button is present in the channel
- Document is created once the user clicks 'Create new document'
- User is asked to name document
- Document is added to the list of existing documents
- A list of all of the existing documents is shown when the user clicks 'List existing documents'
- The document is opened when the user clicks the name of the specified document in the list of all existing documents.
- Launches a document where all members of the channel are able to collaboratively write on
- Only members of the channel in which the document was created are able to collaboratively write on it
- Members are also able to draw on the document
- All changes in the document made by one member are reflected when any other member access the document

---

User Story for Message Tagging (Problem 2):

As a member of a team, I want to be able to tag messages with specific categories so that I can easily find and filter certain messages based on different topics for quick access.

Acceptance Criteria:

- An 'Add tag' button is present next to each message
- A list of the existing tags along with the option to type a new tag is displayed 'Add tag'
- The message is now tagged with the tag from the list of tags that the user specified
- The tag must be between 1 and 50 characters
- Cannot create a tag that already exists
- Only people with owner permissions in channel and the user who sent the message are able to tag a message
- Tags can be removed from a message by people with owner permissions in channel or the user who sent the message
- A 'Filter by tag' drop-down list is present at the top of the list of messages
- Only messages that match the tag specified in the 'Filter by tag' drop-down list are shown to the user

---

Once documented, generate at least one use case that attempts to describe a solution that satisfies some of or all the requirements elicited. You can generate a visual diagram or a more written-recipe style, as per lectures.

Main Success Scenario for Collaborative Document (Problem 1):

1. User asks to create create new collaborative document in channel
2. Streams validates that user is a member of that channel
3. Streams asks user for the name of the document
4. User enters document name
5. Streams displays to user a list of all documents in the channel
6. User asks to open specified document from list
7. Streams opens specified document for user to edit

Use-Case (Background)

Use Case: Create and write on a document

Goal in Context: User wants to collaborate on a single document with their channel members in real time

Scope: Streams

Level: Primary Task

Preconditions: The user has registered and is a member of the channel

Success End Condition: A doc has been created for the channel

Failed End Condition: The doc has not been created for the channel

Primary Actor: Streams User

Trigger: User clicks on "create new document"

Main Success Scenario for Message Tagging (Problem 2):

1. User asks to add a tag to a message
2. Streams validates that user either sent the message or has owner permissions in that channel
3. Streams asks user for the name of the tag
4. User enters tag name
5. Streams displays to user the list of tags with new tag created
6. Streams tags original message with newly created tag
7. User asks to see all messages with that tag
8. Streams displays all messages with that tag

Use-Case (Background)

Use Case: Tag a message

Goal in Context: User wants to be able to filter messages by topic

Scope: Streams

Level: Primary Task

Preconditions: The user has registered and is a member of the channel. User sends a message and has sent the message or has owner permissions in the channel

Success End Condition: Message becomes tagged with the specified tag

Failed End Condition: Message is not tagged

Primary Actor: Streams User

Trigger: User clicks on “Add tag”

[Requirements] Validation

With your completed use case work, reach out to the 2-3 people you interviewed originally and inquire as to the extent to which these use cases would adequately describe the problem they're trying to solve. Ask them for a comment on this, and record their comments in the PDF.

Bob: I think this does generally address the problems I’m currently facing in trying to collaborate with my team within a document. I think if we could also use this document in conjunction with other features such as calls, that would be a really nice implementation as well.

Clarissa: This does answer my problem significantly and will be super helpful for my later use of streams, especially whilst collaborating within a team project. As I mentioned before, it doesn’t necessarily have to be a document and it could be a whiteboard to create flow charts with my team.

Tom: I think this does clear a lot of things up and will make it a lot easier to traverse the stream of messages. I would also like to tag messages for myself personally so I can organise the messages to suit my own study patterns, without affecting the main tags.

[Design] Interface Design

Now that we've established our problem (described as requirements), it's time to think about our solution in terms of what capabilities would be necessary. You will specify these capabilities as HTTP endpoints, similar to what is described in 6.2. There is no minimum or maximum of what is needed - it will depend on what problem you're solving.

Variable name	Type
has suffix id	integer
named exactly token	string
contains substring name	string
named exactly messages	List of dictionaries, where each dictionary contains types { message_id, u_id, message, time_created, reacts, is_pinned }
named exactly documents	List of dictionaries, contains types { document_id, time_created, document_content }

