Thesis for The Degree of Licentiate of Engineering

Mapping a Landscape of Developer Assisting Software Bots

Linda Erlenhov

Division of Interaction Design & Software Engineering
Department of Computer Science & Engineering
Chalmers University of Technology | University of Gothenburg
Gothenburg, Sweden, 2021

Mapping a Landscape of Developer Assisting Software Bots

Linda Erlenhov

Department of Computer Science & Engineering
Division of Interaction Design & Software Engineering
Chalmers University of Technology | University of Gothenburg
Gothenburg, Sweden

This thesis has been prepared using LaTeX.
Icons used in figures made by
Icongeek26, Pixel perfect and Freepik from www.flaticon.com
Printed by Chalmers Digitaltryck,
Gothenburg, Sweden 2021.

"It is a mistake to think you can solve any major problems just with potatoes."
- Douglas Adams— , Life, the Universe and Everything

Abstract

Bots in software development have gained traction in research and in practice. However, there is no consensus on what properties and characteristics that define a bot. The term is used to describe a plethora of different tools with different usages, benefits and challenges. In this thesis we focus on bots for software developments (DevBots) with the goal to aid researchers in future studies involving DevBots. We aim to assist with the scoping and planning of such studies regarding what tools and related work to include or exclude from them. We do so by synthesising the different definitions of DevBots, combining views from literature and practitioners.

To achieve this goal, quantitative and qualitative research methods are used including literature review and semi-structured interviews. We have created a faceted taxonomy for DevBots which categorises DevBots by their most prominent properties. In addition we investigated what delineated DevBots from plain old development tools. Our analysis shows that achieving one single definition is not possible. Instead we identify and name three personas, i.e., practitioner archetypes with different expectations and motivations. The chat bot persona (Charlie) mostly sees DevBots as information integration tools with a natural language interface, while for the autonomous bot persona (Alex) a DevBot is a tool that autonomously handles repetitive tasks. Lastly, for the smart bot persona (Sam), the defining feature of bots is its degree of "smartness".

We have identified a process in the form of a flowchart, which researchers can use to test whether their tool is considered a DevBot by any of our personas. We have concluded that this definition is not congruent with contemporary definitions as only 10 of 54 investigated tools from a large dataset were considered DevBots by our process. Finally we have shown how the definitions and process can be used in practice by using them in the scoping and planning phase of two recently conducted studies.

Keywords

Software bot, Taxonomy, Software engineering, Empirical research

Acknowledgment

Thank you to my supervisors Philipp Leitner and Francisco Gomes De Oliveira Neto - "the world's most okayest mentors".
Thank you to my examiner Robert Feldt.
Thank you to my colleagues at the department, especially my friends of Kuggen 351 and my co-chairs in the CSE PhD council.
Thank you Göteborg Marvels ladies players & coaches, and the Swedish division one ladies American football for allowing me to "puckla på" you every year.
Thank you Brogyllen for not judging me when I buy too much fika every Friday.
Thank you to my ironing board who served me as a working desk for 1.5 years during the pandemic and in that timespan only broke down once (ok, twice.).

This research has been partially funded by Chalmers University of Technology Foundation and the Swedish Research Council (VR) under grant number 2018-04127 (Developer-Targeted Performance Engineering for Immersed Release and Software Engineers).

List of Publications

Appended publications

This thesis is based on the following publications:

[A] L. Erlenhov, F.G. de Oliveira Neto, R. Scandariato, P. Leitner
"Current and future bots in software development"
2019 IEEE/ACM 1st International Workshop on Bots in Software Engineering (BotSE), 2019.

[B] L. Erlenhov, F.G. de Oliveira Neto, P. Leitner
"An empirical study of bots in software development: characteristics and challenges from a practitioner's perspective"
Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE), 2020.

[C] L. Erlenhov, F.G. de Oliveira Neto, P. Leitner
"Dependency Management Bots in Open-Source Systems - Prevalence and Adoption"
Under revision at PeerJ Computer Science, 2021.

[D] L. Erlenhov, F.G. de Oliveira Neto, M. Chukaleski, S. Daknache
"Challenges and guidelines on designing test cases for test bots"
Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering (ICSE) Workshops, 2020.

Other publications

The following publications were published during my PhD studies, or are currently in submission/under revision. However, they are not appended to this thesis, due to contents overlapping that of appended publications or contents not related to the thesis.

[a] F.G. de Oliveira Neto, R. Feldt, L. Erlenhov, J. De Souza Nunes
"Visualizing test diversity to support test optimisation"
2018 25th Asia-Pacific Software Engineering Conference (APSEC), 2018

Research Contribution

Following the Contributor Roles Taxonomy(CreditT)[1]

Table 1: The authors contribution to the different papers that comprise this thesis.

Role	**Paper A**	**Paper B**	**Paper C**	**Paper D**
Conceptualization	x	x	x	x
Data curation	x	x	x	
Formal Analysis	x	x	x	
Funding acquisition				
Investigation	x	x	x	
Methodology	x	x	x	x
Project administration	x	x	x	x
Resources	x	x	x	
Software			x	
Supervision				x
Validation	x	x	x	
Visualization	x	x	x	x
Writing – original draft	x	x	x	
Writing – review & editing	x	x	x	x

