Using an AI Chatbot to Refine Gherkin Specifications Based on Stakeholder Comments

Jianwei Shi
Leibniz Universität Hannover
Software Engineering Group
Hannover, Germany
0000-0001-6228-2478

Alan Dryaev
Leibniz Universität Hannover
Hannover, Germany
alan.dryaev@stud.uni-hannover.de

Kurt Schneider
Leibniz Universität Hannover
Software Engineering Group
Hannover, Germany
0000-0002-7456-8323

***Abstract*—A software project begins with capturing visions and requirements in an understandable format, e.g., vision videos, which represent complex software-based processes. This video invites comments from stakeholders for validating the vision. However, the easy-to-watch videos must be translated to easy-to-validate requirements, which can be written in the semi-formal Gherkin specification.**

Requirements engineers use a video and obtained comments to refine Gherkin specifications. However, this is a demanding task. AI chatbots such as ChatGPT can be used in the refinement. We investigated the effectiveness of using ChatGPT. Two requirements engineers used textual description of a vision video and comments of 12 stakeholders to refine Gherkin specifications with and without ChatGPT 3.5. We asked for stakeholders' opinions on the refined specifications with and without ChatGPT.

Results show that (1) the understandability, and (2) the stakeholders' satisfaction on refined specifications with and without ChatGPT do not differentiate significantly; (3) ChatGPT generated detailed specifications but made formulation errors. We suggest using an AI Chatbot and learning from its answers to achieve stakeholders' satisfaction.

***Index Terms*—Gherkin, ChatGPT, specification, comment**

I. Introduction

Organizations often have a vision for the long-term development. A product vision guides the research and development of new features. A prioritization of these features can be oriented around the (product) vision. For a software project, a vision also directs the definition of a software product at the beginning, before the software implementation. By communicating the vision among stakeholders, requirements are discussed, defined, concretized and validated. The communication helps shape a shared understanding of the vision and the requirements during the project.

A requirements engineer can use a textual vision statement, and simulate interactive scenarios with user interface mock-ups in the communication. Another medium for communicating the vision is video, which visualizes the vision, and shows interactive scenarios with the mock-ups. Schneider et al. [1] define *vision videos* for Requirements Engineering (RE): "A vision video of a software-based system typically shows a problem, an envisioned solution, and its impact, pretending the solution already exists." Figure 1 shows two frames from a vision video of a software called CV Parser which converts a LinkedIn profile to an editable online CV. Figure 1a shows entering verification code for registration; Figure 1b shows editing an online CV. In the vision video, the hand cursors hovering on the "create" button and the "About" text field show the clicking and editing interactions.

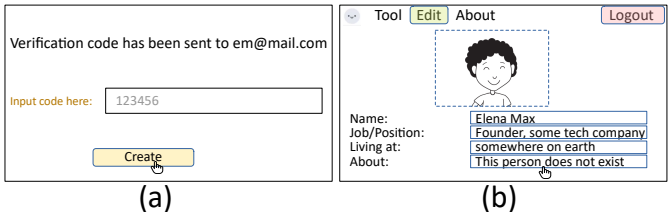


Fig. 1. Two frames of a vision video.

A requirements engineer can use a vision video to communicate requirements with stakeholders. The requirements should be written as a specification, which can be understood by stakeholders, developers, and testers. One such semi-formal specification is a Gherkin specification, which is defined by "Given (preconditions) When (actions) Then (expected results)". Using Gherkin specifications to obtain feedback can identify inconsistencies with customer requirements [2] and reveal hidden assumptions [3]. Generally, Gherkin specifications have two levels of detailing: high-level and low-level. High-level Gherkin specifications describe functionalities without technical details; while low-level ones contain detailed steps which can be directly translated to test code. We use high-level Gherkin specifications in this study, as stakeholders can understand them and give feedback. From frame b in Fig. 1, high-level raw Gherkin specification could be: "**Given** the user is on the edit page **When** the user adapts the CV **Then** the changes should be saved."

However, translating video sequences and related stakeholder feedback to Gherkin is a demanding manual task. A requirements engineer needs to be familiar with the video content, and must consider the feedback properly and write the specification in Gherkin correctly. Since the advent of large language models (LLM) and AI chatbots such as ChatGPT,

How to cite this working paper: J. Shi, A. Dryaev, and K. Schneider, *Using an AI Chatbot to Refine Gherkin Specifications from Stakeholder Comments*, Working Paper, 2024, Available: http://dx.doi.org/10.15488/17361

Copilot etc., there are many new approaches to let the AI chatbots carry out demanding manual tasks instead of humans. In this study, requirements engineers use ChatGPT to generate raw Gherkin specifications by feeding textual information of a vision video and stakeholders' comments from a previous study. Next, the requirements engineers show stakeholders the vision video with the raw Gherkin specifications to obtain stakeholders' comments.

Then, the requirements engineers refine the raw Gherkin specifications based on obtained comments. We want to investigate this Research Question (RQ):

> **Research Question**: How do requirements engineers refine Gherkin specifications based on stakeholders' comments with the help of an AI chatbot effectively?

The aim of this paper is to propose approaches of using AI chatbots in generating and refining raw Gherkin specifications. In our evaluation, we investigate the effectiveness on the **refinement** of raw Gherkin specifications using ChatGPT.

In this paper, we make the following contributions:

1) We propose a systematic approach with concrete steps for working with AI chatbots, to generate and refine Gherkin specifications based on stakeholder comments by using a vision video.
2) We compare specifications which are refined either with or without the help of ChatGPT, to evaluate the effectiveness on refinement of Gherkin specifications.
3) We discuss the evaluation results and suggest to learn from refined specifications by AI chatbots.

The rest of this paper is structured as follows: Section II lists fundamentals and related work. The approach and the concrete steps are explained in Section III. Section IV explains how the experiment is designed and evaluates the results. Section V discusses the refined specifications from an AI chatbot and requirements engineers. The paper is concluded in Section VI.

II. Fundanmentals and Related Work

This section starts with an introduction of the LLM and current AI chatbots. Then we list applications of the AI chatbots generally for Software Engineering and specially for Requirements Engineering (RE). Next, fundamentals of Gherkin specifications are illustrated, including related work about best practices in formulating Gherkin specification. Last, we indicate the novelties in this work.

