UNIVERSIDAD POLITÉCNICA
DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INFORMÁTICOS

MÁSTER UNIVERSITARIO EN INGENIERÍA DEL SOFTWARE –
EUROPEAN MASTER IN SOFTWARE ENGINEERING

Chatbot-based Interview Simulator: A Feasible Approach to Train Novice Requirements Engineers

Master Thesis

Muhammad Laiq

Madrid, July 2020

This thesis is submitted to the ETSI Informáticos at Universidad Politécnica de Madrid in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering.

Master Thesis
Master Universitario en Ingeniería del Software – European Master in Software Engineering
Thesis Title: Chatbot-based Interview Simulator: A Feasible Approach to Train Novice Requirements Engineers
Thesis no: EMSE-2020-7
July 2020

Author: Muhammad Laiq
Master of Science
Universidad Politécnica de Madrid

Supervisor:
Óscar Dieste
Associate Professor
ETSI Informáticos
Universidad Politécnica de Madrid

Co-supervisor:
Minna Isomursu
Associate Professor
Department of Information Processing Science
Univesity of Oulu

ETSI Informáticos
Universidad Politécnica de Madrid
Campus de Montegancedo, s/n
28660 Boadilla del Monte (Madrid)
Spain

Dedication

To my beloved parents, who instilled in me the virtues of perseverance and commitment, and relentlessly encouraged me to strive for excellence.

Acknowledgements

Intellectual Debt - I would like to thank my thesis advisor, *Prof. Óscar Dieste*, who have been a constant source of motivation and knowledge along the way, and this work would have not been possible without his support.

Abstract

Introduction: Although the interview is the most important and widely used requirements elicitation technique, novice engineers do not receive adequate training in Requirements Engineering (RE) courses.

Objectives: The goal of the research is to develop an AI-based interview simulator for helping novice requirements engineers in gaining interview skills.

Methods: The research is based on the Design Science Methodology for Information Systems. The simulator is the outcome of six cycles; in each cycle, a proof of concept with additional features is created. Each cycle finishes with evaluation and improvement suggestions.

Results: The simulator has been tested with students and results have been promising. The interview simulator understands context-free questions, retrieving the right information related to RE concepts such as Tasks, Goals, Users, Benefits, Stakeholders, Constraints, and Integration. Furthermore, the simulator also makes summaries of conversation, answers meta-questions, and questions based on the context.

Conclusions: The simulator has received good response from students, they were able to complete the tasks and feedback suggests that simulator will be surely helpful for novice engineers to improve interview skills. Moreover, the interview simulator will be tested in a real RE course in the academic year 2020-2021. Once it proves effective in the classroom, it will be opened to the RE community for free use and improvement.

Keywords: Requirements elicitation, chatbots, assistants, interviews, interview training.

List of appended papers

An early version of this thesis has been accepted to a workshop:

Laiq, Muhammad and Oscar Dieste (2020). **Chatbot-based Interview Simulator: A Feasible Approach to Train Novice Requirements Engineers**. In: Proceedings of the 10^{th} International Workshop on Requirements Engineering Education and Training (REET), accepted for publication (Laiq and Dieste 2020).

The manuscript is available in Annex D.

Contents

List of Figures

Chapter 1

Introduction

Requirements elicitation (REl) is a crucial phase of requirements engineering (Ian Sommerville and Sawyer 1997), which strongly influences the quality of final product (Wohlin et al. 2005). Communication skills are fundamental to elicitation; hence people with strong communication skills are more effective in the requirements elicitation (Anwar, Razali, and Ahmad 2011). Users and customers should have enough skills to express their needs to the requirements engineers, and obviously, requirements engineers should be specialists in acquiring the necessary domain-specific knowledge.

REl comprises diverse activities and multiple techniques to perform these activities. There exist many requirements elicitation techniques such as *task analysis*, *questionnaires*, *domain analysis*, *interviews*, *observation*, etc. Elicitation techniques are selected depending on system type, purpose, time, budget, etc.(Carrizo, Dieste, and Juristo 2014). However, interviews are the most widely used elicitation technique for requirements acquisition (Zowghi and Coulin 2005), and they are considered as one of the most effective elicitation techniques (Davis et al. 2006).

Interviews look easy to conduct because they are, in essence, a one-to-one conversation. However, an *effective* interview is not a free conversation. On the one hand, interviews require the realization of several tasks, e.g., scheduling, script preparation, and consolidation. On the other hand, their success significantly depends on the skill of the interviewer to know what are right questions and when to ask them, and from whom to ask (Zowghi and Coulin 2005). Interviews are a soft skill but it is not acquired for free.

It is difficult for REl students to develop interview skills during their courses. The problems are multi-fold: (1) students usually have limited domain knowledge, (2) interview practice is time-consuming, and (3) large groups prevent instructors to conduct even a single interview with each student. At least in our experience, interview training in academia restricts to theory or short-time exercises.

Similar problems have been reported in other sciences.(Fitzmaurice et al. 2007) highlight that young doctors have little chances for interviewing patients to acquire interview skills. (Yang and Evans 2019) discuss the issues of students, teachers, and staff; students and teachers face problems while using the newly introduced application and help desk staff struggle to handle an enormous number of queries. (Kowalski, Pavlovska, and Goldstein 2009) reveal that the security personnel struggle to acquire effective communication skills using the traditional learning method.

An approach that has been explored by some researchers to improve the engineers' interviewing skills is to develop interview simulators, likewise e.g., airline pilots are trained in flight simulators. However, we are not aware that any of these simulators are currently available. It is even possible that they did not proceed beyond the research stage, as the natural language processing technology has developed at a rather slow pace in the past years. The situation has changed with the rise of computing power and the expansion of AI-enabled technologies, e.g., digital assistants such as chatbots are commonplace (Maedche et al. 2019) for customer assistance in many web pages and traditional phone systems.

In this thesis, we aim to develop an AI-based interview simulator for helping novice requirements engineers to acquire interview skills. More specifically, our research goals are:

1. Create a chatbot that engages in a natural conversation with requirements engineers.
2. Emphasize good interview practice during the conversation, e.g., using context-free questions (Gause and Weinberg 1989; Wiegers and Beatty 2013), making summaries (Kato et al. 2001), etc.
3. Introduce natural language pitfalls, e.g., ambiguity, incompleteness, etc. in the engineer-chatbot conversation.
4. Elaborate on a design theory for chatbots applied to REl interviews.

This research applies the *Design Science* (Hevner and Chatterjee 2010) methodology, due to its requirement of developing a working software product. We have carried out six *Design Science* cycles and the results are promising. The interview simulator understands context-free questions, retrieving the right information related to RE concepts such as Tasks, Goals, Users, Benefits, Stakeholders, Constraints, and Integration. Furthermore, the simulator also makes summaries of conversation, answers meta-question, and questions based on the context, for instance when discussion is going on about tasks and the user asks *anything else*; simulator will identify and responds with tasks. Moreover, the interview simulator will be tested in a real RE course in the academic year 2020-2021. Once it proves effective in the classroom, it will be opened to the RE community for free use and improvement.

This thesis is organized as follows: In chapter 2, we provide the relevant background information. In chapter 3, we elaborate the research goals. In chapter 4, we introduce the *Design Science* methodology, that we use for the design and implementation of the assistant. In chapter 5, we explain the results of each iteration. In chapter 6, we discuss the achieved research goals. Finally, in chapter 7, we report the conclusions and future work.

Chapter 2

Background

In this chapter, we present the background knowledge necessary to understand the rest of this document. In section 2.1, we discuss the role of interviews in requirements engineering to understand their importance in the requirement elicitation process. In section 2.2, we talk about the interview training process in requirements engineering. In 2.3, we explain the different tools to support the interviews in the requirement engineering process (i.e. help novice requirements engineers). In 2.4, we talk about different tools to support the interviews in other sciences. In the remaining sections, we discuss the use of chatbots in education and Watson Technologies used for the development of the product, i.e., the educational value of Chatbots (section 2.5), and IBM Watson Technologies (section 2.6).

2.1 Interviews in Requirements Engineering

The process of requirements elicitation is an important phase of requirements engineering (Ian Sommerville and Sawyer 1997), and greatly impacts the quality of the final product (Wohlin et al. 2005). In order to produce correct software requirements specifications (SRS), one requires to have strong communication skills (Anwar, Razali, and Ahmad 2011). The process of getting requirements includes multiple activities and there exist multiple techniques to perform those activities such as *task analysis*, *questionnaires*, *domain analysis*, *interviews*, *observation*, etc. Any of the requirements elicitation technique can be selected depending on the type of system, purpose, budget, time, etc. (Davison, Martinsons, and Kock 2004). However, interviews are the most widely used elicitation technique for requirements acquisition (Zowghi and Coulin 2005), and they are considered as one of the most effective elicitation techniques (Davis et al. 2006).

Whatever software development methodology is used doesn't matter; the final product should be according to user requirements. Hence, if there are inaccuracies in the SRS produced after requirements analysis, the software developed using those requirements will include errors, and possibly the creation of the system that does not meet the customer's purpose. In consequence, the software requirements work would have to be revised, which will ultimately add a delay in the process and cause an increase in development cost (Yamanaka and Komiya 2011; Kamata and Tamai 2007; Dvir, Raz, and Shenhar 2003).

(Yamanaka and Komiya 2011) states that the interview technique is widely adopted among various requirements elicitation techniques for requirements eliciting process. One of the reasons this technique is widely adopted is easy to use and the technique proved to be efficient; as they were able to elicit requirements without exclusions or inaccuracies. Interviews play a crucial part in several domains in making vague information clear. For instance, in building expert systems the heuristics are usually accomplished through an interview of a knowledge engineer with a domain expert. In software specification, interviews are vital for the extraction of requirements from customers (Kawaguchi et al. 1987). Furthermore, (Wood 1997) recommend using interview in the normal working conditions to uncover any hidden or incomplete requirements. Interviewing and inspecting users in their work is a critical ingredient for the user-centered design of the system. It will be helpful for designers to obtain accurate information about the user's work that will be supported by the new application.

2.2 Interview training in Requirements Engineering

Interview training is based on the following three ideas that engineers shall acquire and practice:

1. Interviews have a structure **:** Like other elicitation techniques interviews also have a structure that should be followed for the successful execution of the elicitation session. (Wiegers and Beatty 2013) presents activities for the interview structure; activities include 1)decision on elicitation scope and agenda 2)preparation of resources 3)preparation of questions and straw man models 4)execution of process 5)organization and sharing of notes, and 7)documentation of open issues.

2. There are some good practices **:** (Wiegers and Beatty 2013) suggests good practices for conducting the interview.

    - At the start of an interview, it's good to have an introduction with participants; and a review of the agenda to recall objectives and clarification of any matters participants may have.
    - In order to keep participants focused on the main topic, it's necessary to keep discussion limited to objectives only.
    - It's good practice to draft questions or any guidelines ahead of an interview they will be helpful for users.
    - Innovation is key everywhere so during the interview analyst must ask and proposes ideas and choices to uncover as much as possible requirements about the problem domain and to make user express.

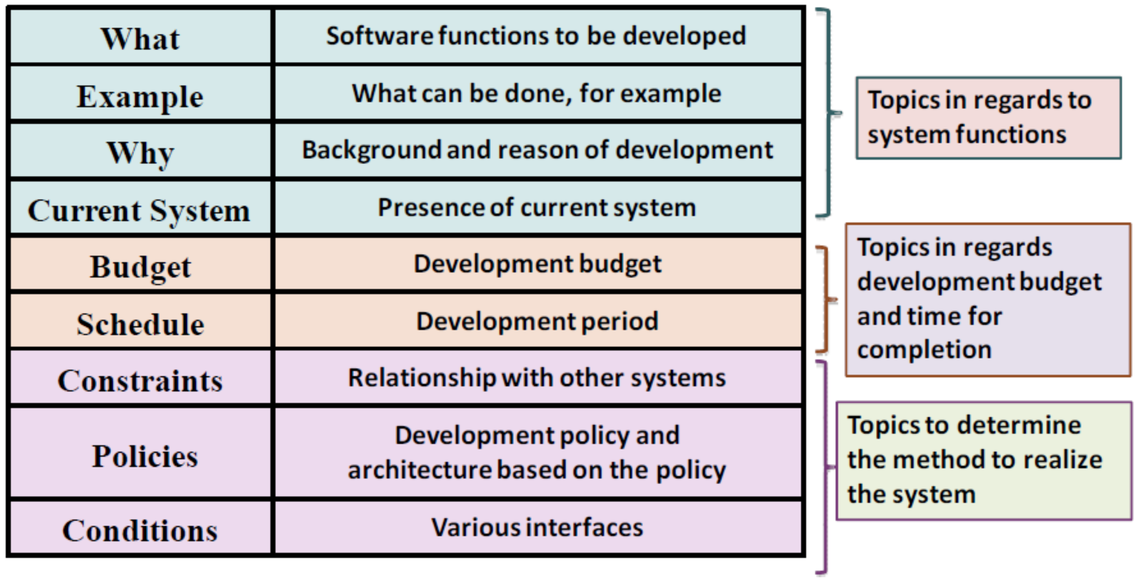


Figure 2.1: Topics for REl adopted by software engineers reproduced from (Yamanaka and Komiya 2011).

- Last but not least is to listen actively and carefully with the patience to know what people have to say and extend the conversation by providing feedback and inquiring if something is unclear.

3. There are different types of questions that could be prepared beforehand **:** (Yamanaka and Komiya 2011) categorized questions for requirements elicitation into nine topics as shown in figure 2.1; further, these categories have been sub-categorized into three classes that include questions related to 1) topics related to functionalities of the system, 2) topics related to time and budget and, 3) topics related to policies or constraints in making system.
The context-free question has been proposed by multiple authors including (Kotonya and I. Sommerville 1998), (Gause and Weinberg 1989) ,(Lundeberg, Goldkuhl, and Nilsson 1981) and (Bolton n.d.) can found at Annex A, Annex C.1, Annex C.2, and Annex C.3 respectively. (Gause and Weinberg 1989) states that these context-free questions that are valid for any product that doesn't matter it's aircraft, boat, home, a song for a one-minute TV advertisement for chewing gum, a lifetime light bulb, or a trek in Nepal. Context-free questions are independent of the design of the product. Those high-level questions can be used to obtain requirements of the any system.

Interviews are not easy to conduct it involves the realization of several tasks and their success is subject to the skills of interviewers to know when to ask what and whom to ask (Zowghi and Coulin 2005). The problems that novice engineers face have been discussed in the chapter 1 and section 3.1 of the chapter 3.

2.3 Computerized support for interviews in Requirements Engineering

Nowadays people encounter many difficulties when going for an interview in any field such as a young doctor interviewing patient, novice requirements engineer interviewing clients, or applicant having a job interview. In such situations training is highly recommended for performing well in the interview.

The problems associated to interview training suggest the need for some computerized support to the education/practice of novice requirements engineers. Computer-Aided Software Engineering (CASE) tools have been applied to software engineering in a variety of ways. Only a few tools have been specifically created for REl (Kato et al. 2001). In this section, we will discuss the systems proposed by different authors for supporting the process of requirement elicitation (i.e, helping novice requirements engineers).

2.3.1 Work by (Gilvaz and Prado Leite 1995)

(Gilvaz and Prado Leite 1995) present an interview assistant to help software engineers during the interview process. The assistant uses a conceptual model which is based on Wetherbe's indirect questioning approach (Wetherbe 1991) for interviewing and provides automated support to software engineers in eliciting necessary information for the development of information systems. The prototype applies generic heuristics for enhancing the quality of the knowledge base and information collected during the interview process is also added back to the knowledge base. Wetherbe states that typical blunder in getting information requirements is to inquire about the incorrect question: "*What information do you need from the new system?*". It's an explicit question but not useful for customers; because determining what information they need is not always helpful for clients. To lessen this problem, Wetherbe proposes a new approach towards interviewing, i.e., an approach that uses indirect questions.

The idea behind the interview assistant is to help software engineers during the interview process not replace. The main features of the assistant are:

- The support for software engineers during the interview process by providing questions.
- The assistant also provides support for analyzing answers and propose new questions.
- It provides support for adding notes.
- Reports can be generated from the knowledge base.
- It provides support for storing observations as well.
- It uses the dictionary for analysis.

Moreover, researchers claim that results have been encouraging so far; and their prototype is operational and has been tested in multiple case studies. According to them, the main result of the research is the automation of the assistant to help software engineers.

2.3.2 Work by (Yamanaka and Komiya 2011)

(Yamanaka and Komiya 2011) illustrates a method to navigate interview-driven software requirements elicitation. (Yamanaka and Komiya 2011) argues that the requirements elicitation process is critical to software development and novice software engineers don't have enough knowledge and don't know when to ask what. Consequently, they will create error-prone SRS that will cost badly to both parties. To help novice software engineers they have designed a transition based method that helps engineers to develop their skills. Moreover, the method provides support to ask relevant questions and keep users on track with the help two-tier model and progress management table. The navigation process for their interview training method is defined as:

1. At first, the system offers a question to the software engineer, who uses that question to inquire from the customer.
2. The customer answer to the software engineer's question.
3. The software engineer checks if the customer's answer is expected or near to expected to answers provided by the system in terms of sense. a)if answers match it will be selected b)if it's hard to decide that answer match with the system provided answers the questions will be repeated until a decision is taken.
4. Then the system will automatically prepare the next question relevant to the answer chosen previously.
5. The system shows the next question to the software engineer.
6. The process will be repeated from Repeat 1 to 5 until there are no more questions to ask.

To check the effectiveness of the proposed method an experiment was conducted on software engineering students with and without navigation method. The experiment focused on collecting requirements for a medical image information system through interviews with a customer. As a result, the effectiveness of the method was verified with concluding remarks; regardless of domain information or work experience elicitation process can be done effectively with quality using an interview-driven navigation method.

2.3.3 Work by (Kato et al. 2001)

(Kato et al. 2001) presents a model of navigating interview processes for eliciting requirements to help novice analysts. They argue that there is a need for methods to support the unsolved problem of interviews such as insufficient navigation techniques for novice engineers, no guidance for making SRS. To overcome such problems they decide to provide a method that supports novice analysts to conduct interviews like experts. The main characteristics of the model are:

- It is a mixture of a blackboard and a state transition model.
- The model stores expert analyst's data in the forms of thesaurus, state transition model, and template.
- It utilizes the stored information for the interview navigation process.
- The navigation process provides support to analysts to specify SRSs in IEEE 830-1993 format.
- The blackboard model keeps the information in IEEE 830 format.
- The state model is used to keep the state of blackboard such as what has been asked and what is remaining to be asked.
- After each event state gets updated.

They conducted an experiment and validated their model by comparing the SRSs created by both novice and expert analysts and results confirm that the navigation method will help both novice and expert analysts as well. However, the development of supporting system based on the method is yet to be done.

2.4 Computerized support for interviews in other disciplines

In this section, we discuss the use of different tools in different fields to help beginners in gaining interview skills. In short, the goal of researchers in later papers is to utilize chatbots technology for helping the community in acquiring interview skills.

2.4.1 Work by (Bollweg et al. 2019)

MOOCs face issues such as high dropout rates and the creation of interacting individual task handlers. (Bollweg et al. 2019) have developed a business-interview simulator type chatbot for helping online education providers to improve their service. The primary purpose of their research is to train students for interviewing companies as part of their course so that they can efficiently conduct interviews (face-to-face or online) with managers from real companies. The conversation is about Enterprise Resource Planning (ERP) of different companies such as Retailer, Manufacturer, Service Provider, with three different levels: *small*, *medium*, and *large*. Students can select the company of any level(small, large, or medium) and ask questions to the simulator.

The Google *Dialogflow* platform (Sehgal n.d.) has been used to develop a chatbot with the support of a webhook mechanism for communication. "*Dialogflow is a Google natural language understanding platform used to design and integrate a conversational user interface into mobile apps, web applications, devices, bots, interactive voice response systems, etc*".

The chatbot has been tested on 38 students enrolled in the course. The evaluation related to the students' acceptance for the chatbot is based on (Ritzmann, Hagemann, and Kluge 2014), and performance is evaluated on the basis of "Chatbot Performance Measure" (Kuligowska 2015). Students were asked to fill questionnaires right after using the chatbot. The results indicate that chatbot performed well technically to help students gaining interview skills. Researchers claim that most students responded that the chatbot was helpful in many ways such as learning, acted as a natural conversation, accurate answers, easy to understand, etc. They have concluded that chatbot logic was successful in interview training to students as part of their practical assignment but needs more work to be done as it's kind of an initial prototype.

2.4.2 Work by (Fitzmaurice et al. 2007)

A variety of hurdles such as patients' reluctance to be interviewed repeatedly by doctors cause future doctors to get fewer opportunities to learn more about their profession. Researchers at the Trinity College have conducted a survey to know about the learning situation of young doctors and found out there is much need for something interactive like an interview simulator to help students in getting interview skills prior to interacting with patients.

In order to overcome this issue, (Fitzmaurice et al. 2007) have developed an online interactive e-learning simulator called "*Virtual Interviews for Students Interacting Online for Psychiatry (VISIOn)*" that lets students select an interview path through a psychiatric disorder. The process is basically a kind of requirement elicitation; students ask questions to a virtual simulator/patient and it answers accordingly. Interview simulator also supports follow up questions depending on the chosen path, e.g., if a patient states that he/she has been feeling depressed than simulator will present questions to the interviewer (doctor) such as "*For how long have you been feeling depressed?*" and "*Tell me more about what it is like for you to be depressed*".

The researchers have also conducted a follow-up survey about the effects of the interview simulator, and they got very good responses. In the follow-up poll, 65% of students responded that they used more than once the simulator, and 45% of the students replied that they used the simulator for more than an hour. Moreover, 55% of the students said that they felt that they were actually interviewing a virtual patient, and all of the participants stated that they will more likely to use the internet because of simulator availability.

They have concluded that it's quite helpful for medical schools to use simulators for teaching students. They also stated that further work will be done with a focus on connecting the simulator with an expert using video links and with further relevant material.

2.4.3 Work by (Stanica et al. 2018)

The goal of researchers is to provide an innovative interview simulator to train software engineers for job interviews so that they perform well in real interviews. By using multiple advanced technologies such as virtual reality (VR), chatbots, and emotions recognition, (Stanica et al. 2018) have created two different VR environments to simulate interviews for two mega-companies (i.e Apple and Microsoft).

The simulator uses the concept of job interview performance evaluation criteria: technical knowledge (also known as "hard skills"), personality characteristics (also known as "soft skills") and *impression management*. The latter describes the capability of a person in making an impression on interviewers through his/her appearance. In order to perform well in the interview, a candidate must possess hard and soft skills.

Machine learning and computer vision techniques have been used collectively to accomplish certain tasks, such as facial detection, semantic analysis, or emotion recognition. One of the most famous, the Tone Analyzer of Watson IBM's AI platform, has been used for emotional analysis. Moreover, Pandorabots platform has been used for the development of chatbot service that uses API endpoints to connect with unity-based VR applications.[1].