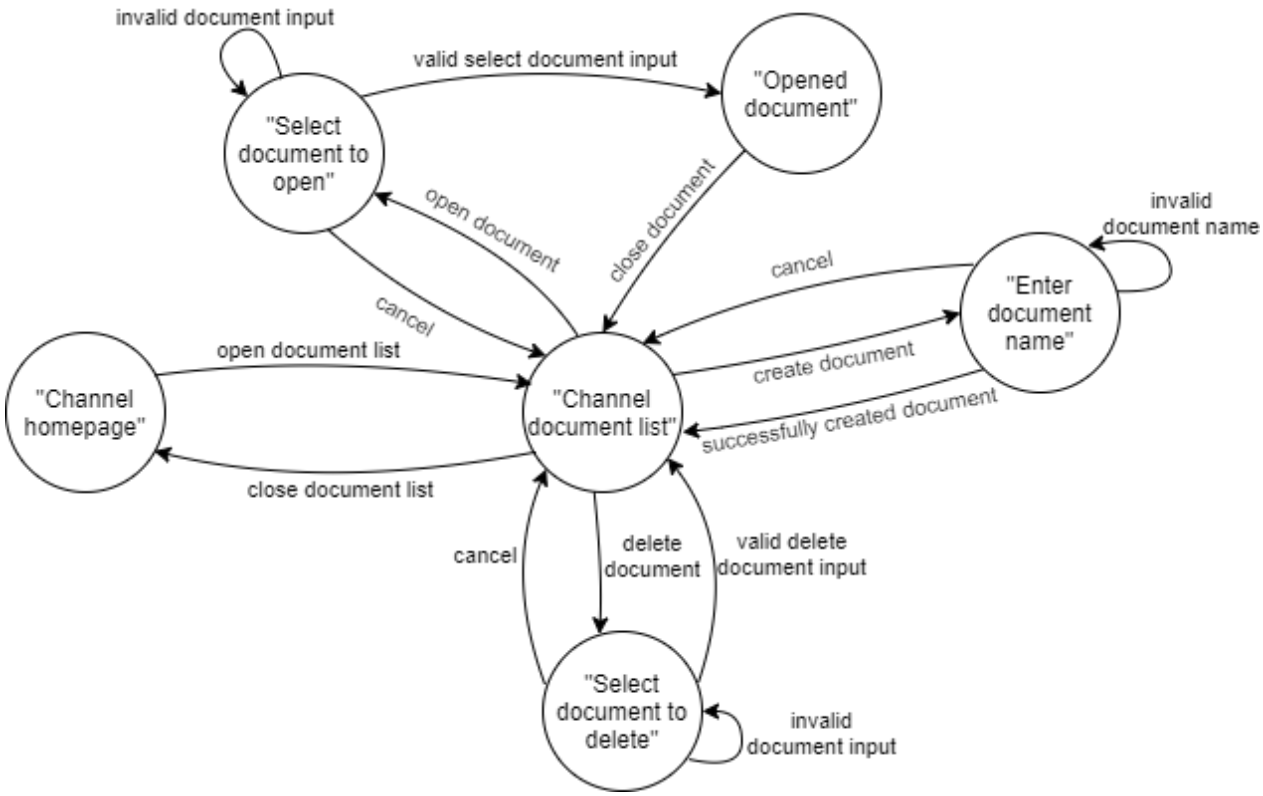
Name and Description	HTTP Method	Data Types	Exceptions
message/createtag/v1 Given a message within a channel the authorised user is part of, creates a new tag and tags that particular message	POST	Parameters: { token, message_id, tag_name } Return Type: { tag_id }	InputError • Message_id is not a valid message within a channel that the authorised user has joined • Tag already exists • Tag_name is not between 1 and 50 characters AccessError • User did not send the message or doesn't have owner permissions in channel
message/tag/v1 Given a message within a channel the authorised user is part of, tags that particular message with the chosen tag	POST	Parameters: { token, message_id, tag_id } Return Type: { }	InputError • Message_id is not a valid message within a channel that the authorised user has joined • Tag_id is not a valid tag_id • Message already contains the tag given by tag_id AccessError • User did not send the message or doesn't have owner permissions in channel
message/filtertag/v1 Given a tag_id and channel_id that the authorised user is a part of , returns a collection of messages that contain the tag	GET	Parameters: { token, channel_id, tag_id } Return Type: { messages }	InputError • Tag_id is not a valid tag_id • Invalid channel_id AccessError • Channel_id is valid, but the user isn't a member of the channel
message/removetag/v1 Given a message with a channel or DM the authorised user is part of, untags that particular message with the chosen tag	POST	Parameters: { token, message_id, tag_id } Return Type: { }	InputError • Message_id is not a valid message within a channel that the authorised user has joined • Tag_id is not a valid tag_id • Message doesn't contain the tag given by tag_id AccessError • User did not send the message or doesn't have owner permissions in channel
channel/createdocument/v1 Create a new collaborative	POST	Parameters: { token, channel_id, document_name}	InputError when: • channel_id does not refer to a

document in the channel for all users to access and edit.		Return Type: { document_id }	valid channel • document_name is already used • document_name is not between 1 and 50 characters AccessError when: • channel_id is valid, and the authorised user is not a member of the channel
channel/listdocuments/v1 Provides a list of all of the documents created in the specified channel.	GET	Parameters: { token, channel_id } Return Type: { documents }	InputError when: • channel_id does not refer to a valid channel AccessError when: • channel_id is valid, and the authorised user is not a member of the channel
channel/opendocument/v1 Opens the specified document for the user to access and edit	GET	Parameters: { token, document_id } Return Type: { }	InputError when: • document_id is not a valid document within the channel that the authorised user has joined AccessError when: • document_id refers to a valid document and the authorised user is not a member of the channel
channel/removedocument/v1 Deletes the specified document from the channel	DELETE	Parameters: { token, document_id } Return Type: { }	InputError when: • document_id is not a valid document within the channel that the authorised user has joined AccessError when: • document_id refers to a valid document in a joined channel and the authorised user does not have owner permissions in the channel

[Design] Conceptual Modelling (State)

Now that you have a sense of the problem to solve, and what capabilities you will need to provide to solve it, add at least one state diagram to your PDF to show how the state of the application would change based on user actions. The aim of this diagram is how to a developer understand the different states the user or application.

Collaborative Document (Problem 1):



Message Tagging (Problem 2):