A. Fundamentals of LLMs and AI Chatbots

A large language model is a generative artificial intelligence based on transformer models [4], which has been trained on existing texts to answer user requests (e.g., summary, classification) [5]. Some examples of LLMs are Generative Pre-trained Transformer (GPT) from OpenAI[1], llama from Meta[2], General Language Model from Zeng et al. [6]. Based on the LLMs, AI chatbots are emerging in the last years, like ChatGPT from OpenAI, Copilot from Microsoft, Wenxin Yiyan from Baidu. A human chats with an AI chatbot to get instant answer of a question. The answer is based on the training data and the algorithms of the LLM.

To make the answer from AI chatbots better, the LLMs behind the AI chatbots can be 1) fine tuned by updating the LLMs' parameters, or 2) given one or many demonstrations (called one-shot or few-shot settings in Brown et al. [7]). For the second approach, the demonstrations should clearly indicate what the requested answer looks like. Then the one-shot or few-shot settings can help the LLMs learn from examples and make fewer mistakes.

B. Application of AI Chatbots

Ozkaya [8] points out opportunities and challenges of using AI Chatbots in Software Engineering activities, including specification generation, provision of instant feedback for developers, generation of tests and documentations, translation of programming language. Meanwhile, they emphasize that the next-generation software engineers must know how to assess trustworthiness of answers from the LLMs and deal with the incorrect answers.

In specification generation, Xie et al. [9] investigated the effectiveness of 15 LLMs by feeding documentation (e.g. API documentation, Javadoc comments) as input and receiving specifications in real programming code as output. They compared the LLMs with tranditional approaches and found that the best-performing LLM *StarCoder* outperforms the traditional approaches. On the other hand, Xie et al. reported that *StarCoder* generated a small portion of ill-formed, incorrect, or incomplete specifications. The root causes of the failure were among others 1) unclear/poorly written request messages, 2) missing domain knowledge.

Many studies were presented at the 31st IEEE International Requirements Engineering 2023 conference. Chen et al. [10] have used GPT-4 for creating goal models. They have fed different messages (long/short domain description, open/closed questions) to evaluate the effectiveness of GPT-4. For the same message, the prompt has been run for several times. The answers of GPT-4 were similar between long and short domain descriptions. Results of GPT-4 for open questions received a better grade than results for closed questions. Chen et al. also find out GPT-4 made less syntax error if a syntax description is given. Moreover, they report a large variation of the answers in an experiment: each answer covered different useful ideas. Hence, they suggest to run the prompt for several times and aggregate the received answers.

Also at the RE 2023 conference, Fantechi et al. [11] have used ChatGPT 3.5 to detect inconsistency in requirements. By comparing results from ChatGPT with ones from students with RE experience, the students performed better detection. Ruan et al. [12] propose to use ChatGPT to extract requirements models by using structured input text. Zhang et al. [13] have explored to feed GPT-4 with user feedback for generating persona templates.

[1]https://openai.com

[2]https://ai.meta.com/llama/

C. Fundamentals and Related Work of Gherkin

Smart [14] explains that Gherkin can be used in communication about an example of a software function to be developed. The requirements engineers (called business analysts in [14]) write down the example in Gherkin and hand in the Gherkin specifications to developers and testers. The Gherkin specifications are written in feature files. Each feature file contains a user story and corresponding scenarios. A feature file related to Fig. 1b may look like this:

Listing 1. A feature file example.

```
Feature: Edit CV
# User Story
As user, I want to edit my CV to change the details.

# Corresponding scenarios
  Scenario: Instant saving while editing the CV
    Given the user is on the edit page
    When the user adapts the CV
    Then the changes should be instantly saved.
  Scenario: Editing the fonts in the CV
    Given the user is on the edit page
    When the user chooses font type, size, color
    Then the chosen fonts are instantly shown.
```

For writing the scenario steps, Wynne and Hellesøy [15] propose to avoid incidental details to make the steps easy to read, i.e. write "When the user chooses font type and size" (high-level, declarative) instead of "When the user chooses font 'Calibri', size 9 pt" (low-level, imperative). In a high-level scenario step, every detail of the step are described; in a low-level step, the details are hidden and need to be clarified for the developers and testers. Wynne and Hellesøy point out that the choice of the styles depend on among others 1) the kind of application to be developed, 2) "domain expertise of the programmers", and 3) "the level of trust that the nontechnical stakeholders have in the programming". Similar argumentation is found in Binamungu et al. [3]: on the one hand, the higher level (declarative) steps are often "closer to the domain concepts that end users are familiar with"; but, writing the "glue code" between the Gherkin specifications and real test is required.

In addition, Binamungu et al. [3] investigated quality aspects of Gherkin specifications and found among others the importance of understandability. In a survey, they have asked community members (60.7% of them are developers) with Gherkin experience and found out over 75% of the participants agree that a scenario step should contain domain vocabulary which is already used in other steps.

This work has following **novel contributions**: 1) We use ChatGPT on generating specifications in Gherkin, a natural language, not specifications in programming code [9]; 2) We consider the good practices of using ChatGPT from Chen et al. [10] in our methodology; 3) We evaluate empirically using not only objective but also subjective metrics, and analyze the refined specification regarding the two related work [3], [15].

III. Methodology

In requirements elicitation, viewing a vision video often triggers a discussion between stakeholders. Requirements engineers can then collect comments from the discussion. They can work with an AI chatbot to derive requirements in Gherkin. In this work, we focus on **functional** requirements from the vision video.

We have compared three free AI chatbots in autumn 2023: *ChatGPT* (GPT 3.5), *Copilot* (was called Bing Chat), and *perplexity.ai*. We used among others these criteria in the comparison: max. number of request, time for waiting an answer, and vocabulary consistency. We found that ChatGPT 3.5 had the best performance in the mentioned criteria. Hence, we decided to use and evaluate ChatGPT 3.5 in our work. The GPT 3.5 model has been trained with the data up until Sep. 2021[3].