However, the product is still under development, but soon it will be validated on real applicants.

2.5 The educational value of Chatbots

Chatbots have been widely used in different fields but their use in the field of education is very less despite their potential (Yang and Evans 2019). In the following sections, we discuss two examples of chatbots being used training in academia and industry.

2.5.1 Work by (Yang and Evans 2019)

The main focus of the research is to raise awareness about the potential use of Artificial intelligence (AI) based conversational chatbots in the field of education to the people interested in utilizing chatbots in the educational area. The authors have created three initial prototypes to support daily activities of their department:

1. **To assist the Master's course(simulation game):** The Simulation bot has

[1] *Pandorabots, Inc. "is an artificial intelligence company that operates a web service for developing and deploying chatbots*" (Pandorabots n.d.). The Pandorabots Platform is "one of the oldest and largest chatbot hosting services in the world."Customers can build "AI-driven virtual agents" for human-like text or speech chats with customers. (Pandorabots n.d.).

been developed for the delivery of a simulation game course in a taught Master's module. Chatbot serves the purpose of the virtual customer in dealing with different companies. Students act as salespeople of the current company and talk to chatbot smartly so that they can retain the client by successfully presenting selling points provided by the current company (i.e., to satisfy chatbot – a virtual customer).

2. **To provide assistance for anew proposed educational application**: The second chatbot serves as a virtual mentor to advise teachers and students about how to use the app and prepare them to solve problems related to the newly developed educational software – allowing teachers to add a digital reading list for their course-based students.

3. **To help in the processing of helpdesk questions**: Helpdesk bot addresses requests from students and staff and answers their frequently asked questions, responds to specific requests, and guides users in the successful submission of helpdesk requests to the appropriate teams.

Moreover, researchers also discuss challenges in developing the chatbot, and lack of available features provided by chatbot engines as well. In short, most prominent chatbot development platforms are not created with the idea to help educational enterprises such as privacy concerns when integrating them, user-friendly interface, automation of request filling, and submission to the relevant person. They have concluded that their chatbots are not ready yet, due to technical challenges, they will be sorted out soon in collaboration with industrial partners.

2.5.2 Work by (Kowalski, Pavlovska, and Goldstein 2009)

The paper discusses two case studies performed to help security personnel for internal security training (i.e Information Security related knowledge) using chatbots. The results of case studies imply that the characters of participants seem to be more positive while chatbots are adopted rather than current conventional e-learning security education provided by the company.

However, researchers clearly state that based on the two case studies, we can not conclude whether the training approach of using chatbots instead of traditional learning methods is false or useful, but there seems to be a positive indication that we can use them as a complement over computer-based online training; because of the interactive nature of chatbots rather sequential, i.e., traditional learning.

2.6 IBM Watson technologies

With the increasing use of technology in every aspect of society, there has been an exponential increase in data that can be utilized to benefit in many aspects such as health, e-Commerce, education, etc. Many companies, research institutes, and nonprofit organizations are in the race to make use of available data effectively to generate useful insights, and IBM is one of the leading companies actively working for years to provide tools to utilize widely available data for different purposes.

IBM has developed many products under the umbrella of Watson technology, that can be used for multiple purposes such as prediction, automating complex processes with less effort, customer service, decision making, health service, and business analytics, etc. Watson services have been expanded so widely, such as Watson Assistant, Watson Discovery, Watson Natural Language Understanding, Watson Knowledge Studio, IBM cloud Function, etc. Moreover, some of the issues mentioned in the background research have been addressed by IBM Watson such as support for emotional data analysis, redirection to external services (search skills), and use of AI-based models to improve the overall process.

Our application utilizes Watson Assistant, Watson Discovery, Watson Knowledge Studio, and IBM cloud Functions together in developing interactive interview simulator. Below we will describe these services briefly, i.e, in section 2.6.1 we discuss IBM Watson Assistant, section 2.6.2 describes IBM Watson Discovery, section 2.6.3 deals with IBM Watson Knowledge Studio and in section 2.6.4 we brief IBM cloud functions.

2.6.1 IBM Watson Assistant

Watson Assistant is IBM's artificial intelligence-based product that lets developers build, train, and deploy smart assistants for businesses. The assistants can be deployed standalone or as part of an application. Unlike traditional chatbots that frustrate humans when complexity increases, the Watson Assistant is rather smart and provide external search support (Miller n.d.).

2.6.1.1 Watson Assistant Architecture

Figure 2.2 shows the Watson Assistant architecture. This architecture is common for the implementation of any kind of chatbot developed using IBM Watson services. In this architecture:

- Users communicate with the assistant by one of the integrated systems:
  - It can be a virtual assistant deployed on a social media communication platform, e.g., Slack or Facebook Messenger.
  - Any custom developed application, e.g., a smartphone app or a robot with a voice interface.
- The assistant will get the user input and send it to a dialog skill.
- The input will further be processed by the dialog skill using intents and entities. Based on the results of processing the relevant dialog will be identified and executed.

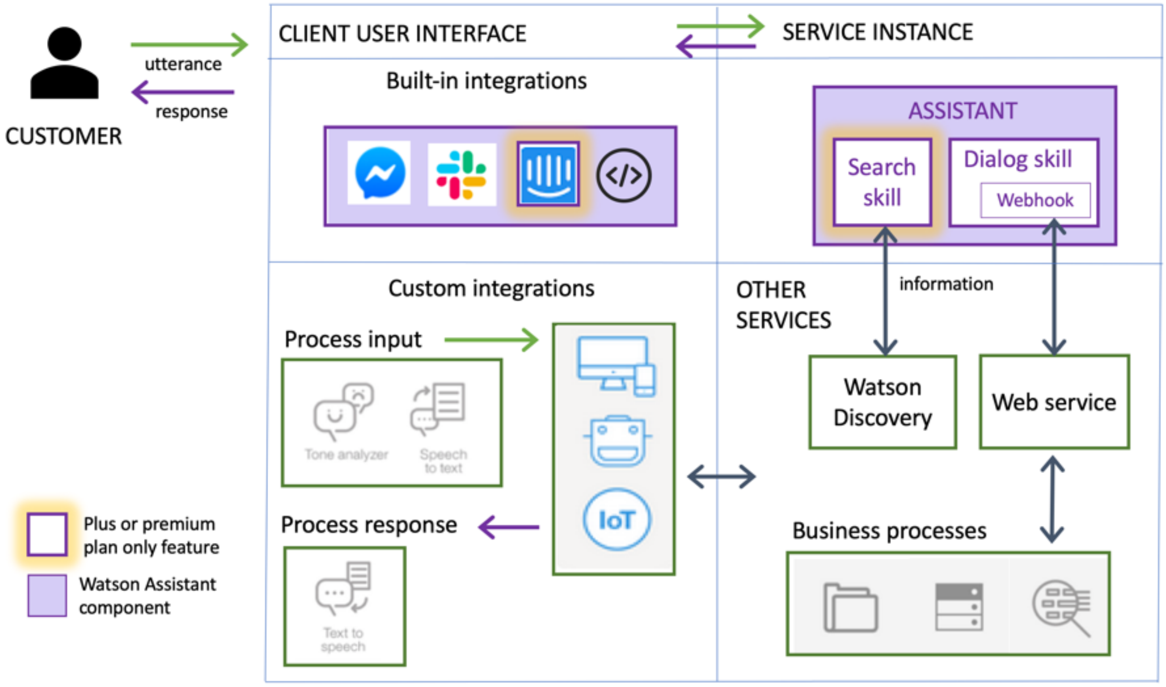


Figure 2.2: Watson Assistant architecture for a chatbot application (Miller n.d.).

- Depending on the system requirements, a search skill can be used to answer questions that are not possible for a dialog skill to answer. Other services like Watson Discovery can also be used.

2.6.1.2 Watson Assistant flow

The flow of typical Watson Assistant application employing Watson services shown in Figure 2.3; including the Watson Discovery service (that we describe in Section 2.6.2).

1. At first, we have to enter the relevant data to answer the user questions. IBM Watson does not require pre-processed data. In turn, Watson accepts a variety of unstructured formats, typically .docx or .pdf documents, that are processed inside the platform. Documents are annotated either using Watson Discovery Smart Document Understanding or Watson Knowledge Studio Machine Learning. AI models are trained on the annotated corpus and later used to discover the relevant information on unseen documents.

2. The user sends a query to the chatbot application using the provided interface.

3. Watson Assistant processes the query and chooses a dialog skill to interact with the user.

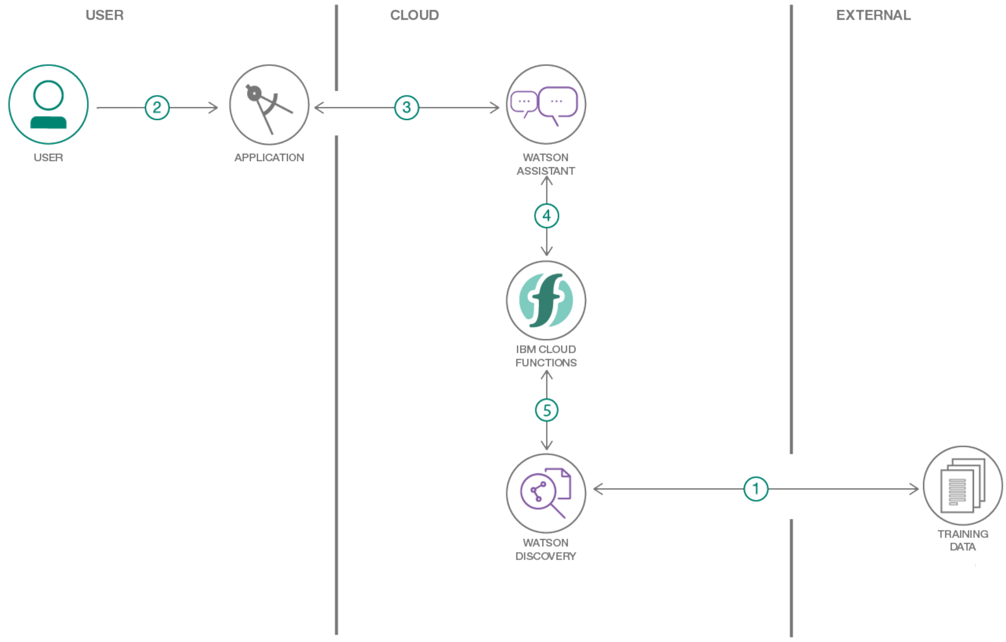


Figure 2.3: Watson Assistant flow
(Miller n.d.).

4. If necessary, the dialog can invoke IBM Cloud Functions using webhooks. Cloud functions enable the programmers to add logic to the assistant beyond the pre-packaged IBM's AI. For instance, a user of a transit system could inquiry about the departure time of a train; the webhook would invoke an external scheduling web service and return the required information.

5. Typical usage of the Cloud Functions is to query the Watson Discovery Service using parameters obtained by the Watson Assistant. For instance, a policy owner wants to find out if an appliance malfunction is covered in his/her policy. The webhook would send the details of the question, e.g., the type of appliance to the Discovery Service, which in turn would use the trained AI models to answer the question.

2.6.1.3 Watson Assistant components

Below we will discuss one by one all the elements of Watson assistant.

Intents: *Intents* are the users' intentions or purposes, i.e., what they aim when interacting with the chatbot. When Watson detects an intent, the appropriate dialog is invoked to interact with the user. Intents are declared and identified with the "#" symbol, For example: **#goals** is an intent that, in the present thesis, means that the interviewee wants to find out about the goals or objectives of the future software system. Watson needs examples of user's utterances to identify intents. For example "What are the goals of the system" is a user example for **#goals**. Figure 2.4 shows the different intents that have been created for developing an interview simulator.

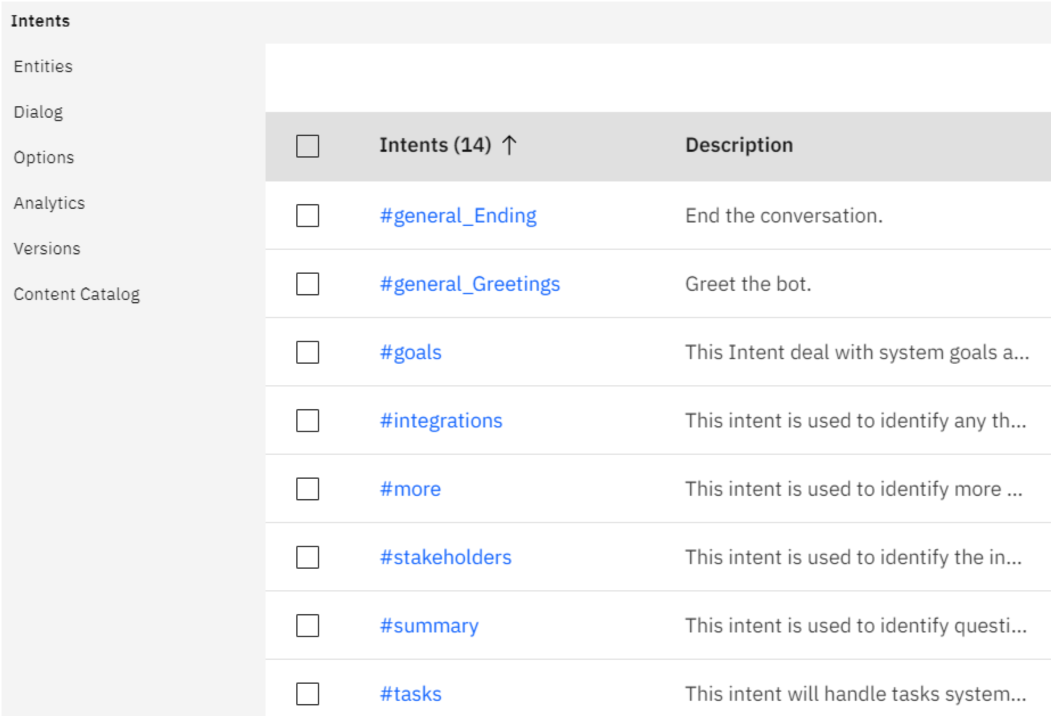


Figure 2.4: Intents.

Entities: Entities are like nouns that describe information in the user input that is appropriate to the user's intent. Entities are declared and identified with the ”@” symbol, For example, ”**@users**” entity used for identifying different types of users. For instance, in figure 2.5 we show the different entities that have been created to identify the intention of the user.

Dialogs: The dialog is used to interact with the user. It first gets the input then identify intent. Based on the identified intent and or context of application it provides a useful response. The dialogs are executed from top to bottom i.e. condition meet first will be executed. Figure 2.6 shows different dialogs of a system such as welcome, goal, tasks, etc.

The dialogs can use the webhook for communicating with discovery service using cloud services. Cloud function can be in any language such as node.js, python. In response to the request cloud function first authenticates all the parameters i.e Watson discovery and Watson assistant environment data, then performs the query and delivers the result to the caller.

2.6.2 IBM Watson Discovery

Watson Discovery is an AI-based search technology that seems to work far better than traditional search. It includes machine learning and natural language processing capabilities that generate meaningful results when applied to huge enterprise data (Bleiel n.d.).

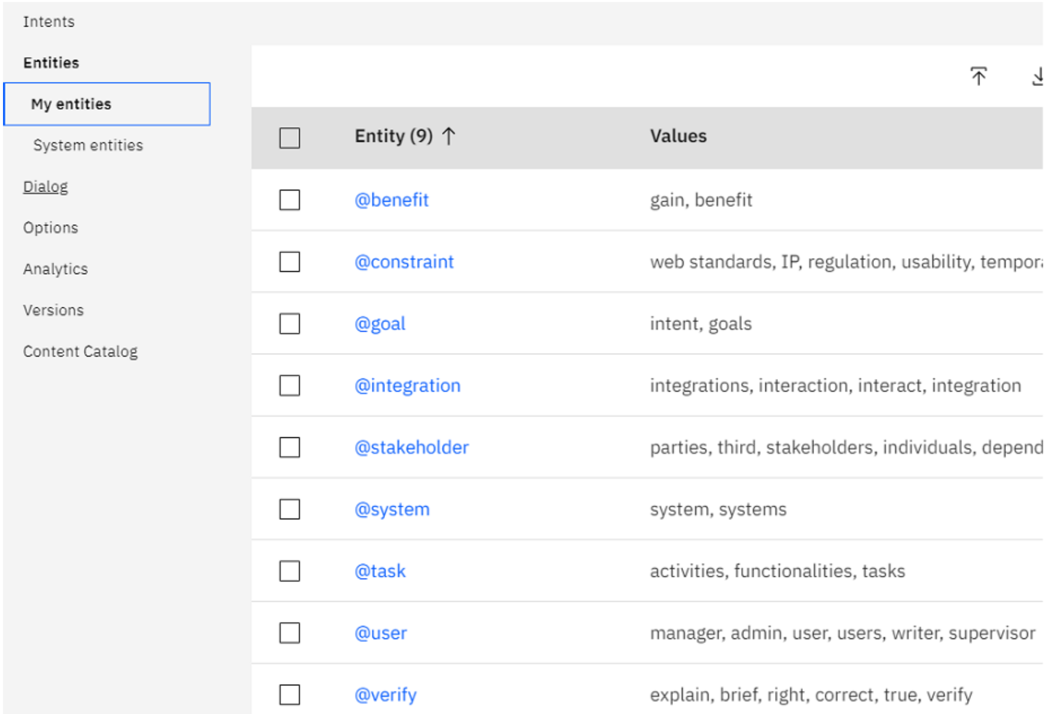


Figure 2.5: Entities.

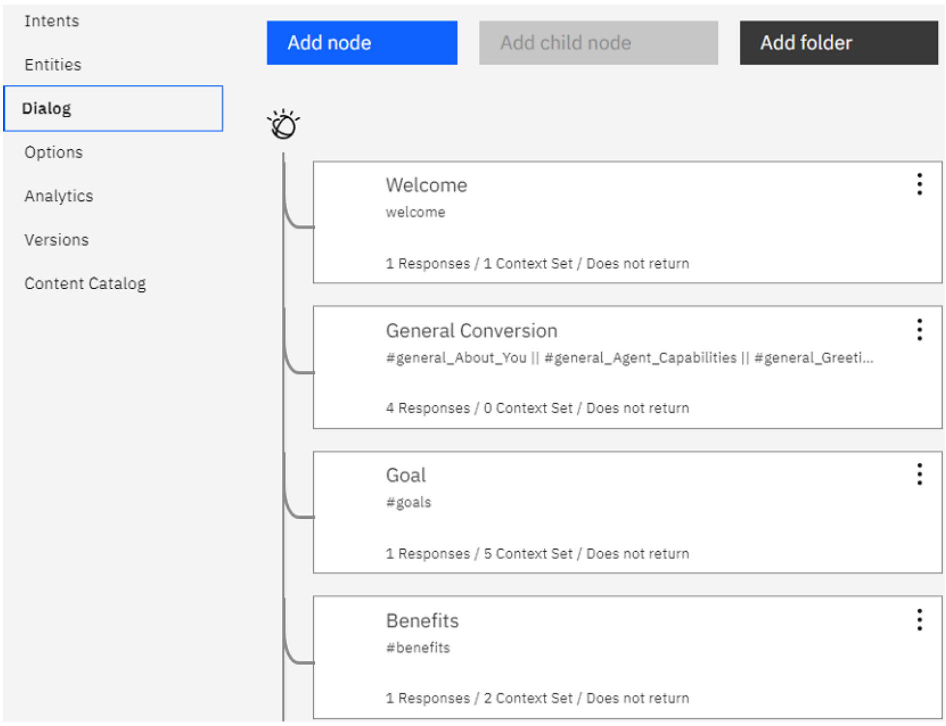


Figure 2.6: Dialogs.

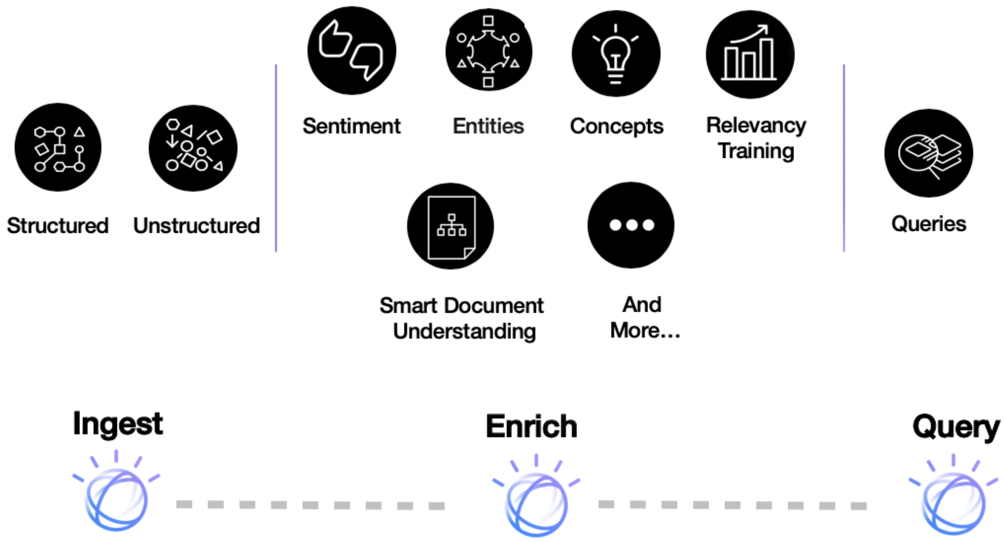


Figure 2.7: Watson Discovery
(Bleiel n.d.).

IBM Watson Discovery provides service to effortlessly create cognitive, cloud-based search services to uncover meaningful insights embedded in unorganized data, such as text documents or web pages.

As shown in figure 2.7, with Discovery, it doesn't take long in sketching the unstructured data ready to be queried for required information. Once the preparation step is completed we can utilize the service for providing relevant knowledge in new or existing applications. Below we highlight the IBM Watson Discovery features:

- It provides support to crawl, convert, enrich, and normalize data.
- It provides support to keep content as restrictive or licensed public.
- It also provides support for adding supplementary enrichment such as concepts, relations, and sentiment using natural language understanding. The definition, training, and deployment of concept/relation models are performed in Watson Knowledge Studio (see Section 2.6.3).
- It simplifies the development process by providing access through APIs.

A typical usage scenario with IBM Watson Discovery is shown in figure 2.8:

- The first step after the creation of an instance of discovery is to upload the data set. It can be private or public.
- Once data is available next step is to enrich data.

