Threaded Messages

Goal Definition Phase

Desired Impact

Lifting Mattermost's NPS score is a top goal for the Core product team for 2020.

Mattermost customers' number top paint point, and number one product request with over 1600 votes, is to Make Threads Collapsible (the second most-requested feature has 900 votes). Make Threads Collapsible has been in the top 5 for at least eighteen months and has been the top request for about a year or more.

Several customers have cited the lack of threading functionality as an impediment to migration or product adoption and it has been specified as the top reason behind low NPS scores for USAF Kessel Run.

The specific pain is users' difficulty processing messages in channels, where the center panel is a conveyor belt of interwoven messages that is continuously moving making conversations fragmented, overlapping, difficult to parse into separate threads. New reply activity is hard to keep up with, and threads are hard to relocate.

This project aims to improve users' ability to process channel content; more easily find, follow and resume discussion threads; and prioritize discussion threads.

Projected impact:

- Reduction in customer pain. Confirmation from above customers, as well as free-tier accounts, that the reply thread changes have improved overall efficiency in day-to-day communications and collaboration. This feedback will be obtained through direct feedback sessions.
- In-product survey - possibly?
- Increase in NPS score of 10 points by Q3
- Unblock Slack migrations specifically attributed to lack of reply threading
- Specific product analytics
    - Adoption of new features and interaction with new UI elements TBD
    - Increase in number of reply posts relative to 'parent' posts

Functional Goal

Quick background: why do people use channels?

- stay informed, i.e. graze new message activity
- read and respond to specific discussions (synchronously and asynchronously)
- coordinate with other people, bots (sync and async)

Currently their ability to accomplish these tasks is compromised. The issue:

- The center panel is a conveyor belt of interwoven messages making conversations fragmented, overlapping and difficult to parse into separate conversations.

The consequent pain:

- Reading and processing information is slow and onerous
    - ‘Can’t find the thread I’m looking for’
    - ‘Too difficult to see which messages belong to which threads’
- Information relevant to any single user is buried or lost
    - ‘too many messages that don’t apply to me’
- It is difficult for users to keep their discussions organized, esp as those discussions evolve into action items

There are five areas that will be explored for pain alleviation and experience improvements:

1. Improving the way that messages are default organized in the center panel and RHS to facilitate faster scanning and processing of distinct threads
2. Ensuring that any solution to re-organizing the center panel and RHS does not increase the risk of users missing replies or updates to threads. For eg. improved notifications, or a ‘Threads’ view.
3. Exploring new product features that offer some level of custom organization of threads
4. Better communicating to users (esp novices) the difference between creating a parent post and a reply post, when and why to do so. Or better yet, alleviating the user from the burden of having to think about the difference.
5. Stretch: Reducing amount of channel switching that is required to keep up with unread messages
6. Stretch: Exploring ways that threads might more naturally and effortlessly turn into action items

Slack and other competitor solutions will be closely reviewed.

The solution will apply to Mobile, Webapp and Desktop App.

Background Details & Use Cases

Link to UX kickoff deck

Prioritization and “Mana Budget”

Folded Reply Threads is the top priority for App team

Requirements Definition Phase

Stakeholders

Core Team
PM - Adam Clarkson
UX - Andrew Brown
Apps Dev Lead - Dean Whillier
Mobile Dev Lead - Elias Nahum

Stakeholders
Executive Team - Lindsay, Corey, Ian, Lauryn, Allison
Product Team - Eric Sethna, Eric Sadur, Aaron, Katie, Ian
CSM Team - all
Account Executive / Sales team - TBD
Sales Architects - all

Customer Beta Partners

Prior to any Beta releases there will be a design research and exploration phase, at the end of which there will be wireframes, mockups and other artifacts that can be brought to strategic customers for input. AN internal list has been compiled.

Beta partners are TBD. They will need to be beta-tolerant orgs, willing to contribute feedback, willing to deal with change.

Competitive Research

Slack

Slack and their approach to threads is thoroughly reviewed in the UX Kickoff doc

Highly recommended read on Slack's approach to thread design:
Threads in Slack, a Long Design Journey
March 23, 2018, Slack Blog

Also recommended:
The Unexpected Design Challenge Behind Slack's New Threaded Conversations
Jan 18, 2017, Fast Company
Weaving Threads - A human-centric approach to organizing conversations
Jan 18, 2017, Medium

Twist

Twist - competitor review notes
Twist design philosophy: Designing Twist: The Challenge of Making Teamwork Less Stressful

