Understanding Chatbot-mediated Task Management

Carlos Toxtli
West Virginia University

Andres Monroy-Hernandez
Snap

Justin Cranshaw
MSR

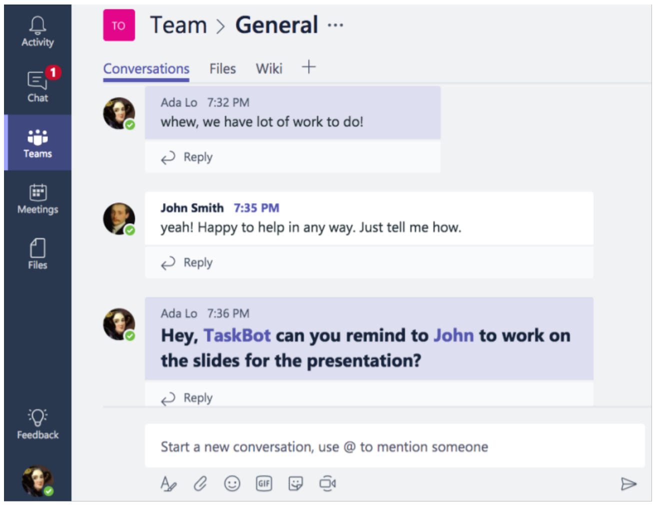
ABSTRACT
Effective task management is essential to successful team collaboration. While the past decade has seen considerable innovation in systems that track and manage group tasks, these innovations have typically been outside of the principal communication channels: email, instant messenger, and group chat. Teams formulate, discuss, refine, assign, and track the progress of their collaborative tasks over electronic communication channels, yet they must leave these channels to update their task tracking tools, creating a source of friction and inefficiency. To address this problem, we explore how bots might be used to mediate task management for individuals and teams. We deploy a prototype bot to eight different teams of information workers to create, assign, and keep track of tasks, while reminding people about their pending tasks, all within their communication channels. We share the lessons we learned about how chatbots can mediate collaborative work.

Author Keywords
Conversational User Interfaces; chatbots; bots; productivity; tasks;

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION
Information workers are constantly switching between communication and productivity tools [13, 17]. For example, people might have an email conversation to decide on a time to meet, then switch to their calendaring tool to add the meeting to their schedule. Similarly, teams might discuss a project over Slack, decide who gets to work on what, then add those tasks to a task management system. These interruptions can add up to reduced productivity and increased stress in the workplace [12, 16, 20].

In order to reduce the cost of switching contexts, a new category of productivity technologies have emerged in the form of chatbots that can be summoned from within communication channels. These chatbots, or bots for short, allow people to delegate in-situ, without having to leave the chatroom, messenger app, or email client. For example, a scheduling



Team > General ···

Ada Lo 7:32 PM
whew, we have lot of work to do!

John Smith 7:35 PM
yeah! Happy to help in any way. Just tell me how.

Ada Lo 7:36 PM
Hey, TaskBot can you remind to John to work on the slides for the presentation?

Start a new conversation, use @ to mention someone

Figure 1. A group chat conversation on *Microsoft Teams* showing a user assigning a task to her teammate and asking Task Bot to help track of it.

meeting bot service allows people to delegate the work of scheduling a meeting by cc'ing the bot in an email conversation [11]. Similarly, there is a myriad of bots available in work-centric chat platforms like Slack and Microsoft Teams, from bots that help teams order food [18], to those that integrate with software development tools [19].

Although there is a lot of research on technologies for team coordination of tasks [22, 7], the emergence of bots as mediators of work-related activities presents new challenges and opportunities that we decided to investigate.

In this work, we focus on trying to understand how bots can mediate task management, and we do this by building and deploying one such bot: Task Bot. Members of a team can, for example, interact with Task Bot by mentioning it in a conversation in the team's chatroom an delegating to the bot the job of tracking and reminding people to complete their tasks (see Figure 1).

We deployed Task Bot in eight different teams of information workers, who delegated to Task Bot the tracking of 88 tasks. The key findings from our deployments were that people feel comfortable to assign tasks to others using Task Bot independently of their hierarchical level; people treated Task Bot politely as a team member; people found useful to remind them and other about their pending tasks; dealing with multiple tasks is challenging through conversational interfaces; people prefer to set soft deadlines instead hard deadlines; people wanted to use Task Bot in their most used communication channel.

Intent	Examples
Deadline Setting	*I'll be done tomorrow*, or *Next week*, or *Thursday*.
Task Completion	*I'm done already*, or *It's finished*, or *Yes*.
Task Rejection	*Cancel the task*, or *Unassign*, or *No*.