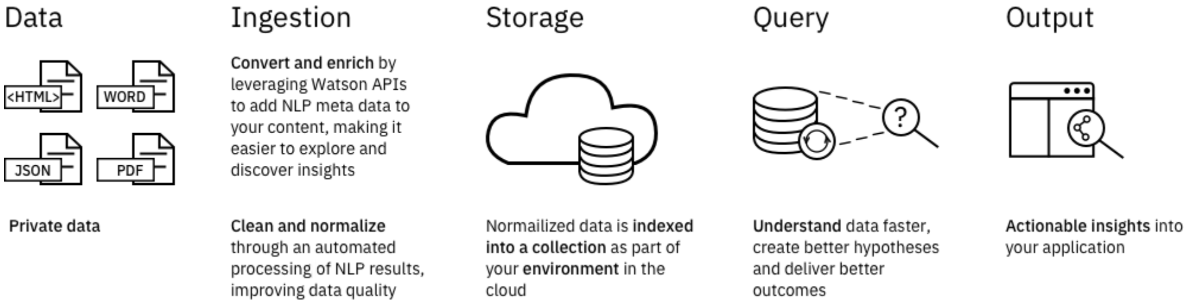


Figure 2.8: Standard development path for using Discovery service (Boyer n.d.).

- New fields can be created to annotate the document at this stage and documents can be divided into sub-documents using any field.
- During the enrichment process model can also be deployed such as the Machine learning model of knowledge studio.
- Once done with enrichment and configuration data will be indexed as part of the collection.
- After the pre-processing of data it's time to try different queries using discovery service to select the best possible query according to requirements.
- Selected query can be run on discovery service using API support.

2.6.3 IBM Watson Knowledge Studio

IBM Watson Knowledge Studio is a cloud-based application that lets developers and domain specialists work together to create custom annotators that could be used to recognize relevant information in unstructured text. It also provides support to deploy custom annotators on multiple services such as IBM Watson Natural Language Understanding, IBM Watson Discovery, and IBM Watson Explorer. (*Watson Knowledge Studio* n.d.).

Fig. 2.9 shows the architecture of the *machine learning model*, a training process used to identify entities, co-references, and relationships of interest in new documents. Below we describe steps for creating machine learning model using IBM Watson Knowledge Studio:

2.6.3.1 Create project

This is the first step in building a custom annotator model. The project can also be called work-space and includes a type system, source documents, and dictionaries.

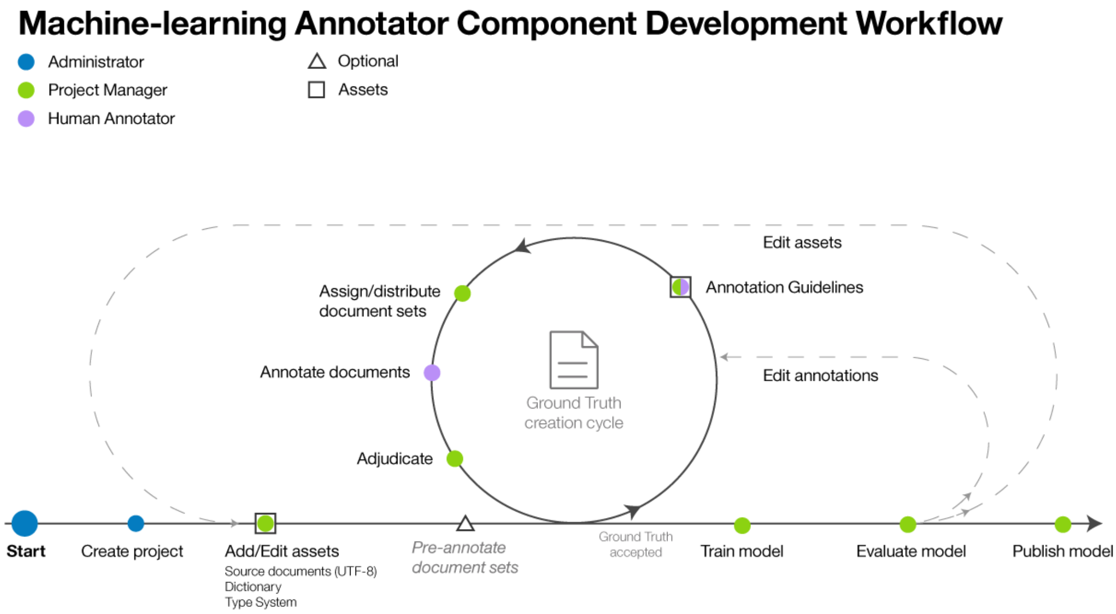


Figure 2.9: Machine learning model view
(Jessica Docarlo n.d.).

a. **Type system**

The type system is used to specify the entity types and relation types. Those types will be used by annotators when annotating documents.

An entity type is a classification of a real-world thing; for instance, the mention "IBM" can be annotated as "ORGANIZATION" entity type.

The relation type describes the relationship between entities; for example, the "Maria works for IBM" indicates the "employedBy" relationship between organization and person entity type.

Moreover, an annotator can use the co-occurrence feature to ensure consistency by identifying mentions that mean the same thing but their wording is different. For instance, the mention "IBM" can be interpreted as "International Business Machine".

b. **Source documents**

The custom annotators are usually trained in unstructured text. Text is provided as domain-specific documents that are uploaded to Knowledge Studio. These documents are divided into sets and assigned to specific annotators depending on their availability and expertise.

c. **Dictionaries**

The Watson Knowledge Studio provides this feature to speed up the annotation process. Before human annotation, we can use a dictionary to pre-annotate our documents. For instance, the human annotator can create a dictionary for seven days of the week and use it to annotate the mentions such as "Monday" and "Friday" for the entity "DAY-OF-WEEK". That same dictionary can be used to pre-annotate the un-seen documents.

2.6.3.2 Pre-annotate documents

This is an optional but important step for generating the machine learning model. The pre-annotation process can be done using multiple options such as IBM Watson Knowledge Studio dictionary, rule-base, or machine learning-based annotator component.

Each dictionary is mapped to a particular entity type; during the process of pre-annotation, mentions will be annotated accordingly using the dictionaries. On the other hand, the already available machine learning or rule-based model can be used to automatically annotate the documents.

After selecting the component, we have to run the component and it will find and annotate the types automatically. The pre-annotation process makes it easier for human annotators to perform the remaining annotation steps.

2.6.3.3 Annotate documents

The process of annotation must be performed by people who know the domain because it is critical for model performance. Annotating wrong mentions or defining incorrect relations will affect the results. During the annotation, the annotator finds mentions of interest and label them with the relevant entity/relationship/co-reference type. For instance, figure 5.22 shows different mentions; such as "writer" is annotated as a "User" entity type, and "recording" is annotated as a "Tasks" entity type.

2.6.3.4 Adjudicate and promote documents

After the completion of annotation, it is time to decide on accepting or rejecting the ground truth generated by humans during the process of annotation. Any conflicts or substantial differences can be resolved at this stage.

In case of conflicts or accuracy checks, senior human annotators can decide to either re-do annotation or move forward towards training the model.

Moreover, annotators work with overlapping sets; hence inter-rater reliability can be calculated and used as a quality measure.

2.6.3.5 Train the model

Once we create a machine-learning annotator component, we can select the previously annotated document sets that we want to use to train the annotator component and specify the percentage of documents that we want to use as training data, test data, and blind data. *Blind data is a set of documents that is kept apart and utilized for system testing repeatedly after multiple improvements and testing of the system to avoid the influence of test data on the model.*

2.6.3.6 Evaluate the Model

In order to assess the accuracy of the annotator component, we can check the accuracy analysis generated while making a model. The level of accuracy or results can be improved by repeating this step, previous steps will be also repeated once the model deployed, and desired results didn't achieve i.e. steps will repeat until the desired accuracy achieved.

2.6.3.7 Publish the model

This is the step where we can leverage a machine-learning model that we trained with Watson Knowledge Studio by making it available to other Watson applications. We can deploy or export a machine-learning annotator on other Watson services such as, in our case, Watson Discovery.

2.6.4 IBM Cloud Functions

IBM cloud functions are a Function-as-a-Service (FaaS) platform based on Apache OpenWhisk. They run the program without servers and scale services automatically. Users can define multiple web actions depending on requirements. Most of the popular languages are supported such as python, node.js, c-sharp, etc. Cloud functions act as an API; they receive requests (using webhooks) from the Watson assistant and then run a query according to the data sent by the assistant. The cloud functions return JSON results to the Watson assistant (*Cloud Functions* n.d.).

Webhook: A webhook is a procedure that can be utilized in calling out an external application depending on a certain event of the system. In dialog skills, webhooks are triggered during the node processing, provided that webhooks are enabled for that node.

The webhook is used to get defined data or date stored using context variables. Then it uses that data when sending the HTTP POST request to on the defined URL, that URL is known as a listener, that has subscribed for the event, in response to event listener is activated. The URL refers to some cloud code that performs certain actions such as querying discovery service according to received parameters and sending results of the query (*Web-hook* n.d.).For instance, we have enabled a webhook feature in the Watson Assistant through cloud code to communicate with the Watson Discovery Service.

Chapter 3

Problem Definition and Research Goals

3.1 Problem Definition

Interviews are not easy to conduct. They involve the realization of several tasks and their success is subject to the skills of interviewers, e.g., to know when, what, and whom to ask (Zowghi and Coulin 2005).

Novice engineers struggle to elicit correctly; (Gilvaz and Prado Leite 1995) highlight the issues of the requirements elicitation process. (Yamanaka and Komiya 2011) argues that the requirements elicitation process is critical to software development, and novice software engineers don't have enough knowledge and don't know when to ask what. (Kato et al. 2001) states that novice analysts struggle in the interview process due to insufficient navigation techniques and no guidance for making SRS.

Effective interviewing requires proper training, and it's difficult for requirements engineering students to gain interview skills. The problems for novice engineers are multi-fold: (1) students normally have insufficient domain knowledge, (2) interview practice is time-consuming, and (3) large groups prevent instructors to conduct even a single interview with each student. At least in our experience, interview training in academia restricts to theory or short-time exercise.

Similar problems have been reported in other sciences as well. (Fitzmaurice et al. 2007) highlights that young doctors have little chances for interviewing patients to acquire interview skills. (Kowalski, Pavlovska, and Goldstein 2009) reveal that the security personnel struggle to acquire effective communication skills using the traditional learning method.

3.2 Research Goals

Our objective is to develop an AI-based interview simulator to assist novice requirements engineers. More precisely, our research goals are:

1. Create a chatbot to develop novice requirements engineers' interviewing skills.

   The primary goal of our research is to create the initial proof of a concept (i.e., a chatbot) to train novice requirements engineers in obtaining interviewing skills. The chatbot will be developed using various IBM Watson technologies such as Watson Assistant, Watson Discovery, Watson Knowledge Studio, and IBM Cloud Functions.

2. Introduce natural language pitfalls, e.g., ambiguity, incompleteness in the analyst-chatbot conversation.

   The process of getting requirements (i.e., using an interview technique) is not straightforward. It involves a complicated conversation between interviewer and interviewee, and most of the time the information interchanged is either ambiguous or incomplete. Keeping in mind that concept, our second goal is to introduce pitfalls in the conversation in order to make a simulator act in a realistic manner.

3. Emphasize good interview practices during the conversation.

   One of our research goals is to provide advice to the novice requirements engineers in communicating with stakeholders to get requirements for the system.

   Our focus will be on using good interview practices, context-free questions, making summaries, and provide meta-questions answering support so that engineers can get essential skills necessary to conduct a good interview with real clients.

   Moreover, there are no standard questionnaires for conducting RE interviews, but at least there is an agreement on some good practices that are described in section 2.2 with context-free questions.

4. Elaborate a design theory for chatbots applied to Requirements Elicitation.

   Our last goal is to elaborate an initial design theory that may hep other researches in the process of designing and developing chatbots in the requirements elicitation domain.

Chapter 4

Research Method

In this chapter, we introduce the design science methodology, that we use for the design and implementation of the chatbot. In section 4.1, we explain the choice of methodology. In section 4.2, we describe seven guidelines that Hevner recommends when conducting design science research. In section 4.3, we present the Telos meta-model for Requirements Engineering concepts. Finally, in 4.4, we discuss the evaluation techniques.

4.1 Choice of Methodology

AI-enabled technologies, particularly chatbots, have been used for interview support and training. Some efforts (Kato et al. 2001; Gilvaz and Prado Leite 1995; Bollweg et al. 2019; Yamanaka and Komiya 2011) have been made in RE to provide support to requirements engineering during REl, but very little in what regards novice engineers' training and interviewing skills acquisition.

We aim to develop an interview simulator, in line with the works of (Bollweg et al. 2019; Fitzmaurice et al. 2007; Stanica et al. 2018). As this research's goal is to design a new artifact, we will apply the Design science methodology (Hevner and Chatterjee 2010).

Design science was created in the Information Systems (IS) discipline. IS involves people, organizations, and technology. Developing IS requires proper analysis of all connected entities. Hevner (Hevner and Chatterjee 2010) argues that this discipline is characterized by two separate paradigms: behavior science and design science. The first one seeks the creation or justification of theories to explain or predict human behavior. The second one is the engineering approach focused and aims to build inventions that solve people's and organization's problems effectively and efficiently.

While the routine design is the application of existing knowledge to organizational problems, Design science involves finding new solutions to previously unsolved problems or better and more efficient solutions to previously solved problems (Karmokar and Singh 2012). Our research matches one of these two situations, depending on each

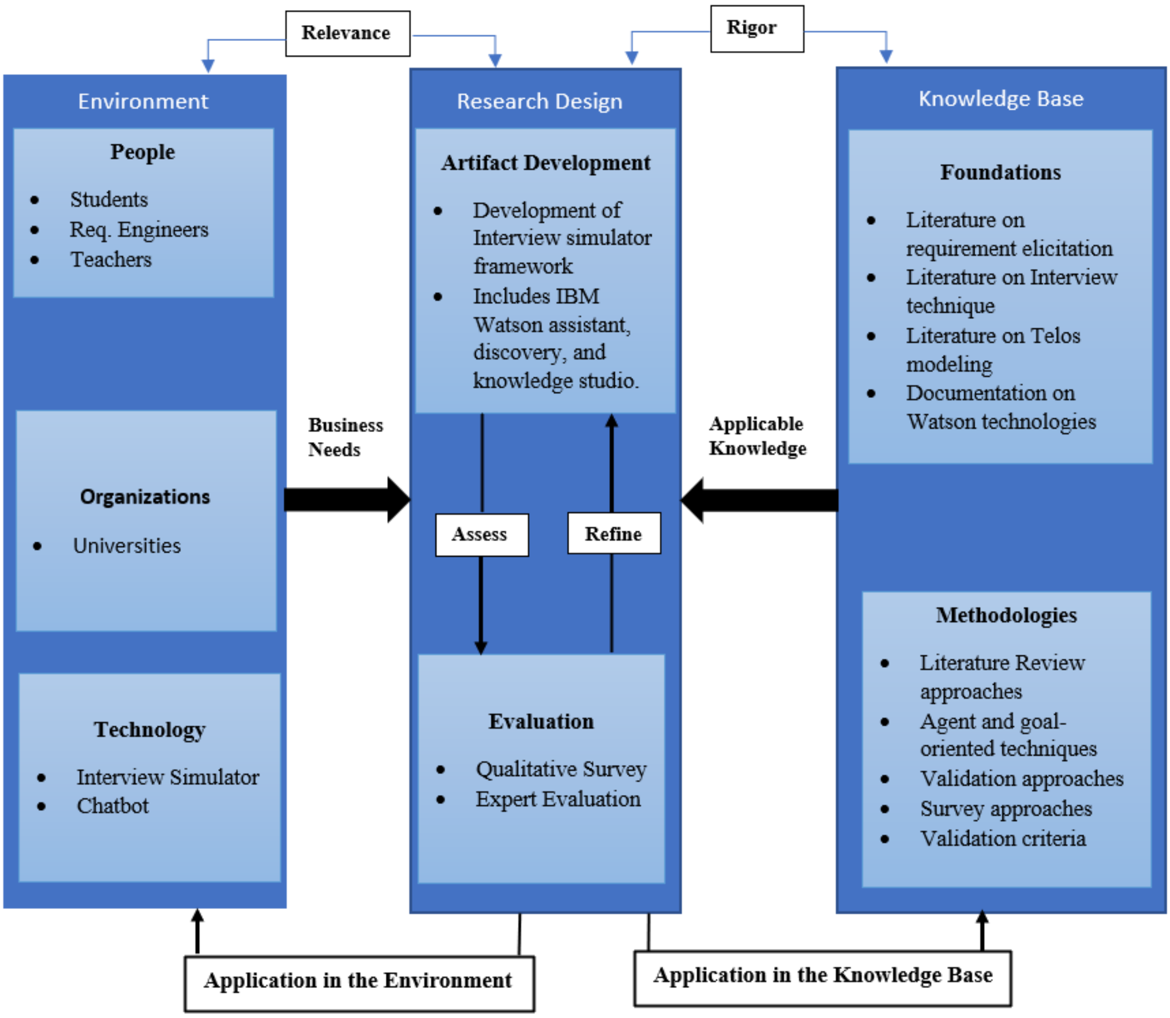


Figure 4.1: Research design science framework based on (Hevner and Chatterjee 2010).

one's interpretation of the current state of the art.

Fig. 4.1 shows the design science framework for this research. The interview simulator is developed using an iterative approach, creating a sequence of proof-of-concept artifacts. Artifacts are refined through the feedback received from different evaluation mechanisms. Once evaluated, the information flows back both to the environment (reporting about e.g., feasibility, scope limitations, etc.) and to the knowledge base (increasing our knowledge about the research area).

4.2 Design science Methodology

Hevner (Hevner and Chatterjee 2010) proposed seven guidelines for conducting design science research:

1. **Design as an artifact**

"The result from research in information systems is by definition a purposeful IT artifact created to address an important organizational problem. design science research must produce a viable artifact in the form of a construct, a model, a method or an instantiation".

The first guideline focuses on the development of an artifact using innovative technologies rather than people or organizations. Hence the result of the research should be useful to an organization; For example solution to a problem. The developed artifact can be model, method, prototype, or an instantiation.

In this system, **the artifact is an interview simulator** (prototype or "proof of concept"), that uses Watson technologies such as Watson Assistant, Watson Discovery, Watson Knowledge Studio, and IBM cloud functions. For details see the results chapter 5.

2. **Problem relevance**

   "The objective of design-science research is to develop technology based solutions to important and relevant business problem".

   The goal of the design science approach is to produce a design for solving a problem. According to (Hevner and Chatterjee 2010), the problem is described as "the difference between a goal state and the current state of the system".

   In this project, we aim to create an **interview simulator to help novice requirements engineers gaining interview skills in addressing the problems** described in section 3.1.

3. **Design evaluation**

   "The utility, quality, and efficiency of a design artifact must be rigorously demonstrated via well-executed evaluation methods".

   In order to confirm the performance of the developed artifact, the proper methods should be used. According to (Hevner and Chatterjee 2010), artifact evaluation must include the use of proper metrics, data collection, and analysis. Artifacts should be evaluated concerning functionality, completeness, coherence, correctness, performance, safety, usability, fit with the business, or additional relevant characteristics.

   The initial prototype of an artifact is evaluated using **quantitative and qualitative techniques: expert-opinion, surveys, and small-scale experiments**. The artifact is refined through the feedback received in the evaluation. For details see the evaluation sections in chapter 5.

4. **Research contribution**

"Effective design science research must provide clear and verifiable contributions in the areas of design artifact, design foundations, and/or design methodologies".

(Hevner and Chatterjee 2010) challenge the researchers to ask themselves: "What are the new and interesting contributions" of their research. The idea is that design science research develops something that contributes to the knowledge base. Contribution can be novelty, generality, or significance of the new IT artifact.

We contribute with the artifact itself (novelty) to the knowledge base in REl. **We also provide guidelines to those who are interested in using chatbot technology for the requirements engineering domain**; in this regard, see the discussion in chapter 6.

5. **Research rigor**

"Design science research relies upon the application of rigorous methods both in the construction and evaluation of the design artifact".

The design science guidelines highly emphasize the researchers to use proper and well-defined techniques for the design, development, and evaluation of artifacts. It's clear from the guideline that the way of conducting research should be proper, i.e., a systematic and scientific approach.

The development process of our artifact is iterative and incremental in nature. We work in cycles to make sure the product is error-free and according to the research goals. The theoretical models used for the artifact is based on work done by other people and proved effective, such as literature review, Telos model, context-free question and selection of Watson technologies for the development of chatbot (Gause and Weinberg 1989; Lundeberg, Goldkuhl, and Nilsson 1981; Bolton n.d.; Koubarakis et al. 2020; Maiden and Rugg 1996; Therkelsen-Terry n.d.).

6. **Design as a search process**

"The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment".

The guideline states that researchers should explore a variety of implemented systems/prototypes to discover more and more ways of generating the best possible artifact.

In order to fulfill the above guideline, we have used an iterative development methodology that allows the system to **grow smoothly cycle by cycle to generate the best possible outcome**.

7. **Communication of research**

"Design science research must be presented effectively both technology-oriented as well as management-oriented audiences".

The last but not least, this guideline recommends researchers to establish a broader level of communication to target more audiences. Therefore the central communication of this research will be this thesis. The thesis will be intended at an academic audience(RE professors and students), so this thesis has been created using academic standards. Moreover, the research paper has created applying research standards for research audiences.

4.3 RE Conceptual Model

For the implementation of the chatbot, we have to manage RE knowledge. Telos (Koubarakis et al. 2020) is a conceptual modeling language meant to represent software information, such as domain information, system requirements, architectures, design decisions, and more. In order to express diverse sorts of information, Telos was created to be extensible with meta-classes so that one can specify in the language the ideas used to model diverse topics. For instance, we can create a model for software that includes meta-classes for requirements, design, and implementation notions.

Individuals and characteristics are considered as first-class subjects in Telos models and are known as propositions. Propositions include an identifier, an origin, a label, and a target. They are formed along four dimensions i.e. instantiation/classification, specialization/generalization) and a couple of temporal dimensions for historical time and transaction. In Telos models, classes are known as instances of meta-classes and their hierarchy is endless i.e. one can make meta-meta-classes and so on (Koubarakis et al. 2020).