Zulip

Why Zulip (company marketing)
Zulip - competitor review notes

MS Teams

Work in progress

Preliminary User Research

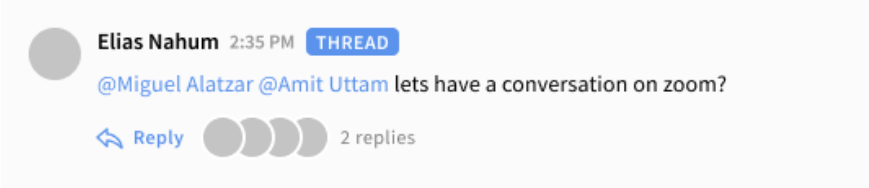
Nine heavy users of Slack were interviewed to better understand their personal and organizational communications approach to Slack. Report here: **User interviews**

Design Sprint

The UX design sprint yielded several concept prototypes: Figma prototypes

Summary of key feature proposals

1. **Collapsed Reply Threads** - replies removed from the center panel, parent messages with replies show number of replies, users who replied, date and time of most recent replies. Some designs showed an abbreviation of the most recent reply. To read the reply the user has to open the RHS.
   Eg.

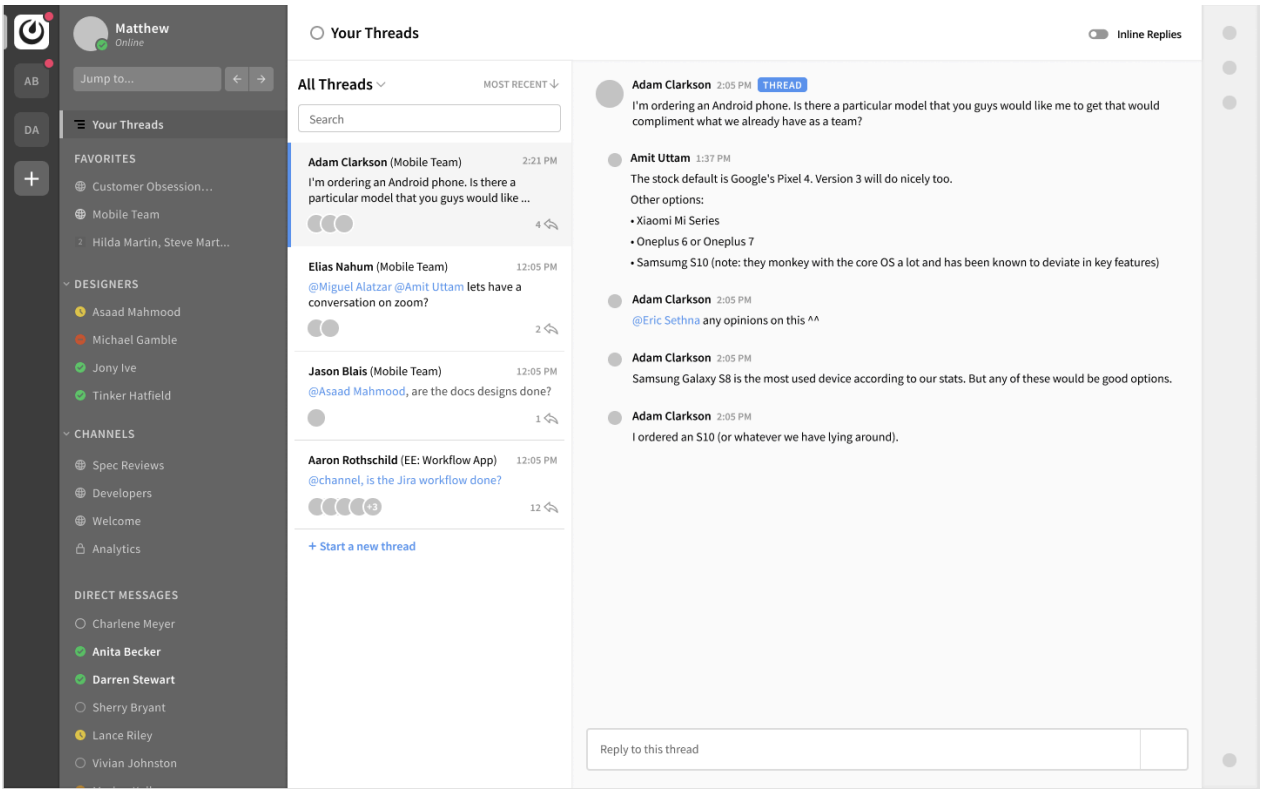


2. **Reply summary system message** - to mitigate the possibility of users missing replies to older threads, a concept was proposed of a system message outputting a succinct summary of the reply activity to the center panel

   Eg.