The general activity is described in Fig. 2:

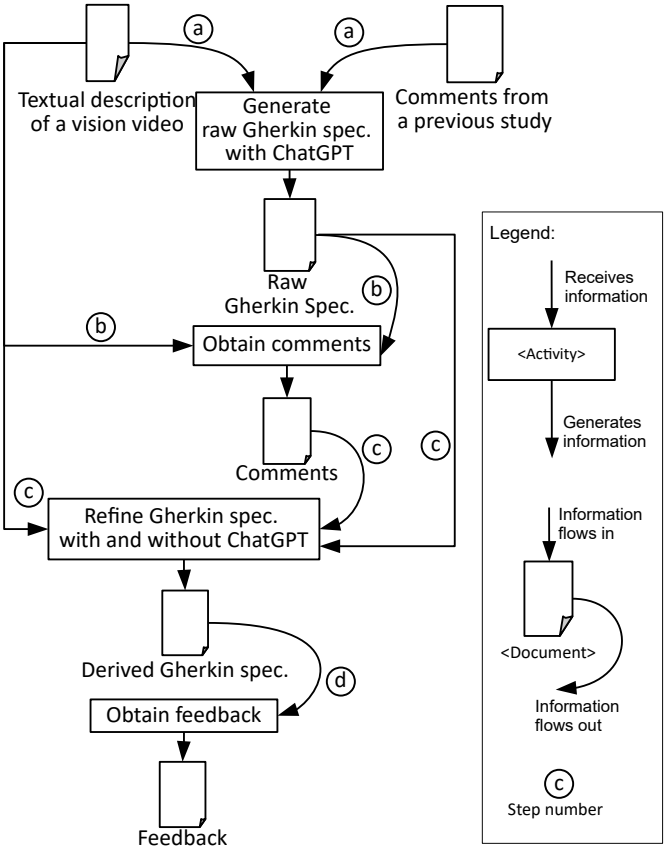


Fig. 2. The information flow diagram of our approach.

In **step a**, requirements engineers use ChatGPT to generate raw Gherkin specifications of functional requirements from a vision video and comments from a previous study. The raw Gherkin specifications remain abstract, as the shown functionalities in the vision video are abstract. In **step b**, the raw Gherkin specifications and the vision video are shown to

[3]https://platform.openai.com/docs/models/gpt-3-5

stakeholders; the requirements engineers obtains comments. In **step c**, the requirements engineers use the comments and textual description of the vision video to refine the raw Gherkin specifications, with and without ChatGPT (different account than the one in step a). In **step d**, the potential stakeholders express their opinions on the refined Gherkin specifications. In this Section, we describe steps a, b, and c in detail. We evaluate steps c and d in Sec. IV.

A. Generating Raw Gherkin Specifications

We feed following texts to ChatGPT: 1) a textual description of the vision video and 2) corresponding user stories, 3) format description of a feature file and an example (one-shot setting [7]). We regenerate the Gherkin specifications from ChatGPT and select the best ones.

We use the vision video CV Parser in this step as a preparation for the evaluation. The description of the vision video consists of all the features presented in the vision video. User stories are derived systematically: 1) User stories are derived from the comments of the previous study by Fokam Piam [16]; 2) The derived user stories are organized in a tree: Epic descriptions are at the top of the tree; general user stories are under an epic; detailed user stories are under a general user story. 3) Missing general/detailed user stories are added to the tree. Last, we have nine general user stories and feed them to ChatGPT.

The raw Gherkin specifications are abstract, i.e. they are plain descriptions of the vision video, without interpretation of the details and error handling. Example: Vision video CV Parser shows that the verification code is sent to an e-mail address (Fig. 1a). The raw Gherkin specification of this function does not handle the case of an invalid e-mail address.

B. Obtaining Comments

A requirement engineers shows stakeholders the raw Gherkin specifications and the vision video. Following motivating questions are asked to encourage feedback:

- Which new functions do you want / are missing for you?
- Do you want a functional change? Which one?
- Is a process too complicated? To what extend should the process be simpler?
- Do you want an alternative scenario? Which one? What should happen?
- What should happen, if something does not work?
- Do you want a more exact process description? What should happen?

The raw Gherkin specifications consist of scenarios. Each scenario is written in format "Scenario ... Given ... When ... Then ..." (see also Listing 1). For good readability to potential users, we write multiple times of "When ... Then ..." under a Given-step (see also message 1 in Tab. I), because the Gherkin specifications are more compact than the specifications which contain the same *Given* step many times. The requirements engineer collects textual comments of each scenario. This collection can be conducted through an online survey or an interview meeting.

C. Refining The Raw Gherkin Specifications

A requirements engineer interacts with ChatGPT, as Fig. 3 shows. We utilize a different ChatGPT account than the one in step a (generation of raw specifications), to avoid possible deviations in refinement.

1) Feed Context Information and Comments to the AI chatbot: At the beginning of this step, the requirement engineer feeds the AI chatbot context information, i.e. the textual description of the vision video and the raw Gherkin specifications (see message 1 in Tab. I). Subsequently, the requirements engineer provides the AI chatbot with comments, requesting an enhancement of the Gherkin specifications for each comment individually. In this second message, a clear reference from a comment to corresponding scenario is given (message 2a in Tab. I). If the requirements engineers have an impression that the comment does not have a connection to a single scenario, they use another message without the clear reference (message 2b in Tab. I). For step c, we use the zero-shot setting [7].

TABLE I
MESSAGES TO CHATGPT (TRANSLATED FROM GERMAN)

No.	Content
1	On the website of CV Parser, you can convert your LinkedIn profile into a CV. This website is now described. Description: ””” At the beginning you are not logged in and are on the login page. Here you can click on “create link”, which will redirect you to a registration form. *(Some content is omitted)* ””” For this context, there is now an excerpt from the test specification in Gherkin: ””” Feature 1: Registration on CV Parser As an unregistered user, I would like to register so that I have an account on CV Parser. Szenario 1: Registration of a new user Given the user is on the registration page When the user clicks the “Create” link Then should the user be redirected to a registration form When the user enters his e-mail and password And repeats the password And clicks “Create” Then should the user receive a confirmation e-mail *(Specifications of features 2 to 6 are omitted)* ”””
2	You are a requirements engineer and should enhance the test specification in Gherkin.
2a	For scenario *1*, following comment is written: “*For using this service, I don’t want to register.*” Enhance the test specification accordingly.
2b	Following comment is written from a stakeholder: “*Could you add an option that allows the user to receive a new confirmation email if it has not arrived?*” Enhance the test specification accordingly.

