2019 IEEE/ACM 1st International Workshop on Bots in Software Engineering (BotSE)

Defining and Classifying Software Bots: A Faceted Taxonomy

Carlene Lebeuf*, Alexey Zagalsky†, Matthieu Foucault† and Margaret-Anne Storey†
*Microsoft, USA
Email: calebeuf@microsoft.com
†The CHISEL Group, University of Victoria, Canada
Email: alexeyza@uvic.ca, mfoucault@uvic.ca, mstorey@uvic.ca

***Abstract*—While bots have been around for many decades, recent technological advancements and the increasing adoption of language-based communication platforms have led to a surge of new software bots, which have become increasingly pervasive in our everyday lives. Although many novel bots are being designed and deployed, the terms used to describe them and their properties are vast, diverse, and often inconsistent. Even the concept of what is or is not a bot is unclear. This hinders our ability to study, understand, design, and classify bots.**

In this paper, we present a taxonomy of software bots, which focuses on the observable properties and behaviours of software bots, as well as the environments where bots are deployed and designed. We see this taxonomy as a focal point for a discussion in our community so that together we can deeply consider how to evaluate and understand existing bots, as well as how we may design more innovative and productive bots.

***Index Terms*—software bots, taxonomy, classification, software engineering**

I. Introduction

"The bot family tree is a confused and contradictory plant, a warped and twisted structure, as unlike Darwin's great Tree of Life as a blackberry bush is unlike a weeping willow." [1]

From the earliest days of computer programming, people have dreamed about creating software programs that could think and behave like humans [1]. Such programs would not only automate tasks that humans perform, they would also work with humans to solve intellectual problems that cannot be entirely automated. The term "bot" was used to describe a realization of this vision quite early on [1].

In just the past few decades, we have witnessed an uprising of the next generation of software bot technologies, which are becoming increasingly pervasive in our everyday lives. Recent technological advancements and the adoption of language-based messaging platforms (e.g., Microsoft Teams, Slack) have led to a surge of new ubiquitous software bots [2]. Bots are replacing not just simple tasks but also very complex software applications. However, we worry that a poor understanding of this shift could lead to unanticipated challenges and risks experienced by both bot users and creators.

Although many novel bots have been designed and widely deployed, the terminology used to describe them and their properties is inconsistent. This is due to their development (and adoption) across multiple technical disciplines (including HCI, AI, SE, and systems) and diverse domains (e.g., health, software engineering, commerce, and entertainment). Indeed even definitions about what is or is not a bot vary considerably. This is not surprising given the wide range of software applications that are commonly referred to as "bots". The use of the term *bot* varies from describing simple scripts that automate a task in the background, to complex applications that interact with one or more humans and autonomously adapt to activities that people and other systems do, and even all the way to software applications that use AI and natural language processing (NLP) to mimic human behaviour and intelligence. We believe that, as a research and design community, we need an agreed upon taxonomy so that we can together more effectively compare, classify, evaluate, and design new bots.

In this paper, we provide a **definition** of software bots and present a **taxonomy** developed through an extensive literature review and consideration of a broad landscape of bots [3]. We used Usman *et al.*'s process for taxonomy generation [4], part of which involved a card sorting activity to develop the main facets (and subfacets) in the taxonomy [3]. We propose a **faceted taxonomy** so that bots may be classified using multiple perspectives.

Faceted taxonomies are commonly used in software engineering and are particularly suitable for classifying complex entities where "each facet is independent and can have its own classes, which enable facet-based taxonomies to be easily adapted so they can evolve smoothly over time" [4]. As bots and their properties tend to evolve rapidly, any taxonomy about them needs to be easy to expand and adapt. The taxonomy we propose has three main facets that refer to:

- the properties of the **environment** that the bot was created and operates in;
- the **intrinsic properties** of the bot itself; and
- the bot's **interactions** within its environment.

We hope it will serve as a focal point for discussion so that our community can share insights and knowledge about how to evaluate and understand existing bots, as well as how we may design innovative bots for the future while being aware of the limitations and risks they may introduce. We invite the community to reflect on the definition and taxonomy we put forward as we expect other insights may help us improve and refine both of these contributions.