Fig. 4.2 shows a simple Telos model graphically. Individuals are expressed by nodes and attributes are indicated by dashed arrows. In Fig. 4.2, TelosPaper is an individual proposition and an instance of another proposition, the class Paper. Instantiation connections in Fig. 4.2 are indicated by pointers of regular density. Below we reproduce TelosPaper by following (Koubarakis et al. 2020):

```
TOKEN TelosPaper (at 10/05/2018...*)
IN Paper
WITH
    hasTitle
        title: "Telos is great!"
END
```

When the above statement is executed, the subsequent knowledge base propositions are produced:

```
P1 = (TelosPaper, instanceOf, Token, T1)
P2 = (TelosPaper, instanceOf, Paper, T1)
P3 = (TelosPaper, title, "Telos is great!", T1)
```

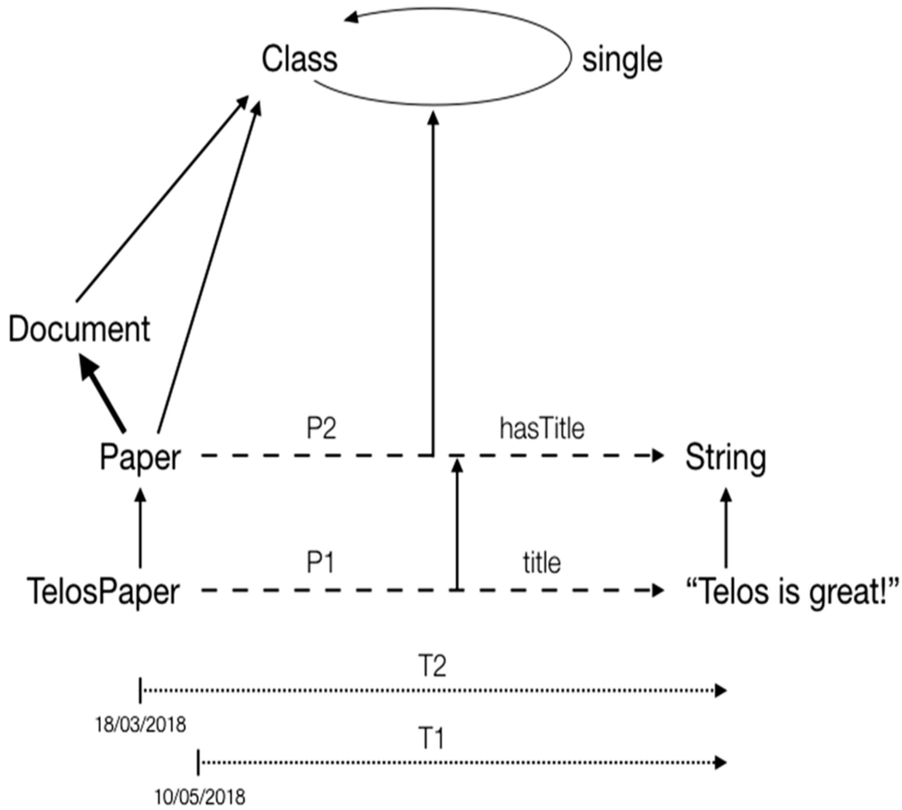


Figure 4.2: An graphical example of a Telos model
(Koubarakis et al. 2020).

```
T1 (at 10/05/2018...*)
```

Fig. 4.3 shows the Telos model that we have designed to represent/manage the RE knowledge that will be used by the chatbot. This model consists of three levels:

- **M0:** At level 0, we have the domain-specific concepts. For instance, in the Display Management domain, there are concepts such as writer, supervisor, preview, upload, download, etc. For more details check Appendix B.
- **M1:** At level 1, we have general RE concepts. For this project, we have defined the following classes: Task, User, Goal, Summary, Benefit, Stakeholder, Verify,and Constraint for our application.
- **M2:** At level 2, we have the Telos meta-classes *class* and *AttributeClass*, which are the root of the our model's hierarchy.

If necessary, the model could be extended with another level to describe individual instances of the problem domain, e.g., *Laiq* is a *Writer*. However, such level of details has been unnecessary for this research.

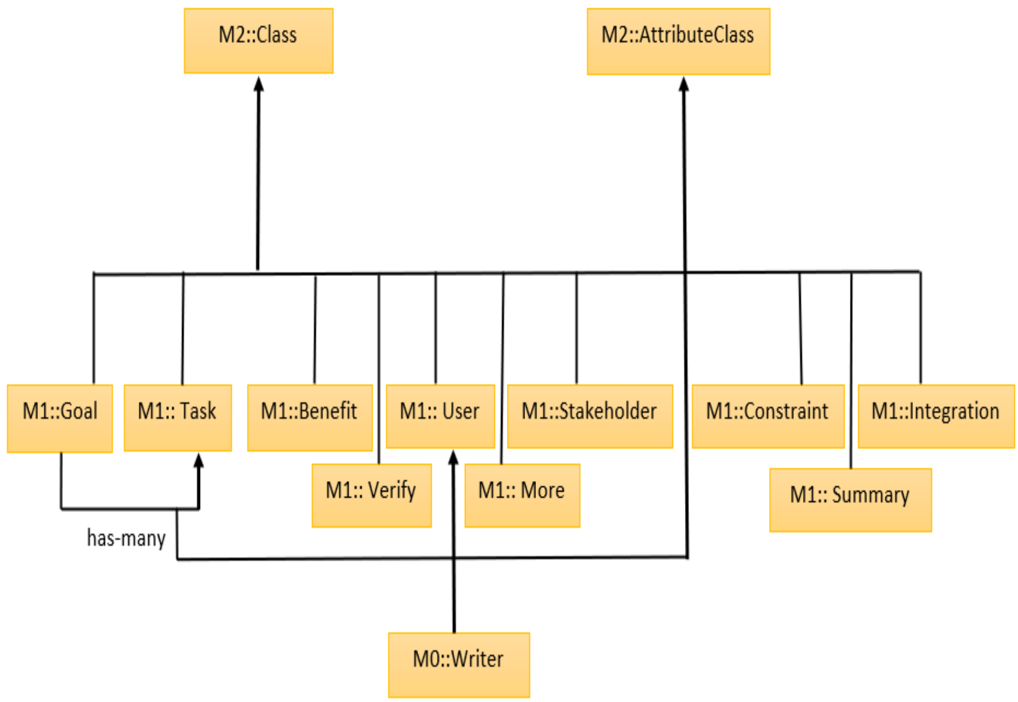


Figure 4.3: Telos Meta Model for Simulator
(Koubarakis et al. 2020).

4.4 Evaluation of the Proof of Concept

In this section, we explain the different evaluation techniques used for the evaluation of the artifact at different levels (i.e. cycles).

4.4.1 Expert Evaluation

During the evaluation of the first two cycles of the design science approach, we have used the expert evaluation technique also known as Qualitative effects analysis. This type of evaluation provides: ”an individual assessment of the quantitative effect of methods and tools, based on expert opinion” (Kitchenham and Linkman 2000).

4.4.2 Qualitative and Quantitative Surveys

In the third and fourth cycles, we have conducted qualitative and quantitative surveys with students. In a qualitative survey, feature-based evaluation is conducted with the help of people who have studied or used the tool (Kitchenham and Linkman 2000). Similarly, in a quantitative survey, quantitative results of methods/tools are checked using the survey (Kitchenham and Linkman 2000). In our case, we asked M.Sc students to use the tool and fill the survey forms to check the simulator functionality and usability. Surveys can found in evaluation sections i.e. section 5.3.3, section 5.4.3 and, section 5.6.3.

4.4.3 Small-scale experiment

(Kitchenham and Linkman 2000) states that an experiment is a method in which subjects are invited to do a task or a variety of tasks using the method/tool under test. For the fifth cycle, a more formal approach was used, i.e., an experiment. We selected second-year Master students to use the chatbot and create the SRSs. More detail about the small experiment can be checked in section 5.5.3. Moreover, SRSs created by students can found at Annex I.

Chapter 5

Results

The methodology selected for this research is *design science*. This methodology proceeds though several iterations, each one composed of:

- Requirements
- Design and implementation
- Evaluation

5.1 Iteration 1

5.1.1 Requirements

Before developing any software system, getting to know what to develop is very important. Many development projects have been failed or abandoned or delayed due to poor process of the requirements engineering. So our goal is to make sure we use a systematic process while developing the system to achieve research goals.

In the first iteration, we focus on developing a basic chatbot using Watson Assistant at first; then we try to use the Watson Discovery service as well in order to move towards the generalization process i.e providing answers to user queries based on the collection of data present in discovery service rather than hard-coded answers. Below is the summary of the requirements for the first iteration:

- Creating a basic Chatbot using Watson Assistant.
- Adding the functionality for answering queries related to Tasks, Goals, and Users of the system.
- After adding basic functionality of Watson Assistant, we use the Watson Discovery service to annotate the documents to improve query results.

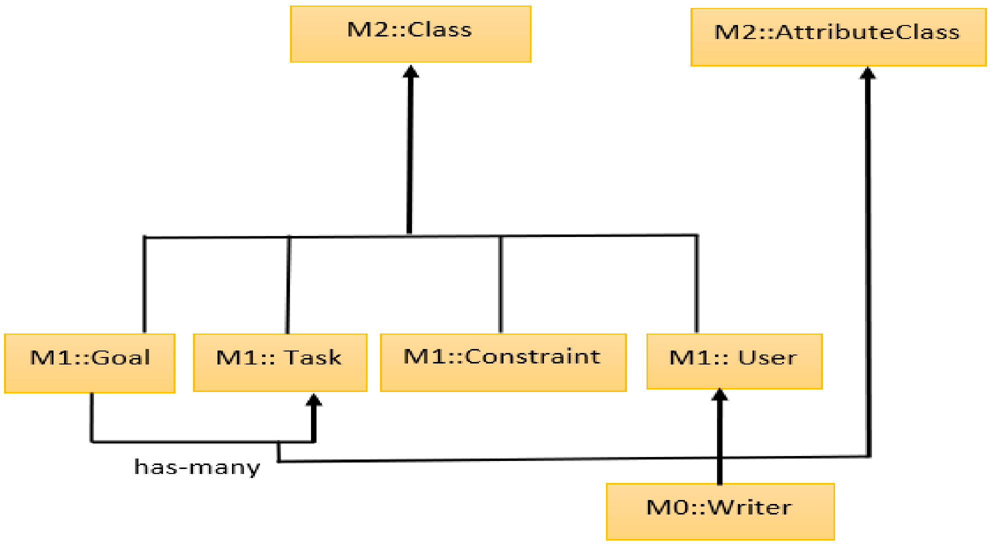


Figure 5.1: Telos Meta Model for first cycle.

- Use of IBM cloud functions and web-hook for communication between discovery and assistant.

To achieve the above requirements we have created a conceptual model using Telos guidelines (Koubarakis et al. 2020). Conceptual Model and Telos guidelines have been discussed in detail in Chapter 4. Fig. 5.1 shows the Telos model for first iteration. At the first iteration, we have Task, User, Constraint, and Goal as general RE concepts i.e. classes. These are identified intents that assistant will support after the first cycle.

5.1.2 Development

The process of development includes the development of basic chatbot using Watson Discovery, Watson Assistant, and IBM cloud functions. Fig. 5.18 shows the final system architecture. Below we describe in detail the system architecture at the first iteration and also show the demo of the prototype.

5.1.2.1 Architecture

System architecture dictates the flow of different services to achieve a common goal. As shown in figure 5.18, we have Watson Assistant, Watson Discovery, Watson Knowledge Studio,and IBM cloud functions. It's important to note that we do not use the knowledge studio in the first iteration. The user initiates the process and interacts with the system using the provided interface. Watson Assistant collects the inputs and identifies the intent first then forwards the request to Watson Discovery using IBM cloud function and webhook mechanism, cloud function a node.js based application to query the discovery using parameters received from assistant and send responses to assistant accordingly, below we describe the use of each service in detail.

5.1.2.2 User Examples

In order to identify intents, the assistant needs to have some initial knowledge. In Watson terminology, they are called user examples and from the requirements engineering point of view, they may be called questions. To define generic questions we have used the context-free questions proposed by multiple authors including (Kotonya and I. Sommerville 1998), (Gause and Weinberg 1989) ,(Lundeberg, Goldkuhl, and Nilsson 1981) and (Bolton n.d.) can found at Annex A, Annex C.1, Annex C.2, and Annex C.3 respectively. A generic protocol for questions is defined, for instance, the user may ask 1) What are the goals of the system? 2) What are the constraints of the system? (constraints type), and 3) Could you explain to me the tasks of the system? etc.

5.1.2.3 Watson Assistant

The Watson Assistant consists of intents, entities, and dialogues. Below we discuss each of them in detail.

- **Intent**: In the first iteration, the Watson Assistant consists of four generic intents:
  1. **#goals**: This intent is used to identify the goals of a particular system.
  2. **#tasks**: This intent is utilized to identify the tasks of a system. For instance, when a user asks for "what are the tasks of the system".
  3. **#identify_constraints**: This intent is used to identify particular constraints that must be followed while developing a system, i.e language, platform, etc.
  4. **#identify_users**: This intent is used to identify users of a particular system, i.e admin, writer, etc.
  5. **#identify_system**: This intent is used to identify the system, users can call it using multiple names such as an app, application, software, etc.
- **Entities**: Assistant consists of four generic entities:
  1. **system**: Used to identify system from user input i.e app, application, software etc.
  2. **goal**: Used to identify goal from user input i.e intent, objective etc.
  3. **user**: Used to find type user i.e admin, writer and supervisor.
  4. **constraint**: Used to identify specific constraint on system. i.e technology, language etc.

- **Dialogues**: Assistant consists of multiple dialogues, but two important and very generic dialogues are Goals and Tasks. Dialogues are evaluated from top to bottom i.e conditions first match will be evaluated first. Our application consist of the following dialogues:

  1. **Welcome**: Very basic dialog used to say welcome to the user.
  2. **Goal**: One of the generic dialogs used to respond to the user once **#goal** intent is identified, the text `''goal''` will be sent to IBM cloud function to run a query on Watson Discovery using text, a dialog will display received results from API.
  3. **Task**: Very similar to goal dialog used to respond user once **#task** intent is identified, the text "functional requirements" will be sent to IBM cloud function to run a query on Watson discovery using text and return back results.
  4. **Identify System**: As described in the intent section this dialog is used for identifying the system's name.
  5. **Identify Users**: A generic dialog used to get knowledge about the users of the system.
  6. **Identify Constraints**: A generic dialog used to get knowledge about the constraints of the system.
  7. **AnyThing Else**: If nothing matches this dialog will be called. i.e I can not understand.

5.1.2.4 IBM Cloud Functions

Our prototype uses node.js -based IBM cloud function, that acts as API to receive/send requests. Watson Assistant request the API and API responds after querying on Watson Discovery. Watson Discovery responds query in JSON format; we process JSON data using IBM cloud function and forward it to Watson Assistant. In later iteration we plan to:

- deploy the Machine learning model generated using Watson Knowledge Studio on Watson Discovery for improving our results.
- use cloud function in the later iteration to create incompleteness and ambiguity in the system.

5.1.2.5 Watson Discovery

As shown in Fig. 5.18 the architecture of our application, we use the Watson Discovery Smart Document Understanding to get better results from the document, for that we have first trained Watson Discovery i.e. how to interpret our document. Training in the first cycle includes the annotation of a document such as a title, headers, text, subtitle,

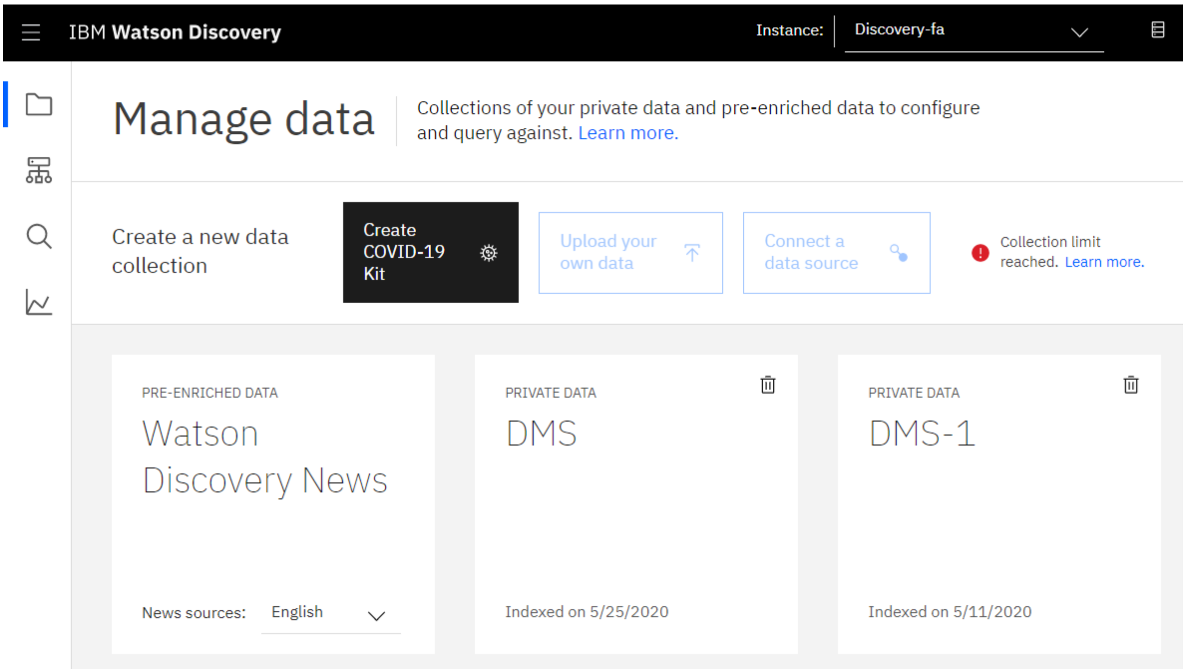


Figure 5.2: Discovery interface.

etc., after annotation we have used breaking document features of discovery that actually break documents into sub-parts, in our case we break text into paragraphs. After training Watson,we query accordingly to get results from specific sections i,e *text:"goal"* this will look for word `''goal''` and return passages containing word `''goal''`. Figure 5.2 shows the interface of the Watson Discovery interface with data collections.

5.1.2.6 Demo

Figure. 5.3 shows the demo of the chatbot, a conversation between user and simulator about requirements of the system, the user asks the chatbot to tell me about the goal, tasks or constraints of systems and chatbot identify the intent and forward request to Watson discovery using cloud functions and, IBM cloud functions runs a query on Watson Discovery and return results. Watson Assistant responds according to results received from the discovery.

5.1.3 Evaluation

5.1.3.1 Evaluation strategy

The evaluation approach selected for the first *Design Science* cycle is "Expert Evaluation". This type of evaluation aims to provide the subjective evaluation of the quantitative impact of methods and tools, based on expert opinion (Kitchenham and Linkman 2000). In the academic peer-reviewing world expert assessment is considered as the predominant approach i.e reviewing papers and dissertation evaluations. Expert as-

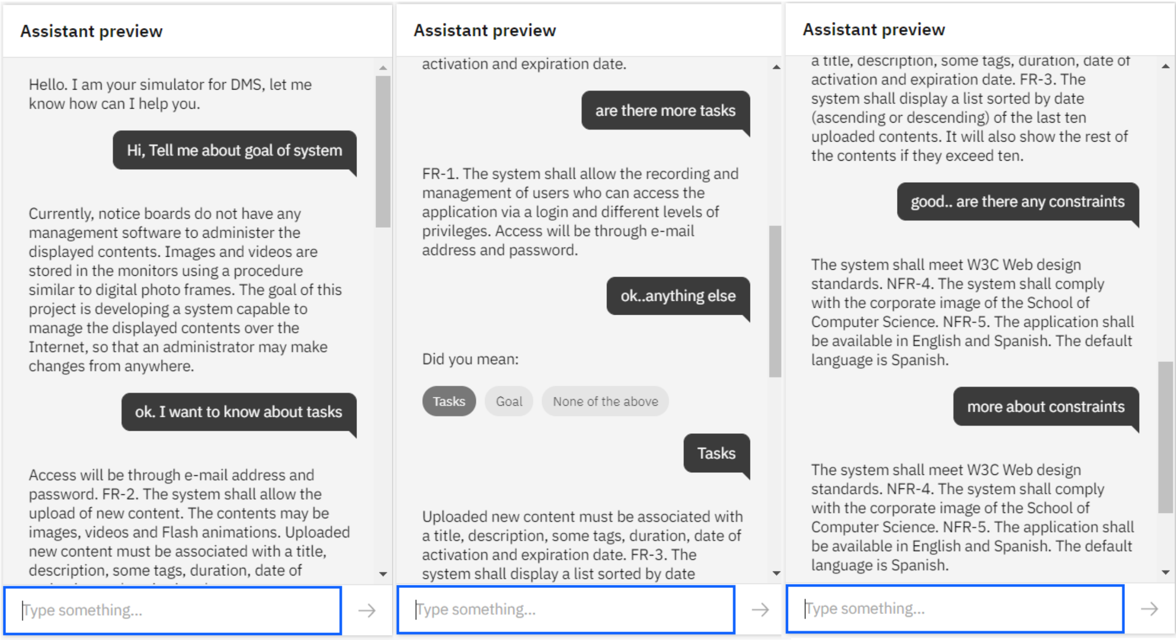


Figure 5.3: ChatBot Demo Cycle-1.

sessment is beyond the bibliometric data provided a good panel of experts are being selected and given enough time. One can expect that a good expert can give very helpful opinions and guidance on how to enhance research. (Iivari 2008).

Common sense suggests that the effort spent in the expert evaluation during the early stages of the project shall be limited. Therefore, the only expert involved has been the M.Sc. Thesis UPM supervisor, O. Dieste. The evaluation procedure has consisted of:

- **Selection of an interview script**: This script is based on context-free questions, described in the Appendix A. These questions, according to (Gause and Weinberg 1989; Wiegers and Beatty 2013), elicit information about global characteristics of both the business problem and the potential solution. More specifically, the questions are directed to identify the following requirement-related concepts:
  - Goals (question 1).
  - Tasks (questions 2 and 5).
  - Users (questions 6 and 7).
  - Constraints (question 11).

  which are the ones targeted by the first research cycle. Questions 3, 4, 8, 9, and 10 refer to other requirement-related concepts, and thus should not generate useful results during the evaluation.
- **Execution of the interview script**: This step is as simple as entering the questions into the Watson Assistant, and record the results.
- **Analyze the interview answers**: This is the most complex step. It implies to assess whether the answers provided by the assistant are (reasonably) correct.

To make the evaluation unbiased, we have identified the relevant concepts (goals, tasks, users, and constraints) in the example specification (the *Display Management System*). Such concepts, i.e., a *gold standard*, have been reported in Appendix B, along with the *Display Management System* SRS. The assessment using the gold standard is straightforward. The answers will be compared to the corresponding concepts and checking whether they match.

In addition to the concept matching, we have also used the feedback provided by the Watson Assistant to improve the identification of defects and the recommendations for improvement.

5.1.3.2 Evaluation results

Below we describe the evaluation of each interaction with assistant:

IT-1 The dialogue with the assistant started with a greeting. The response was a little discouraging.

IT-2 The response of the assistant is irrelevant when asked about general information such as your name, this implies that the assistant doesn't provide support for general info.

IT-3 The response of the assistant in interaction three is similar to interaction two, again this implies that the assistant doesn't provide support for general info.