2) Selection of Gherkin Specifications: After feeding a comment, the AI chatbot gives updated Gherkin specifications. We follow the recommendation of requesting multiple answers from Chen et al. [10] to receive alternative answers. In our case, we click the regenerate button on ChatGPT to request alternative answers. The requirements engineer selects the best answer as *Selected Gherkin Specifications* in Fig. 3.

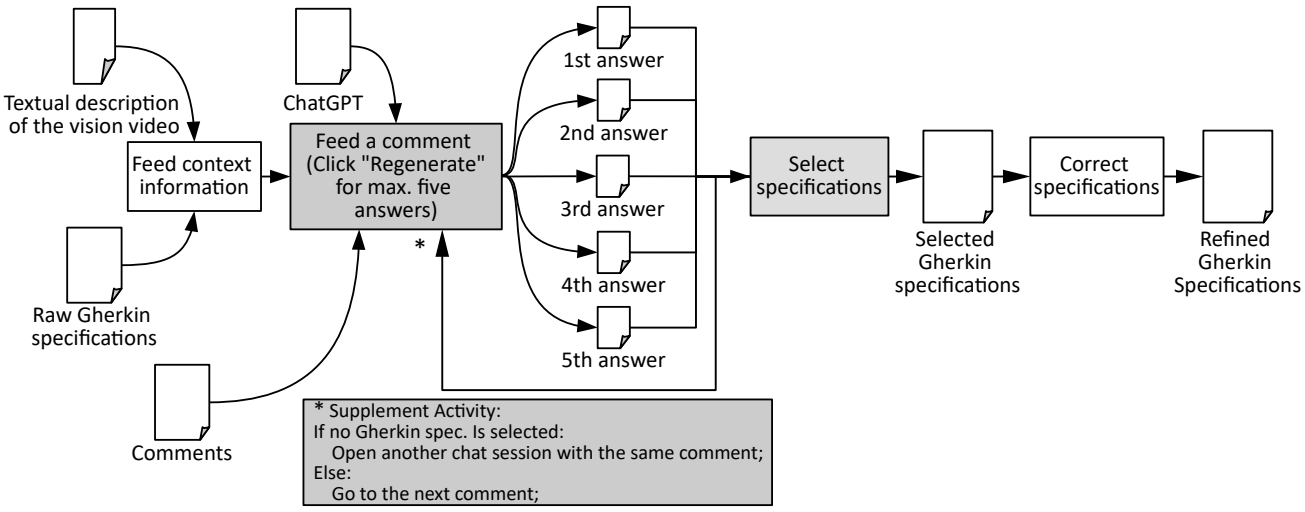


Fig. 3. Detailed step c in Fig. 2 refining specifications with ChatGPT. Note that the activities (rectangles) in grey background can be repeated.

As Fig. 3 shows, the requirements engineer requests max. **five** answers for message 2 (Tab. I). We have set this number in a test with a separate ChatGPT account. In our test, we found that we could receive a good enhanced scenario by having up to five answers.

We suggest the usage of multiple chat sessions, for the worst case that the requirements engineers are not satisfied with any received answers on current chat session. Then another chat session with message 1 is opened and the above described processes are repeated. On ChatGPT, the requirements engineer clicks the regenerate button on the answer of message 1 to create another chat session. We observe that the regenerate button for the answer of message 1 appears only before the message 2 is sent. Hence, the requirements engineer clicks the regenerate button two times **before** sending message 2. This mechanism is shown as asterisk and the grey rectangle "Supplement Activity" in Fig. 3.

3) Correction: However, the selected Gherkin specifications could have some errors in formulation and syntax. A requirements engineer corrects the specifications. We analyze the selected Gherkin specifications in Sec. IV and discuss them in Sec.V.

IV. Evaluation

We want to compare and evaluate the specification refinement with and without ChatGPT. Based on the goal template by Basili and Rombach [17], we define our evaluation goal:

Goal definition:
We analyze the refined Gherkin specifications from stakeholder comments with and without ChatGPT 3.5
for the purpose of evaluation
with respect to the effectiveness
from the viewpoint of (1) the requirements engineers who can write specifications in Gherkin and interact with ChatGPT and (2) the potential stakeholders
in the context of a controlled online experiment by using a vision video.

A. User Study Design

Starting from the evaluation goal, we ask:

EQ$_1$: Are opinions of potential stakeholders the same about Gherkin specifications refined from ChatGPT and from only requirements engineers?

Concretely, we want to ask stakeholders their satisfaction, understandability of derived specifications, and their acceptance in the first appointment. Satisfaction means to which extend a stakeholder comment is considered in a refined specification. To investigate possible influencing factors of an acceptance, we ask:

EQ$_2$: Does stakeholders' satisfaction correlate with acceptance?

EQ$_3$: Does understandability correlate with acceptance?

Figure 4 summarizes our study design in a goal-question-metric paradigm. Table II explains metrics 1 to 3.

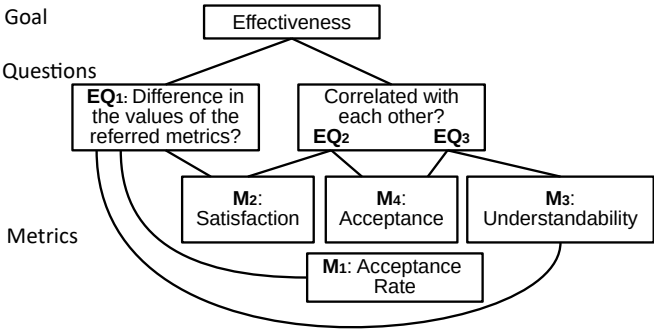


Fig. 4. The GQM paradigma.

We want to test the following hypotheses:

TABLE II
EXPLANATIONS OF METRICS 1 TO 3.

Metric	Meaning
M_1	Acceptance rate of each participant (in percentage)
M_2	Participants' satisfaction of a refined specification (see below)
M_3	Understandability of an updated specification (see below)

Participants state their agreement for M_2 and M_3 in Likert scale (1: Strongly Disagree - 2: Disagree - 3: Rather Disagree - 4: Rather Agree - 5: Agree - 6: Strongly Agree):
M_2: The enhanced specification has taken my comment well into consideration.
M_3: The enhanced specification is well formulated and easy to understand.