978-1-7281-2262-5/19/$31.00 ©2019 IEEE
DOI 10.1109/BotSE.2019.00008

II. An Overview and the Origins of Software Bots

Early descriptions of bot-like helpers include Socrates' *daimonion* in 399 BC [1] and James Clerk Maxwell's hypothetical *demon* in 1871 [5]. However, the first real digital helpers were created for the Multics operating system by programmers at MIT in 1963 [1]. They adopted the term **daemon**, which is still used today, to describe small programs running unobtrusively as background processes instead of being directly controlled by users on Unix-like operating systems.

While the first appearance of the term robot is credited to a 1921 science fiction play entitled Rossum's Universal Robots [6], where the author replaced the classical term **automata** with **robot**, real robots only began to appear in the early 1970s [7], [8]. The term *bot* originated as an abbreviation of *robot*, however, unlike software bots, which are digital, robots are mechanical. And while robots are used in the physical world much in the same way software bots are used in the digital world, they have tangible, mechanical bodies that perform tasks by manipulating the physical world, often helping automate repetitive tasks.

The Turing Test (1950) sparked the development of **chatbots**, computer programs designed to act humanly by talking to users [9], [10]. Created in 1966 by MIT professor Joseph Weizenbaum, Eliza was the first computer program to converse with humans. Eliza attempted to cover her limited vocabulary by imitating a psychotherapist: Eliza searched for keywords in the user's speech and responded with preprogrammed questions, shifting the focus of the conversation back onto the user.

Eliza inspired a variety of notable chatbots, including: Perry (1972), the paranoid schizophrenic [11]; Alice (1995), the natural language processing bot with lots of personality [12]; and SmarterChild (2000) [13]. While these earlier chatbots' interactions were purely text based, advances in natural language processing allowed chatbots to begin using spoken language or a combination of text and speech. The personal assistant chatbot Julia (1994) was the first verbal chatbot [14]. A couple of years later, Sylvie (1997) became the first "virtual human" with an animated face and voice [15].

Related to these are **software agents**, which are often confused with bots. The word agent originates from the Latin word *agere*, meaning *"to do"* or *"to act on someone's behalf"* [10], [16]. The first software agents can be traced back to Hewitt's Actor Model [16]. However, agents were brought into the public eye by the famous "Knowledge Navigator" video that portrayed the interaction between a software agent and its user [17]. Later, software agent research began to diversify and a variety of agents emerged to support a broad range of tasks across many domains. Software agents also began to take on various names, based on either some significant property (e.g,. collaborative, interface, mobile, internet, reactive, or smart) or their purpose (e.g., personal assistants, guides).

The recent re-emergence of software bots and subsequent increase in new bots being designed, deployed, and used has been a result of technological advancements, the mainstream adoption of both text messaging and voice-based platforms, the transition to service-oriented architectures, and the abundance of public APIs and datasets [3]. However, due to the complicated and dispersed history of bots, bot design and research communities are confronted with a vast, diverse, and often inconsistent terminology, and lack a proper classification scheme.

There are existing bot taxonomies and classification schemes [16], [18]–[29], but we believe they are ineffective or unsuitable for classifying many of the modern software bots. Some classifications organize bots into subtypes (e.g., agents, chatbots) or based on their roles (e.g., informational bots, transactional bots). Other taxonomies focus only on the properties of specific subtypes of bots. In terms of granularity, some taxonomies are too low level and describe a specific property or behaviour of bots in extreme detail, while others are too high level and combine many properties together into a single dimension, often providing no description of the property or behaviour at all. Moreover, the rapidly changing bot landscape means that classification schemes become out of date if they don't consider recent technological advances that are a central part of the bots we see today (e.g., NLP, voice).

We believe that a taxonomy should not be a *one-size-fits-all* solution. However, we strived to combat these issues through the design decisions behind our taxonomy: (1) it was designed as a multi-faceted taxonomy to allow for selective use and graceful expansion as each facet is independent; (2) it can be used to classify the observable properties and behaviours of software bots; and it (3) provides a consistent set of terminology (i.e., variant terms [30]) that map between our controlled vocabulary and the terms used to describe the same content in other related work. It is the product of many software bot taxonomies being merged together, yet we strongly believe that it should never be considered complete.