IT-4 The mention to the "software system" is strongly related to the `#goals` intent. This happens because the word "system" is used very often in interviews to mean the "software to be". In the case of the `#goal` intent, there are several "user examples" containing the word "system"; this is the origin of the association "system" → `#goals`. This can create problems to identify other types of intents.

The assistant contains a `@system` entity. Watson identifies correctly this entity (actually twice, due to the usage of the words "software" and "system" in the entity definition). However, this entity is not used to define intents.

IT-5 The response of assistant is relevant and intent is correctly identified but an error has occurred while calling API, probably some problem with request syntax or API code.

IT-6 The assistant correctly identifies the `#task` intent.

IT-7 The assistant does not understand what "benefits" mean. It makes complete sense, as such intent does not exist.

IT-8 The assistant understands "characteristic" as "goal". It may be correct, but we will have to reflect whether the intermediate "feature" concept should be introduced between "goal" and "task".

IT-9 The assistant identifies that the question is about "tasks", but the word "system" is likely interfering, preventing Watson to provide the right information.

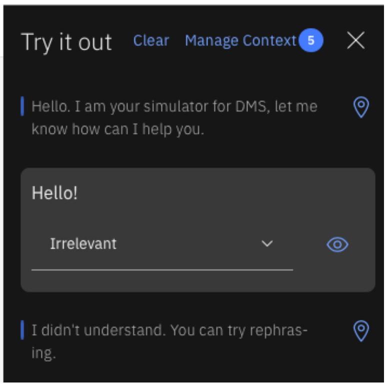


Figure 5.4: First interaction with the assistant

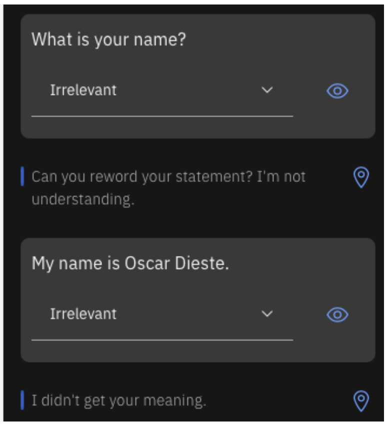


Figure 5.5: Second and third interactions with the assistant.

IT-10 Correct identification, but the information returned is not correct (it is about "tasks", not "users").

IT-11 This question is essentially equivalent to IT-10, with the same results.

IT-12 The assistant does not understand the idea of interacting with external systems. It makes sense since such intent does not exist. However, the word "system" is interfering again, suggesting to Watson that the question is about "goals".

IT-13 Same than IT-12.

IT-14 Correct identification.

IT-15 Correct identification, but the word "system" is creating noise again.

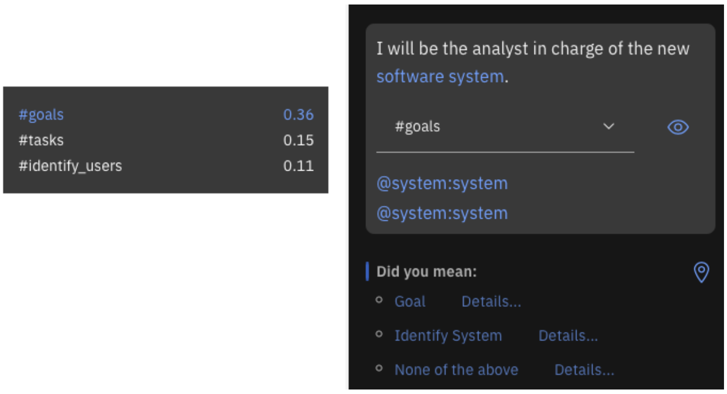


Figure 5.6: 4^{th} interaction with the assistant.

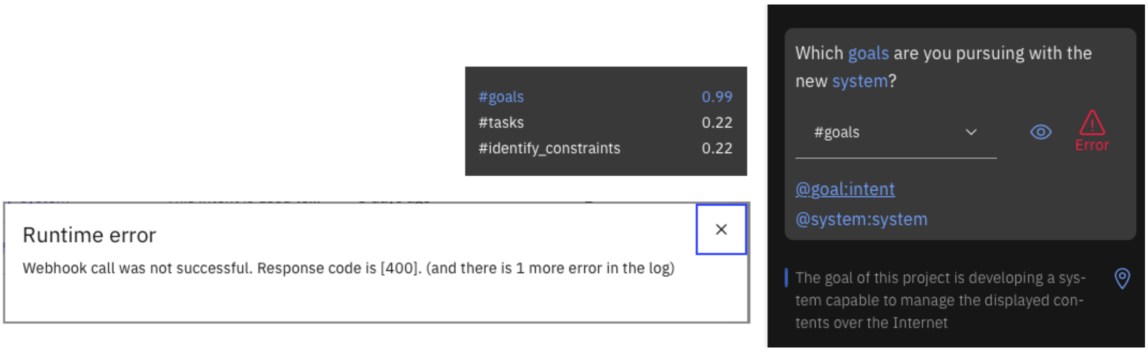


Figure 5.7: 5^{th} interaction with the assistant.

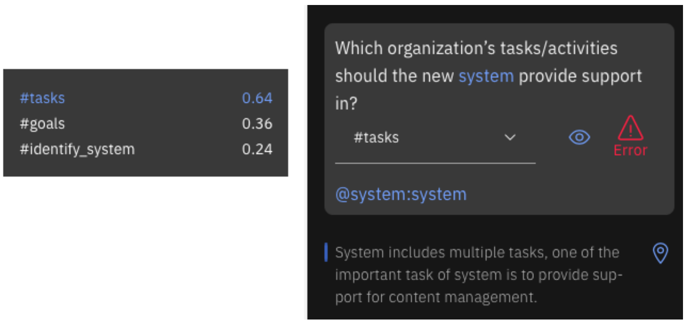


Figure 5.8: 6^{th} interaction with the assistant.

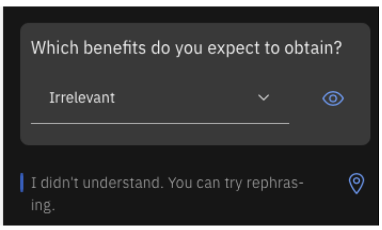


Figure 5.9: 7^{th} interaction with the assistant.

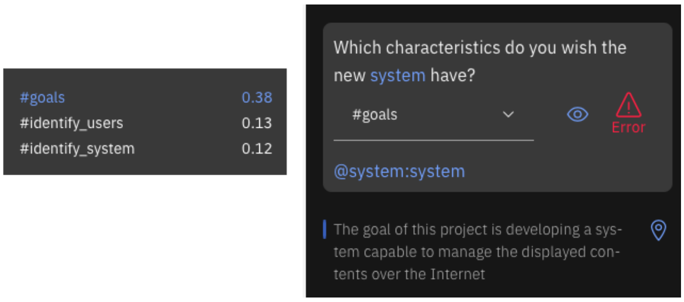


Figure 5.10: 8th interaction with the assistant.

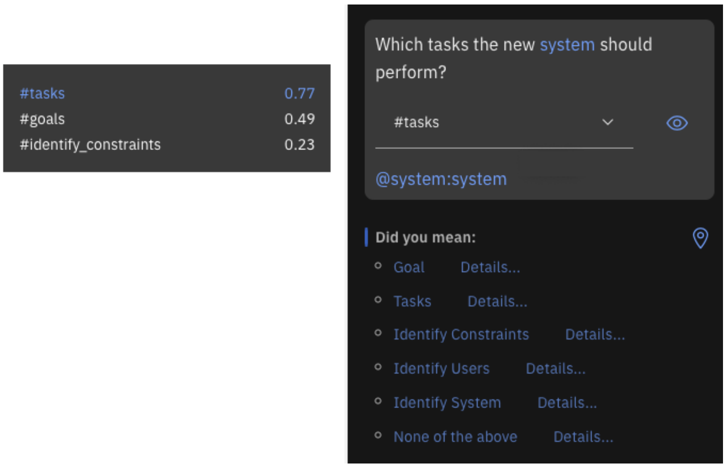


Figure 5.11: 9th interaction with the assistant.

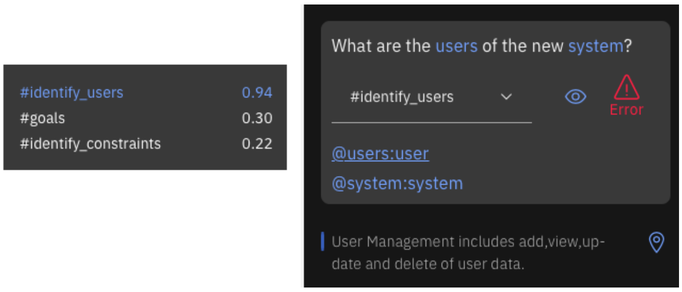


Figure 5.12: 10th interaction with the assistant.

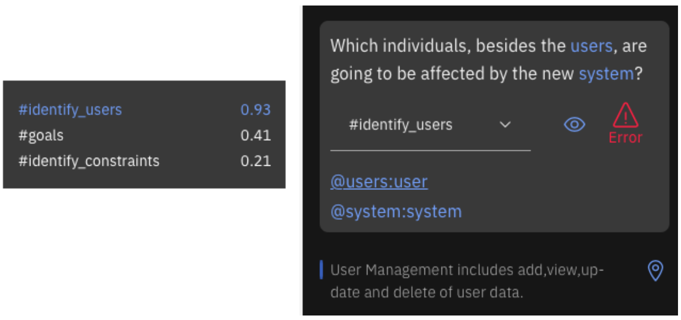


Figure 5.13: 11th interaction with the assistant.

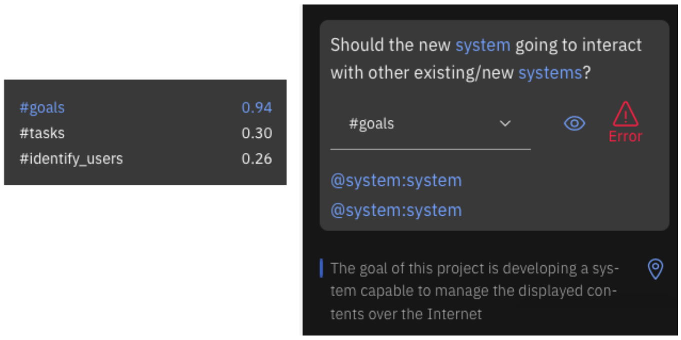


Figure 5.14: 12th interaction with the assistant.

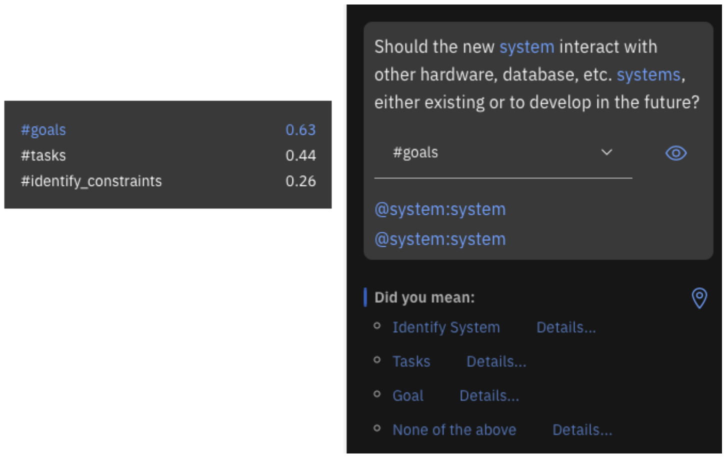


Figure 5.15: 13th interaction with the assistant.

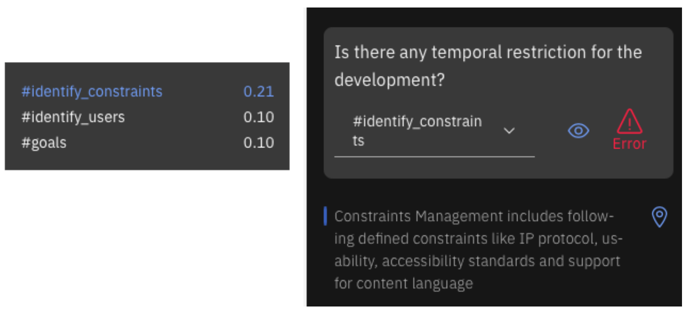


Figure 5.16: 14th interaction with the assistant.

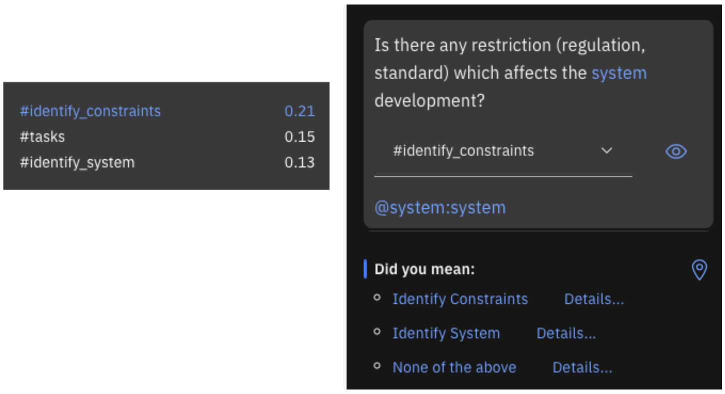


Figure 5.17: 15$^{\text{th}}$ interaction with the assistant.

5.1.3.3 Recommendations for improvement

1. Add some social abilities to the assistant. However, this is not a critical issue and could be deferred to later research stages.

2. The `#identify_system` intent has an unclear purpose. It has never identified by Watson, probably because there are few associated user examples. On the other hand, this intent is not clearly associated with any requirement concept. It seems that the `#identify_system` intent should be removed.

3. In turn, the `@system` entity looks appropriate. It shall be introduced in the user examples of the different intents. The inclusion of the `@system` entity may avoid intent recognition problems, such as interaction #12 and #13 (Figs. 5.14 and 5.15).

4. The same applies to the other entities (`@goal`, `@user`, and `@constraint`).

5. The definition of the `@constraint` intent looks too specific. It shall be made general.

6. By symmetry, it seems logical to introduce a new entity `@task`. In general, creating an entity for each requirements concept looks like an intelligent strategy, because it simplifies the definition of the intents (synonym management and simplification of the user examples).

7. The `#task` intent should be refined. In interaction #6 (Fig. 5.8), one correct task is returned. In interaction #9 (Fig. 5.11), none is returned.

8. Questions #2 and #5 could make reference to different concepts (organization processes and requirements, respectively). This difference shall be taken into account in the assistant's design.

9. The `#user` intent returns information about tasks, not users. It happens both in interactions #10 and #11 (Figs. 5.12 and 5.13). The reason may be a simplistic parameter passed to the Discovery service. The problems associated with the `#task` intent may have a similar origin.

5.2 Iteration 2

5.2.1 Requirements

This cycle aimed at making the assistant more robust; thus, we didn't introduce new features but improve the current ones. The main lines of action were the introduction of Watson Knowledge Studio and its machine learning model. The model was deployed on Watson Discovery. The Assistant and the Cloud functions were updated to accommodate the changes. Below we summarize our requirements for the second iteration.

- Introducing Watson Knowledge Studio to the system.
- Connecting Watson Knowledge Studio with Watson Discovery.
- Creating a basic Machine Learning Model and deploying it on Watson discovery.
- Updating overall system architecture to accommodate the suggestion of expert evaluation.

The conceptual model for the second iteration remains the same as we focus on improving the assistant.

5.2.2 Development

The process of development in the second iteration adds increments on first, which includes the use of Watson Knowledge Studio and the connection between Watson Discovery and Knowledge Studio. Fig. 5.18 shows the system architecture of application. Below we describe the system architecture and also present a demo of the prototype for the second iteration.

5.2.2.1 Architecture

System architecture dictates the flow of different services to achieve a common goal. As shown in the figure 5.18, we have Watson Assistant, Watson Discovery, Watson Knowledge Studio, and sample data sets. The user initiates the process and interacts with the system using the provided interface. Watson Assistant collects the inputs and identifies the intent first then forwards the request to Watson Discovery using IBM cloud function and webhook mechanism, cloud function a node.js based application to query the discovery using parameters received from assistant and send responses to assistant accordingly, below we describe the use of each service in detail. The overall architecture flow in the second iteration is summarized below:

1. We load document set into Watson Knowledge Studio for the annotation process.

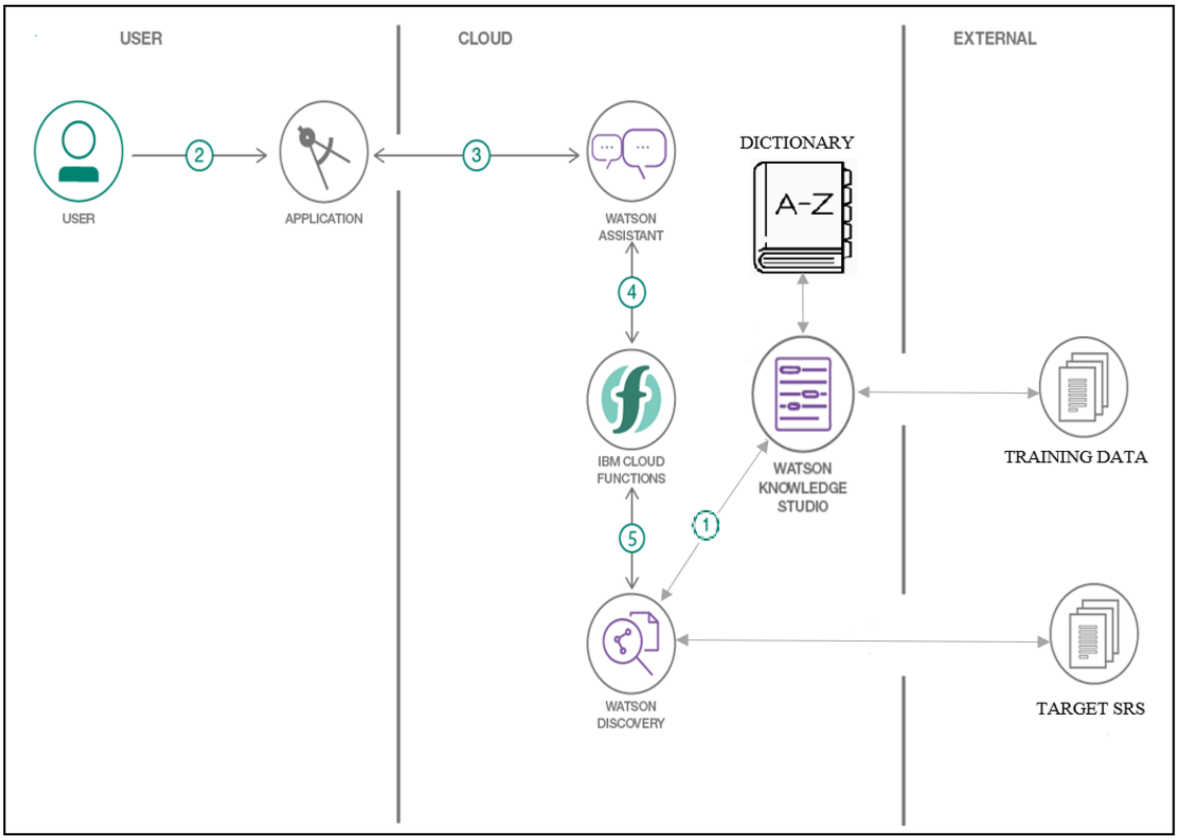


Figure 5.18: ChatBot Architecture
(following (Hagarty n.d.)).

2. We create a machine learning model using the knowledge studio.
3. The machine learning model is deployed on the Watson Discovery service instance for enrichment.
4. The display management SRS is added to the Discovery collection.
5. User sends dialog to the cloud-based application using the provided web interface.
6. The communication between the user and discovery service is ordered using a Watson Assistant dialog skill with the help of cloud functions.
7. Assistant communicates with IBM Cloud Functions action using the webhook.
8. The cloud function query the Watson Discovery Service using parameters of assistant and pass the results.
9. Replies from Watson Assistant are returned to the user via an application interface.

5.2.3 Watson Knowledge Studio

Fig. 2.9 shows the architecture of the Machine learning model, a training process used to identify entities, co-references, and relationships of interest in new documents and for creating machine learning model using IBM Watson Knowledge Studio. Below we describe in detail the important steps for creating a machine learning model using IBM Watson Knowledge Studio:

5.2.3.1 Entity Type

Our application consists of four entity types and one relationship(i.e Goal has Tasks), later we will use them with the help of dictionaries for the pre-annotation process with more data samples:

- **Goal**: Entity type used to identify and annotate goals of application(new system) from document (i.e requirements specification document).
- **Tasks**: Entity type used to identify and annotate tasks/activities of application(new system) from document (i.e requirements specification document).
- **User**: Entity type used to identify and annotate Users of application(new system) from document (i.e requirements specification document).
- **Constraints**: Entity type used to identify and annotate constraints on application(new system) from document (i.e requirements specification document).

5.2.3.2 Source documents

We have uploaded a sample set of documents that are representative of our domain content to the project. Documents are related to the display management systems; these documents will be divide into sets during the process of model generation after the annotation process.

5.2.3.3 Dictionaries

To reduce the workload of annotation we have created dictionaries relevant to the requirements engineering domain. It includes the generic terms for goals, tasks, users, and constraints so far. For example requirements specification might have tasks like manage, create, add, upload, download, view, remove, delete, etc. so during the pre-annotation process identified terms will be annotated in advance. As stated we will use them in a later iteration for the larger data set.

5.2.3.4 Annotate documents

The process of annotation can be performed by domain experts, background section contains the details about the annotation process i.e section 2.5.3. In our case, the authors themselves belong to the software engineering community and have enough expertise to do the annotation process itself.

5.2.3.5 Adjudicate and promote documents

After the completion of annotation, it is time to decide on accepting or rejecting the ground truth generated by humans during the process of annotation, any conflicts or difference can be resolved at this stage, In our case, again the authors themselves belong to the software engineering community and have enough expertise to perform this step itself as well.

5.2.3.6 Train the model

Once we create a machine-learning annotator component, we can select the document sets that we want to use to train the annotator component and specify the percentage of documents that we want to use as training data, test data, and blind data, in this iteration we only created a single set of documents and used that for model generation keeping other settings on default as recommended by Knowledge Studio.

5.2.3.7 Evaluate the Model

In this iteration we didn't go in much deep, we created the basic model only. In later iterations, more work will be done for creating a robust and general model.

5.2.3.8 Publish the model

This is the step where we can leverage a machine-learning model that we trained with Watson Knowledge Studio by making it available to other Watson applications. We can deploy or export a machine-learning annotator on Watson other services such as Watson Discovery, Natural Language. In our case, we have deployed the model on the Watson Discovery service.