[1] `https://casrai.org/credit/`

Contents

Abstract v

Acknowledgement vii

List of Publications ix

Personal Contribution xi

1 Introduction 1
- 1.1 Problem statement . . . 1
- 1.2 Research scope . . . 2
- 1.3 What delineates DevBots from plain old development tools? . . . 5
  - 1.3.1 Method . . . 5
  - 1.3.2 Contributions . . . 6
- 1.4 What process can researchers use to determine whether a tool is a DevBot? . . . 8
  - 1.4.1 Method . . . 8
  - 1.4.2 Contributions . . . 8
- 1.5 To what extent is this definition congruent with definitions used in contemporary research? . . . 9
  - 1.5.1 Method . . . 9
  - 1.5.2 Contributions . . . 10
- 1.6 How can we practically apply the definition and process to future research? . . . 11
  - 1.6.1 Empirical analysis of dependency bots . . . 11
  - 1.6.2 Empirical analysis of test bots . . . 11
  - 1.6.3 Contributions . . . 12
- 1.7 Discussion . . . 13
  - 1.7.1 [RQ1:] What delineates bots from plain old development tools? . . . 13
  - 1.7.2 [RQ2:] What process can researchers use to determine whether a tool is a bot . . . 13
  - 1.7.3 [RQ3:] To what extent is this definition congruent with definitions used in contemporary research? . . . 14
  - 1.7.4 [RQ4:] How can we practically apply the definition and process to future research? . . . 14
- 1.8 Threats to validity . . . 14
- 1.9 Related work . . . 15

1.9.1 A brief history of bots . . . 15
1.9.2 Contemporary bots . . . 16
1.9.3 Benefits and challenges when adopting bots . . . 16
1.9.4 Bot Identification . . . 17
1.9.5 Taxonomies . . . 18
1.10 Conclusions . . . 18
1.11 Outlook on the future . . . 19

2 Paper A **21**
2.1 Introduction . . . 22
2.2 Related Work . . . 22
2.3 DevBot Taxonomy . . . 23
2.3.1 Facet-Based Taxonomy . . . 24
2.3.1.1 Purpose . . . 25
2.3.1.2 Initiation . . . 25
2.3.1.3 Communication . . . 26
2.3.1.4 Intelligence . . . 26
2.4 A Vision of Future DevBots . . . 27
2.5 Conclusions . . . 29

3 Paper B **31**
3.1 Introduction . . . 32
3.2 Related Work . . . 33
3.3 Study Methodology . . . 35
3.3.1 Interviews . . . 35
3.3.2 Survey . . . 36
3.4 Results . . . 39
3.4.1 Overview . . . 39
3.4.2 Characteristics of DevBots . . . 42
3.4.3 Potential Benefits and Bot Use Cases . . . 45
3.4.4 Challenges . . . 48
3.5 Discussion . . . 51
3.6 Threats to Validity . . . 52
3.7 Conclusions . . . 53

4 Paper C **55**
4.1 Introduction . . . 56
4.2 Related Work . . . 58
4.2.1 Impact of Bot Adoption . . . 58
4.2.2 Bot Identification . . . 59
4.3 Study Methodology . . . 60
4.4 Distinguishing Bots and Automation Tools . . . 60
4.4.1 Data Collection . . . 60
4.4.2 Analysis and Interpretation Approach . . . 61
4.4.3 Results . . . 63
4.5 Activity Analysis of Dependency Management Bots . . . 65
4.5.1 Data collection . . . 65
4.5.2 Analysis and Interpretation Approach . . . 66
4.5.3 Results . . . 66

4.6 What are the discussed challenges and preferences when adopting, switching or discarding bots? . . . 70
4.6.1 Data Collection . . . 70
4.6.2 Analysis and Interpretation Approach . . . 71
4.6.3 Results . . . 73
4.7 Discussion . . . 74
4.7.1 Threats to Validity . . . 76
4.8 Conclusions . . . 78

5 Paper D **79**
5.1 Introduction . . . 80
5.2 Related work . . . 80
5.3 Research methodology . . . 81
5.4 Findings and Discussion . . . 82
5.4.1 Analysis of RQ1 . . . 83
5.4.2 Analysis of RQ2 . . . 85
5.4.3 Guidelines for Designing Tests for Test Bots . . . 86
5.4.4 Threats to Validity . . . 87
5.5 Conclusion . . . 87

Bibliography **89**

Appendix **97**

A Appendix - Paper C **97**

B Appendix - Paper D **99**

Chapter 1

Introduction

Even though bots in software development recently have gained attention in research, the idea of software bots is not new. The use that Maes [1] or Lieberman [2] described during the 90s for software, under the umbrella term of agents, however never gained extensive public use beyond the published research and did not to the best of our knowledge incorporate the use case of aiding software developers in development related tasks. Several studies did however discuss classifications of agents in addition to what an agent is and where to draw the line between an agent and a program [3–5]. There was a fear that without proper definitions the word agent would become a noise term causing confusion within the research community.

1.1 Problem statement

Current research on bots is also done under an umbrella term and the debate on where to draw the line between a bot and a program is also still alive. When researchers write about bots they mean different tools that do different things and interact in different ways. Even though this does not mean that the research done is not interesting and meaningful, without structure it makes it difficult to map out (i) what research has actually been conducted, (ii) what research is related (iii) and what gaps still needs to be filled. Lacking clear definitions also hinders comparisons between different studies, what could be considered further support for indications in a previous study and what could be considered new findings.

1.2 Research scope

This research is motivated by an observation of an emerging problem with the wide use of the term bot to describe a plethora of different tools with different usages, benefits and challenges. We first introduce the term DevBots as a shorthand for "bots for software development". A Venn diagram illustrating where a DevBot belongs can be found in figure 1.1. The thesis has two goals.

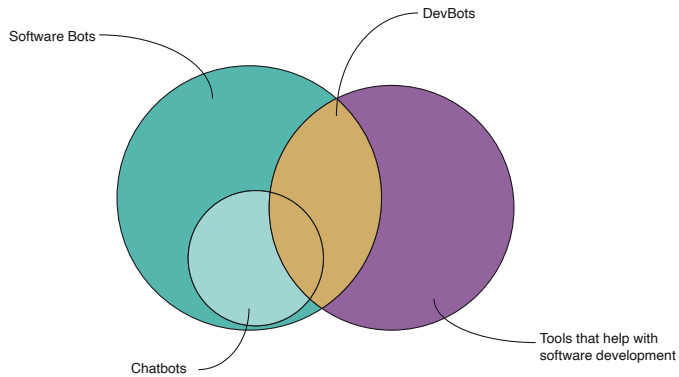

Figure 1.1: A Venn diagram showing the relations between different type of bots and tools that help with software development.