III. Defining Software Bots

Despite their increasing popularity, analyzing and understanding bots is challenging. Existing research of software bot technologies spans across multiple areas and disciplines, and has resulted in the lack of a generally accepted definition of software bots. Over the years, researchers and practitioners have tried to define bots in accordance with their specific applications. For example, some define bots by their ability to automate tasks [31]–[33] or behave autonomously [34], while others define bots by their conversational capabilities [35], [36] or human-like behaviors [37], [38].

While a multitude of definitions exist, some contradicting or overlapping, there are several themes that are consistent across many of the interpretations. We used these central themes to develop our proposed definition of software bot-hood. We also compared software bots to other related technologies to further understand what makes a bot a *"bot"*. For brevity, we do not include this comparison here and kindly refer the interested reader to Lebeuf's thesis [3].

We view software bots as a new **interface paradigm**; bots *connect* users with software services. While bot users are often humans, they are not required to be: users can be programs, systems, or even other bots. A bot is the interface that *provides the services to the user*, i.e., a bot is everything required to present the service to the user. However, the bot and the service can and should be decoupled from each other.

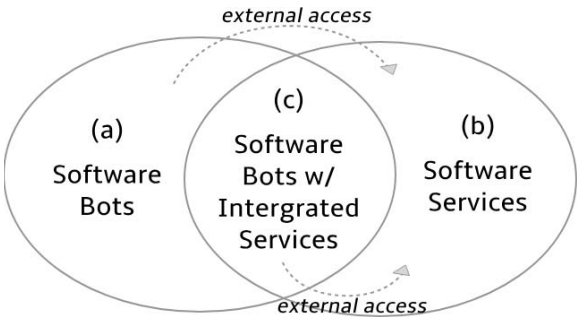

Fig. 1: The relationship between (a) software bots, (b) software services, and (c) software bots with internal services.

Software services are *"a mechanism to enable access to one or more capabilities"* [39]. Software bots utilize software services for the raw value they provide; services provide software functionality (or a set of software functionalities) in a format that can be reused by multiple clients for a variety of purposes. Modern services come in many forms, ranging from the retrieval of information to the execution of a set of operations. Often, a bot performs tasks that rely on these services repetitively, saving the user time through automation. Software services can be external to the bot, internal, or a mixture of both types. Figure 1 illustrates this bot–service relationship.

If software bots provide an alternative interface to services, then what exactly does a software bot interface entail? A software bot is the interface where the user and the bot's services meet. Furthermore, the software bot interface usually leverages recent advances in interface paradigms and **provides additional value** on top of its services (e.g., lowering the barrier of access, consolidating multiple services, providing automation).

Nowadays, users can interact with software bots via the command line, graphical interfaces, touch interfaces, spoken/written language, or a combination of interaction paradigms. It should be noted, however, that these interfaces are not required to be interfaces that humans use; the software bot's interface can be used by other bots or other types of software systems. Another common way that software bots provide additional value is through anthropomorphism—making the user's interactions with the software services more enjoyable by making it more human. There are many ways in which people anthropomorphize software bots: giving them names, personalities, emotions, etc. We discuss more of the additional value that bots provide as we introduce our taxonomy in the next section.

IV. The Taxonomy

This taxonomy aims to update and organize the emergent properties of software bots to provide a deeper understanding of existing software bots as a whole. More specifically, it presents a controlled vocabulary (i.e., variant terms [30]) for discussing the observable properties and behaviours of software bots. This taxonomy also provides a range of possible values for each category of properties or behaviours. It was designed specifically as a faceted taxonomy to allow for the subject matter (software bots) to be classified from multiple, independent perspectives (called facets) that can be combined to create a full classification of a software bot.