5.2.4 Demo

Figure. 5.19 shows the demo of the chatbot, a conversation between user and simulator about requirements of the system, the user asks the chatbot to tell me about the goal, tasks or constraints of systems and chatbot identify the intent and forward request to Watson discovery using cloud functions and, IBM cloud functions runs a query on

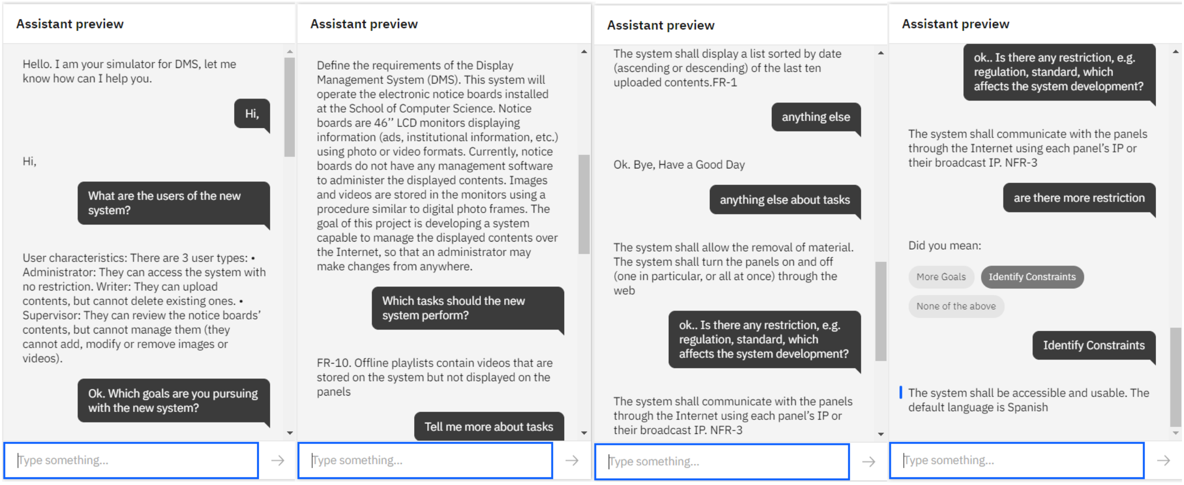


Figure 5.19: ChatBot Demo Cycle-2.

Watson Discovery and return results. Watson Assistant responds according to results received from the discovery.

5.2.5 Evaluation

5.2.5.1 Evaluation strategy

Due to the yet simplistic assistant's behavior, the same approach as in the first evaluation will be used.

5.2.5.2 Evaluation results

The responses given by the assistant have improved as compared to the previous version, but the behavior is still simple, and some problems remain. We are not including a full report here, as the number of interactions (~ 20) is pretty large and the detail does not have much interest. In turn, we will proceed immediately to the recommendations for improvement.

5.2.5.3 Recommendations for improvement

1. The set of intents shall be enlarged. For instance, the questions about *benefits* and *characteristics* have been mapped by the assistant to the `#goals` intent. The questions about connections with other stakeholders or systems have been mapped to the `#identify_users` and `#goals` intents. This is a clear demonstration of the current limited assistant's abilities.
2. The concept of `task` shall be refined to clearly identify:
   - An organization's task.

   - A system's task.

   This could be done in parallel to the introduction of new intents, as described in the previous bullet point. It is highly likely that it will require the improvement of the M1 (requirements) model.

3. The assistant does not provide consistent responses. On the one hand, when asked for users, it reports *all* the user classes. In turn, when asked for tasks, it reports *only one*. In the 3^{rd} cycle, the assistant could be designed for completeness, i.e., report all relevant information. In the 4th cycle, such information could be further elaborated, e.g., split in smaller chunks.

4. The `@constraint` entity has a mixed character. On the one hand, some of the synonyms used in its definition are completely general, e.g., `restriction`, `limitation`, etc. On the other hand, other terms are too general, e.g., `regulation`, `temporal`, etc., and can lead to misunderstandings.

   Regarding these latter terms: The problem arises due to the different contexts where these words can be used. For instance, `regulation` could be used in sentences about (the list is by no means complete):

   - Goals, e.g., *Our goal is to meet regulation requirements.*
   - Requirement sources, e.g., *Regulations say that taxpayers are eligible for.....*
   - Functional or non-functional requirements, e.g., *The minimum frame rate shall be 60 fps as required by regulations....*

   The `@constraint` definition shall be kept general, and other strategies shall be applied to identify the `#identify_constraints` intent. It looks like (although it is soon to make strong claims) that identification could be based on M2 (domain) models trained using the *Knowledge Studio*.

   Domain-specific terms, e.g., `IP`, `web standards`, etc. shall not be included in the entity definition. Even if the precision of the responses might increase (which didn't happen), the portability of the assistant gets negatively affected (due to the domain specificity of the terms). Again, the use of M2 (domain) models looks promising.

5. Similar considerations, in all respects, could be made regarding the `@users` entity.

5.2.5.4 Final considerations

1. Some undesired numbers, bullets, codes, etc. appear in the responses. They are probably due to imprecision's in the begin/end of the examples (both in *Knowledge Studio* and *Discovery*).

2. `System` cannot be used as a synonym of the `goal`. It leads to undesired identification of the `#goals` intent quite often.

3. The `@constraint` entity has been associated with the word `g`. The reason why is unclear.

4. Likewise, the `@users` entity has been associated with the words `software` and `hardware`.
5. The `@constraint` and `@users` entities could be renamed in the singular number. In general, all identifiers, e.g., `#identify_users`, `#goals`, etc. could be redefined in a uniform way to improve code understanding and maintenance.
6. The entity values (of any entity, not `@constraint` and `@users` only) could be defined in the singular number. It is highly likely (although it would have to be checked) that Watson can infer plurals from singular names.

5.3 Iteration 3

5.3.1 Requirements

The third cycle is focused on adding more intents and improving the performance of the machine learning model with the use of a bigger training data set. To speed up the annotation process for model training and reduce the human annotator workload, we created a domain-specific dictionary for each entity type (goals, tasks, etc.). Dictionaries are equivalent to *L0*-models, that is, instances of the *L1*-model (the RE concept model) which contains domain-specific concepts. Summary of requirements for the third cycle:

- Adding few more intents.
- Improving Machine learning model.
- Pre-annotation using dictionaries.
- Improving query process.

5.3.2 Development

As stated in the requirements section, this cycle aims to incorporate expert suggestions in addition to new features. Below we describe the development work in detail that has been carried out to complete requirements. In sub-section 5.3.2.2, we list new intents added on assistant. In sub-section 5.3.2.3, we talk about new entities for intents. In sub-section 5.3.2.4, assistant new dialogues have been discussed. In sub-section 5.3.2.5, we presents detail about used dictionaries in third iteration for pre-annotation process. In sub-section 5.3.2.6, we describe how pre-annotation is works. Finally, in sub-section 5.3.2.7, we describe annotation process on Watson Knowledge Studio.

5.3.2.1 Architecture

The chatbot architecture remains same as shown in Fig. 5.18 , but we add three more intents so RE concept model for 3rd iteration changes as shown in Fig.5.20.

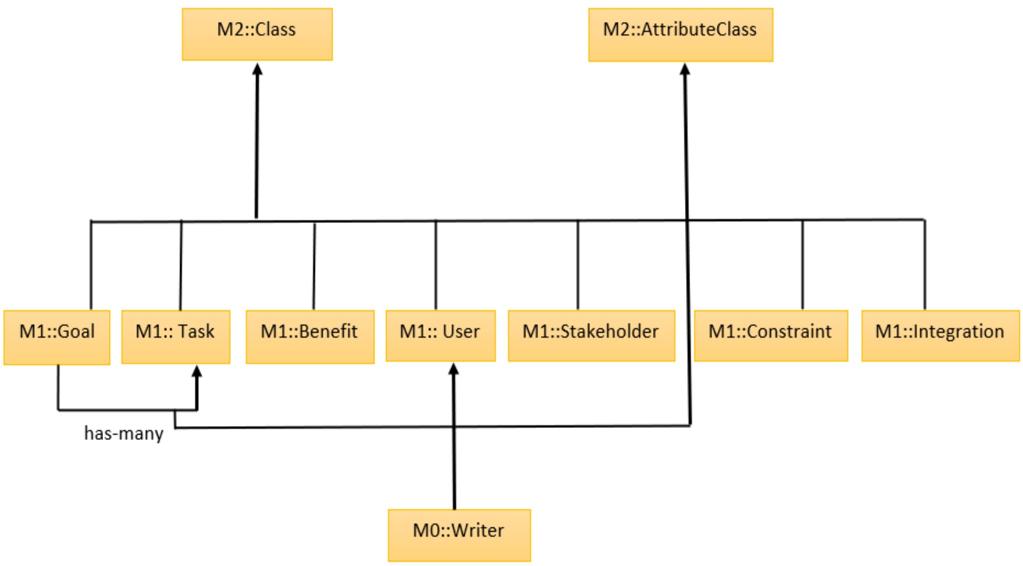


Figure 5.20: Telos Meta Model for third cycle.

5.3.2.2 Intents

We added three more intents in Watson Assistant to incorporate expert suggestions.

- **#benefits**: This intent is used to identify benefits of system.
- **#integrations**: This intent is used to identify any integration of the new system with existing hardware or software or system.
- **#stakeholders**: This intent is used to identify stakeholders of the system i.e. users that are going to be affected by the system.

5.3.2.3 Entities

Similarly, we created entities for each intent to improve the intent recognition process.

- **benefit**: Used to identify benefit related questions from user input i.e system will display ads etc.
- **integration**: Used to identify integration related questions from user input.
- **stakeholder**: Used to find affected users other than end-user i.e Owner of system or third party.

5.3.2.4 Dialogues

We added three more dialogues for new intents.

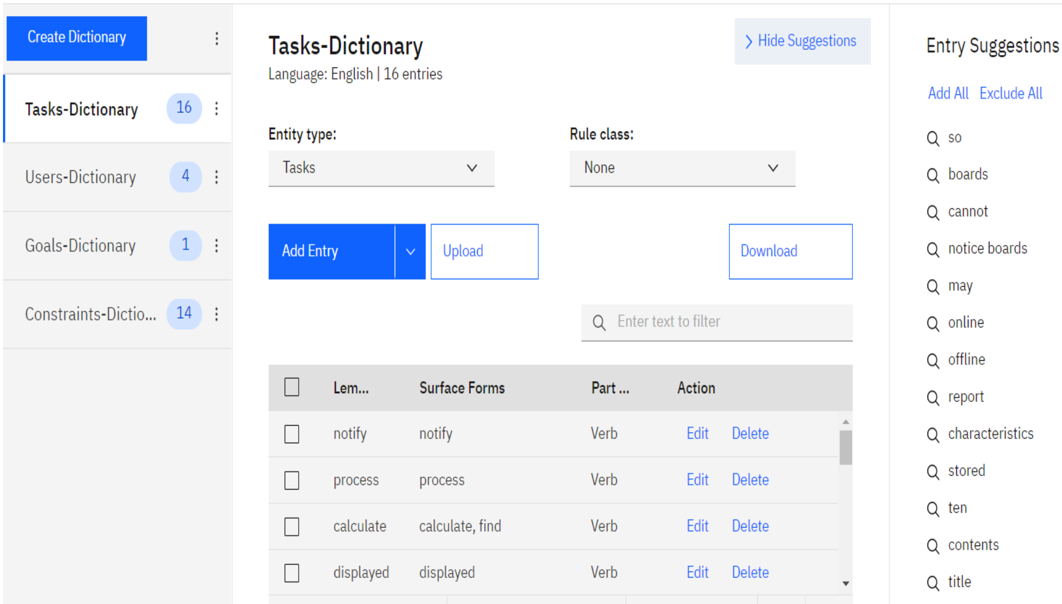


Figure 5.21: Dictionaries for Pre-annotation.

- **#Benefits**: This dialogue is used to display answers related to benefits of system.
- **#Integration's**: This dialogue is used to display answers related to any integration of the new system with existing hardware or software or system.
- **#Stakeholders**: This dialogue is used to display answers related to stakeholders of the system i.e. users that are going to be affected by the system.

5.3.2.5 Dictionaries

We have created four dictionaries so far as shown in Fig.5.21 for the pre-annotation process, these dictionaries have been mapped on Watson Knowledge Studio entity types and then have been utilized during the pre-annotation process.

- **Tasks-Dictionary:** Tasks dictionary is used to find and annotate the words related to tasks.
- **Users-Dictionary:** This dictionary is used to find and annotate the words related to users, such as writer, admin, etc.
- **Constraints-Dictionary:** Constraints dictionary is used to find and annotate the words related to Constraints.
- **Goals-Dictionary:** This dictionary is used to find and annotate the words related to goals, such as goal, intent, etc.

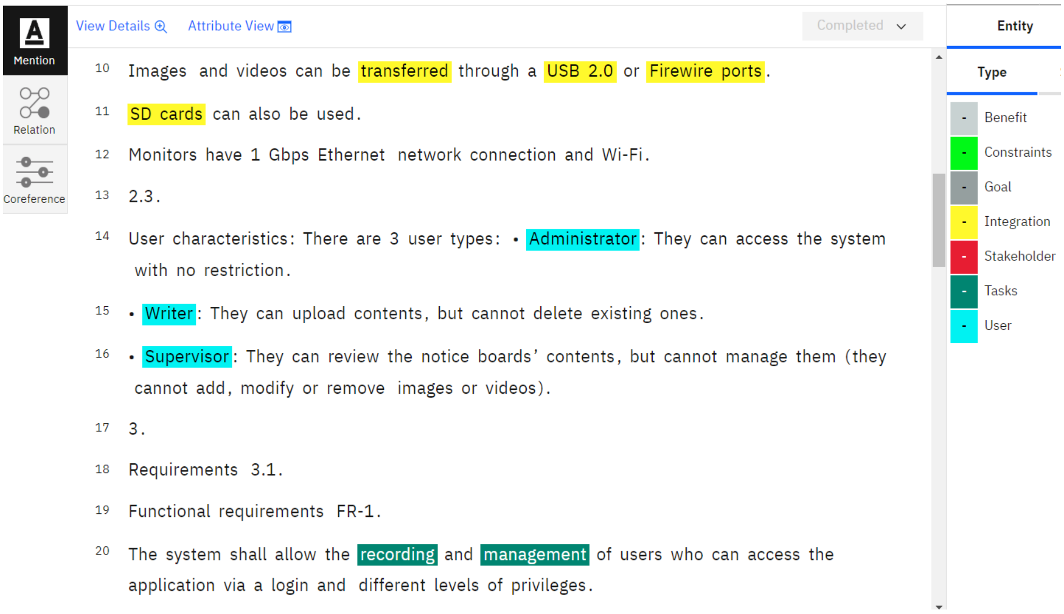


Figure 5.22: Annotation process.

5.3.2.6 Pre-Annotation

The pre-annotation process has been used to reduce the annotator workload, we have used dictionaries as discussed in the previous section but we will also test the machine learning model for the pre-annotation process in the later cycles. In the process of pre-annotation, Watson Knowledge Studio uses the defined terms and find the relevant words and then annotate them.

5.3.2.7 Annotation

The annotation process is rather simple but laborious unless having support from pre-annotation techniques. The process includes finding domain-specific terms and their annotation as shown in Fig.5.22.

5.3.2.8 Demo

Figure. 5.23 shows the demo of the chatbot for the third cycle, a conversation between user and simulator about requirements of the system, the user queries the chatbot about the goal, users, tasks, and constraints of systems. In response, at first assistant identify intents and then forward request to Watson discovery using cloud functions and, IBM cloud functions run a query on Watson Discovery and return results. Watson Assistant responds according to results received from the discovery. We can see results have been improved as compared to the previous iterations, we don't have irrelevant text like FR or NFR.

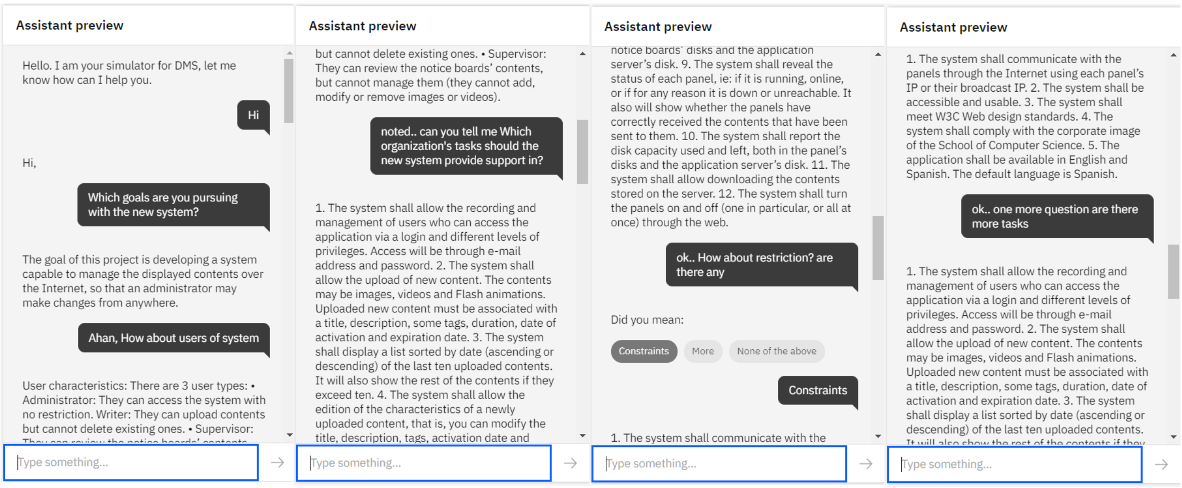


Figure 5.23: ChatBot Demo Cycle-3.

5.3.3 Evaluation

In the first and second research cycles, we have used *Expert evaluation* as recommended by Kitchenham et al. (Kitchenham and Linkman 2000). In this section, we will describe the evaluation conducted during the third cycle, where the expert has been replaced by a student. This represents an intermediate step towards the application of more formal evaluation procedures. The open character of the feedback provided by the expert has been limited in the case of the student evaluation. Instead of submitting his/her opinion, the student fills out two surveys:

- Conversation survey: `https://bit.ly/3gcQOVt`
- Opinion survey: `https://bit.ly/3e7LgcR`

The student was selected by convenience. He is a first-year MSc student with the Universidad Politécnica de Madrid. He has taken a Requirements Engineering course in the semester previous to this evaluation, including (short) interview training.

To facilitate the process of evaluation, context-free questions were given to the student. The simulator-student conversation lasted 5 minutes approximately. After interacting with the simulator, the student was asked to fill the surveys.

The first survey aims to assess the degree to which the student has acquired knowledge about the future software system during the interaction with the simulator. This survey contains sentences, e.g., *The system shall allow the upload of new content*, that are either *true* or *false*. The student was able to answer almost all (9 out of 10) questions correctly. This result shall be taken with obvious caution due that (1) just one student participated in the evaluation and (2) the Display Management System is a rather simple system.

The second survey collects the student's impressions regarding the interaction with the simulator (not the knowledge acquired). The responses are given on a 5-point Likert scale. Quite not surprisingly, the simulator has received a very good rating (most of the responses are 4's and 5s'). Further evaluation efforts are necessary.

Both surveys include one open-ended question. The student suggested that the simulator should answer questions more precisely; in the current version, all information related to one user intent is displayed at once. The simulator should also provide support for clarification questions. Student suggestions have been recorded and will be incorporated in the next cycles. Most of the suggestions match our future work; hence the research project will not experience changes in the short term.

5.4 Iteration 4

5.4.1 Requirements

In the third cycle, the system was tested with a student, as a result of evaluation it was decided that the focus of the fourth cycle should be on: "Providing partial instead of complete information to the requirements engineer". Hence this iteration will focus on making changes in Watson Discovery and cloud code to respond assistant in chunks for relevant intents.

5.4.2 Development

The development work includes the changes in code to return results in chunks rather than single paragraphs, for that we made changes in cloud function and query as well. In order to get results in chunks, we first get all entity types for user input, and then we pick random entity type to return specific results in chunks. Moreover, we also have added functionality of highlighting entities relevant to intents, below we describe the changes step by step:

1. At first, we query Watson Discovery to return all data related to identified intent.
2. We get entities from metadata.
3. We sort out entities only select entities with relevant types. i.e for intent "Task" we select all entities with type "Task", the type concept has been explained in Watson Knowledge Studio section 5.2.3.
4. We pick random entities from the sorted entity list.
5. We query to discovery again to return more precise results using identified type and entity text i.e.

$$'query' :' text' : +entityText+', enriched\_text.entities.type :' +queryText; \tag{5.1}$$

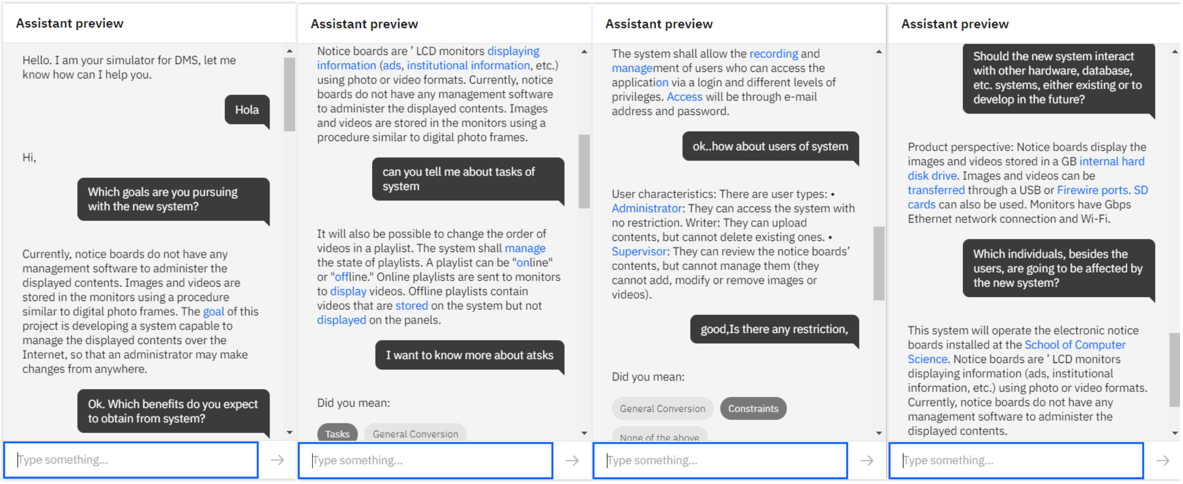


Figure 5.24: ChatBot Demo Cycle-4.

here "entityText" is text of enriched entity and "queryText" is identified intent type received as parameter from Watson Assistant. This Discovery language query will return concise results; it will look for content where entity type is "queryText" and it contains the "entitytext" text as well.

6. We use the sorted list of entities to highlights the relevant results (i.e. results that we are going to return), the process of highlighting is simple we just update entity text with HTML tag. Watson Assistant understands some HTML tags including the one we have used.