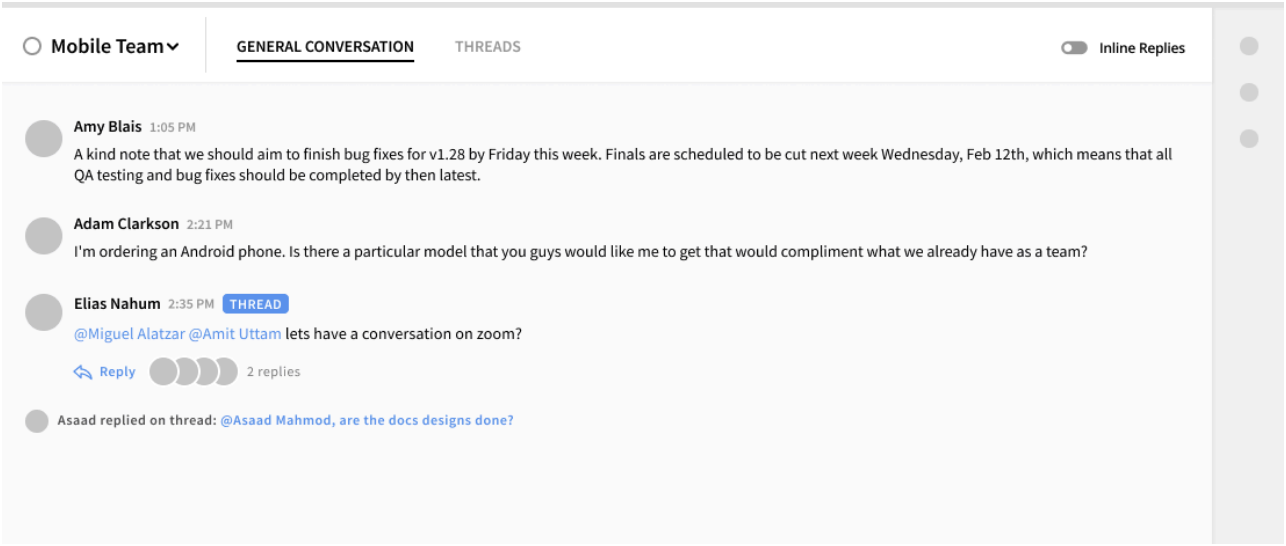
3. **Global Threads -** a repository of all threads that the user has a) participated in b) been mentioned in or c) explicitly followed. The repository contains threads from across all channels, DMs and GMs (hence ‘global’) and is accessed from the navigation item in the channel sidebar.

Eg.

