How can ChatOps improve DevOps pipelines?

Automated Software and DevOps
DD2482

Elin Ryman

eryman@kth.se
April 30, 2021

1 Introduction

As more and more companies are incorporating the philosophies, practices, and tools of DevOps, the need for improved communication within and between teams grow. Traditionally, the two worlds of development and operations were separated and siloed but in DevOps, these parts are integrated creating a smooth connection and workflow [4]. The two teams, development and operations, are sometimes even merged into one team where the engineers work across the entire application lifecycle. In such merged teams, the engineer can work in all parts of the pipeline, such as development, testing, deployment and operations, which leads to a range of skills not limited to a single function [1].

Because DevOps requires cross-functional teams and improved collaboration between multiple teams, communication becomes essential. The increased usage of chat applications and instant messaging applications for developers and other team members makes it natural to integrate tools into the existing workflow to improve the collaborative work environment [2]. Integrating bots into an existing, frequently used chat environment to improve communication is the idea behind ChatOps.

2 Background

ChatOps is an abbreviation for chat-based operations and centers around conversation-driven development [11]. It is an unobtrusive way to connect humans with the technical systems and the processes they work with, in a familiar work environment that they use on a daily basis. ChatOps is made up of three main components, namely the collaboration tool, the bot and the system integration. The integration of these components enable everything to be centralized into one single point of interaction, which in turn facilitates collaboration. Below, these components will be introduced.

2.1 Collaboration tool

Most people have probably used a chat client like Slack, HipChat or Flowdock at work or in other projects, and these are the collaboration tools needed for ChatOps to work. The collaboration component is the foundation for ChatOps and is located between the DevOps team and the bot.

The corporation behind the collaboration tool Slack made a user survey and found that, on average, teams using Slack increased their productivity with 32%. The average usage of email was reduced with 48.6% and time spent in meetings was reduced with 25%. The survey also showed that transparency within teams was increased by an average of 80% [12]. These results shows that collaboration tools alone can provide immense benefits for collaboration and productivity.

2.2 Chatbots

Another component within ChatOps is bots, which are executing commands in the chat client. The first, and the most widely forked, ChatOps bot is Hubot [11]. Hubot is a bot by GitHub written in CoffeeScript and Node.js. The bot was initially developed as an internal tool to automate GitHub's intra-company chat room which was later rewritten and made publicly available as open source [6]. Two other commonly used chatbots are Lita and Errbot. Lita is written in Ruby and Errbot in Python [11]. These are the most mentioned bots, but there are many other bots available with different functionality.

Chatbots can be customized to do a broad variety of task with the help of open-source plugins and scripts. For instance, there are plugins for everything from interesting cat facts to programs for managing coffee orders or a queue with users who need help with debugging [8]. Most chatbots can also be customized by writing scripts that fits the specific needs of the organization that intends to use the bot.

2.3 System integration

The last component in ChatOps is the system integration already used by the DevOps team. It is the DevOps tool for continuous integration, issue tracking, monitoring etc. Some examples of the systems integrated with the chatbots are Jenkins, Docker, Ansible, AWS CloudFormation, GitHub and Travis.

By connecting the chat platform used by the DevOps team to their build system with a chatbot, teams can communicate with their applications and infrastructure as they would with their teammates. By using ChatOps, teams can get notified when something happens in the project, deploy code and execute processes on their CI servers through the chat application [13]. When several tasks can be centralized to one place, like a chat application, the tasks get automated and the teams can work more efficiently, cheaper and collaborate better.

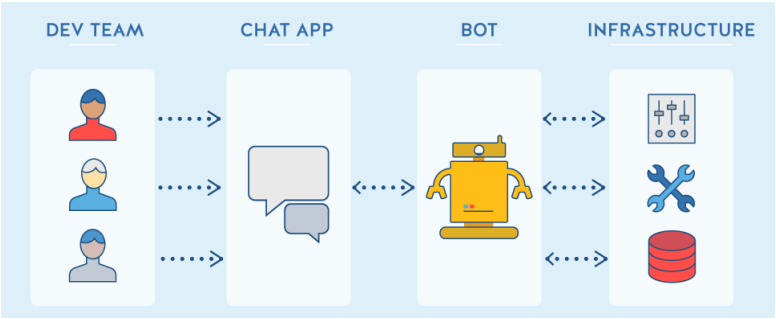

Figure 1: Components of ChatOps. [3]

3 How ChatOps can be used in DevOps pipelines

ChatOps is simply a tool to integrate a team's discussions and their workflow with a chat room to improve collaboration and communication. As previously discussed, bots can be tailor-made to a specific company by plugins and custom-made scripts. This means that there are a wide variety of tasks the bots can do. There are plenty of examples and scenarios where ChatOps can be used within a DevOps team, a selection of which is discussed below. The examples are divided into three different categories, namely information sharing, incident management and automation.

3.1 Information sharing

One of the most fundamental reasons for using collaboration tools is information sharing between and within teams. In the comment stream, employees can discuss problems and ideas, share their expertise, and ask questions. One way ChatOps can improve collaboration is to make the information shared in the comment stream more accessible. Even though there is a search function in the collaboration tools, searching and scrolling through endless of comments might not be the most efficient way to find answers. This is where chatbots like Guru can assist [5]. Guru is a bot that integrates with the chat client Slack to capture, share, and access the knowledge from teams and keep everything in one single location. When someone posts something in Slack, an option to create a Guru card becomes available. The Guru card holds the information from the post and the person creating the Guru card can categorize it. The information will then be stored in a categorized and searchable wiki [5]. This wiki can be useful for all employees and can result in fewer repeat questions and less time searching for an answer within Slack.