7. We then return the most relevant results i.e top paragraph.

5.4.2.1 Architecture

The chatbot architecture and RE concept model both remain the same.

5.4.2.2 Demo

Figure. 5.24 shows the demo of the chatbot for the fourth cycle, a conversation between user and simulator about requirements of the system, the user queries the chatbot about the goals, users, tasks, benefits, the affected party, integration, and constraints of systems. In response, at first assistant identify intents and then forward request to Watson discovery using cloud functions and, IBM cloud functions run a query on Watson Discovery and return results. Watson Assistant responds according to results received from the discovery. We can see results have been much improved as compared to the previous iterations, we don't have the irrelevant text and large paragraphs and importantly the relevant text is highlighted.

5.4.3 Evaluation

In this section, we will report the evaluation conducted during the fourth cycle, where we have selected three students instead of single in the previous iteration This represents an intermediate step towards the generalization of the validation process for the system. The single student evaluation can be bias or an exceptional case for the system behavior. Again instead of submitting their opinions, the students fill out two surveys used in the previous iteration but questions for "Conversation survey" have been modified :

- Conversation survey: `https://forms.gle/KR3sXUimeC6cjCTR8`
- Opinion survey: `https://forms.gle/MgMte15Fz9WRxKRJ9`

The students were selected by convenience. Two of them are first-year MSc students with the Universidad Politécnica de Madrid and the third one is a second-year student at the same university. Two of them have taken a Requirements Engineering course in the semester previous to this evaluation, including (short) interview training and the third one has taken the course in first-year and also holds some development experience.

To facilitate the process of evaluation, again context-free questions were given to the students. The simulator-students conversation lasted 8 minutes approximately. After interacting with the simulator, the students were asked to fill the surveys.

The first survey aims to assess the degree to which the students have acquired knowledge about the future software system during the interaction with the simulator. This survey contains sentences, e.g., *The system shall allow the upload of new content*, that are either *true* or *false*.

As a result, the students were able to answer 86 percent of questions correctly. Results are good but not unusual, as Display Management System is a slightly simple system.

The second survey collects the student's impressions regarding the interaction with the simulator (not the knowledge acquired). The responses are given on a 5-point Likert scale.

In the second survey, the simulator also received a positive rating (4.2 avg). In open-ended questions, the students highlighted that simulator performs well but there is still a need for some improvements, for instance, the simulator should highlight the words properly, it should provide support for making summaries, and simulator should also provide support for clarification questions. Student suggestions have been recorded and will be incorporated in the next cycles. Again most of the suggestions match our future work; hence the research project will not experience changes in the short term.

5.5 Iteration 5

5.5.1 Requirements

In fourth cycle system was tested with more than one student and evaluation results were quite satisfactory. Hence this cycle will focus on incorporating the student's suggestions and adding features to improve the conversation abilities, allowing the interview simulator to understand engineers' meta-questions, such as "is there anything else", "am I correct", or summaries. Moreover, changes in Watson Assistant and cloud code will be made to complete the above requirements.

5.5.2 Development

In fifth cycle development work is focused on making changes in cloud code and Watson Assistant for improving the conversation abilities of an assistant. We have created three more intents and dialogues i.e. "more", "verify" and "summary". The intent "more" is used to handle questions such as "anything else" or "are there more", the intent "verify" is used to answer meta-questions such as "am I correct about this" and "summary" is used to provide a summary of the conversation. Moreover, the context variables have been used in Watson Assistant to keep the context of user conversation and respond accordingly i.e. depending on the value of context variable. In addition to user context, we are also keeping the record of full conversation so that users can see a summary of the interaction with the assistant.

On the cloud side, we have modified code to query Watson Discovery based on parameters received from assistant, if the assistant sends just a single parameter(i.e. intent type input) we query using discovery language else natural language query. The Watson Discovery language query is used to utilize annotated types which is possible in case of intents such as Tasks and Goals but when the user asks a question like "am I correct system will show the preview of playlist", it is not possible to answer such questions using discovery language query so we use natural language query and return top-most results to clarify user questions. Below we describe the working of functionality step-by-step:

- User interacts with assistant using a provided web interface.
- Assistant keeps the context using variable i.e.

$$"usercontext" = "Goal"; \tag{5.2}$$

  , dialog node will update the value of "usercontext" depending on dialogue execution in response to a user question.
- We use context variables to respond accordingly when the user asks questions such as "anything else" or "are there more" i.e. depending on the value of context of variable such as "Tasks", "Goals" etc.

- In addition to user context, we also use another context variable "summary" that keeps user interaction data.
- During the execution of dialogue node it appends question and answers to current data i.e.

$$"summary+' < br > Q.' + input.text+' < br >' +webhook\_result.data"; \quad (5.3)$$

here we append data into the "summary" variable that includes "Q." as an input of the user and webhook result is the assistant response shown to the user.
- We show the summary using a summary variable when the user asks for it.
- When a user asks verification related questions such as "am I correct about this", we send additional parameter to cloud function i.e natural
- Cloud function check for parameters if there is a "natural" parameter in JSON object it will run natural language query.
- In case no intent has been detected by Watson Assistant we also run a natural language query, instead of replying we couldn't find anything we let Watson Discovery decide.
- We show the most relevant result for natural language queries.

5.5.2.1 Architecture

The chatbot architecture remains same as shown in Fig. 5.18 , but we add three more intents so RE concept model for fifth iteration changes as shown in Fig.5.25.

5.5.2.2 Demo

Figure. 5.26 displays the demo of the chatbot for the fifth cycle, a dialogue between user and simulator about requirements of the system, the user queries the chatbot about the goals, users, tasks, benefits, the affected party, integration, and constraints of systems. In response, the Watson Assistant responds according to results received from the discovery. Results after the fifth cycle are more natural and accurate, the assistant can answer meta-questions, the question related to more information and a summary of a conversation between user and assistant.

5.5.3 Evaluation

In this section, we discuss the evaluation method used in the fifth cycle for the simulator. After expert opinions and surveys with students, we introduce a more formal method to finish the thesis, i.e., a formal experiment. Similar approaches have been used by (Niknafs and Berry 2017; Pitts and Browne 2004; Agarwal and Tanniru 1990; Marakas

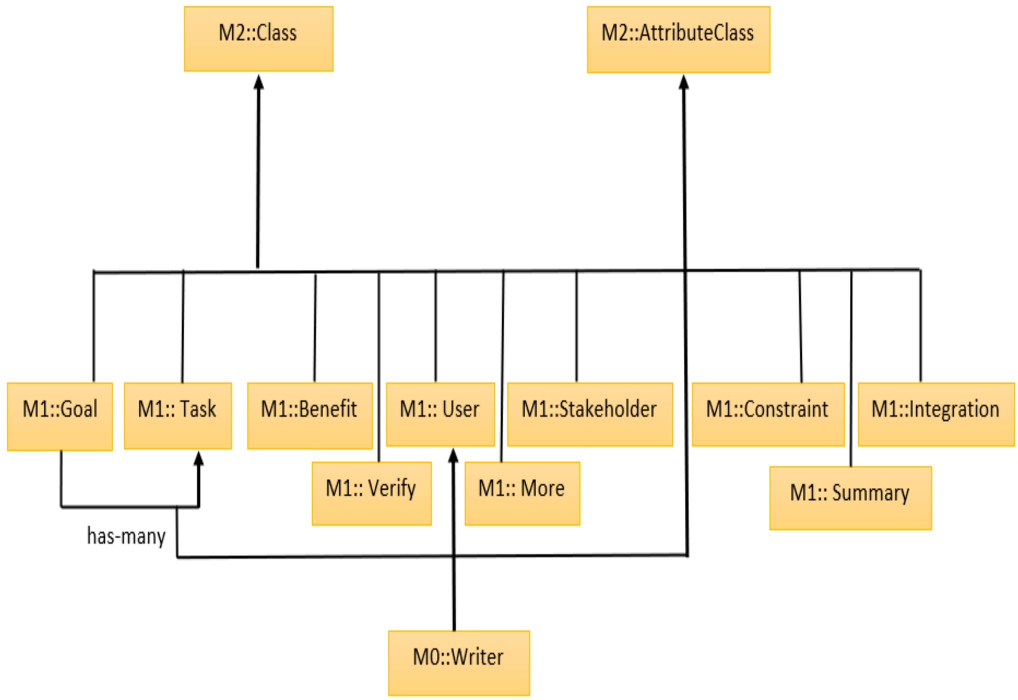


Figure 5.25: Telos Meta Model for fifth cycle.

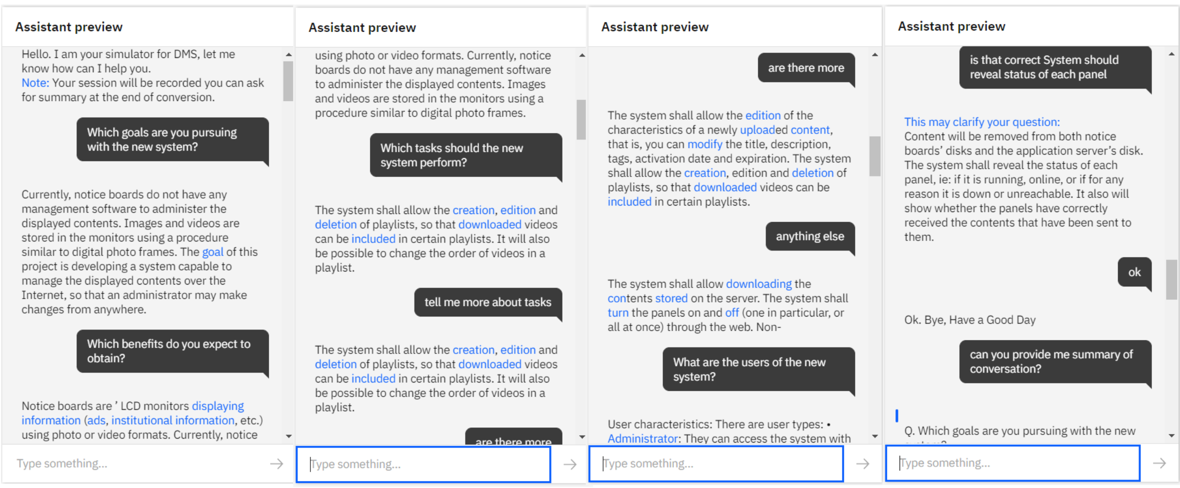


Figure 5.26: ChatBot Demo Cycle-5.

and Elam 1998). We will use a hybrid approach. First, the experiment will check the correctness and completeness of of the information provided y the chatbot. Second, we will use a survey, as in the previous cycles to get students' feedback related to the simulator performance.

5.5.3.1 Hypotheses

The aim of our research is helping students to gain interview skills. An interview is successful when all relevant information is correctly acquired. Thus, our hypotheses are:

- $H1_0$: The students cannot retrieve the correct information using the chatbot.
- $H1_1$: The students can retrieve the correct information using the chatbot.

and

- $H2_0$: The students cannot retrieve complete information using the chatbot.
- $H2_1$: The students can retrieve complete information using the chatbot.

5.5.3.2 Independent Variables

The independent variable is the chatbot usage. In a full-fledged experiment, this variable would have two different levels or values: using chatbot vs. conducting an actual interview. Due to the prototype character of the chatbot, a comparison with a real interview would be unfair. Likewise, the number of experimental subjects is limited. Therefore, we decided to perform an exploratory quasi-experiment instead of a real experiment. The goal is understanding whether Hx_0 could be rejected in larger-scale studies, instead of formally testing the hypotheses using statistical methods.

5.5.3.3 Dependent Variable

There are two dependent variables:

- The subjects' capability to correctly retrieve information using the chatbot.
- The completeness of the information acquired.

Moreover, subjects will also provide feedback on how the system behaved in response to their questions; this will help to check the performance of the system as well i.e in terms of usability and performance.

5.5.3.4 Experimental task

We have used the same SRS than in previous evaluations. This may be a threat to validity to some extent but students do not have complete information about SRS and interaction in the previous iterations had a different purpose. Moreover, the simulator results were different as well. In this iteration, we ask students to use the system and create SRS for a system using the simulator.

5.5.3.5 Subject Selection

The students who participated in the previous cycle were selected for this cycle as well. Two of them are first-year MSc students with the Universidad Politécnica de Madrid and the third one is a second-year student at the same university. Two of them have taken a Requirements Engineering course in the semester previous to this evaluation, including (short) interview training and the third one has taken a course in first-year and also holds some testing experience.

5.5.3.6 Experiment Execution

We select students and ask them to use the simulator and create SRS for Display Management System. The SRSs created by students will be checked in comparison to the original SRS of the Display Management System. We have created google form using IEEE-830 format for SRS; form also has questions related to the opinion:

- Evaluation form: `https://forms.gle/11jZmr29SaNYDZpo9`

To facilitate the process of evaluation, again context-free questions were given to the students. The simulator-students conversation lasted 30 minutes approximately.

5.5.3.7 Analysis

As stated previously, we have checked the correctness and completeness of the information acquired by the students using the chabot. After the qualitative analysis of each SRS created by students, we conclude that all students were able to create correct/complete SRSs in comparison to the original SRS. They were able to put things in the right place as well. We did not perform statistical analysis because of small data points. Moreover, the simulator received 4.6 ratings as well. All students agreed that the system will be helpful for novice requirements engineers in gaining interview skills. We didn't receive any issue regarding the performance of the simulator. However we received a suggestion for making improvements in a simulator to act like a real interviewee, that will be addressed in the final iteration.

5.6 Iteration 6

5.6.1 Requirements

In this last iteration, we will focus on introducing natural language pitfalls in assistant, e.g., ambiguity, incompleteness, etc, and to incorporate suggestions noted during the evaluation of the fifth cycle. Our goal is to make conversation natural as much as possible. We will make changes in the cloud code to introduce natural language pitfalls.

5.6.2 Development

In this cycle development work is focused on making changes in cloud code to achieve above-stated requirements, below we describe step by step how we add ambiguity and incompleteness:

- User interacts with assistant using a provided web interface.
- Assistant forwards request to cloud function.
- Cloud function query discovery.
- Cloud function does processing on data.
- We remove unwanted terms.
- We generate two random conditions for introducing ambiguity and incompleteness.
- If the first condition met we call add ambiguity function else if the second condition met we call add incompleteness function.
- In add ambiguity function we remove the random entity from the response.
- In add incompleteness function we remove random sentence.
- Finally, we return a response.

5.6.2.1 Architecture

The chatbot architecture and RE concept model both remain the same.

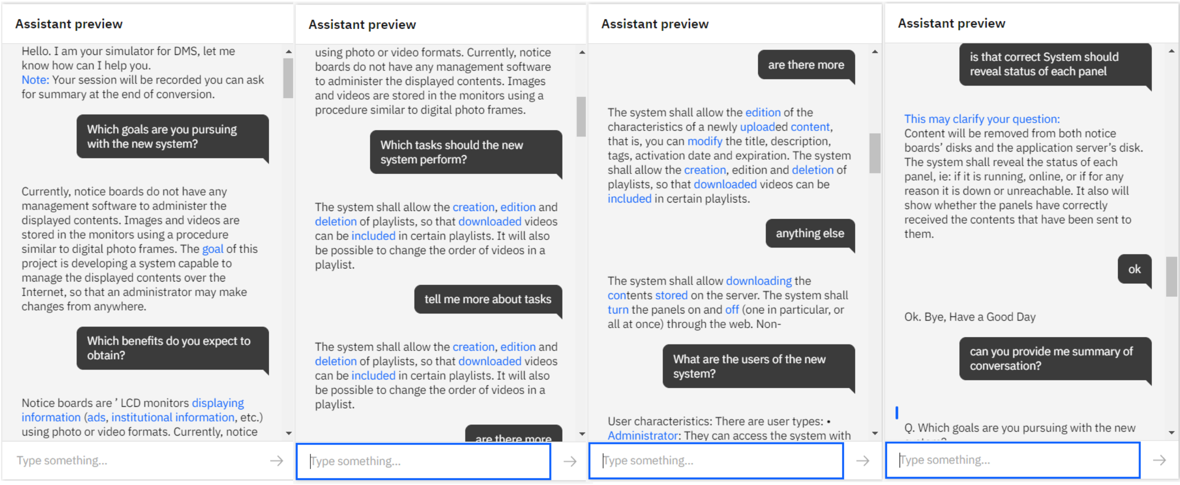


Figure 5.27: ChatBot Demo Cycle-6.

5.6.2.2 Demo

Figure. 5.27 displays the demo of the chatbot after the final cycle, a dialogue between user and simulator about requirements of the system, the user queries the chatbot about the goals, users, tasks, benefits, the affected party, integration, and constraints of systems. In response, the Watson Assistant responds according to results received from the discovery. In the final cycle, we focused on introducing natural language pitfalls; we can see simulator return an inconsistent response when asked about the goals of the system. Moreover, when we ask any meta-question; simulator response with clarification rather than yes/no answer. This implies that simulator responds realistically, in real interviews customers often provide ambiguous and incomplete information.

5.6.3 Evaluation

In this section, we discuss the evaluation method used in the final cycle. Formal evaluation of the system was carried in the fifth cycle in this cycle system is not much changed so we will use opinion survey only to evaluate simulator. Five students were selected to interact with the system and provide their opinion on the performance of the system.

- Opinion Survey: `https://forms.gle/PnCEHmG5EEkBZniK7`

The students were selected by convenience three from the Universidad Politécnica de Madrid, and the remaining two from Technische Universität Kaiserslautern, Germany, University of Oulu, Finland respectively. They have taken a Requirements Engineering course.

In final the survey, we collects the student's impressions regarding the interaction with the simulator . The responses are given on a 5-point Likert scale. Quite not surprisingly, the simulator has received a very good rating (4.7/5). Moreover, rating should be taken with caution due to small set of subjects.

Chapter 6

Discussion

In this chapter, we explain the attainment of research goals. At first, we outline each research goal and then provide detail on how we achieved it. The justification and explanation are based on the results of previous chapters.

6.1 Create a chatbot

The first research goal is achieved by developing an interactive and robust chatbot using Watson Technologies such as Watson Assistant, Watson Discovery, IBM Cloud Functions, and Watson Knowledge Studio. An iterative and incremental approach has been observed for the development process based on the Design Science approach.

The interview simulator can answer diverse types of questions related to RE concepts such as Tasks, Goal, Users, Benefits, Stakeholders, Constraints, and Integration. Moreover, the simulator also presents a summary of conversation when asked and answers meta-questions such as "Am I correct system should reveal the status of each panel".

The simulator can also answer the questions based on the context, for instance when the discussion is going on about tasks and the user asks "anything else" or "are there more". Likewise, it will identify other contexts such as constraints or goals.

6.2 Introduce natural language pitfalls

The interview simulator has been developed with the idea of achieving a natural conversation between client and requirements engineer. To achieve this goal we have added logic in cloud code that makes simulator to:

- Detach entities from response randomly.
- Adds incompleteness randomly, i.e., it detaches some text from the response depending on random condition.

- Response randomly to the questions.
- Repeat the same answer sometimes as the same thing happen in real life interviews.
- When asked about yes/no questions it tries to clarify answer rather answering in yes/no format.

6.3 Emphasize good interview practices during the conversation

Interviewing customers is not an easy-going job; it requires skills and expertise. Senior requirements engineers possess such skills but novice requirements engineers struggle. To help them we use good practices while developing the assistant:

- We use context-free questions.
- Our system is capable of making summaries and answering meta-questions.
- Novice engineers can check those summaries and reflect on what went wrong or right.
- They can ask meta-questions such as "Am I correct system should reveal each panel" or "anything else" or "are there more" for clarifying their doubts and getting complete requirements.
- Once novice engineers feel they done with interviews they can ask for a summary and reflect on that.

6.4 Elaborate a design theory

One of our goals is to provide a skeleton for creating chatbots in requirements elicitation. The experience gained in this project enable us to sketch an architecture that can be used in different domains, and probably with different elicitation techniques such as surveys or focus groups.

Fig.6.1 shows the proposed chatbot architecture, that includes all the required components for building chatbot application for requirement elicitation domain. Context-free identified intents in architecture present Telos M1 model, while a dictionary representing the Telos M0 model that can be used to annotate the documents before model generation. The Watson Assistant takes input(i.e Telos M0 model) and identifies the purpose of using intents, then it calls API/Cloud Function to run a query on discovery service and return results. On the other hand, Watson Discovery uses Machine Learning Model provided by Watson Knowledge Studio. For utilizing the architecture following guidelines may be followed:

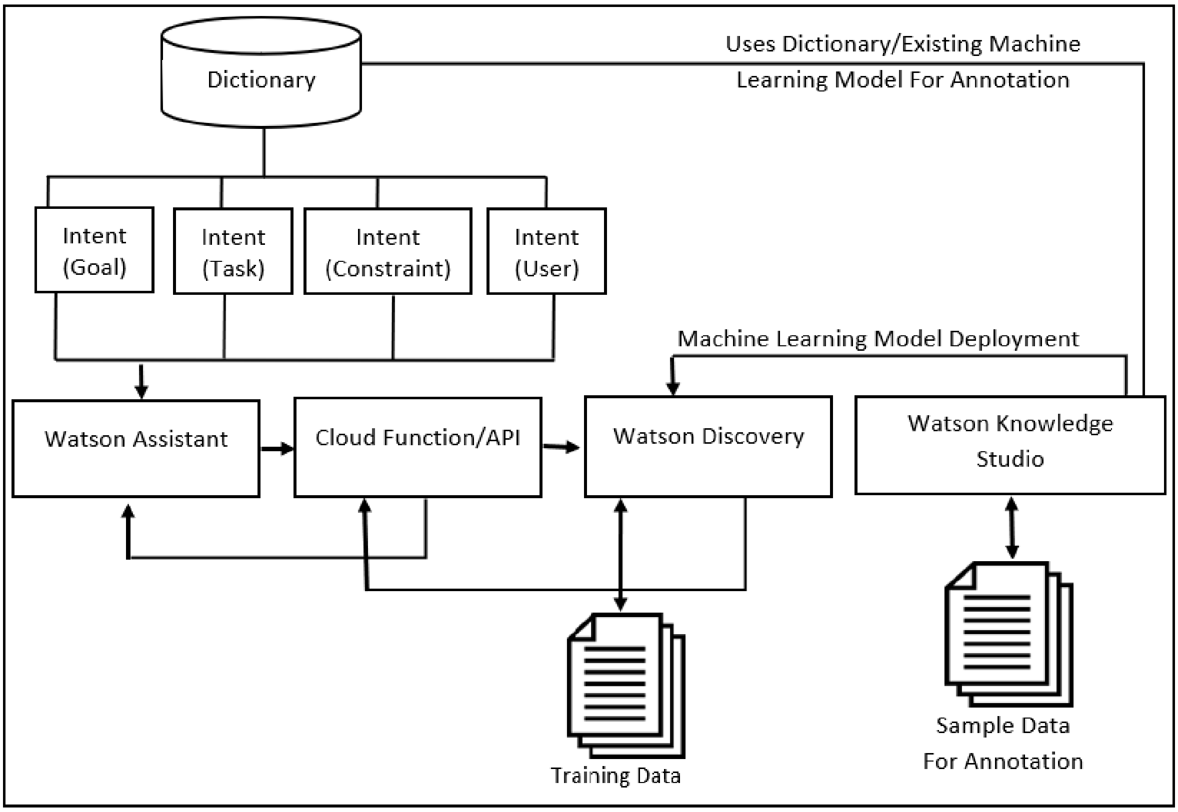


Figure 6.1: Abstract Architecture for developing Chatbot in REl domain.