Hypotheses 1 to 3: There is a difference in
H_1: the acceptance rate (M_1)
H_2: the stakeholders' satisfaction (M_2)
H_3: the understandability (M_3)
between the refined Gherkin specifications with and without ChatGPT.
Hypotheses 4 and 5: The following metrics do not correlate with each other:
H_4: the stakeholders' satisfaction (M_2) and the acceptance (M_4)
H_5: the understandability (M_3) and the acceptance (M_4)
The corresponding null hypotheses assume that there is no difference or no correlation.

Figure 5 shows the activities for a participant. Two requirements engineers (with IDs 1 and 2) are involved in this experiment. Before the first appointment, a participant learns Gherkin in an online training and must pass the comprehension test. In the first appointment, requirements engineer 1 shows the participant the vision video with the raw Gherkin specifications (6 scenarios). After that, the participant is asked to write comments (1 comment per 1 scenario) in an online survey form. The participant does not have to write a comment for all six scenarios. Subsequently, one requirements engineer refines Gherkin specifications with ChatGPT; another requirements engineer refine Gherkin specifications without ChatGPT.

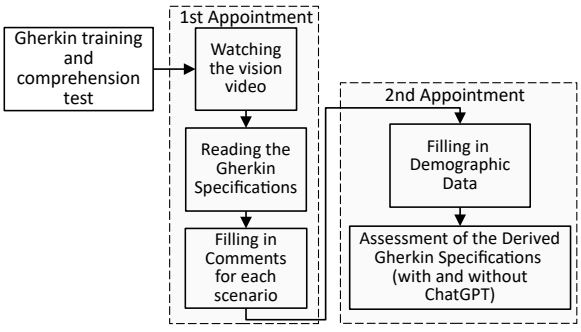


Fig. 5. Activities of a participant.

From one comment, specifications are refined in two ways: one requirements engineer interacts with ChatGPT; another requirements engineer refined without ChatGPT. To minimize the individual influence in writing specifications, we have mixed the assigned person, as Tab. III shows:

TABLE III
ASSIGNMENT OF COMMENTS.

Requirements Engineer	For following participants	
	101,102,103,104,105,112	106,107,108,109,110,111
1	With ChatGPT	Without ChatGPT
2	Without ChatGPT	With ChatGPT

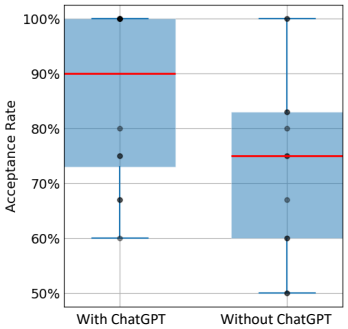
Before the second appointment, requirements engineer 1 corrects the refined specification from ChatGPT and from requirements engineer 2. In the second appointment, the participant fills in demographic data. Requirements engineer 1 shows the participant the original comments which the participant wrote in the first appointment. For each comment, the corrected Gherkin specifications refined from ChatGPT and from only requirements engineers are shown and the participants state their opinions (M_2, M_3, M_4) on both.

B. Demographics

Twelve participants took part in our experiment voluntarily and showed their interest in using CV Parser. They are potential stakeholders. At the time of the study in Nov. 2023: One participant worked as trained electronics engineer; One worked as trained physiotherapist; Six were undergraduate students (number in brackets) from subjects of English and politics (1), mathematics (1), computer science (3), and economics and management (1); Three were postgraduate students from subjects of applied computer science (1) and computer science (2). One was a PhD student in electrical engineering. Five participants were between the ages of 20 and 24 years old, six participants were between 25 and 29. One participant did not state the age. Two requirements engineers are the first and second authors: Requirements engineer 1 is the second author who was a undergraduate student, engineer 2 is the first author who is a PhD student. Both have studied computer science and were between 24 and 30. Both have experience in Gherkin and ChatGPT.

C. Answering Evaluation Questions 1

Figure 6 shows the results of M_1:



(a) Box Plot of M₁

	With ChatGPT	Without ChatGPT
Mean	85%	74%
Standard Deviation	16%	17%
Mann Whitney U Test		
U	46.5	
p	0.134 (Two-tailed)	

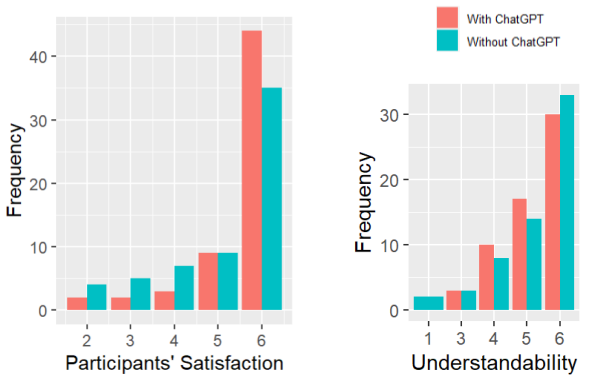
(b) Statistical Results of M₁

Fig. 6. Box-plot and statistics for the acceptance rate.

Participants have accepted more refined specifications from ChatGPT, 85% on average. However, the Mann Whitney U

Test does not show the difference between both options. We cannot reject the null hypothesis of H_1.

Histograms show results of M_2 (Fig. 7a) and M_3 (Fig. 7b). We have observed a similar distribution for both metrics and cannot find the statistical difference between both options. We cannot reject the null hypotheses of H_2 and H_3.



(a) Histogram of M2 (b) Histogram of M3

	M2			M3	
	With ChatGPT	Without ChatGPT		With ChatGPT	Without ChatGPT
Median	6	6		5.5	6
U	1491.0			1745.5	
p	0.054 (Two-tailed)			0.754 (Two-tailed)	

(c) Statistical Results (Mann Whitney U Test) of M2 and M3

Fig. 7. Histograms and statistics for satisfaction and understandability.