G1: To assist researchers in future studies involving DevBots, regarding what tools and related work to include or exclude from such a study.

G2: Synthesise the different definitions of DevBots by combining views from literature and practitioners

To reach the goals of this thesis, we then formulate the following research questions:

RQ1: What delineates DevBots from plain old development tools (PODT)?

RQ2: What process can researchers use to determine whether a tool is a DevBot?

RQ3: To what extent is this definition congruent with definitions used in contemporary research?

RQ4: How can we practically apply the definition and process to future research?

A visual overview of the research questions and how they connect with the input data source, to the output deliverables and how these deliverables are connected to the research questions can be found in figure 1.2.

This thesis comprises four papers, distributed on a theoretical-empirical scale as shown in figure 1.3. A paper being placed more towards the theoretical side of the scale indicates that the results are more conceptual while it on the

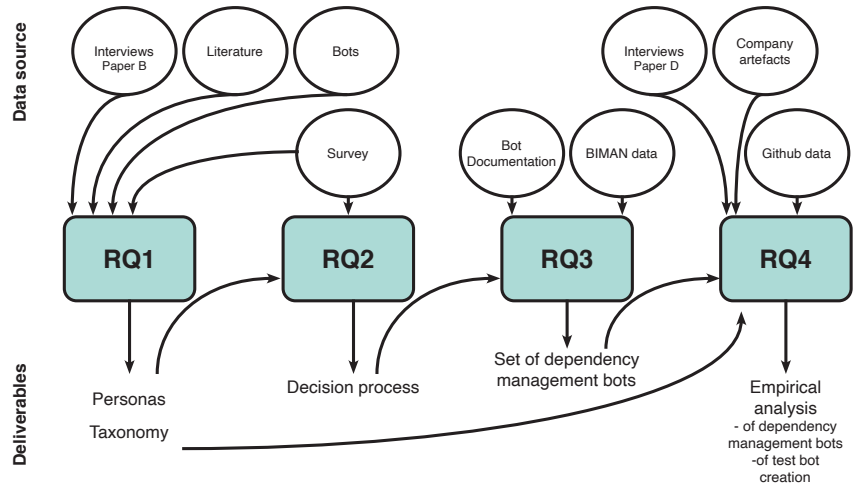

Figure 1.2: The input data source and output deliverables of respective research question.

empirical part of the scale is more real-world based. All four papers include some parts of both which inspired to place the papers on a scale instead of e.g. using a table where empirical/theoretical is a boolean. Mapping our research to that scale can help other researchers use our studies to characterise or prevent challenges when defining their DevBot evaluation constructs (theory side) or DevBot data availability for analysis (empirical side)

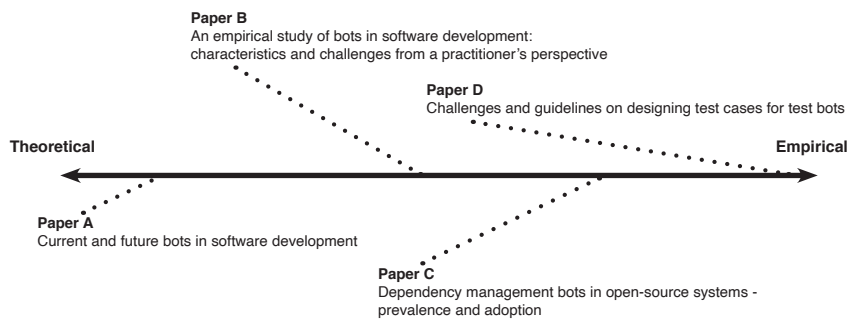

Figure 1.3: The four papers distributed on a theoretical-empirical scale

To answer the research questions we used a variety of data. An overview of the data collection methods, the associated paper and towards which research question they contributed can be found in table 1.1 The data was analysed by a mix of quantitative and qualitative methods. The methods with regards to both data collection and analysis are explained in more detail in the sections covering each research question and in even further detail in each of the papers.

Table 1.1: Overview of the data collection methods in relation to the studies and research questions.

RQ	Paper	Source of evidence	Data Collection method
1	A	Research on software agents	Literature study
1	A	Collection of bots	Web search
1	B	Software professionals	Semi-structured interviews
1,2	B	Software professionals	Survey
3	C	Existing dataset (BIMAN)	Systematic search
3	C	Bot related documentation	Web search
4	C	Github	Repository mining
4	D	Software professionals	Semi-structured interviews
4	D	Company Artefacts	Intranet & Repository search

1.3 What delineates DevBots from plain old development tools?

To answer the first research question, what delineates bots from plain old development tools, we created a faceted taxonomy for DevBots, performed a themed analysis on semi-structured interviews and described statistics of survey answers both visually and textually.

1.3.1 Method

To create an initial taxonomy we identified and extracted data from literature followed by design and construction using a faceted analysis [6] Facet-based taxonomies have been found particularly useful for emerging fields [7], such as ours, since it does not require a complete knowledge on the subject of matter and is easily adapted so they can evolve over time.

Subsequently we collected various bots to validate the taxonomy via utility demonstration [7]. The early-stage nature of our field of study complicated applying standard systematic data collection procedures. Hence, we opted for a pragmatic data collection procedure. The DevBots collected thereby are a mix of previously known bots, bots found by searching the internet and found via advertisement in social media. The criteria for the bots being included was that they were involved in some type of software development task and that the creator or user talked about them as bots.

Following this we did an interview study. Here we sampled industry practitioners that, at some point, worked with tools that they self-identified as DevBots. We began by inviting practitioners from our personal industry network, who then referred us further to other potential participants. Then, we used a saturation approach [8] where we kept inviting new participants in parallel to data analysis while the data offered new information. We conducted interviews over a period of three months in the fall of 2019. Each interview took between 15 and 45 minutes and was done either face-to-face or via video conference depending on the participant's availability.