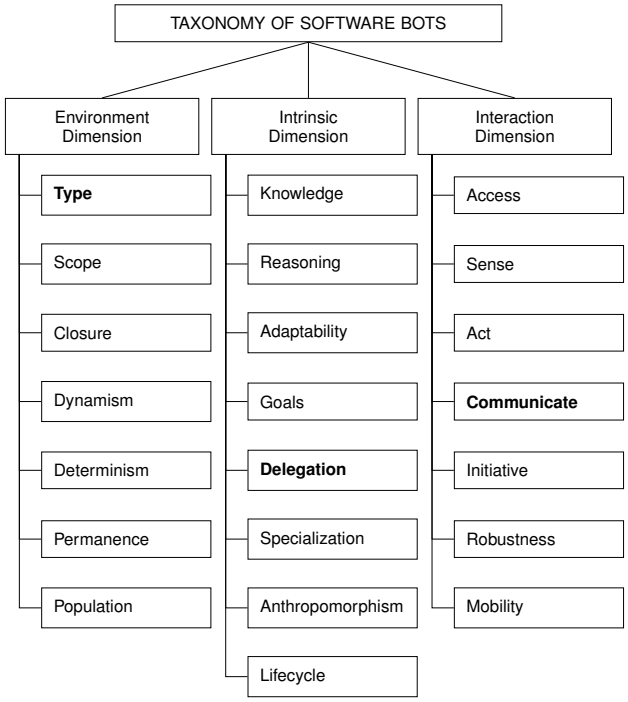

Fig. 2: A high-level view of the Software Bot Taxonomy's structure. The bolded facets are used as examples in the following sections.

As mentioned, we used an adaptation of Usman *et al.s* iterative software engineering taxonomy generation methodology in order to ensure rigour and to allow a multi-stage data collection, term extraction, taxonomy construction, and validation process [4]. We collected articles that discussed the characteristics of software bots following a systematic literature review. We extracted any terms used to describe the observable properties and behaviours of software bots. The extracted terms were then reduced (through mapping and merging variant terms) and card sorted to allow the new taxonomys dimensions, facets, and facet values to emerge from the data. More information regarding the full methodology used to generate this taxonomy is available in Lebeuf's thesis [3].

Using this methodology, we arrived at a holistic taxonomy of software bots. At its top level, the taxonomy has three main

dimensions: (a) the bot's **environment**, (b) the **intrinsic** properties of the bot itself, and (c) the bot's **interactions** within its environment. Figure 2 shows an overview of the taxonomy's dimensions and their top-level facets. In the following, we provide a high-level description of our proposed taxonomy. We refer interested readers to Lebeuf's thesis [3], where each of the taxonomy's dimensions, facets, sub-facets, and facet values are described in greater detail, along with many examples of bots classified using the taxonomy.

A. Environment Dimension

To better understand a bot, we have to first understand its environment. The environment dimension describes the surroundings in which the bot lives and operates. What we can observe about bots is how they behave with the environment around them, therefore, the environment has an influence on the bot's behaviours. In the case of a bot operating in many distinct environments, each of these environments should be classified independently to provide a more complete picture of the bot's environmental influences.

As shown in Fig. 2, there are seven top-level facets that fall under the environment dimension: **type**, the bot's setting; **scope**, the size of the bot's environment; **closure**, who is able to access the bot's environment; **dynamism**, the degree to which the bots environment changes; **predictability**, the degree to which outcomes of the bot's actions are deterministic; **permanence**, how long the effects of the bot's actions last; and **population**[1], who else resides in the bot's environment. Each of these facets have a set of sub-facets or facet values.

For instance, the **type** facet represents the setting (often a system) that the bot inhabits, participates, or accesses. This facet can have one of two values:

- **Standalone**: The bot is not tied to a specific platform. Instead, the bot is hosted independently but can access platforms in the same way as users. For example, most video game bots and IRC bots are standalone.
- **Platform**: The bot can be hosted independently or through the platform, but accesses the platform through non-user methods. These kinds of bots augment a system's behaviour. For example, most GitHub and Microsoft Team's bots inhabit specific platforms.

B. Intrinsic Dimension

The intrinsic dimension is composed of facets that describe the properties belonging to the bots themselves. These properties are controlled at *design-time* by the bot developers. Although some of these intrinsic facets touch on the inner workings of the software bot itself, they are still relatively visible from a *black-box* approach[2]. For the most part, we try to focus on the externally observable, intrinsic properties of software bots.

As shown in Fig. 2, the intrinsic dimension has a total of eight facets[3]: **knowledge**, what the bot knows or understands; **reasoning**, the bot's logical capacity; **adaptability**, the bot's ability to modify its own behaviour; **goals**, the type of future state the bot is attempting to achieve; **delegation**, the bot's authority to act on behalf of others; **specialization**, the degree to which the bot specializes its efforts in a specific area; **anthropomorphism**, the degree to which the bot is given human-like traits; and **lifecycle**, the phases the bot goes through in its life.