D. Answering Evaluation Questions 2 and 3

Correlations between a dichotomous feature (M_4 accept or reject) and a ordinal scale (M_2 and M_3 in Likert scale) should be calculated. Hence, the biserial ρ for shared rankings in the ordinal scaled data [18] is calculated to test H_4 and H_5. We observed the positive ρ values which mean positive correlations. Subsequently, we used the Mann Whitney U test to test the significance of the positive correlations, according to Bortz and Schuster [19]. Because we have noticeable distribution of acceptance (N_2=96) and rejection (N_1=24), we used the Vedooren formula (cited in Bortz [18]) for corrections of the u value in Mann Whitney U Test. Table IV shows the results:

TABLE IV
CORRELATION ANALYSIS.

	Test for H_4	Test for H_5
Biserial ρ	0.665	0.302
U	379	836
$u_{corrected}$	-5.509	-2.097
p (One-tailed)	<0.0001	0.018

The p values of the Mann Whitney U test show the significance of the correlations (significance level α=0.05). We accept H_4 and H_5.

V. DISCUSSION

Besides quantitative analysis in Sec. IV, we examine the refined specifications and then list threats to validity.

A. Possible Reasons of Negative Opinions

We have examined all refined specifications which were either rejected, do not consider a given comment properly ($M_2 \leq 3$), or had low understandability ($M_3 \leq 3$). First, we analyze specifications with low understandability. From these specifications, we list examples where obvious understanding problems can be observed. Second, we compare rejected specifications with accepted ones refined from a given comment. Meanwhile, we consider the satisfaction.

1) Possible Reasons for Low Understandability: Listing 2 shows three examples of low understandability:

Listing 2. Gherkin step examples of low understandability.

```
# Example 1, refinement without ChatGPT
  And the user has a public LinkedIn profile
  When the user pastes the address of her LinkedIn
       profile

# Example 2, refinement with ChatGPT
  When the user click on the "create" link
  And enters the false entered e-mail

# Example 3, refinement without ChatGPT
  Then the user does not find a confirmation e-mail
  And the user is forwarded to a website with a
       resend button
```

In the first example ($M_3 = 1$), "public LinkedIn profile" may cause confusions in understanding, because the privacy setting of the LinkedIn profile is mentioned neither in the video nor in the raw Gherkin specifications. In the second example ($M_3 = 3$), the meaning of "false entered e-mail" is unclear. Instead, "not existing e-mail" which is written from a requirements engineer conveys the intended meaning. The third example ($M_3 = 3$) is about a fallback if a user does not get a confirmation e-mail on registration. Hence, "the user does not find a confirmation e-mail" with the use of "find" may cause understanding problems. For other refined specifications of low understandability, we cannot identify obvious understanding problems. Maybe the specifications contain difficult vocabulary or did not fit the stakeholders' comments.

2) Possible Reasons for Rejection and Low Satisfaction: We want to estimate the possible reasons behind a rejection or low satisfaction, which correlate with each other (as H_4 is accepted).

Table V shows examples of refined scenarios from stakeholder comments. We have identified following possible reasons of rejections or low satisfaction:

- misunderstanding: The requirements engineers made a wrong suggestion or understood the comment wrong. Example: Participant 101, scenario 1.
- too abstract: The specifications are not concrete or not detailed. Example: Participant 101, scenario 6.

TABLE V
COMPARISON OF SCENARIOS (TRANSLATED FROM GERMAN)

Comment	Refined scenario without ChatGPT	Refined scenario with ChatGPT
Participant 101, scenario 1: I do not want to register to use the service.	(Rejected, comment not properly considered) Scenario 1: Account creation without registration Given the user is on the login page When the user ticks "I don't want to register" And clicks on the "Create" link Then the website shows "we need your email address to send you a confirmation link" When the user enters her email And clicks on "Create" Then the user should receive a confirmation email	(Accepted) Scenario: Account creation without registration Given the user is on the login page When the user clicks on "Continue without registration" Then the user should be redirected directly to the tool to create their CV
Participant 101, scenario 6: Legal notice missing. Further information is missing. Text tutorials would also be useful.	(Rejected, comment not properly considered, the specification is more abstract than the right one) Scenario 6: Displaying the "About" page Given the user is logged in to the CV Parser website When the user clicks on the "About" menu bar Then the website should show further links for tutorials in text form, tutorials in video form, legal notice, privacy information When the user clicks on a link mentioned above Then the website should show the corresponding information	(Accepted, the specification is structured in three scenarios) new scenario 6.1: Display of legal notice Given the user is on the "About" page When the user clicks on "Legal notice" Then the legal notice should be displayed New scenario 6.2: Displaying further information *(Steps of this scenario are omitted)* Then further information should then be displayed New scenario 6.3: Displaying textual tutorials Given the user is on the "About" page When the user clicks on "Textual tutorials" Then textual tutorials should be displayed
Participant 111, scenario 5: It is unclear whether I should enter the old or new password. And if it is the new one, I would personally prefer receiving an e-mail with a link that takes me to the page where I can enter the new password.	(Accepted) Scenario 5: Password reset Given the user is on their profile When the user clicks on "Reset Password" Then the user should receive an email When the user enters the new password And repeats the new password And clicks on "confirm" Then the user's password should be reset	(Rejected, comment not properly considered) Scenario 5: Password reset Given that the user is on their profile When the user wants to change their password Then the user has two options: - Option A: The user enters their current password and selects a new password. - Option B: The user requests a password reset e-mail. - In this case, the user receives an e-mail with a link to reset the password. - The user clicks on the link and is redirected to a page where they can enter a new password. *(Some steps are omitted)*
Participant 106, scenario 3: The possibility to change the font and font color, as well as font size.	(Accepted) Scenario 3: Given the user is on the "Tool" page When the user uses a menu bar to adjust font type, color, and size in the extracted resume Then the changes should be saved immediately	(Rejected, comment not properly considered) Scenario 3: Extracting a CV with customization options *(Some steps are omitted)* When the user changes the font, font color and/or font size And clicks on "Confirm" Then the customized font information should be saved in the CV
Participant 106, scenario 4: additional buttons where you can undo edited steps or restore the change. And remove the Confirm button so that it saves automatically	(Accepted) Scenario 4: Editing the CV Given the user is on the "Edit" page When the user customizes their resume And clicks "Confirm" Then the changes should be saved immediately When the user clicks the "Undo" button in the menu bar Then the last edited step should be undone When the user clicks the "Restore" button in the menu bar Then the step should be restored	(Rejected) Scenario 4: Editing and undoing changes in the CV Given the user is on the "Edit" page When the user makes changes to the CV And clicks on "Undo" Then the last change made should be undone When the user clicks on "Redo" Then the undone change should be restored New scenario 5: Automatic saving of the edited CV Given the user is on the "Edit" page *(Some steps after the given step are omitted)* Then the changes should be saved automatically

- too detailed: The specifications provide too many options or contain unnecessary steps. Example: Participant 111, scenario 5.
- comment not properly considered: We can see obvious inconsistencies between the comment and the specifications. Example: Participant 106, scenarios 3 & 4.