In parallel to doing the interviews, we performed open and axial coding based on the Straussian variant of Grounded Theory [9, 10]. In open coding we fracture the data to find relevant excerpts. In axial coding, we aggregate and connect those excerpts into categories and themes until achieving saturation. This also allows us to identify limitations of the interview guide, such as missing relevant aspects that were not clear upon creation of the interview guide.

In order to triangulate results, different pairs of authors performed open coding on the first 4 interviews to check for consistency and agreement in our coding process. A total of 13 interviews were openly coded independently by two authors, whereas the remaining 8 interviews were open coded by only one author. The second part, the axial coding in which we identified themes, memoed and performed card sorting, was done by all authors together in different sessions lasting between 2–3 hours each. The resulting categories and findings, which include the three personas we will describe in the contributions section, are supported by statements from multiple participants.

Based on the interview results we designed a Web-based survey using

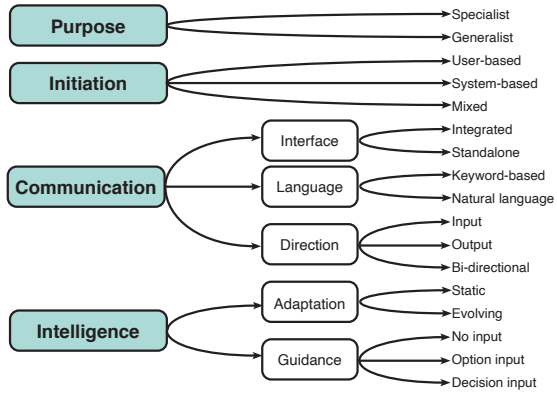

Figure 1.4: Our proposed DevBot taxonomy with facets and respective levels.

Typeform[1], with 48 questions in total. After five questions collecting basic demographic information, the main part of our survey consisted of two top-level sections covering, respectively, the definition and usage of bots in software development. We used the results from our interview study to devise the questions in each survey section. We distributed the survey through our industry network as well as social media. Further, we invited all interviewees to participate and distribute the survey further. We received 111 complete responses.

To analyse the first block of questions, we mapped each described system to the three personas that emerged from interviews. During analysis, for each respondent, we assigned points based on each answer to verify which persona that respondent corresponded to. We referred to the final score as *persona association scores*, where a score ≤ 0 would indicate no association with this persona, while higher (positive) values represent an association of increasing strength. Further, we analysed both survey sections using descriptive statistics and visually using diverging plots.

1.3.2 Contributions

Firstly, we provide a facet-based taxonomy of existing ("contemporary") DevBots, orthogonal to the one provided by Lebeuf [11]. The taxonomy is found in figure 1.4. Benefits of using this taxonomy over e.g. the one provided by Lebeuf is that it is simpler with fewer facets, making it easier to classify a DevBot by its prominent characteristics. A detailed description of the taxonomy and its facets can be found in paper A.

Secondly from our interviews and the results of the survey we find that a single definition of DevBots is unachievable, as different developers associate widely different characteristics with the term. However, we are able to identify three different personas, i.e., practitioner archetypes with different expectations and motivations [12]. The **chat bot persona** (Charlie) sees DevBots mostly

[1] `https://www.typeform.com`

as (information) integration tools with a natural language interface, while for the **autonomous bot persona** (Alex) a DevBot is a tool that autonomously handles, often quite simple, tasks for human developers. Finally, for the **smart bot persona** (Sam), the distinguishing characteristic of a bot is a "smartness" that goes beyond other tools. The personas are described in more detail in paper B.

1.4 What process can researchers use to determine whether a tool is a DevBot?

To answer the second research question, what process can researchers use to determine whether a tool is a DevBot, we created a flowchart from the survey results.

1.4.1 Method

For RQ2 we used the same data collection and analysis as in RQ1. Additionally, we used the survey responses to rank the first block of questions in the survey, the questions that described potential DevBots. The flowchart was then created step by step using the facets of the taxonomy and the personas from the interviews, until all 32 described tools could be mapped using the flowchart.

1.4.2 Contributions

Figure 1.5 depicts the flowchart that has emerged from the study. This model can be used to decide on a high-level whether a given tool is likely to be considered a DevBot, and for which persona. The limitation of this flowchart is that it stops at the first match. So, imagine a DevBot that would be considered a DevBot by both Charlie and Alex. The flowchart would only detect it as a DevBot recognised by Charlie since every persona is an endpoint. To see if a DevBot has multiple persona associations, one would have to traverse down the flow chart, ignoring the previous match. But the flow chart will still give an answer to the question "Is this tool considered a DevBot by any persona?", which answers the research question. The flow chart was found useful in later empirical work to obtain a homogeneous sample of DevBots.

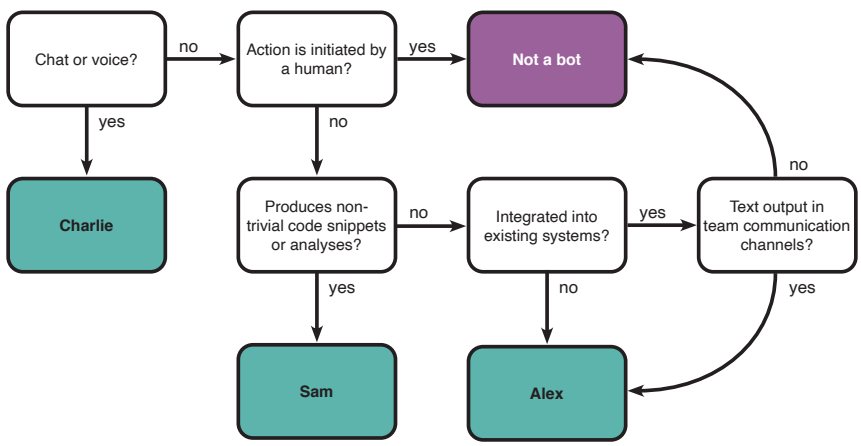

Figure 1.5: Simplified model to classify DevBots. This model assumes that the bot being classified is helping with some type of software development activity

1.5 To what extent is this definition congruent with definitions used in contemporary research?

To answer the third research question, to what extent is this definition congruent with definitions used in contemporary research, we have looked at one dataset.