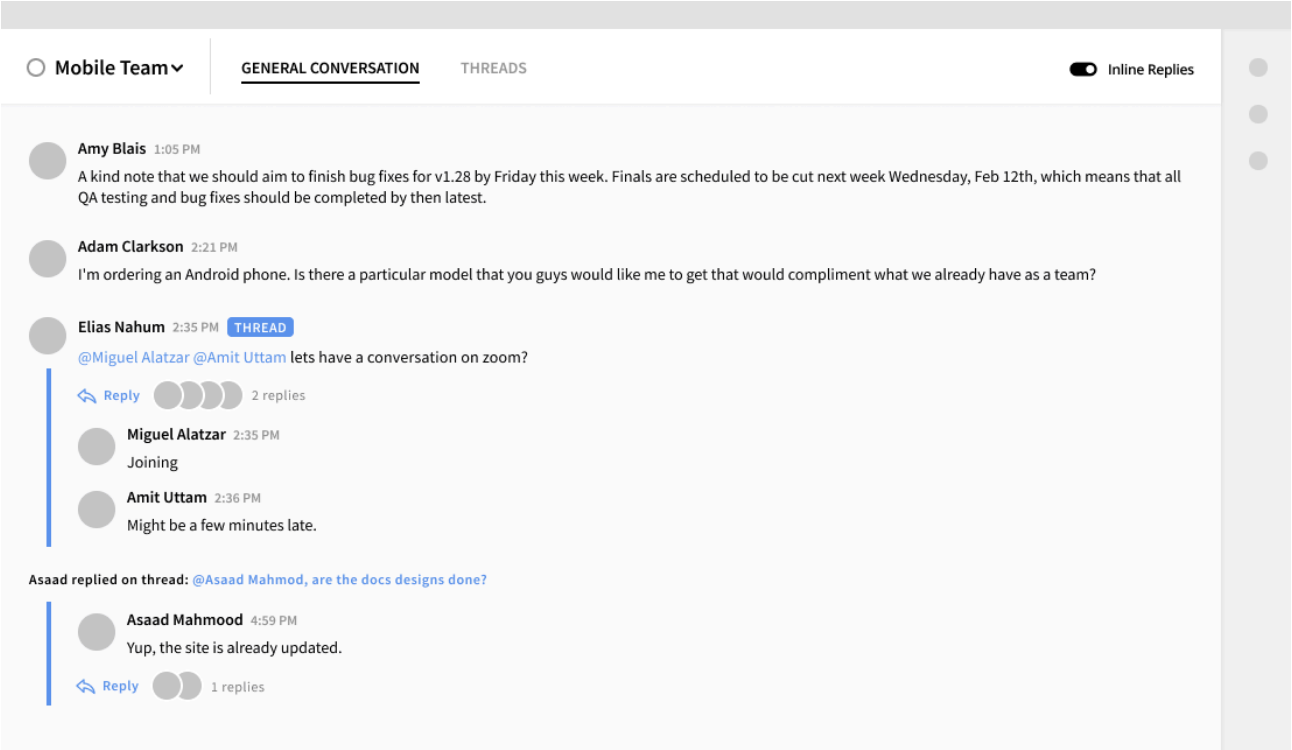


4. **Expand / Collapse Threads toggle** - to address change management and soften the impact of substantial changes to the center panel a toggle was proposed that would change the center panel from hiding all replies in a collapsed view under the parent message, to showing them expended in chronological order (MM classic, or MM as it presently functions)

Eg. 1 - collapsed view

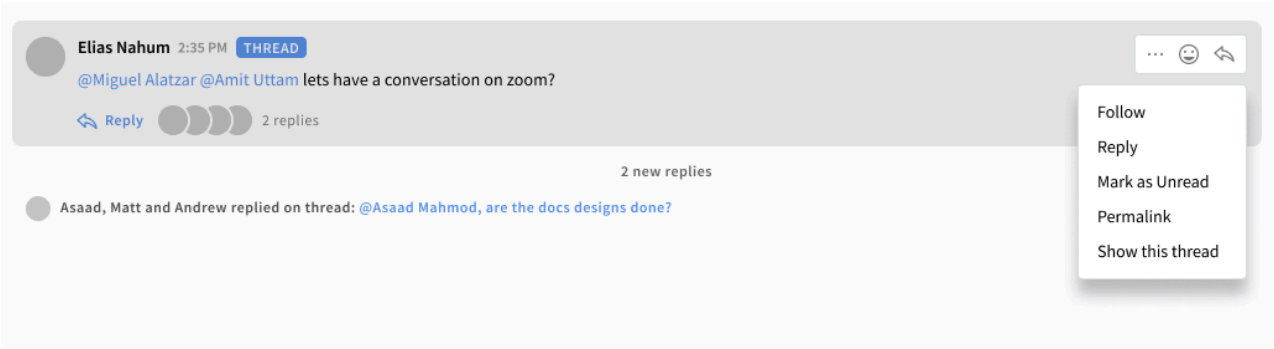


Eg. 1 - expanded view



5. Follow Thread - a feature that would allow users to follow a thread which would include it in the Global Threads view. Included in this proposal was the ability to unfollow a thread.

   Eg. 1



6. **In-channel Threads View -** An alternate or complimentary view within a channel that organizes discussions by thread using the same information architecture proposed in the Global Threads view.

7. **Editable Thread subjects** - The ability to edit the ‘subject’ line of a thread to help convey purpose, audience and scope of a discussion to other users

8. **Start a thread button -** A feature that initiates a discussion. Exact functionality TBD. At minimumum would present a unique message composition screen with editable subject

Several other features were suggested, not captured here, but available in the Figma prototypes

Three Approaches to Product Solution

Coming out of the design sprint there are three possible approaches. It's important to remember the fundamental project goals here:

1. Alleviate congestion in the center panel, facilitate scanning
2. Ensure users don't miss messages
3. Improve thread formation (encouraging users to reply instead of posting new parents)
4. Stretch - reduce overhead of channel switching (ie. consolidating important messages & features to better track messages)

Approach 1

Scope:

1. Collapsed Reply Threads - address acute customer pain with message congestion in center panel
2. Reply Summary System Message (RSSM) - capture reply activity in a concise way in center panel to ensure users don't miss replies.