Table 1. Reactive messages intents in response to "Are you done with the task? If not, when do you think you'll finish?"

RELATED WORK

Chatbots began attracting attention in the formative days of artificial intelligence research [24] as a vehicle for demonstrating machine understanding [25]. While early experiments focused on free-form conversations [26], bots have recently become more popular as single purpose tools, performing a complex tasks for people via a conversational interface [11, 14]. We build on this more utilitarian take on bots, offering a convenient way for teams to track and manage their tasks from within a conversation.

Previous research has looked at the intersection between task tracking and communication. Some systems seek to extend or enhance the email client with extra capabilities to track tasks [8, 9, 27]. Our work differs from this approach, in that it does not rely on any particular messaging client, making it more inter-operable with the tools and workflows of peoples' choosing. Other work has looked at how to proactively identify tasks in messages [15, 28, 23, 10], and for example, classify them and present in a task-centric interface from where users could easily find people who had performed that task before [15]. While these seamless, proactive approaches show promise, in practice, it is difficult to accurately detecting the users' precise intentions about a task, so we opt to give them control over when and how they invoke the bot.

A number of products exist for collaborative task tracking for teams, including Trello [3], Asana [1], Wunderlist [6], and Visual Studio Online [5]. While these products vary considerably in terms of offerings, generally they allow teams to create tasks records, add implementation and execution details, assign tasks to team members, track the task progress, and develop time-lines or plans of execution for pending work. Task Bot extends this basic approach, offering a conversational interface to the core functionality of these tools, using Wunderlist as its backend. Some companies have begun exploring bots for tracking task, including To-do bot for slack [2] and the Trello bot for Microsoft Teams [4], however, these bots have taken an command-driven approach to building a conversational user experience, rather than focusing on free form conversational interactions, as does Task Bot.

TASK BOT USER EXPERIENCE

Task Bot helps teams manage their tasks from within platforms they use to communicate. Although Task Bot is available on several different platforms, such as email, Skype for Business, and Slack, in this work we focus on Microsoft Teams, a group chat platform for the workplace.

Once an administrator adds Task Bot to a chatroom, anyone on the team can then interact with it by @-mentioning Task Bot and some of their teammates in a message. For example, "Ada" might summon Task Bot to help her assign a task to her colleague "John" and follow up with him until he completes the it:

```
Ada:  Prepare the slides for the presentation
@John @TaskBot
```

We adopt the model that if Ada had mentioned two or more people in her message, then Task Bot assigns the task to all mentioned parties. We settled on this approach after trying, and discarding, other methods, including a round-robin approach where Task Bot first asked each person if they could work on the task.

Shortly after Ada's message is received, Task Bot acknowledges the request and starts interacting with John through the direct messaging functionality found in group chat tools like Slack or Microsoft Teams. Because these conversations are one-on-one with the bot, they do not add noise to the to the rest of the team's conversations:

```
Task Bot:  Hi John, Ada recently asked you to
take this task:  "Prepare the slides for the
presentation." When do you think you can have
it done?

John:  Tomorrow
```

Once John promises to finish the task at a particular time, Task Bot passes on this approximate completion date to Ada:

```
Task Bot:  Hi Ada, John said that the task
"Prepare the slides for the presentation" will
be done by "Tomorrow"
```

Task Bot reaches out to John again if the task has not been completed by the deadline set by John himself:

```
Task Bot:  Are you done with the task "Prepare
the slides for the presentation"?  If not,
when you think you'll finish?
```

It is worth noting that in one of our early versions, Task Bot would check on people's tasks twice a day (in the morning and in the afternoon) during workdays, but we dropped this feature because participants considered it too annoying.

John can then respond he's done with the task. Task Bot will mark it as complete, ending the life-cycle of the task.

```
John:  Yes, I'm done

Task Bot:  Great job, John!  I marked this
task as complete.
```

IMPLEMENTATION

We built Task Bot using the Microsoft Bot Framework's Bot Builder SDK for Node.js. The state of the tasks were stored in Wunderlist and accessed by Task Bot via API. For this study, all tasks were stored under a central Wunderlist account for

Intent	Example
Task assignment	*@Bob, don't forget to finish the report. @TaskBot*
Self-Reminders	*@TaskBot, please remind me to leave at 5pm.*
Greetings	*Thanks, Task Bot!*
Task Termination	*I've completed this task,* or *Please cancel.*
Help	*Help me!*
Other	Catch-all: Task Bot replies with "I didn't understand."