An example of an intrinsic property is the **delegation** facet, describing the bot's permission to act on behalf of or to represent others. This facet is defined according to the following ordinal scale:

- **None**: The bot does not have the authority to act on behalf of others. However, bots with this property can appear to be acting on behalf of users, but do so without their permission and often with malicious intent (e.g., 'doppelganger' bots [40]).
- **Partial**: The bot has authority to act on behalf of the user, but does not pretend to be the user, e.g., bots that complete pull requests on behalf of users.
- **Complete**: The bot has the authority to both act on behalf of and pretend to be the user, e.g., bots that complete pull requests using the users' credentials.

C. Interaction Dimension

The interaction dimension is composed of facets that describe the bot's interactions with different entities in its environment. More specifically, they try to focus on the wide range of externally observable behaviours that the bot can exude when interacting with the various elements in its environment. Some of these interaction facets touch somewhat on the inner workings of bots, yet, they are needed for a *black-box* approach[2] when examining bot behaviours.

As shown in Fig. 2, there are a total of seven facets[4] that fall under the interaction dimension: **access**, the bot's ability to sense and act within its environment; **sense**, the degree to which the bot can perceive environmental stimuli; **act**, the bot's ability to act upon its environment; **communicate**, the degree to which the bot can have meaningful interactions with others; **initiative**, the way the bot's environmental interactions are initiated; **robustness**, the bot's error or ambiguity handling; and **mobility**, the bot's ability to move around in its environment.

An example of an interaction sub-facet that falls under the **communication** facet is **cardinality**. It represents the number of users that the bot is capable of interacting with simultaneously. Its values are along the following ordinal scale:

- **One-One**: The bot is capable of interacting with one user at a time. For example, most Microsoft Teams bots that a user can have direct, private messages with are one-one.
- **One-Many**: The bot is capable of interacting with many users simultaneously. For example, most Microsoft Teams bots on public channels are one-many.

[1]The population facet has two additional subfacets: cardinality and diversity [3].

[2]We adopted a black-box approach so that software bots could be classified using the taxonomy even if their inner structures were not known.

[3]With an additional 27 sub-facets for these eight intrinsic facets [3].

[4]With an additional nine sub-facets for these seven interaction facets [3].

- **Many-Many**: The bot is capable of interacting with many users who are also interacting between themselves, e.g., Xiaoice [41].

D. Validation

Following Usman *et al.*'s guidelines [4], we ensured that the taxonomy correctly captured the range of observable software bot properties and behaviours by validating it through: (i) benchmarking our taxonomy against existing classifications of software bots; (ii) demonstrating the utility of the taxonomy through the tagging of three publicly available software bots [3, p. 126]; and (iii) testing the utility and usability of the taxonomy through a domain expert tagging session. More details regarding our validation efforts can be found in Lebeuf's thesis [3].

V. Why Defining & Classifying Bots Is Important

Ability to identify bots: The taxonomy we developed helps determine what may or may not be a bot, identify different *bot species*, and clarify the differences between bot subtypes and other types of non-bot programs. For example, it could be used to better answer questions like *how do agents differ from other bots?* Additionally, using the definition we provide in this paper, we may also be able to more easily ascertain the number of and nature of bots *"in the wild"* for specific domains.

Capture and describe diversity: When validating our taxonomy, participants were asked to classify bots they had built themselves, and were able to do so with little guidance [3]. However, when we attempted to classify bots from previously published research and literature, we found it extremely challenging (and in many cases impossible) since key details were lacking in the authors' descriptions of their bots. This indicates to us that having a taxonomy such as the one we propose can help researchers and designers more clearly describe bots and compare them with existing bots. Having a set of consistent terminology in an emerging field (like SE) also makes it easier to understand and build upon each other's work, which in turn speeds up research [4].

Indeed, a taxonomy should allow us to capture the diversity of bots we see today. In software development, for example, we often classify bots by their ability to support different development activities [32], [42] (e.g., assist in code review, testing, bug fixing, quality checks, deployment), but describing bots by their goals alone may lead to an impoverished description. Classifying these bots using other dimensions can help reveal important similarities and differences between them. For example, identifying the different mechanisms bots use to "collaborate" with developers can help reveal insights into designing bots that focus on mediating the collaboration between two or more developers.