Approach 2

Scope

1. Collapsed Reply Threads
2. Global Threads - to provide ensure users don't miss replies; reduce channel switching by providing a venue for consolidating personally relevant conversations
    a. Update logic for unreads, badges and notifications (rules here)
3. *Optional* - Follow/unfollow Thread - to improve ability to track important conversations

Approach 3

Scope

1. Collapsed Reply Threads
2. Reply Summary System Message (RSSM)
3. Global Threads
    a. New logic for unread, badges and notifications
4. Follow/unfollow Thread

Design and Evaluation process

Phase 1 - Design, evaluation and estimation

Goal - determine which Approach will have the most Impact and what associated costs will be (dev effort) in order to better determine the best path forward (RICE exercise).

We know the most about Approach 2 (since it's basically exactly what Slack does), which means we have more to learn from evaluating Approach 1 and 3. We believe Approach 1 to be lower effort and to solve core user problems, and pave the way for improvements beyond what Slack does now.

1. Design two lo-fi prototypes in Figma
    a. Approach 1 - thoroughly design out functions and edge cases
    b. Approach 3 - use to convey directionally what a more robust and fully featured solution would look like that includes RSSM and Global threads.
2. Evaluate both Approaches with engineering team for 1) design input and 2) mana estimates
3. Evaluate both Approaches with stakeholders
4. Revise and iterate as nec

Phase 2 - Build and Experimental release

Goal - evaluate most promising Approach in live customer environments, incorporate feedback

1. Release to Daily Community server
2. Make available through a switch in Account settings
3. Consider shipping a custom build to some customer organizations that are appropriately beta tolerant and have means to revert if nec.

Proposed Project schedule

In order to ship and experimental release by August release, the following schedule would need to be roughly adhered to:

Week 1	UX - build out Figma prototypes according to Product Spec
Milestone	Dev review
Week 2, 3	Incorporate Dev feedback; iterate on prototypes
Week 4	Stakeholder reviews, customer reviews
Week 5	Consolidate stakeholder and customer feedback
Milestone	Finalize scope for initial release; update spec and prototypes
Weeks 6 - 9	Dev cycle (4 weeks), functionality & telemetry
Milestone	Release POC to Community Daily Server

Week 10-11	Collect community feedback
Week 11	Possibly ship custom builds to beta-tolerant customers
Week 11	Revise project plan and scope for phase 2
Week 12	Feature complete
Milestone	Enter QA and release management process

Initial Proposed Requirements

Collapsed Reply Threads

Terminology
Post = message
Parent post = message entered through the center panel
Child post = reply to a parent post, entered through the RHS
Parent posts with child posts = thread
CP = center panel
Chronological order = oldest at top, most recent at bottom
Reverse chronological = most recent at top, oldest at bottom

Proposed functionality:

The requirements below pertain only to desktop and web app views of the product (for now).

1. Parent posts display in CP - Parent Posts are sorted chronologically (most recent at the bottom) by date and time of the parent post. Date and time of child messages does not alter the sort order of parent posts, which stays locked by datestamp of PP.
2. Child posts display in the RHS only . Child messages are displayed in RHS under the parent post to which they are associated (as per current state). Child messages are sorted chronologically as the currently are in the product.
3. Display of parent posts - For threads, in the CP, the PP's metadata elements should be displayed as they currently are: author, time, body copy, formatting, images, attachments, etc.
    a. Long messages should be abbreviated as they currently are.
    b. Messages with attachments above *N* should be truncated as they currently are.
    c. Reaction to PP and positioning remain the same
    d. Hover state remains the same
    e. parent posts with children will be visually distinct from childless parent posts in some way that indicates the existence of a thread (not simply a childess parent).

4. Thread metadata - For threads, in the CP, the PP should be appended with metadata that helps define the purpose, audience and scope of the thread:
    a. Number of child posts
    b. Avatars of all unique individuals who contributed to the thread, including bots, truncated at point where the design breaks
    c. Time and date of the most recent reply (maybe username of most recent child post author?)
    d. *A possible design consideration:* For longer threads (above a certain number of child posts) the design phase could experiment with a concept that truncates the oldest child through the penultimate child, but shows the most recent child in full (eg. Matt's design, inbox style, to facilitate faster 'catching up' of where the thread left off).
    e. An interaction with the UI that opens the thread's child posts in the RHS
    f. TBD - the design could consider a unique treatment for threads that contain unread replies, eg. 'X new replies) and/or that could be captured in the *Reply Summary System Message* (spec'd below). Use case = "how do I quickly catch up on what I missed"
5. Reply button - as one of the primary goals of the project is to prevent users from creating new parent posts when their actual intention is a reply (KPI is *reply rate*), the reply action in the UI should be prominent. The design phase should propose a solution emphasizing the reply as the primary action for a parent post. Some possibilities include making the reply button display explicitly (without requiring a hover as it currently does), or perhaps making the entire parent post open up in the RHS when clicked and drawing attention to the reply post draft area.
    a. In instances where the thread is longer than the vertical viewing area of the RHS the most recent child posts are displayed (earlier messages out of view at top)
6. Unread parent messages - new parent post activity would be visible at the bottom of the CP as in current design.
7. Unread child messages - TBD - design could consider a state for unread threads
8. Unread toast - show only for new parent posts? Or for any new child or parent? Hide only when new child and parent messages are seen? Or hide after new parent messages are viewed and disregard new child messages?
9. Search results - search results are displayed in the RHS. If a user clicks on a search result to a child post it would open in the RHS (since it can not longer appear in the CP). The design must therefore offer users an easy and clear way to return to search results and resume scanning their results.
10. Permalink to child posts - if a child post contains a permalink to another child post, clicking on the permalink would load the linked-to content in the RHS. The design must therefore offer users an easy and clear way to undo their action (return to the originating thread).
11. Compact mode - *Do we have data on how many users use this? This becomes expensive to support!*
12. Mobile - requirements to be added as desktop and webapp design phase progresses