Information sharing and knowledge management can benefit a broad range of employees and not only the technical teams. For example, the information from a wiki can speed up the onboarding process so that new employees easily can read about processes and get tips from their colleagues. Teams like marketing, sales and finance can also reap benefit since they have better insight into what is going on in a project, when code will be deployed and who is responsible for certain tasks. Instead of walking over and interrupting their colleagues with a question, the relevant information can be extracted directly from the bot.

3.2 Incident management

ChatOps can also be used for incident management. When a technical issue arises, a bot can automatically issue an alert to the integrated communication tool where team members can collaborate to resolve the issue. The bot brings the incident workflow into one location for better communication and faster management. By keeping everyone up to speed in real-time and centralizing the information flow to one single point, incidents can be resolved faster. Afterwards, the information in the chat from the incident management can become a

detailed record of what went wrong and why. With a bot like Guru, the record can be collected and stored in a searchable wiki [5]. If the chatbot is proactive, it can even set up a new chatroom dedicated to the incident, populate it with necessary information and invite relevant team members. By providing a fast and clear overview of what happened and what the most recent changes were, the chatbot enables the team members to focus on the problem solving within the dedicated chatroom.

For instance, Shopify's incident manager is assisted by their proprietary chatbot called Spy during incidents. Spy is a bot that steams from Lita and features several incident commands that can be ran right from the chat client Slack. Integrating Spy has helped to streamline the incident response by reducing the manual effort. Spy allows status pages to be updated from the chat platform, keeps records of the event times, and generates a summary of the incident [9].

3.3 Automation

One of the most useful features of ChatOps is task automation, where manual tasks are automated with the help of a bot. By automating tasks such as deployment, builds and testing, a lot of time can be saved. Bots can also automate less complicated tasks, such as notifying everyone that have not yet reacted to a specific Slack post, instead of requiring someone to manually check and follow-up. Instead of asking teammates the same questions separately, teammates can be provided with a poll with the help of the aptly named bot Polly [10]. Tracking the time spent with various tasks in a project can be done with help from the bot Harvest [7].

4 Advantages and disadvantages

Several advantages associated with using ChatOps in DevOps pipelines have already been mentioned in this essay. These advantages include increased transparency, better communication and greater efficiency in teams as well as faster incident management to only mention a few. Another benefit is that the interface of a chat client is more user-friendly than the tools software developers are used to work with. Some bots even utilize natural language. By asking the bot "what is wrong with this program", it can answer by fetching the information, making it very intuitive and easy to use. As mentioned in section 3.3 about task automation, ChatOps results in time savings. Bots also help save time by virtue of centralizing the entire workflow into one single location, the chat room. Automation also limits the risk of human error, which subsequently decreases the number of bugs and incidents.

The advantages of ChatOps in DevOps teams are many, but are there any issues? While there are no obvious or serious downsides associated with usage of ChatOps, certain smaller issues can arise. For example, moving the entire workflow and the employees to one location could lead to a noisy environment. The number of posts and conversations will inevitably increase as more and

more employees start using ChatOps within an organization. There is a fine line between a useful chat filled with important information that facilitates collaboration, and a less valuable tool where there is too much information so that relevant data might get lost. The level of noise can however be kept low even with many participants if the chat tools are used correctly. For instance, establishing rules for which rooms, channels and functions should be used for certain tasks could prove helpful.

5 Discusson and conclusion

As this essay shows, ChatOps can improve DevOps pipelines in several different ways. Advantages include improved information sharing, faster and more efficient incident management and less human errors. It is clear that DevOps teams could generally benefit from incorporating ChatOps into their workflow, and the possibilities are vast. An integration of ChatOps without proper thought and planning might take more time to set up, but even then, there are very limited potential disadvantages. It will take time to set up and educate the workforce, but once ChatOps is integrated and up and running, the time that has been invested will yield great improvements on a continuous basis.

It is not easy to exhaustively set out all the ways in which ChatOps can be used to improve DevOps pipelines, but it clear that investing in ChatOps will lead to material improvements in many different areas. The possibilities with ChatOps are ultimately limited only by the imagination of the developers.

References

[1] Aws. What is DevOps? URL https://aws.amazon.com/devops/what-is-devops/.

[2] G. Barnott. ChatOps, DevOps, ScrumOps and 5 Other Ops religions, 2020. URL https://www.devopsonline.co.uk/chatops-devops-scrumops-and-5-other-ops-religions/.

[3] B. Doerrfeld. 12+ Frameworks to Build ChatOps Bots, 2016. URL https://nordicapis.com/12-frameworks-to-build-chatops-bots/.

[4] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano. Devops. *IEEE Software*, 33(3):94–100, 2016. doi: 10.1109/MS.2016.68.

[5] GetGuru. Features. URL https://www.getguru.com/features .

[6] GitHub. Hubot. URL https://hubot.github.com.

[7] Harvest. Harvest + Slack. URL https://www.getharvest.com/integrations/slack.

[8] Lita. Plugins. URL https://plugins.lita.io.

[9] D. Niyonkuru. Implementing ChatOps into our Incident Management Procedure, 2018. URL https://www.getguru.com/features .

[10] Polly.ai. Polly for slack. URL https://www.polly.ai/slack-poll?utm_source=slackutm_campaign=installutm_medium=app_directory.

[11] E. Sigler. What is ChatOps?, 2014. URL https://www.pagerduty.com/blog/what-is-chatops/.

[12] Slack. Slack Survey Results. URL https://a.slack-edge.com/7b00/img/survey/slack_survey_results.pdf.

[13] R. ZYANE. How ChatOps Can Help You DevOps Better, 2017. URL https://chatbotsmagazine.com/how-chatops-can-help-you-devops-better-5-minutes-read-507438c156bf.