Table 2. Proactive message intents to new requests.

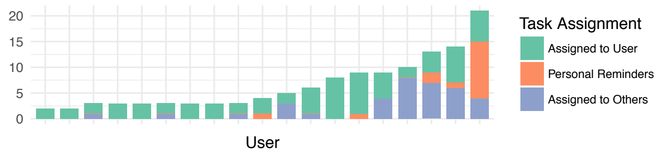


Figure 2. Usage statistics of 19 users in the study, by assignment type, showing number of tasks assigned to the user by others, assigned to others by the user, and personal reminders (tasks assigned to one's self).

which the bot was the owner, but in practice, we envision people integrating Task Bot with their own accounts on task tracking platforms of their choosing.

We implemented a model for detecting intents from users' utterances using the Microsoft Language Understanding Intelligent Service (LUIS), focusing on two possible categories of messages Task Bot receives from a user:

1. Reactive messages in response to a key Task Bot's question: "Are you done with the task? If not, when do you think you'll finish?" (see Table 1).
2. Proactive messages, that is, unprompted messages people post in a group chat mentioning Task Bot, or send to Task Bot via direct message (see Table 2).

We trained this model by hand at first, based on lists of keywords we developed to capture these intents, and then we iteratively improved it with users' actual utterances in several rounds of pilot testing, first within our own team, then in our extended research group.

METHODS

We conducted a week-long field deployment of Task Bot with eight teams of information workers ranging in size from two to five people, with 19 people in total. Participants were recruited from a large Fortune 500 company, and were contacted via email using company mailing lists. Five of the eight teams were hierarchical, consisting of a manager and one or more of their subordinates, while the others were peers at the same level. All teams were collocated, and communicated via synchronous, asynchronous, and face-to-face interactions.

A requirement for participation was that the teams of two or more people had to be working closely together on the same project, and all members of the team had to agree to be part of the study. Each team was given a brief tutorial on how to use Task Bot, including the creation of an example task. Over the course of one work week, each participant was asked to create at least three tasks using Task Bot. We also administered pre- and post- study surveys. In gratuity, each participant received a $20 gift card.

After the field study, all the messages written by users to Task Bot (177) were independently coded by two of the authors into 27 categories, that were inductively and iteratively developed to characterize usage behavior. The two independent classifications agreed on 95.6% of all the messages (Cohen's kappa = .76). The remaining discrepancies were resolved through aggregation or discussion as appropriate.

RESULTS

Over the course of the field deployment, 88 tasks were created by participants, averaging 4.4 per user, of which 65 were ultimately marked as complete by the assigned person. Figure 2 shows how these tasks are distributed across users and types of task assignment. The median number of tasks created per team was seven. In tracking the progress of these 88 tasks, 177 messages were received by Task Bot from the participants. 22% of these messages were tasks assignments. On average, users sent 12 messages to Task Bot, while users received 54 messages from Task Bot. Lastly, on average, users received 16 reminders to complete their assigned tasks.

From the post study survey, we learned that six of the 11 people who answered felt more productive, and 8 people will use it in the future, however, 4 people reported finding it annoying. When noting features that were important to them, most said that reminding other people about tasks (10/11), and tracking the progress of tasks (9/11) as important. Six people did not find it important that Task Bot to interface with external task tracking software.

Users particularly enjoyed the ease with which they were able to create and track tasks from the conversation in-situ: "*I liked being able to jot tasks down and have them get tracked without needing to go to a separate tool to track them.*" However, people expressed concerns about the complexity of using a conversational user interface for handling many tasks: "*when I received multiple messages I didn't know how to respond to a specific message/reminder.*" People also requested to use Task Bot on other channels instead of Teams that they use more, for example, Skype for Business. Five people perceived reminders from Task Bot as invasive or annoying. Some cited the frequency of reminders as the main problem: "*The reminders were annoying and too frequent.*" Others pointed to their lack of context: "*reminders are not context sensitive and would appear at random times and often caused me to task switch when I was focused in doing something else.*"

DISCUSSION

In this section, we highlight seven different patterns we observed from people's interactions with Task Bot that are relevant lessons for designing work-centric chat bots.[1]

Handling human-like interactions with bots

People often treated the bot in a human-like manner, conversing with it naturally, and politely, using words like "please" and "thanks," which was somewhat unexpected. We coded all messages for signals of this type of behavior and we found that almost all the participants (93%) interacted with Task Bot as if it was a human at least once. In fact, some users (20%) never interacted with Task Bot as if it was a bot. For example, one of the participants said:

```
P1:  Hi @TaskBot, could you remind @P2 to read
the article I shared with him?  Thanks!
```

Unfortunately, when participants interacted with the bot in polite or human-like ways, the bot often failed to understand

[1]Our assessment of generalizability is based on prior experience building bots (which we redact for anonymity).

them (44%) because the natural language model hadn't been trained to handle those utterances, or they asked for somethign beyond its capabilities.