Document patterns of change: Integrating bots into software developers' workflows can lead to changes in the behaviour and practices of individual members, their teams, and their organizations. However, documenting these patterns of change without a description or an understanding of the various *bot species* and their characteristics is bound to be very difficult. For example, Lebeuf *et al.* [43] used a socio-technical model to explore how chatbots can help reduce the friction points software developers face when working collaboratively.

Provide a basis for future guidelines: Although our taxonomy is currently non-prescriptive (i.e., it does not provide guidelines or recommendations for selecting between facet values), we feel the taxonomy could be used to help develop a set of best practices for designing bots. Since the taxonomy describes the set of possible values for each of the facets or sub-facets, recommendations could be derived by exploring the variety of bot facets. However, much future work is needed to develop and evaluate recommendations and best practices for general and specific domains, as well as to further validate our proposed taxonomy.

VI. Discussion Points

To conclude this paper, we provide a set of questions that we hope may spark discussion on the classification of software bots both in this workshop and across the software bot community.

1) What is a bot to you? Do you agree with our proposed definition of software bots? What do you see as their defining characteristics?
2) Do you see a need to have a more formal definition of software bots?
3) What other facets can/should be added to the taxonomy? Which ones do you think will become more or less important as we continue to design, develop, and research software bots?
4) What kinds of guidelines are needed to guide the design of bots, especially for specific domains and activities such as software development?
5) What is your vision for the future role bots may play in software engineering (or other domains)? Does the taxonomy prompt you to think of future bot capabilities?

References

[1] A. Leonard, *Bots: The Origin of New Species*. Penguin Books Limited, 1998.
[2] J. Cabot, "Best bots to improve your software development process," https://livablesoftware.com/best-bots-software-development, 2018, [Online; accessed 28-Feb-2019].
[3] C. Lebeuf, "A taxonomy of software bots: towards a deeper understanding of software bot characteristics," Master's thesis, University of Victoria, 2018.
[4] M. Usman, R. Britto, J. Börstler, and E. Mendes, "Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method," *Information and Software Technology*, vol. 85, pp. 43–59, 2017.
[5] W. Thomson, "The sorting demon of maxwell," in *Proceedings of the Royal Society*, vol. 9, 1879, pp. 113–114.
[6] K. Čapek, *R.U.R. (Rossum's Universal Robots)*. Oxford University Press, 1951, english translation.
[7] I. Kato, "Development of wabot-1," *Biomechanism*, vol. 2, pp. 173–214, 1973.
[8] P. Mowforth and I. Bratko, "AI and robotics; flexibility and integration," *Robotica*, vol. 5, no. 2, pp. 93–98, 1987.
[9] A. M. Turing, "Computing machinery and intelligence," in *Parsing the Turing Test*. Springer, 2009, pp. 23–65.
[10] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education Limited, 1995, vol. 25, ch. 2.
[11] V. Cerf, "Parry encounters the doctor," Tech. Rep., 1973.
[12] R. Wallace, "Artificial linguistic internet computer entity (alice)," 1995.
[13] R. Hoffer, T. Kay, P. Levitan, and S. Klein, "Smarterchild," ActiveBuddy, 2001.
[14] M. L. Mauldin, "Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition," in *AAAI*, vol. 94, 1994, pp. 16–21.
[15] "Verbot sylvie," Virtual Personalities Inc, 1997.

[16] H. S. Nwana, "Software agents: An overview," *The Knowledge Engineering Review*, vol. 11, no. 3, pp. 205–244, 1996.

[17] J. Sculley, "The knowledge navigator," 1987, educom Keynote.

[18] S. D. Bird, "Toward a taxonomy of multi-agent systems," *International Journal of Man-machine Studies*, vol. 39, no. 4, pp. 689–704, 1993.

[19] S. Franklin and A. Graesser, "Is it an agent, or just a program?: A taxonomy for autonomous agents," in *International Workshop on Agent Theories, Architectures, and Languages*, 1996, pp. 21–35.

[20] Z. Huang, A. Eliens, A. van Ballegooij, and P. de Bra, "A taxonomy of web agents," in *Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, 2000, pp. 765–769.

[21] S. Munroe and M. Luck, "Agent autonomy through the 3m motivational taxonomy," in *Proceedings of the International Conference on Agents and Computational Autonomy*, 2003, pp. 55–67.

[22] P. T. Tosic and G. A. Agha, "Towards a hierarchical taxonomy of autonomous agents," in *IEEE International Conference on Systems, Man and Cybernetics*, 2004, pp. 3421–3426.