In our experiment, ChatGPT did not misunderstand the comments, while requirements engineer misunderstood. We observe that the specifications from ChatGPT are generally more detailed than the specifications from two requirements engineers. We also found that the requirements engineer considers comments 3 and 4 from participant 106 together, ChatGPT not always. In comment 4, participant 106 wrote "saves automatically": This requirement is considered in scenario 3 ("saved immediately") from requirements engineers, but not explicitly mentioned in scenario 3 by ChatGPT. Participant 106 accepted scenario 4 from requirements engineers, even the "Confirm" button was not removed as required.

B. Examples of Formulation Errors from ChatGPT

During refinement of Gherkin specifications, requirement engineer 1 has corrected the specifications from requirements engineer 2 and ChatGPT. We have identified and corrected formulation errors from ChatGPT. Table IV shows two examples:

TABLE VI
EXAMPLES OF FORMULATION ERRORS FROM CHATGPT

When the user decides not to register and clicks on "Continue without registration" instead
Error explanation: The sentence "The user decides not to register" does not contain an action.
Given the user is on the "Edit" page When the user wants to add their own categories or edit existing ones Then the user should be able to create new categories or edit existing ones
Error explanation: The use of modal verbs "wants to", "be able to" does not mean an action (when step) or expected results (then step).

We observed that ChatGPT tended to write Gherkin steps which do not have a real meaning: The step is neither an action (when step) nor an expected result (then step), but an intention. This is a weakness of ChatGPT and should be by all means avoided in refinement.

C. Reflection of Gherkin Specification Quality

In the study, we tried to follow the best practices [3], [14] in writing Gherkin specifications. After the study, we reflect on the refined specifications according to Binamungu et al. [3].

- The specifications should be "concise, testable, understandable, unambiguous, complete, and valuable". Most refined specifications fit this quality aspect, but some abstract specifications do not.
- The specifications should "use the terminology understood by all project stakeholders". We have some specifications, which have low understandability and may have difficult vocabulary for participants.
- We adapted the practice that "each scenario should test one thing". In our study, each scenario tests one process, which contain many "When ... Then ..." structures. The scenario is then easy to read for potential users. For later development and testing, the scenario can be extended to multiple scenarios, each having only one "When ... Then ..." structure.

D. Threats to Validity

Our results are subject to some threats to validity which we discuss in the following according to Wohlin et al. [20].

1) Construct Validity: The formulation style of refined Gherkin specifications without ChatGPT can vary between requirements engineers. If a requirements engineer writes the Gherkin specifications in a style which a participant exactly does not understand, the comparison between ChatGPT and requirements engineers could be biased. Hence, in deriving without ChatGPT, the comments from six participants are assigned to requirement engineer 1 and the comments from another six to requirements engineer 2.

Tiredness of a participant due to long participation time may influence their opinions. In our pilot study, we found that six scenarios are appropriate to prevent the tiredness. In our study, the first and second appointments took about 30 minutes each on average. Hence, we have minimized the participation time to avoid the tiredness.

2) Internal Validity: ChatGPT could learn after clicking "Regenerate". In addition, the preparation of the raw Gherkin specifications with ChatGPT could also influence the result of step c in Fig. 2. As prevention, after refining specifications from comments of a participant, a new chat section is opened for comments of another participant. In addition, the experiment has been carried out on a ChatGPT account which was not previously used in the context of this work.

3) Conclusion Validity: The questions and activities for participants should be clear, especially for the stakeholders' satisfaction (M_2). Instead of asking "Are you satisfied with the enhanced specification?", we ask the participants how well the enhanced specifications take their comments into consideration (cf. Tab. II).

During refinement of specifications, requirements engineer 1 has corrected the errors from ChatGPT, such as formulation errors mentioned before. This individual correction could lead to biased evaluation. To mitigate that, requirements engineer 1 has communicated with requirements engineer 2 about the error types. Both have agreed on the correction methods, i.e., deleting the modal verbs or unnecessary steps, correction according to the Gherkin syntax and German grammar.

4) External Validity: If a participant did not provide any comments, we could not elicit requirements. To encourage comments, we provide motivating questions (see Sec. III-B). These motivating questions are generally formulated to prevent influence on participants' opinions.

VI. CONCLUSION AND FUTURE WORK

We have proposed a systematic and concrete approach by using an AI chatbot for deriving specifications from stakeholder comments. Although we chose ChatGPT for evaluation, we argue our approach can be applied to other AI chatbots, with adapted practices in the mentioned "regenerate" function for requesting several answers at one time. The regenerate function can be conducted on *Copilot* by sending the same message again; on *perplexity.ai* by clicking the "Rewrite" function.

We have evaluated the effectiveness of refinements with and without ChatGPT. The evaluation results show that the effectiveness does not differentiate significantly. In addition, the stakeholders' satisfaction and the understandability of a specification are possible positive factors for an acceptance. ChatGPT 3.5 wrote detailed specifications: Requirements engineers can learn from that. We argue that communication with stakeholders by discussing detailed specifications helps clarify the misunderstandings, although rejections were made on specifications from ChatGPT because of too much detail. However, ChatGPT 3.5 wrote modal verbs and unnecessary Gherkin steps in specifications: Requirements engineers

should correct and avoid those errors. After knowing strengths and weaknesses from AI chatbots, requirements engineers can choose to work with AI chatbots if needed in refining Gherkin specifications.

We answer the research question how requirements engineers refine specifications with AI chatbots effectively:

> **Answer to Research Question**: Requirements engineers write a request which contains a comment and a reference to the Gherkin specifications. They require several answers of a request, select the proper ones and correct the answers.

