How Bots may Play a Role in Your Development Project

XXX
XXX

1. THE EMERGENCE OF BOTS

From the earliest days of computer programs, developers have imagined the emergence of programs that can act, talk and think like humans. Such programs would not only automate tasks that humans might do, but they could also work with humans to solve intellectual tasks that cannot be entirely automated. The hope, even as far back as 1966, was for these programs to pass the Turing Test (proposed in the 1950 paper Computing Machinery and Intelligence [s]), where humans could be fooled into believing that they are interacting with an intelligent human rather than a mere program (reference Eliza Bot [x]).

The term "bot" was used to describe the realization of this vision quite early on, but today the term is used to describe a conversational style user interface or an anthropomorphized script or agent that can automate rote and tedious tasks. Such bots today may fetch or share information, extract and analyze data, detect and monitor events and activities in communication and social media, connect users with each other or with other tools, or they may provide feedback and recommendations on individual and collaborative tasks. Bots are not (typically) intended to fool the end user into believing a bot is a real person, but many bots do have a personality that is engaging and pleasant to interact with.

The term "chatbot" is commonly used to emphasize the conversational style user interface of many bots, especially those that use natural language processing rather than simple commands. Bots typically reside on popular platforms where users work or play with other users, and frequently integrate other services and micro-services into these channels, providing a conduit between users and other software services.

Bots are rapidly becoming a *defacto* interface for interacting with services in just the past few years. In part, this is because of the widespread adoption of messaging platforms (e.g., Facebook for social networking and Slack by developers) and in part because of the advancement of natural language processing, which many bots (but not all) leverage.

Another driver is the prevalence of "big data" and machine learning algorithms for analyzing data across many domains. Bots provide a convenient way for developers to generate a user interface for interacting with these algorithms and data.

Developers use bots as well as create them. For developers, the transition from command line interfaces to interacting with bots through the messaging interface feels intuitive and has the advantage of bringing transparency of invoking and customizing services in a communication channel, while non-technical users are also embracing the notion of bots as opposed to installing and relying on apps that are not well integrated with their messaging environment.

All of the major companies clearly recognize the value that bots bring in terms of integration of services, users and communication channels. Facebook aims to "replace apps" one bot at a time in their messaging platform [x], while Microsoft claims that "conversation as a platform" is the operating system of the future rather than Windows [x]. Amazon's Alexa, Apple's Siri and Google's Now platform are also showing agreement with this rapid shift towards bots. Software developers will also recognize many bots in the platforms they frequently use to connect with other developers and services, such as Slack [x] and HipChat [x].

In this column, we first discuss how bots can be used to automate many end user tasks and to fill numerous roles across a wide variety of user domains. We provide some background for developers on how they can create bots and how they can host these bots on various platforms. These platforms support one or more frameworks that developers can leverage to support the rapid creation and design of bots. We also discuss some design guidelines for bots, pulling ideas from the emerging area of human-robot interaction. In a sidebar, we look at how developers themselves use bots in their development work. We call out this domain because to quote XXX from StackOverflow, "developers are writing the script for tomorrow" and thus we see examples of sophisticated and innovative bots emerging that pave the way for how bots may be created for other domains. Finally, we conclude this column with some considerations and advice for developers wishing to develop or use bots within their own development projects.

2. BOTOLOGY: UNDERSTANDING THE LANDSCAPE OF BOTS

It has been surprising to see the rapid development and widespread adoption of bots in just a few years, but what is more surprising is the vast array of bots that support

WOODSTOCK '97 El Paso, Texas USA

a variety of tasks and roles. Rather than attempting to narrowly define bots or chatbots, instead we look at how they may be categorized.

One way to characterize bots is through the **interaction model** they provide. Some bots support a domain specific language where the user interacts with the bot using specific commands. For example,.... Other bots rely more on natural language processing, such as Apple's Siri. Bots also differ in terms of whether they support a pull based approach, where the user initiates the interaction (e.g., Hey Siri), or the bot may initiate the interaction based on some system or user context (e.g., ...).

Another way to characterize bots is in terms of their **intelligence**. Some bots follow very simple logic rules, passing control to a human user if needed (e.g., IFTT), whereas others tend to be more autonomous and learn over time. For example, ...

Finally, bots may be characterized by the kind of **tasks** that they support, as we summarize here:

- *transactional* bots provide support for user transactions such as...
- *informational* bots provide support for...
- *productivity* bots aim to improve a user's productivity by automating rote or tedious tasks such as ...
- *collaboration* bots support how users communicate, coordinate or collaborate their tasks. For example, ...

3. HOW TO CREATE BOTS AND WHERE TO HOST THEM

[Carly]

Although simple bots can be built from scratch and self-hosted, many developers choose to leverage third-party technologies to streamline the process. With the explosion of new tools to match in the bot development domain, if is important to distinguish between the tools used to build bots (Creation Platforms) and the platforms where they live (Distribution Platforms).

Many large companies, such as Microsoft and Facebook for example, offer a comprehensive set of tools to support both the creation and distribution of bots. Other companies, provide specialized solutions for specific aspects of bot creation and distribution.

3.1 Distribution Platforms

The distribution platforms define where and how the bots are accessed by users. They range from general purpose social networks (e.g. Facebook Messenger) to domain specific channels (e.g. Slack, Teams, HipChat).

Distribution platforms offer many benefits for developers: access to an existing user base[1], defined interaction mechanisms, discovery[1], and monetization[1].

Launching a bot on an existing platform, helps overcome the cold start problem that many new applications face. Selecting the wrong user base, however, can be equally detrimental. Carefully consider the size and demographics of the user base as well as the cost to access the platform.

[1]https://medium.com/mobile-lifestyle/messenger-vs-skype-vs-slack-vs-telegram-how-to-spot-the-winners-adc34b4ca066

Distribution platforms also define and standardize how users can interaction with bots. Commonly support interaction mechanisms include: text, user can communicate with the bots using natural language; commands, users can trigger actions with a set of keywords or phrases; selection, users can choose from a set of presented options; and voice, users can talk to the bots.

Distribution platforms also offer mechanisms promotion or for user discovery of the bots. Similar to Apple's App-Store [x], many distribution platforms offer virtual "bot stores" where users browse for bots. Third-party sites (e.g. BotList) also offer online catalogs of bots for many popular distribution platforms.

Mature distribution channels may also offer means of collecting payment from users, particularly usefully when developing transactional bots.

A summary of popular distribution platforms and technologies is included in figure X.

3.2 Creation Platforms

Developer ecosystem for the different platforms... plug in to these communities...

The creation platforms support the design and development of bots. This support often comes in the form of software foundations (e.g. frameworks, tool kits, APIs, or other services). These bot creation services can be platform specific or produce bots that can be deployed across multiple platforms (Microsoft Bot Connector). The services they provide range from documentation and code templates, to no-code-required bot building interfaces. Many of the popular bot creation platforms also offer vibrant developer ecosystems. Developers can plug into these communities for developer expertise in the form of tutorials, articles, discussions, and support.

A summary of popular creation platforms and technologies is included in figure X.

3.3 Design guidelines

4. TAKEAWAYS

[Peggy]

Bots are everywhere: pay attention to how bots are used around you, in cars, in the home but also watch how other developers use Bots in their development activities (reference FSE visions paper)

Bot ethics: Reference Asimov's rules for Robots here.. Should this be updated for Bots?

Bots may ease or increase collaboration friction... choose Bots not to remove collaboration (and in turn creativity) but use them to enhance collaboration (reference CSCW workshop paper)

5. SIDEBAR

[Peggy]

Software bot usage in software development.

6. REFERENCES