Reply summary system message

The Reply Summary System Message is intended to inform users of new child posts in a succinct and non-invasive way that doesn't clutter the CP, and also a navigational route back to threads that might have scrolled out of the viewing window and would be tedious to relocate. It's both a FOMO mitigator and a navigational aid.

The initial spec for this feature attempts to define a simpler logic for the RSSM. Complexity arises when trying to maintain a chronological sequence of parent and RSSM messages in the CP while at the same time separating what a user has ‘seen’ or ‘read’ from what they haven’t. Here are two examples:

Example 1 - if the RSSM summarizes all reply activity on a thread, then it will inevitably contain references to child messages the user has already read. The RSSM may indicate that 5 people have replied to a parent message and a user may have already read up to 4 of those messages, but the existence of a new fifth message would put the RSSM under the new messages line in a channel.

Example 2 - if the product evolves to include a Global Threads feature, then it is possible that a user could read every replies to a thread, but when they return to a channel might see an RSSM under the new messages line (if there was at least one new parent post followed by new child posts).

That in mind, here are recommended initial requirements:

Slides in Figma that help illustrate proposed specs

1. The RSSM has a 1:1 relationship with the thread / parent message meaning that there is always only one RSSM in a channel that consolidates all reply activity to a parent message.
2. Summarizes reply activity on a thread and is comprised of the following elements:
    a. Avatar of reply author(s), listed in reverse chron order of reply (most recent replier, 2nd most recent replier, 3rd, 4th)
    b. Username of reply author(s) listed in reverse chron order of reply (as above)
    c. Thread replied on (using summary of parent thread copy and parent thread author’s username)
    d. If there are more than 5 reply authors the RSSM lists the authors and then appends ‘and *N* others’
3. Has a unique design if the reply incorporates a mention of the user viewing the RSSM
    a. ‘@user mentioned you in reply on thread’
    b. This type of RSSM would appear uniquely on its own and constitutes an exception to the rule of 1:1 RSSM to Threads.
4. Appears in the CP and is inserted into the timeline by the timestamp of the most recent child message.
5. When a new child is added to a thread the RSSM will move positions in the CP. The timestamp of the most recent child post is used to determine the position of the RSSM in the CP’s chronological timeline.
6. Includes a click interaction that opens the thread in the RHS (probably shouldn’t try to scroll the CP to where the parent post resides in timeline of messages)
7. ~~When a user leaves a channel and returns, the RSSM will be below the ‘New Messages’ line if the RSSM contains at least one new reply message (even though the RSSM may also summarize previously read reply posts)~~
   When there is reply activity in a channel and the user is away from a channel, when the user returns to the channel the RSSM messages will be positioned below the ‘New Messages’ line. The RSSM may contain references to replies that have already been seen and ones that have not been seen.
8. The RSSM would not appear in the CP if there was no activity in the channel between the parent post and subsequent replies to that parent message. For eg. A parent post is entered into the channel - Post A - then immediately after a reply. This child post would not generate an RSSM. However if the sequence was Parent Post A, then parent post B, then a reply to A, the reply to A would generate an RSSM. Or if the sequence was Parent Post A, then Parent Post B, then a reply to A (RSSM would be triggered) then a reply to B (also triggering an RSSM)

    a. TBD exception to above rule: if user leaves channel, or is scrolled back into the channel history, and there is a parent post (seen) that is then followed immediately by a reply (unseen), then an RSSM would be generated immediately after the parent post, positioned below a ‘New Messages’ line.
9. Clicking on the RSSM message - ….
10. Unread conditions
    a. Any RSSM generated while the user is either out of the channel will make the channel bold (unread)
    b. If user leaves the channel, and subsequently there are child posts (and only child posts) and all of those child posts are viewed either in the RHS or Global Threads view (if/when we build) then the channel should be un-bolded.