1.5.1 Method

Our origin of data was the BIMAN dataset, which includes git commits detected by the BIMAN approach proposed by Dey et al. [13, 14]. The goal of the approach was to detect bots that commit code, so the commits in the dataset are produced by what the authors recognise as bots. It contains over 13 million commits from 461 authors. We extracted the authors and sorted them by the number of GitHub organisations adopting each tool, as a proxy of popularity or importance. However, initial analysis showed that the dataset contained duplicate tools (the same tool acting under multiple identities). We resorted to manually merging identities of the first 70 tools in the ordered list, which after merging, produced a final table consisting of 54 unique tools associated with 89 different authors.

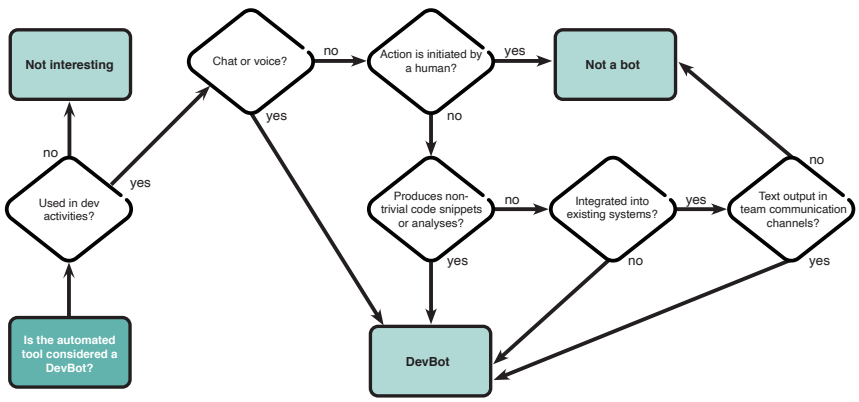

Figure 1.6: Decision flow-chart without delineation of personas and the previously implicit box “software development activity” added.

We categorised the 54 remaining tools manually using the flow-chart from RQ2, but for our categorisation in RQ3 we adapted this decision model slightly (see figure 1.6). We added a decision block to first check if the tool was actually used for a software engineering task, since this was implied in RQ2 and we are only interested in finding DevBots. Further, since the goal of RQ2 is just to decide if a tool is a bot or an automation tool, we were less interested in the specific persona and classified all types of bots simply as “DevBots” with

no further distinction. As the BIMAN dataset only contains commit data, we resorted to manually querying additional information, such as GitHub user profiles, documentation, the tool's external website, developer comments, etc., to arrive at a classification decision for each tool.

1.5.2 Contributions

On the final list of 54 tools, only 10 tools were (clearly) judged as bots according to the persona-oriented classification model. We conclude the following from this classification exercise:

- Only a small fraction (10 of 54, or 18.5%) of analysed tools clearly qualify as "bots" according to a stricter definition. A large majority are, often fairly conservative, automation tools that have been rebranded as bots, and exhibit little qualitative difference to the kinds of scripts that developers have used for a long time as part of their development, build, and deployment processes.
- Interestingly, this includes many tools that are explicitly called "bots" as part of their names, e.g., the Bors bot or docker-library-bot. Hence, researchers that are interested in investigating bots in a stricter sense should not rely on tool names as the primary way to identify bots.
- It is evident that the tools that we actually classify as Devbots (e.g., dependabot, renovate, or greenkeeper) are very similar. More specifically, nine out of these ten bots are dependency management bots in some form. In one case - Snyk and Greenkeeper - one bot was acquired by the other in 2020.[2]

More details on the classification is found in paper C.

[2] `https://snyk.io/blog/snyk-partners-with-greenkeeper-to-help-developers-proactively-maintain-dependency-health/`

1.6 How can we practically apply the definition and process to future research?

To answer the fourth research question, how can we practically apply the definition and process to future research, we used the definitions from RQ1 and the process from RQ2 to scope and plan two empirical studies in papers C and D.

1.6.1 Empirical analysis of dependency bots

For paper C we used the outcome of the exercise in RQ3, which was a list of ten DevBots. Nine out of those ten were dependency management bots and because of this the scope became an investigation of dependency management bots. We selected five of the DevBots from the list. We did a quantitative analysis where our goal was to see if they generated contrasting patterns of activity, if their contributions were well received by developers and if projects used multiple DevBots in parallel. This was done by selecting projects from the BIMAN dataset. In order to compare the activity of different bots, we analysed the issues or PRs authored by those bots in the selected projects over the years. This allowed us to see increasing/decreasing trends of bots usage. Additionally, we analysed how human contributors react to this activity by verifying the proportion of merged PRs that were created by bots and a survival analysis of the issues created by bots. Our survival analysis measured the number of days until an issue is closed. We compared the expected duration of PRs created by bots and those created by humans.

Lastly, we analysed overlapping bot activity by comparing (i) projects using multiple bots, as well as (ii) how the bot activity overlaps over time. Particularly, we filtered projects in which one or more issues were created by two or more bots over the period of, at least, one month.

We also performed qualitative analysis, investigating the factors that guide adopting, switching, using or discarding these bots in open-source software. For the same projects used in the quantitative analysis we looked at issues and PRs where discussions about the bots took place and performed a themed analysis on those discussions.