Designers of chatbots in the workplace should either, explicitly build signals that their bot is not personified, or be resilient to the range of possible responses when when people interact with the bot in more human ways.

Supporting self-communication

Even though our training focused on task assignment to others, five users asked Task Bot to create reminders for themselves:

```
P3:  Remind me at 10:15 to leave
```

In some cases this was a way for people to get started without bothering others, but for some it became a common practice, not unlike emailing oneself was probably not the intended goal of email but emerged as a common practice for note taking.

Designers of social chatbots should assume that bots would also be used for self-communication, either as a way to test the system or as a practical use of the tool.

Defusing hierarchical awkwardness

We observed people using Task Bot to assign tasks to people across different hierarchical levels. The percentage of total tasks that were assigned to managers by their subordinates (35%), was almost the same as those assigned to subordinates by their managers (31%). The rest (34%) were among people at the same level.

We believe that perhaps using Task Bot as proxy helped reduced the social friction of assigning tasks to someone with higher position in the hierarchy.

Designers of social chatbots can leverage bots' ability to defuse potentially awkward interactions with people of higher hierarchical status in a work organization.

Failing gracefully

Since participants did not know about all the features that Task Bot had and Task Bot was not trained to understand any message, there were some messages (10.5%) that Task Bot could not understand, and therefore were not answered successfully, for example two of the messages that Task Bot failed to understand were:

```
P7:  How do I reply to a question?

P8:  Can you send the consent form again?
We've filled out the survey.
```

So, Task Bot simply replied with a canned response, "Sorry, I could not understand. Could you rephrase?"

Designers should be bots that fail gracefully when users ask for novel scenarios, and those failures should be categorized as such in order to use that data to improve future version of the bot. Designers should come up with mitigation strategies such as amusing error messages (e.g. Twitter's fail whale), and provide escalation to humans [11].

Dealing human ambiguity

Out of all the dates and times people mentioned in messages, 29.3% of them were ambiguously defined, with 60% of users mentioned at least one ambiguous date. For instance:

```
P3:  Before tomorrow morning
```

Sometimes people are intentionally ambiguous in conversation [21], so designers of chatbots need to build strategies to resolve certain ambiguities to function according to user expectation.

Identifying people's name in conversations

Some of the Instant Messaging and group communication channels use a specific syntax to mention a person within the messages in order to be notified. The syntax was a problematic factor in how people assigned tasks to others, because Task Bot associates the task to the people mentioned. 40% of the users experienced issues mentioning members in a proper way, sometimes they forgot to type the symbol "@" before the name, sometimes they did not know how it is used in MS Teams. This is a message example:

```
P4:  Hey P5, can you finish your tutorial?  cc
@TaskBot
```

Designers should find ways of nudging users to mention people in the ways communication channels expect (e.g. using the at sign), such that bots can tell when someone is being mentioned.

Handling multi-threaded conversations

One of the biggest challenges for Task Bot and other bots is the difficulty of maintaining multiple active conversations at the same time. For example, some users had multiple tasks assigned to them, and when they told the bot "I'm done with the task," the bot would not know which task they were referring to. We implemented a solution for this in Task Bot, using a menu for canceling and completing tasks, that would list all active tasks.

Designers should invest in technology for determining which active conversation thread a new incoming message belongs to.

CONCLUSION AND FUTURE WORK

In this paper we introduce Task Bot, a bot designed to help teams manage their tasks. Users delegate the tracking of their tasks to Task Bot. We described our approach to designing Task Bot, and shared the lessons that we learned from deploying it with eight teams. We focused on identifying design considerations for other bot designers building conversational user interfaces for workplace.As for Task Bot, future work will focus on the following features: exploring the use of multiple communication channels (e.g. email, Skype, etc.), better handling of multi-threaded conversations, and more sophisticated ways of assigning tasks to people based on the task description.