Our approach can be applied in agile or hybrid software development, where specifications are updated according to stakeholders' feedback during the development. A product owner can replace the vision video with any other requirements documentations (e.g. mock-ups, screenshots of the software under development, user stories, goal models, explanations) and use our approach to generate and refine specifications from stakeholders' feedback.

This work invites the community to evaluate efficiency for specification refinement by AI chatbots. Also, this work can be extended in another set-up of specification generation and refinement. If the Gherkin specifications are already refined from stakeholder comments, the future work could explore the code generation from the Gherkin specifications, as Cheminiz et al. [21] envisioned.

ACKNOWLEDGMENT

This work is funded by Deutsche Forschungsgemeinschaft (DFG) - Project number 289386339 (ViViUse).

REFERENCES

[1] K. Schneider, M. Busch, O. Karras, M. Schrapel, and M. Rohs, "Refining vision videos," in *Requirements Engineering: Foundation for Software Quality*, vol. 11412. Essen, Germany: Springer International Publishing, 2019, p. 135–150.

[2] J. Shi, J. Mönnich, J. Klünder, and K. Schneider, "Using gui test videos to obtain stakeholders' feedback," in *2023 IEEE/ACM International Conference on Software and System Processes (ICSSP)*. Melbourne, Australia: IEEE, May 2023, p. 35–45. [Online]. Available: https://ieeexplore.ieee.org/document/10169066/

[3] L. P. Binamungu, S. M. Embury, and N. Konstantinou, "Characterising the quality of behaviour driven development specifications," in *Agile Processes in Software Engineering and Extreme Programming*, ser. Lecture Notes in Business Information Processing, V. Stray, R. Hoda, M. Paasivaara, and P. Kruchten, Eds., vol. 383. Cham: Springer International Publishing, 2020, p. 87–102. [Online]. Available: http://link.springer.com/10.1007/978-3-030-49392-9_6

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[5] NVIDIA, "Large language models explained," last accessed 17-01-2024. [Online]. Available: https://www.nvidia.com/en-us/glossary/large-language-models/

[6] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, W. L. Tam, Z. Ma, Y. Xue, J. Zhai, W. Chen, P. Zhang, Y. Dong, and J. Tang, "Glm-130b: An open bilingual pre-trained model," in *Proc. of the ICLR*. Kigali Rwanda: OpenReview.net, Oct 2023. [Online]. Available: https://openreview.net/pdf?id=-Aw0rrrPUF

[7] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020, event-place: Vancouver, BC, Canada.

[8] I. Ozkaya, "Application of large language models to software engineering tasks: Opportunities, risks, and implications," *IEEE Software*, vol. 40, no. 3, p. 4–8, May 2023.

[9] D. Xie, B. Yoo, N. Jiang, M. Kim, L. Tan, X. Zhang, and J. S. Lee, "Impact of large language models on generating software specifications," no. arXiv:2306.03324, Oct 2023, arXiv:2306.03324 [cs]. [Online]. Available: http://arxiv.org/abs/2306.03324

[10] B. Chen, K. Chen, S. Hassani, Y. Yang, D. Amyot, L. Lessard, G. Mussbacher, M. Sabetzadeh, and D. Varró, "On the use of gpt-4 for creating goal models: An exploratory study," in *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*. Hannover, Germany: IEEE, Sep 2023, p. 262–271. [Online]. Available: https://ieeexplore.ieee.org/document/10260905/

[11] A. Fantechi, S. Gnesi, L. Passaro, and L. Semini, "Inconsistency detection in natural language requirements using chatgpt: A preliminary evaluation," in *2023 IEEE 31st International Requirements Engineering Conference (RE)*, Hannover, 2023.

[12] K. Ruan, X. Chen, and Z. Jin, "Requirements modeling aided by chatgpt: An experience in embedded systems," in *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*. Hannover, Germany: IEEE, Sep 2023, p. 170–177. [Online]. Available: https://ieeexplore.ieee.org/document/10260857/

[13] X. Zhang, L. Liu, Y. Wang, X. Liu, H. Wang, A. Ren, and C. Arora, "Personagen: A tool for generating personas from user feedback," in *2023 IEEE 31st International Requirements Engineering Conference (RE)*. Hannover, Germany: IEEE, Sep 2023, p. 353–354. [Online]. Available: https://ieeexplore.ieee.org/document/10260883/

[14] J. F. Smart, *BDD in Action: Behavior-Driven Development for the whole software lifecycle*. Shelter Island, NY: Manning, 2014.

[15] M. Wynne and A. Hellesøy, *The cucumber book: behaviour-driven development for testers and developers*, ser. The pragmatic programmers. Dallas, Texas: Pragmatic Bookshelf, 2012.

[16] R. A. Fokam Piam, "Erhebung und validierung von testbaren anforderungen durch visionvideos," Mar 2023, Master Thesis. [Online]. Available: https://www.pi.uni-hannover.de/fileadmin/pi/se/Stud-Arbeiten/2023/MA_FOKAM_PIAM.pdf

[17] V. Basili and H. Rombach, "The tame project: towards improvement-oriented software environments," *IEEE Transactions on Software Engineering*, vol. 14, no. 6, p. 758–773, Jun 1988. [Online]. Available: http://ieeexplore.ieee.org/document/6156/

[18] J. Bortz, *Verteilungsfreie Methoden in der Biostatistik*, 3rd ed. Heidelberg: Springer Medizin Verlag, 2010.

[19] J. Bortz and C. Schuster, *Statistik für Human- und Sozialwissenschaftler*, ser. Springer-Lehrbuch. Springer, 2010. [Online]. Available: https://doi.org/10.1007/978-3-642-12770-0

[20] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012. [Online]. Available: http://link.springer.com/10.1007/978-3-642-29044-2

[21] L. Chemnitz, D. Reichenbach, H. Aldebes, M. Naveed, K. Narasimhan, and M. Mezini, "Towards code generation from bdd test case specifications: A vision," in *2023 IEEE/ACM 2nd International Conference on AI Engineering – Software Engineering for AI (CAIN)*. Melbourne, Australia: IEEE, May 2023, p. 139–144. [Online]. Available: https://ieeexplore.ieee.org/document/10164755/