1.6.2 Empirical analysis of test bots

Paper D was written in proximity to paper A, where we created the taxonomy. Using the taxonomy allowed us to clarify under what context our findings would be valid by highlighting specific properties that help distinguish the test bots from other software bot applications, such as chat bots. E.g. the test bots interact with the user via dashboards and team communication channels in contrast to chatbots. In paper D we performed semi-structured interviews with selected participants. The interviewees were selected based on having previous experience with the test bots, which meant that they were familiar with the overall scope of software test bots and had worked on their development. Four participants agreed to join our study.

Additionally, we also collected data from software artefacts which included test bot code, test case code and requirements for the system under test in

order to investigate the design of the test bots and their test cases.

The interviews were recorded, transcribed and coded. We performed thematic analysis on the interview data in order to find patterns in the raw data later used as the base for the coding. The outcome was categories which summarised the data, gathered and expressed key themes and processes related to their usage of the test bots. Lastly, our findings were later presented to the participants of the interview to clarify and validate our understanding of their process.

The software artefacts (requirements, testbot and test case code) that we collected, were used to get a better understanding of the design of the test bots and their test cases, which was also used to generate insights and inspiration for the optimal test design proposed.

1.6.3 Contributions

For both studies we were able to use the theoretical tools from the previous research questions, that is (i) the taxonomy (ii) the personas (iii) the flowchart to characterise the activity of different types of bots (in this case dependency management bots and test bots). We presented a set of challenges and benefits associated with that specific type of bot and offered recommendations to practitioners on how to improve them.

1.7 Discussion

Here follows a discussion of the implications of the results.

1.7.1 [RQ1:] What delineates bots from plain old development tools?

After creating the taxonomy we noticed one important limitation in its structure. The facets could also be used to map software tools that would not fall under the definition of a DevBot. Hence the taxonomy alone was not sufficient to decide if a tool would be considered a DevBot or not. This raised the need for a more accurate delineation. A key goal was then to systematically identify and categorise what qualities, characteristics, or properties turn a "Plain Old Development Tool" (PODT) into a "DevBot" in the eyes of practitioners. Our hope is that such a characterisation of DevBots will be useful to steer the emerging field of bot-based software engineering research going forward. However, it quickly became evident in our interviews that no single clear-cut definition exists – the same tool may be a bot to some, while it may just be a "plain old development tool" (PODT) for others. However, We find that there are fundamentally three different groups among our interviewees, depending on how they define DevBots for themselves. We name and identify three personas, i.e., practitioner archetypes with different expectations and motivations [12]. What differs between the personas is mainly the association with a different set of human-like traits. First, the **chat bot persona** (Charlie) primarily equates bots to tools that communicate with the developer through a natural-language interface (typically voice or chat), while caring little about what tasks the bot is used for or how it actually implements these tasks. Second, the **autonomous bot persona** (Alex) defines bots primarily as tools that work on their own (without requiring much input from a developer) on a task that would normally be done by a human. Third, the **smart bot persona** (Sam) distinguishes DevBots from PODTs primarily through how "smart" (technically sophisticated) a tool is. Sam cares less about how the tool communicates, but more about whether it is unusually good or adaptive at executing a task.

1.7.2 [RQ2:] What process can researchers use to determine whether a tool is a bot

We created descriptive statements describing a plethora of tools based on different levels of interaction, autonomy and intelligence. The responses from those statements were then used to create a flowchart that can be used to assess whether the combined properties of a software development tool infers that it is considered a DevBot by one of our described personas. A limitation we experienced with the approach during the BIMAN exercise is that the GitHub projects and tool documentation often miss details that would allow us to answer some of the questions in the flow-chart (e.g., the first step asks whether the tool uses a chat, which is often hard to answer conclusively without using the tool).

1.7.3 [RQ3:] To what extent is this definition congruent with definitions used in contemporary research?

Our manual analysis of a sample of 54 widely used tools from the BIMAN dataset [14] showed that only 10 (18.5%) comply with the software development bot (Devbot) characteristics defined by us. It is important to note that this is not intended as a criticism of the dataset, as the remaining 44 tools are certainly not false positives according to *their* definition (which classified all non-human contributors as "bots"). However, it is important for researchers to remain aware that a majority of tools contained in a dataset such as this are relatively simple automation scripts that do not exhibit any specific human-like traits, and are not qualitatively different to the kind of scripting developers have been doing for a long time.

1.7.4 [RQ4:] How can we practically apply the definition and process to future research?

To show how the process could be used we applied it to the BIMAN dataset in accordance with RQ3. It allowed us to find a homogeneous group of tools, which the process had identified as DevBots, that we then could investigate the activity patterns of in several projects. By using several similar bots in the comparison we can, to a certain extent, eliminate effects caused by a specific implementation of the bot and by separating the bots from PODT we will be able to separate phenomenons and problems that are already investigated in other areas from new ones. In the second empirical study we used the definition in the taxonomy to plan our case study and by being able to group test bots we could give recommendations to practitioners building similar bots.

1.8 Threats to validity

Creating a complete taxonomy would require extensive knowledge. Since bots in software development is a new and still developing area it would not be possible to claim that our taxonomy is complete, but our goal was to create an easy to use taxonomy for developers and researchers. To do so we have focused on the prominent properties of contemporary DevBots. To mitigate the threat of unrecognised important properties missing from our taxonomy, therefore making it outdated, we used faceted analysis which is found useful in emerging fields such as ours since it allows the taxonomy to be extended without losing existing structure or invalidating previously mappings of DevBots. This allows future researchers to extend the taxonomy as new important properties emerge.

Designing the investigation of the boundary between a DevBot and "something else" as a mixed-method study allowed us to triangulate the results obtained through interviews with quantitative survey data. However, a number of limitations remain. We cannot claim that our study population for both methods is representative of software engineers in general, as both populations have been sampled through our personal network (convenience sampling). To mitigate this threat, we selected interview participants to cover companies of different sizes and in different domains. However, given our sampling method, a majority of interviewees are working in the same broad geographical region.