~~Expand / Collapse toggle~~

~~The expand/collapse toggle changes the mode of displaying messages in the CP to either Collapsed Threads mode or Expanded Threads mode.~~

1. ~~Collapsed Threads mode - Parent Posts only are displayed in the CP and are sorted chronologically (most recent at the bottom) by date and time of the parent post. Date and time of child messages does not alter the sort order of parent posts, which stays locked by datestamp of PP~~
2. ~~Expanded Threads mode - The product as it functions today - Parent Posts and Child posts are displayed in the CP and are sorted chronologically.~~
    a. ~~Thread metadata - Parent posts in the CP should be displayed as they are in Collapsed mode, i.e. with meta data about extent of children, authors, time, etc.~~
    b. ~~Reply button - Parent posts inthe CP should have a reply function consistent with Collapsed mode.~~
    c. ~~Unread toast - would behave the same as collapsed?~~
    d. ~~Search results in expanded mode - don’t want to maintain two experiences, if we change it for Collapsed, then we should have it also work the same way for expanded).~~
    e. ~~Permalinks for expanded mode - same as above. Should work as redesigned for Collapsed mode~~

Global Threads

Global threads is a view of conversations that the user has been involved in across all their channels, group messages and direct messages. The spirit of Global Threads is to offer a more personalized experience, to ensure replies to threads don’t get missed and to facilitate faster processing of messages by taking them from the many venues (channels, DM, GM) where they canonically reside and consolidating them in one place (less channel switching).

1. Location and appearance in Channel Sidebar
    a. Title TBD - either Threads, or Your Threads, or All Threads.
    b. Exact positioning TBD and in consideration Channel Sidebar project
2. Layout & Architecture

a. Thread List - a new vertical pane that shows all threads that the user has either a) contributed to (parent or child) b) been mentioned in or c) Followed (if/when we implement that functionality), sorted reverse chronologically by most recent child message.
    i. Thread List consists of rows of Thread Items.
        1. Thread Item - is a succinct summary of what the thread is about to help users understand context (what is this discussion about?) and also purpose, audience and scope.
        2. Each Thread Item has a read and unread state to indicate where there are child messages that haven't been displayed in users viewing area.
        3. Thread Item meta data elements:
            a. Heading - copy from the parent post, abbreviated by character count to fit into this view.
            b. Time of most recent child post in the thread
            c. Originating venue - i.e. the Channel, DM or GM that it's from
            d. Number of child posts in the threads
            e. Avatar of each person who has contributed to the thread (truncated as appropriate)
        4. Clicking on a Thread Item - loads the full thread - parent and all child posts - in the Full Thread Panel
        5. Sort options - for now only sorted by date and time (reverse chron). In future could add a 'most replies' or other
        6. Filter options
            a. All - all threads that meet criteria for inclusion in this view
            b. Unread - only threads with child posts that have not been displayed to user
            c. Followed - if/when we implement the follow functionality
            d. Unfollowed - if/when we implement the follow functionality (to undo an unfollow, or go back and read something you thought you were done with)
    ii. Full Thread Panel - works exactly like the RHS, showing all parent and child posts of a thread with a post draft area at the bottom for entering a reply.
        1. Permalink to channel - Full Thread Panel should also contain a link to the channel and the place in the channel where the thread originated. That would mean loading the parent post in the CP and the thread in the RHS
        2. **Question**: Should a thread have a Mark as Read button? This kind of feature might make it possible to use the Global Threads as an inbox, moving things from an 'unread' list to a 'read' list.
3. Header - what is in the header? Should users be able to start a Zoom from here? A Todo list? Recent mentions? Flag posts?
4. Empty state
5. Impact on unreads, badges and notifications - [see matrix of scenarios and associated outputs here] (full spreadsheet of the chart below)