[23] J. Aguero, M. Rebollo, C. Carrascosa, and V. Julian, *Agent Capability Taxonomy for Dynamic Environments*. Springer Berlin Heidelberg, 2012, pp. 37–48.

[24] G. Sakarkar and N. M. Shelke, "A new classification scheme for autonomous software agent," in *International Conference on Intelligent Agent Multi-agent Systems*, 2009, pp. 1–2.

[25] A. Hector and V. L. Narasimhan, "A new classification scheme for software agents," in *Third International Conference on Information Technology and Applications*, 2005, pp. 191–196.

[26] H. V. D. Parunak and M. Fleischer, "A design taxonomy of multi-agent interactions," in *International Workshop on Agent-Oriented Software Engineering*, 2003, pp. 123–137.

[27] M. Huhns and M. P. Singh, "Agents and multiagent systems: Themes, approaches and challenges," in *Readings in Agents*, 1998, ch. 1.

[28] L. J. Moya and A. Tolk, "Towards a taxonomy of agents and multi-agent systems," in *Proceedings of the Spring Simulation Multiconference*, 2007, pp. 11–18.

[29] E. Paikari and A. van der Hoek, "A framework for understanding chatbots and their future," *The 11th International Workshop On Cooperative and Human Aspects of Software Engineering an ICSE workshop*, 2018.

[30] L. Rosenfeld and P. Morville, *Information Architecture for the World Wide Web*. O'Reilly Media, Inc, 2002.

[31] C. Vouillon. (2015) Software bots: From do-it-yourself companion bots to AI powered software. [Online]. Available: https://medium.com/point-nine-news/software-bots-c56aeedcfec3

[32] M.-A. Storey and A. Zagalsky, "Disrupting developer productivity one bot at a time," in *Proceedings of the 24th ACM Sigsoft International Symposium on Foundations of Software Engineering*, 2016, pp. 928–931.

[33] B. Nerds. (2017) Types of bots: An overview. [Online]. Available: http://botnerds.com/types-of-bots/

[34] A. Zantal-Wiener, "Where Do Bots Come From? A Brief History," https://blog.hubspot.com/marketing/where-do-bots-come-from, 2017, [Online; accessed 28-Feb-2019].

[35] R. Dale, "The return of the chatbots," *Natural Language Engineering*, vol. 22, no. 5, pp. 811–817, 2016.

[36] K. Iqbal, I. Berry, L. Spacil, C. Qian, and R. Standefer, "About Azure Bot Service," https://docs.microsoft.com/en-us/azure/bot-service/bot-service-overview-introduction?view=azure-bot-service-3.0, 2019, [Online; accessed 28-Feb-2019].

[37] G. Maus, "A typology of socialbots (abbrev.)," in *Proceedings of the ACM Conference on Web Science*, 2017, pp. 399–400.

[38] "Bot definition," https://en.oxforddictionaries.com/definition/bot, Oxford English dictionary, [Online; accessed 28-Feb-2019].

[39] P. Brown, J. A. Estefan, K. Laskey, F. G. McCabe, and D. Thornton, "OASIS Reference Architecture Foundation for Service Oriented Architecture," https://www.oasis-open.org/committees/soa-rm, 2012, [Online; accessed 28-Feb-2019].

[40] O. Goga, G. Venkatadri, and K. P. Gummadi, "The doppelgänger bot attack: Exploring identity impersonation in online social networks," in *Proceedings of the 2015 Internet Measurement Conference*, ser. IMC '15. New York, NY, USA: ACM, 2015, pp. 141–153. [Online]. Available: http://doi.acm.org/10.1145/2815675.2815699

[41] J. Markoff and P. Mozur, "For Sympathetic Ear, More Chinese Turn to Smartphone Program," https://www.nytimes.com/2015/08/04/science/for-sympathetic-ear-more-chinese-turn-to-smartphone-program.html, 2015, [Online; accessed 28-Feb-2019].

[42] C. Lebeuf, M. A. Storey, and A. Zagalsky, "Software bots," *IEEE Software*, vol. 35, no. 1, pp. 18–23, 2018.

[43] C. Lebeuf, M.-A. Storey, and A. Zagalsky, "How software developers mitigate collaboration friction with chatbots," 2017.