REFERENCES

1. Asana. **http://asana.com**. (Accessed on 09/18/2017).

2. To-do bot by workast — slack app directory. `https://cartheftbot.slack.com/apps/A0HBTUUPK-to-do-bot-by-workast`. (Accessed on 09/18/2017).
3. Trello. `http://trello.com`. (Accessed on 09/18/2017).
4. The trello app for microsoft teams - trello help. `http://help.trello.com/article/1086-the-trello-app-for-microsoft-teams`. (Accessed on 09/18/2017).
5. Visual studio online. `https://www.visualstudio.com/vso/`. (Accessed on 09/18/2017).
6. Wunderlist. `http://wunderlist.com`. (Accessed on 09/18/2017).
7. Anh, N.-D., Cruzes, D. S., and Conradi, R. Dispersion, coordination and performance in global software teams: A systematic review. In *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '12, ACM (New York, NY, USA, 2012), 129–138.
8. Bellotti, V., Ducheneaut, N., Howard, M., and Smith, I. Taking email to task: the design and evaluation of a task management centered email tool. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (2003), 345–352.
9. Bellotti, V., Ducheneaut, N., Howard, M., Smith, I., and Grinter, R. E. Quality versus quantity: E-mail-centric task management and its relation with overload. *Human-computer interaction 20*, 1 (2005), 89–138.
10. Corston-Oliver, S., Ringger, E., Gamon, M., and Campbell, R. Task-focused summarization of email. In *ACL-04 Workshop: Text Summarization Branches Out* (2004), 43–50.
11. Cranshaw, J., Elwany, E., Newman, T., Kocielnik, R., Yu, B., Soni, S., Teevan, J., and Monroy-Hernández, A. Calendar. help: Designing a workflow-based scheduling agent with humans in the loop. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ACM (2017), 2382–2393.
12. Czerwinski, M., Cutrell, E., and Horvitz, E. Instant messaging and interruption: Influence of task type on performance. In *OZCHI 2000 conference proceedings*, vol. 356 (2000), 361–367.
13. Czerwinski, M., Horvitz, E., and Wilhite, S. A diary study of task switching and interruptions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (2004), 175–182.
14. Fast, E., Chen, B., Mendelsohn, J., Bassen, J., and Bernstein, M. Iris: A conversational agent for complex tasks. *arXiv preprint arXiv:1707.05015* (2017).
15. Faulring, A., Myers, B., Mohnkern, K., Schmerl, B., Steinfeld, A., Zimmerman, J., Smailagic, A., Hansen, J., and Siewiorek, D. Agent-assisted task management that reduces email overload. In *Proceedings of the 15th international conference on Intelligent user interfaces*, ACM (2010), 61–70.
16. Iqbal, S. T., and Horvitz, E. Disruption and recovery of computing tasks: field study, analysis, and directions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (2007), 677–686.
17. Iqbal, S. T., and Horvitz, E. Notifications and awareness: a field study of alert usage and preferences. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, ACM (2010), 27–30.
18. Kip. What bots can do, that websites and apps cant, mar 2017.
19. Lin, B., Zagalsky, A., Storey, M.-A., and Serebrenik, A. Why developers are slacking off: Understanding how software teams use slack. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*, ACM (2016), 333–336.
20. Mark, G., Gudith, D., and Klocke, U. The cost of interrupted work: more speed and stress. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ACM (2008), 107–110.
21. Rong, X., Fourney, A., Brewer, R. N., Morris, M. R., and Bennett, P. N. Managing uncertainty in time expressions for virtual assistants. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ACM (2017), 568–579.
22. Sanchez, R., Jin, J., Maheswaran, R. T., and Szekely, P. Interfaces for team coordination. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*, IUI '08, ACM (New York, NY, USA, 2008), 427–428.
23. Sappelli, M., Pasi, G., Verberne, S., de Boer, M., and Kraaij, W. Assessing e-mail intent and tasks in e-mail messages. *Information Sciences 358* (2016), 1–17.
24. Shawar, B. A., and Atwell, E. Chatbots: are they really useful? In *LDV Forum*, vol. 22 (2007), 29–49.
25. Turing, A. M. Computing machinery and intelligence. *Mind 59*, 236 (1950), 433–460.
26. Weizenbaum, J. Elizaa computer program for the study of natural language communication between man and machine. *Communications of the ACM 9*, 1 (1966), 36–45.
27. Whittaker, S. Supporting collaborative task management in e-mail. *Human–Computer Interaction 20*, 1-2 (2005), 49–88.
28. Yang, L., Dumais, S. T., Benne, P. N., and Awadallah, A. H. Characterizing and predicting enterprise email reply behavior. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2017, Tokyo, Japan* (2017).