For the survey, we did not collect detailed company or geographical information to prevent de-anonymising some participants. However, we have to again assume that the respondent population is relatively homogeneous in terms of geographical distribution. Further, a voluntary survey design is always susceptible to self-selection bias: respondents uninterested in using bots for software development are unlikely to participate in our study. This may also explain why we have received relatively few responses not strongly associated with any of our personas.

A threat is that we were, through our previous interest in the field, pre-exposed to existing research (e.g., existing bot taxonomies in addition to our own [15, 16]), which may have biased our interview design. Hence we may have missed potential characteristics (or, worse, entire personas) because they have not been featured prominently in previous research. However, given that most of our survey respondents mapped well into the three personas identified in this work, we judge this threat to be low. Nonetheless, given our study design, we cannot claim that the identified list of personas is necessarily complete. We have also observed that participants tended to become more conservative in their ratings while progressing through the questionnaire. This threat could have been mitigated by randomising the order of question blocks. Unfortunately, our survey tool did not support this feature.

One of the main limitations when initially creating the flow chart is the limited number of tool descriptions used (32 in total); there might exist a need for more blocks in the chart that were missed by our sample. Later in our research we additionally were able to categorise 54 tools. Here we quickly noticed a practical limitation in the process since the GitHub projects and tool documentation often miss details which hindered our classification of real-world bots using the flow-chart. Therefore, there is a risk that leads to false negatives that would affect the outcome of RQ3. For instance, some tools that we did not classify as bots in our list could be bots for, e.g., a Charlie user persona or an Alex persona whose bots use other team communication channels. We mitigate this threat by focusing our analysis on the distinction between true (actual bots) and false positives (tools misclassified as bots) such that the false negatives have smaller impact on our conclusions. The limited availability of tools documentation was also a challenge in the classification done by Dey et al. [13], hence motivating the identification based on activity patterns for the tool, instead of qualitative answers.

1.9 Related work

There has been several studies done on bots in software development and here we conclude some of those most related to our work.

1.9.1 A brief history of bots

Researchers have in the past investigated and created software to help humans in similar fashion under the names of e.g., daemons, agents and expert systems. Using automated tools to assist humans in software development was partly explored in the 1990's under the moniker of software agents [17, 18], but lack of sufficiently powerful computing resources were obstacles to the development

of AI related services and, consequently, ambitious agent-based systems never became mainstream. Interface agents [3] have acted as successful digital personal assistants that optimise their user's time, learning from user actions to assist with emails and scheduling/rescheduling meetings [1]. However, actually assisting in core software engineering tasks, such as coding or testing, has not to the best of our knowledge been explored in the context of agent-based systems.

1.9.2 Contemporary bots

In 2016, Storey and Zagalsky laid the foundation for research on bots in software engineering. They described how bots are increasingly used to support tasks that traditionally required human intelligence [19]. This early work already established that bots may come in very different forms, support a wide range of use cases, and occupy different roles in software teams. Industrial usage of, as well as academic research on, bots specifically supporting software development tasks have since then gotten more prevalent. Examples include agile team management [20, 21], program repair [22, 23], bug triaging [24], creating performance tests [25], software visualisation [26], source code refactoring [27], or pull request management [24].

Wessel et al. have shown that DevBot usage is indeed widespread, at least in the context of open source software development [28]. Through repository mining they found 48 different bots being used in 93 open source projects. Over one fourth of all analysed projects used at least one bot.

This proliferation of bots is slowly creating demand for coordination between bots in a project, which has recently started to receive attention by Wessel et al. [29] through the design of a "meta-bot".

Our work is orthogonal to these studies, as we are not proposing any concrete new type of DevBot. Instead, the subject of our work is how developers define and perceive DevBots in general.

1.9.3 Benefits and challenges when adopting bots

When it comes to adopting tools in the open-source software ecosystem Lambda et al. [30] looked at how the usage of a number of tools spread by tracking badges from the project's main page. They found that social exposure, competition, and observability affect the adoption. In a recent paper by Wessel et al. [31], the initial interview study revealed several adoption challenges such as discoverability issues and configuration issues. The study then continues to discuss noise and introduces a theory about how certain behaviours of a bot can be perceived as noise. It is still an open research question how exactly bot adoption impacts projects. Previous work from Wessel et al. [28] studied 44 open source projects on GitHub and their bot usage. They clustered bots based on what tasks the bot performed and looked at metrics such as number of commits and comments before and after the introduction of the bots. However, no significant change could be discerned. One reason for this may have been that this study did not sufficiently distinguish between different types of bots, which may be used for very different purposes. Hence, follow-up research [32] focussed foremost on one specific type of bot, namely code coverage bots (1190 projects out of 1194), and found significant changes related to the communica-

tion amongst developers as well as a in the number of merged and non-merged PRs. This was subsequently investigated further in an interview study [33]. These results, that less discussion is taking place, also is what was found by Cassee et al. [34] when looking at how continuous integration impacted code reviews. Peng et al. [35] studied how developers worked with the Facebook mention bot. The study found that mention bots impact on the project was both positive in saved contributors' effort in identifying proper reviewers but also negative as it created problems with unbalanced workload for some already more active contributors.

Other research has indicated further potential challenges of using bots. For instance, a well-known experiment by Murgia et al. indicated that users are sceptical of bot contributions on the developer Q&A site Stack Exchange [36]. Recently, this has also been confirmed by Brown and Parnin through an experiment with bot-generated contributions to open source projects [37] — from 52 pull requests submitted by a DevBot, only two were merged in their experiment (and these two were later reverted by the project owners). These experiences have led some bot developers to camouflage their bots as human developers. For instance, the program repair bot Repairnator [22] has, for a while, submitted pull requests using a human profile to improve acceptance. While understandable, this practice may potentially be harmful to software projects. Ferrara et al. have discussed the threat of malicious, non-obvious bots damaging online ecosystems [38]. In other large online ecosystems with longer experience using bots (e.g., Wikipedia), rigid governance rules have been established [39]. Wikipedia bots need to contain the string "bot" in their name, have a discussion page that clearly describes what they do, and can be turned off by any member of the community at any time.