scenario	venue	bold venue	bold threads	badge venue	badge threads	notification
parent post	channel	TRUE	FALSE	FALSE	FALSE	FALSE
parent post mention @user	channel	TRUE	FALSE	TRUE	FALSE	TRUE
parent post @here alert	channel	TRUE	FALSE	TRUE	FALSE	TRUE
reply mention @user	channel	**TRUE**	TRUE	FALSE	TRUE	TRUE
reply to @user parent message	channel	**TRUE**	TRUE	FALSE	FALSE	FALSE
reply to a thread @user has participated in	channel	TRUE	TRUE	FALSE	FALSE	FALSE
reply to thread @user has not participated in	channel	**TRUE**	FALSE	FALSE	FALSE	FALSE
react to a @user parent message	channel	FALSE	FALSE	FALSE	FALSE	FALSE
react to a @user child message	channel	FALSE	FALSE	FALSE	FALSE	FALSE
parent post	GM	TRUE	FALSE	TRUE	FALSE	TRUE
reply to a @user parent message	GM	**TRUE**	TRUE	FALSE	TRUE	TRUE
reply to a thread @user has participated in	GM	**TRUE**	TRUE	FALSE	TRUE	TRUE
reply to a thread @user has not participated in	GM	**TRUE**	FALSE	FALSE	FALSE	FALSE
parent post	DM	TRUE	FALSE	TRUE	FALSE	TRUE
reply to a @user parent message	DM	**TRUE**	TRUE	FALSE	TRUE	TRUE
reply to a thread @user has participated in	DM	**TRUE**	TRUE	FALSE	TRUE	TRUE
reply to a thread @user has not participated in	DM	**TRUE**	FALSE	FALSE	FALSE	FALSE

Follow/unfollow Thread

1. The Follow/unfollow feature is dependent on the Global Threads feature.
2. For each parent post in a channel, GM (and DM?) there should be a control to enable users to ‘Follow’ it. Any subsequent replies to the parent post will then appear in the users’ Global Threads repository.
    a. A followed threads does not appear in Global Threads until there has been a reply
3. Each parent post should also have an ‘Unfollow’ button that will remove the thread from the Global Threads repository
4. The Global Threads repository should have the ability to access ‘unfollowed’ threads
5. TBD: should it be possible to unfollow a thread that is not explicitly ‘followed’ but is in Global Threads by virtue of it containing a mention or being previously participated in?

Out of Scope

The In-channel threading panel was explored in design sprint but not being considered for these initial phases but should continue to be explored for a future version. It’s a real market differentiator and had lots of positive internal feedback and in limited exposure with customers.

The expand/collapse toggle was explored in the design sprint but not being considered

Release Stages

Described above

Questions

HOw do threads work in instances where data retention is turned on? If an organization deletes any posts that are older than 30 days, what does the thread look like if there's no parent post to anchor it in place? (relevant customer: cyber.gc.ca)

What areas does this touch?

Is this built as a plugin?

No

Impact on Bulk Import/Export functionality?

TBD

New architecture would have impact on Slack import tool. Research required.

Design Refinement Phase

Notes on Key Decisions

For questions or decision points that come up during the design process, add notes on what decision was made and why. The goal is to give people reading this document background on how key decisions were made.

Notes on Customer Feedback

If the design is shared with customers or users for feedback, link to notes here.

Design Revision Reviews

Does the design cover everything applicable on the design checklist? Does the design address the requirements and use cases? Are there any technical concerns? Is the proposed design consistent with other parts of the app?

Design Validation

User testing results, sharing with customers or key stakeholders for feedback.

Final Specification Design

High Level Summary:

Short summary / introduction to set context for the design.

Desktop Design

Include functional design notes and mockups/prototypes.

Mobile App Design

Include functional design notes and mockups/prototypes.

Technical Design

Link to the technical design doc. Technical design and implementation needs to consider developing E2E tests for the feature.

Accessibility Design

Ensure implementation of the feature meets accessibility requirements.

Audit Log Events

What events should be recorded in the System Console Audit Log?

Telemetry Design

What events should be recorded. Plan for how the data will be analyzed.

Roll out Plan

Plan for feature roll out. Does this feature have a feature flag?

Spec Review / Sign-off

Team review before signing off on the design: Are all questions addressed? Is this design ready to be implemented? Final check against design checklist.

Launch Plan

Beta Launch Plan

If a feature is being launched as experimental or beta, how do you plan on working with your beta partners to gather feedback? At what point will it be taken out of beta?

Marketing Launch

Work with marketing on a launch plan. Does it need a blog post, videos, anything else? Should this be grouped with other features for marketing as a larger "launch"? When should marketing happen?

Training Plan

Internal Training

Outline what training is needed for internal teams (Sales, CSMs, SAs, Support). Does a meeting need to be scheduled before launch for training purposes?

External Training

Does any external training need to be added to Mattermost Academy? Is this a feature customers may need a training webinar on?

Implementation Jira Tickets

Link to Jira tickets of Jira epic for implementation tickets.

Quality Assurance

Testing Feature Branch

Link to branch as it is available. This is preferred and recommended for all larger features.

Test Plan

Link to test plan.

Test Automation

Is test automation required?