Our studies are complementary to this work as both look at the benefits and challenges related to adoption of one specific bot type, the dependency management bot type, and the perceived benefits and challenges identified by developers associated with their specific persona.

1.9.4 Bot Identification

An area where bot identification is directly useful is in the (automated) study of developer activity. Software repository mining studies, such as the work published every year at the MSR conference[3], frequently struggle to distinguish between contributions of humans and bots (where the study goal often requires to only include human contributions). Different approaches have recently been proposed to automatically identify bot contributions [13, 40], also leading to the BIMAN dataset, i.e., a large dataset of bot contributions [14] which we build upon in our work.

One challenge with identifying bot contributions is the presence of "mixed accounts" [41], i.e., accounts that are used by humans and bots in parallel. Mixed accounts require an identification of bot contributions on the individual contribution level (rather than classifying entire accounts). Cassee et al. [42] have shown that existing classification models are not suitable to reliably detect mixed accounts. In general, existing approaches are sufficient if the goal is to identify human contributions. However, as a foundation to study

[3]https://conf.researchr.org/home/msr-2021

the bot contributions themselves (e.g., to assess bot impact), existing work lacks fidelity, in the sense that they do not distinguish between different types of automation tools and bots, nor between different types of bots. Our work directly connects to these earlier studies. We use a categorisation model to further investigate the BIMAN dataset [14], particularly with regards to the question of how many of these automated contributions are actually "bots" in a stricter sense of the word. We further quantitatively as well as qualitatively investigate the (dependency management) bots we identified in the BIMAN dataset, further contributing to the discussion related to the impact of bot adoption on open-source projects.

1.9.5 Taxonomies

As a consequence of the advances in the field, proposed taxonomies can assist researchers and practitioners to better understand the opportunities and risks of applying bots, or AI tools in general, to software engineering tasks [15, 43, 44]. For instance, the AI-SEAL taxonomy [43] relies on three facets to classify AI applications in SE. The facets are high level and not exclusive to bots. The goal is to expose the risks of AI applications depending on when practitioners choose to adopt a technology (e.g., before deployment) and what is the level of autonomy of the AI technology. Most relevant to our work, Lebeuf proposes a taxonomy to classify software bots [11, 45]. Unlike DevBots, "software bot" is a more inclusive term to bot applications, hence not being focused on software engineering activities, artefacts and roles. Lebeuf's taxonomy is extensive, covering 22 facets organised into three dimensions describing: i) the surroundings in which the bot lives and operates (Environmental), ii) the internal properties of the bot itself (Intrinsic), and iii) the bot's interactions with its environment (Interaction).

A second bot taxonomy has been proposed by Paikari and van der Hoek [16] and specifically investigates chat bots in software engineering and beyond. In contrast to these taxonomies, our work tackles a related, yet not identical, question: what exactly characterises bots, and in what aspects they are perceived as different from PODTs. Further, and unlike the taxonomies proposed by Lebeuf and Storey or Paikari and van der Hoek, the work focuses solely on bots used for software engineering tasks. Finally, the work is conducted from the perspective of bot users, rather than bot developers and vendors.

1.10 Conclusions

This thesis investigates the definition of bots in software development (DevBots). We have created a faceted taxonomy for DevBots which categorises them by their most prominent properties. To overcome the limitation of the taxonomy, in which one could also categorise tools that would not be considered bots, we investigated what delineated DevBots from plain old development tools. Our analysis shows that achieving one single definition is not possible. Instead we identify and name three personas, i.e., practitioner archetypes with different expectations and motivations. The chat bot persona (Charlie) mostly sees DevBots as information integration tools with a natural language interface, while for the autonomous bot persona (Alex) a DevBot is a tool that

autonomously handles repetitive tasks. Lastly, for the smart bot persona (Sam), the defining feature of bots is its degree of "smartness". We have identified a process in the form of a flowchart, which researchers can use to test whether their tool is considered a DevBot by any of our personas. We have concluded that this definition is not congruent with contemporary definitions as only 10 of 54 investigated tools from a large dataset were considered DevBots by our process. Finally we have shown how the definitions and process can be used in practice by using them in the scoping and planning phase of two recently conducted studies.

1.11 Outlook on the future

We envision at least four possible research directions in the future.

The work in this thesis has mostly been focusing on the view of bot users, with a minor exception in paper D. If we instead look from the angle of the bot developer, one future investigation could be to develop a reference framework (or an architecture) to help people when creating new bots. This framework could build on the existing personas and the identified expectations, benefits and challenges.

One other option is to investigate DevBot ecosystems, the environment being any network and the organisms working together or against each other being the DevBots. The idea emerged during the interview study in RQ1 where one participant mentioned that they had two bots "fighting", one bot had the goal to create tickets in a planning system whenever an error occurred and another bot had the goal to close tickets that had not had that much activity. The result was that several tickets were opened and closed without any humans ever seeing them. The investigation could look at how to build DevBots in a multi-bot environment by mapping out different interaction patterns and anti-patterns.

If we instead study how DevBots impact the users and look more into the productivity aspect on a Human computer interaction level, a question could be how people interact with and are affected by a specific type of DevBots. However, related work has indicated that a problem exists here: we currently are lacking useful and measurable metrics of this effect which in turn makes it difficult to evaluate the impact of using those DevBots. For example, one expectation [15, 44, 46] is that DevBots improve productivity. However, so far we lack strong evidence for this. One of the challenges of such an investigation is that it requires a precise definition of productivity, which turns out to be challenging to achieve – particularly since such a definition would need to distinguish between the productivity of an individual developer and the productivity of a team as a whole.

A fourth option could be to create a decision framework for bots that already exist. To do this we would first have to classify more DevBot into groups and create an interface where developers could add their bot to the correct category for inclusion in the system to create meaningful recommendations to potential DevBot users.