- First step after identifying the problem is to define context-free intents for the Watson assistant.
- Next step should be integration with Watson Discovery using IBM cloud function or custom web API.
- Once integrated with discovery service annotation and configuration should be performed in the Watson discovery such as creating new fields and breaking documents.
- After integrating with the discovery service Watson Knowledge studio should be introduced.
- In order to utilize knowledge studio guidelines provided in section 2.6.3 must be followed.
- Once done with the previous step; machine learning model must be deployed on discovery service.

Chapter 7

Conclusion

In this chapter, we present a summary of the thesis outcomes; we also discuss future work.

7.1 Summary

The interview simulator has been created using the *Design Science* approach with the primary goal of helping novice requirements engineers. The results have been promising:

- The interview simulator understands context-free questions, retrieving the right information related to RE concepts such as Tasks, Goals, Users, Benefits, Stakeholders, Constraints, and Integration.
- The simulator also makes summaries of the conversation, answer meta-questions, and answer the questions based on the context. Moreover, the simulator also responds randomly with ambiguity and incompleteness.
- The simulator's knowledge base has been defined at 3 levels: (1) *L2*, i.e., classes and relationships, (2) *L1*, containing RE-related concepts, and (3) *L0*, including domain-specific concepts. This knowledge base is easy to implement in Watson using the Knowledge Studio and Assistant services.
- The modular structure of the knowledge base leverages the simulator's ability to maintain conversations in different domains. Code changes will not be likely required.

The chatbot was evaluated by conducting the experiment in the fifth cycle, students were asked to create SRSs using the simulator for Display Management System. SRSs of students were compared with the original SRS of the Display Management System for checking the correctness of information retrieved using the simulator. Furthermore, students were also asked to give feedback about interaction. The simulator has received a good response from students. They were able to complete the tasks, and the feedback of the final version suggests that the simulator will be surely helpful for novice engineers to improve interview skills.

7.2 Future Research

The future research will focus on applying the interview simulator to different problem domains by utilizing the modular structure of the knowledge base. Due to time and resource constraints we were not able to test the interview simulator to different domains. In addition to different domains, more work will be done to make an interview simulator to act like a real assistant.

Moreover, the interview simulator will be tested in a real RE course in the academic year 2020-2021. Once it proves effective in the classroom, it will be opened to the RE community for free use and improvement.

Bibliography

Agarwal, Ritu and Mohan R Tanniru (1990). "Knowledge acquisition using structured interviewing: an empirical investigation." In: *Journal of Management Information Systems* 7.1, pp. 123–140.

Anwar, Fares, Rozilawati Razali, and Kamsuriah Ahmad (2011). "Achieving effective communication during requirements elicitation-A conceptual framework." In: *International Conference on Software Engineering and Computer Systems*. Springer, pp. 600–610.

Bleiel, Nicky (n.d.). *About Discovery*. URL: `https://cloud.ibm.com/docs/discovery?topic=discovery-about`. (accessed: 04.25.2020).

Bollweg, Lars et al. (2019). "Who Am I, and If So How Many? A Multi-Entity Chatbot Business-Interview Simulator for Individualized Practical Assignments in MOOCs." In: *EdMedia+ Innovate Learning*. Association for the Advancement of Computing in Education (AACE), pp. 928–936.

Bolton, Michael (n.d.). *DialofFlow-Offical Blog*. URL: `https://www.developsense.com/blog/2010/11/context-free-questions-for-testing/`. (accessed: 05.18.2020).

Boyer, Jerome (n.d.). *IBM Watson Discovery*. URL: `https://www.ibm.com/cloud/architecture/tutorials/cognitive_discovery`. (accessed: 07.07.2020).

Carrizo, Dante, Oscar Dieste, and Natalia Juristo (2014). "Systematizing requirements elicitation technique selection." In: *Information and Software Technology* 56.6, pp. 644–669.

Carrizo, Dante, Oscar Dieste, Natalia Juristo, and Marta López (2011). "Estudio experimental de la efectividad de la entrevista abierta frente a la entrevista independiente de contexto." In: *Actas del 14th Workshop on Requirements Engineering*, pp. 297–308.

Cloud Functions (n.d.). URL: `https://www.ibm.com/cloud/functions`. (accessed: 05.21.2020).

Davis, Alan et al. (2006). "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review." In: *14th IEEE International Requirements Engineering Conference (RE'06)*. IEEE, pp. 179–188.

Davison, Robert, Maris G Martinsons, and Ned Kock (2004). "Principles of canonical action research." In: *Information systems journal* 14.1, pp. 65–86.

Dvir, Dov, Tzvi Raz, and Aaron J Shenhar (2003). "An empirical analysis of the relationship between project planning and project success." In: *International journal of project management* 21.2, pp. 89–95.

Fitzmaurice, Brian et al. (2007). “Virtual interviews for students interacting online for psychiatry (VISIOn): a novel resource for learning clinical interview skills.” In: *Psychiatric Bulletin* 31.6, pp. 218–220.

Gause, D.C. and G.M. Weinberg (1989). *Exploring Requirements: Quality Before Design*. Dorset House Pub. ISBN: 9780932633132.

Gilvaz, Ana Paula Pinho and JC Sampaio do Prado Leite (1995). “FAES: A case tool for information acquisition.” In: *Proceedings Seventh International Workshop on Computer-Aided Software Engineering*. IEEE, pp. 260–269.

Hagarty, Rich (n.d.). *Combine Watson Knowledge Studio and Watson Discovery*. URL: `https://github.com/IBM/watson-discovery-food-reviews`. (accessed: 05.06.2020).

Hevner, Alan and Samir Chatterjee (2010). “Design science research in information systems.” In: *Design research in information systems*. Springer, pp. 9–22.

Iivari, Juhani (2008). “Expert evaluation vs bibliometric evaluation: experiences from Finland.” In: *European Journal of Information Systems* 17.2, pp. 169–173.

Jessica Docarlo, Garrett May (n.d.). *Tutorial: Creating a machine-learning model*. URL: `https://cloud.ibm.com/docs/services/knowledge-studio?topic=knowledge-studio-wks_tutml_intro`. (accessed: 04.24.2020).

Kamata, Mayumi Itakura and Tetsuo Tamai (2007). “How does requirements quality relate to project success or failure?” In: *15th IEEE International Requirements Engineering Conference (RE 2007)*. IEEE, pp. 69–78.

Karmokar, Sangeeta and Harminder Singh (2012). “Improving the Website Design Process for SMEs: A Design Science Perspective.” In: *CONF-IRM*, p. 26.

Kato, Junzo et al. (2001). “A model for navigating interview processes in requirements elicitation.” In: *Proceedings Eighth Asia-Pacific Software Engineering Conference*. IEEE, pp. 141–148.

Kawaguchi, Atsuo et al. (1987). “SIS: A Shell for Interview Systems.” In: *IJCAI*. Citeseer, pp. 359–361.

Kitchenham, Barbara A. and Stephen G. Linkman (2000). “DESMET : A method for evaluating Software Engineering methods and tools.” In:

Kotonya, Gerald and I. Sommerville (1998). *Requirements engineering: processes and techniques*. Worldwide series in computer science. J. Wiley. ISBN: 9780471972082.

Koubarakis, Manolis et al. (2020). “A retrospective on Telos as a metamodeling language for requirements engineering.” In: *Requirements Engineering*, pp. 1–23.

Kowalski, Stewart, Katarina Pavlovska, and Mikael Goldstein (2009). “Two case studies in using chatbots for security training.” In: *IFIP World Conference on Information Security Education*. Springer, pp. 265–272.

Kuligowska, Karolina (2015). “Commercial chatbot: performance evaluation, usability metrics and quality standards of embodied conversational agents.” In: *Professionals Center for Business Research* 2.

Laiq, Muhammad and Oscar Dieste (2020). “Chatbot-based Interview Simulator: A Feasible Approach to Train Novice Requirements Engineers.” In: *Proceedings of the 10^{th} International Workshop on Requirements Engineering Education and Training (REET)*, submitted for review.

Lundeberg, M, G Goldkuhl, and A Nilsson (1981). “Information systems development, a systematic approach ‘, Prentice—Hall.” In: *Englewood Cliffs, New Jersey*.

Maedche, Alexander et al. (2019). "AI-Based Digital Assistants." In: *Business & Information Systems Engineering* 61.4, pp. 535–544.

Maiden, Neil AM and Gordon Rugg (1996). "ACRE: selecting methods for requirements acquisition." In: *Software Engineering Journal* 11.3, pp. 183–192.

Marakas, George M and Joyce J Elam (1998). "Semantic structuring in analyst acquisition and representation of facts in requirements analysis." In: *Information Systems Research* 9.1, pp. 37–63.

Miller, Michelle (n.d.). *About Watson Assitant*. URL: `https://cloud.ibm.com/docs/assistant?topic=assistant-index`. (accessed: 04.24.2020).

Niknafs, Ali and Daniel Berry (2017). "The impact of domain knowledge on the effectiveness of requirements engineering activities." In: *Empirical Software Engineering* 22.1, pp. 80–133.

Pandorabots, Inc. (n.d.). *Pandorabots Documentation*. URL: `https://pandorabots.com/docs/`. (accessed: 04.24.2020).

Pitts, Mitzi G and Glenn J Browne (2004). "Stopping behavior of systems analysts during information requirements elicitation." In: *Journal of management information systems* 21.1, pp. 203–226.

Ritzmann, Sandrina, Vera Hagemann, and Annette Kluge (2014). "The Training Evaluation Inventory (TEI)-evaluation of training design and measurement of training outcomes for predicting training success." In: *Vocations and Learning* 7.1, pp. 41–73.

Sehgal, Surbhee (n.d.). *DialofFlow-Offical Blog*. URL: `https://blog.dialogflow.com/`. (accessed: 04.24.2020).

Sommerville, Ian and Pete Sawyer (1997). *Requirements engineering: a good practice guide*. John Wiley & Sons, Inc.

Stanica, Iulia et al. (2018). "VR job interview simulator: where virtual reality meets artificial intelligence for education." In: *2018 Zooming innovation in consumer technologies conference (ZINC)*. IEEE, pp. 9–12.

Therkelsen-Terry, Nick (n.d.). *Why Watson? More accuracy. Less training time. Insanely better CX results*. URL: `https://www.ibm.com/blogs/watson/2017/09/more-accuracy-less-training-time-better-cx-results-with-watson/`. (accessed: 05.21.2020).

Watson Knowledge Studio (n.d.). URL: `https://www.ibm.com/cloud/watson-knowledge-studio`. (accessed: 05.21.2020).

Web-hook (n.d.). URL: `https://cloud.ibm.com/docs/assistant?topic=assistant-dialog-webhooks`. (accessed: 05.21.2020).

Wetherbe, James C (1991). "Executive information requirements: getting it right." In: *Mis Quarterly*, pp. 51–65.

Wiegers, K. and J. Beatty (2013). *Software Requirements*. Developer Best Practices. Pearson Education. ISBN: 9780735679627.

Wohlin, Claes et al. (2005). *Engineering and managing software requirements*. Springer Science & Business Media.

Wood, Larry E (1997). "Semi-structured interviewing for user-centered design." In: *interactions* 4.2, pp. 48–61.

Yamanaka, Tatatoshi and Seiichi Komiya (2011). "A Method for Navigating Interview-driven Software Requirements Elicitation Work: Effectiveness Evaluation of the

Method from the Viewpoint of Efficiency." In: *International Journal Of Systems Applications, Engineering & Development* 5.

Yang, Shanshan and Chris Evans (2019). "Opportunities and Challenges in Using AI Chatbots in Higher Education." In: *Proceedings of the 2019 3rd International Conference on Education and E-Learning*, pp. 79–83.

Zowghi, Didar and Chad Coulin (2005). "Requirements elicitation: A survey of techniques, approaches, and tools." In: *Engineering and managing software requirements.* Springer, pp. 19–46.

Appendices

A Context-free questions

The list of *context-free* questions that appears below has been used during the Expert Evaluation of the *Design Science* cycles 1-2.

1. Which goals are you pursuing with the new system?
2. Which organization's tasks should the new system provide support in?
3. Which benefits do you expect to obtain?
4. Which characteristics do you wish the new system to have?
5. Which tasks should the new system perform?
6. What are the users of the new system?
7. Which individuals, besides the users, are going to be affected by the new system?
8. Should the new system interact with other existing or new systems?
9. Should the new system interact with other hardware, database, etc. systems, either existing or to develop in the future?
10. Is there any temporal restriction for development?
11. Is there any restriction, e.g. regulation, standard, which affects the system development?

These questions are based on (Kotonya and I. Sommerville 1998). They have been used in an elicitation experiment (Carrizo, Dieste, Juristo, and López 2011), as well as in O. Dieste's Requirements Engineering course at the European Master in Software Engineering. These questions are rather simple as compared to other sets of questions, e.g., (Gause and Weinberg 1989), being thus ideal to test the interview assistant in the early stages.

Please notice that the original list (Carrizo, Dieste, Juristo, and López 2011) contained some grammatical errors and typos. They have been fixed during the subsequent expert evaluations.

B Specification used in the chatbot

B.1 Display management system

B.1.1 Introduction

B.1.1.1 Purpose : Define the requirements of the Display Management System (DMS). This system will operate the electronic notice boards installed at the School of Computer Science. Notice boards are 46" LCD monitors displaying information (ads, institutional information, etc.) using photo or video formats.

Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames. $_{goal}$[The goal of this project is developing a system capable to manage the displayed contents over the Internet, so that an administrator may make changes from anywhere].

B.1.2 Overall description

B.1.2.1 Product perspective : Notice boards display the images and videos stored in a 500 GB internal hard disk drive. Images and videos can be transferred through a USB 2.0 or Firewire ports. SD cards can also be used. Monitors have 1 Gbps Ethernet network connection and Wi-Fi.

B.1.2.2 User characteristics : There are 3 user types:

- $_{user}$[Administrator]: They can access the system with no restriction.
- $_{user}$[Writer]: They can upload contents, but cannot delete existing ones.
- $_{user}$[Supervisor]: They can review the notice boards' contents, but cannot manage them (they cannot add, modify or remove images or videos).

B.1.2.3 Constraints The system shall $_{constraint}$[communicate with the panels through the Internet using each panel's IP or their broadcast IP].

B.1.3 Requirements

B.1.3.1 Functional requirements

FR-1 The system shall allow the $_{task}$[recording and management of users] who can access the application via a login and different levels of privileges. Access will be through e-mail address and password.

FR-2 The system shall allow the $_{task}$[upload of new content]. The contents may be images, videos and Flash animations. Uploaded new content must be associated with a title, description, some tags, duration, date of activation and expiration date.

FR-3 The system shall $_{task}$[display a list sorted by date (ascending or descending) of the last ten uploaded contents]. It will also show the rest of the contents if they exceed ten.

FR-4 The system shall allow the $_{task}$[edition of the characteristics of a newly uploaded content], that is, you can modify the title, description, tags, activation date and expiration.

FR-5 The system shall allow the $_{task}$[creation, edition and deletion of playlists], so that downloaded videos can be included in certain playlists. It will also be possible to $_{task}$[change the order of videos in a playlist].

FR-6 The system shall $_{task}$[manage the state of playlists]. A playlist can be "online" or "offline." Online playlists are sent to monitors to display videos. Offline playlists contain videos that are stored on the system but not displayed on the panels.

FR-7 The system shall allow the $_{task}$[preview of playlists].

FR-8 The system shall allow the $_{task}$[removal of material]. Content will be removed from both notice boards' disks and the application server's disk.

FR-9 The system shall $_{task}$[reveal the status of each panel], i.e.: if it is running, online, or if for any reason it is down or unreachable. It also will $_{task}$[show whether the panels have correctly received the contents] that have been sent to them.

FR-10 The system shall $_{task}$[report the disk capacity used and left], both in the panel's disks and the application server's disk.

FR-11 The system shall allow $_{task}$[downloading the contents stored on the server].

FR-12 The system shall $_{task}$[turn the panels on and off] (one in particular, or all at once) through the web.

B.1.3.2 Non-functional requirements

NFR-1 The system shall be accessible and usable.

NFR-2 The system shall meet W3C Web design standards.

NFR-3 The system shall comply with the corporate image of the School of Computer Science.

NFR-4 The application shall be available in English and Spanish. The default language is Spanish.

C Compilation of context-free questions

C.1 1st List

Authors: Gause & Weinberg (Gause and Weinberg 1989)

1. Who is the client?
2. What is a highly successful solution really worth to this client?
3. What is the real reason for wanting to solve this problem?
4. Should we use a single design team, or more than one?
5. Who should be on the team(s)?
6. How much time do we have for this project?
7. What is your trade-off between time and value?
8. Where else can the solution to this design problem be obtained?
9. Can we copy something that already exists?
10. What problems does this product solve?
11. What problems could this product create?
12. What environment is this product likely to encounter?
13. What kind of precision is required or desired in the product?
14. Am I asking you too many questions?
15. Do my questions seem relevant?
16. Are you the right person to answer these questions?
17. Are your answers official?
18. Is there anyone else who can give me useful answers?
19. Is there someplace I can go to see the environment in which this product will be used?
20. Is there anything else I should be asking you?
21. Is there anything you would like to ask me?
22. I notice that you hesitated a long time before answering that question. Is there something else we should know?

23. When I asked X about that, she said Y. Do you have any idea why she might have said Y?
24. I notice that you don't seem to agree with that reply. Would you tell us about that?
25. Are you comfortable with the process right now?
26. Is there any reason you don't feel you can answer freely?
27. What can you tell me about the other people on this project?
28. How do you feel about the other people working with us on this project?
29. Is there anybody we need on this project whom we don't have?
30. Is there anybody we have on this project whom we don't need?

C.2 2nd List

Authors: Lundeberg, Goldkuhl & Nilsson (Lundeberg, Goldkuhl, and Nilsson 1981)

1. What are the problems?
2. For each problem:
   (a) What is the real reason for wanting to solve this problem?
   (b) Can a solution to this problem be obtained elsewhere?
   (c) Which organizational goal is served by solving this problem?
   (d) How bad is the problem? (Quantify if possible)
   (e) How urgent is it?
3. Which stakeholders have which problems?
4. For each stakeholder/problem combination:
   (a) How much is it worth to this stakeholder to solve the problem?
   (b) How bad is it for the stakeholder if the problem is not solved?
   (c) How urgently should this problem be solved?
   (d) How bad is it if this problem is solved one year later?
   (e) What is the trade-off between time and value?

C.3 Third List

Author: Michael Bolton (Bolton n.d.)

1. Who is my client?
2. Are you my only client?
3. Who is the customer of the product?
4. Who are the other stakeholders?
5. What is my mission?
6. What else might be part of my mission?
7. What problems are you aware of that would threaten the value of this product or service?

D Appended paper

The following paper has been submitted to the 10th International Workshop on Requirements Engineering Education and Training (REET), co-located with the RE'20 conference. Notification of acceptance/rejection is scheduled for June 22nd, 2020.

Chatbot-based Interview Simulator: A Feasible Approach to Train Novice Requirements Engineers

Muhammad Laiq
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid
Campus Montegancedo, 28660 Boadilla del Monte, Spain
Email: m.laiq@alumnos.upm.es

Oscar Dieste
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid
Campus Montegancedo, 28660 Boadilla del Monte, Spain
Email: odieste@fi.upm.es

***Abstract*—*Introduction:* Although the interview is the most important and widely used requirements elicitation technique, novice engineers do not receive adequate training in Requirements Engineering (RE) courses.**

***Objectives:* Develop an AI-based interview simulator for helping novice requirements engineers in gaining interview skills.**

***Methods:* The research is based on the Design Science Methodology for Information Systems. Seven cycles have been planned; in each cycle, a proof of concept with additional features is created. Each cycle finishes with evaluation and improvement suggestions.**

***Results:* So far three development cycles have been completed. The simulator can answer context-free questions and provide information about RE concepts, e.g., goals, tasks, etc.**

***Conclusions:* The results have been promising. Research is still in progress. We plan to improve the simulator in the short term adding features like more complex conversation abilities and the capacity to handle different problem domains.**

***Index Terms*—Requirements elicitation, chatbots, assistants, interviews, interview training.**

I. Introduction

Requirements elicitation (REl) is a crucial phase of requirements engineering [1], which strongly influences the quality of final product [2]. Communication skills are fundamental to elicitation; hence people with strong communication skills are more effective in the requirements elicitation [3]. Users and customers should have enough skills to express their needs to the requirements engineers, and obviously, requirements engineers should be specialists in acquiring the necessary domain-specific knowledge.

REl comprises diverse activities and multiple techniques to perform these activities. There exist many requirements elicitation techniques such as *task analysis*, *questionnaires*, *domain analysis*, *interviews*, *observation*, etc. Elicitation techniques are selected depending on system type, purpose, time, budget, etc. However, interviews are the most widely used elicitation technique for requirements acquisition [4], and they are considered as one of the most effective elicitation techniques [5].

Interviews look easy to conduct because they are, in essence, a one-to-one conversation. However, an *effective* interview is not a free conversation. On the one hand, interviews require the realization of several tasks, e.g., scheduling, script preparation, and consolidation. On the other hand, their success significantly depends on the skill of the interviewer to know what are right questions and when to ask them, and from whom to ask [4]. Interviews are a soft skill but it is not acquired for free.

It is difficult for REl students to develop interview skills during their courses. The problems are multi-fold: (1) students usually have limited domain knowledge, (2) interview practice is time-consuming, and (3) large groups prevent instructors to conduct even a single interview with each student. At least in our experience, interview training in academia restricts to theory or short-time exercises.

Similar problems have been reported in other sciences. Fitzmaurice et al. [6] highlight that young doctors have little chances for interviewing patients to acquire interview skills. Yang and Evans [9] discuss the need for training for the use of a newly introduced educational application. Kowalski, Pavlovska, and Goldstein [10] explains the training of security personnel for effective communication.

An approach that has been explored by several researchers to improve the engineers' interviewing skills is to develop interview simulators, likewise e.g., airline pilots are trained in flight simulators. However, we are not aware that any of these simulators are currently available. It is even possible that they did not proceed beyond the research stage, as the natural language processing technology has developed at a rather slow pace in the past years. The situation has changed with the rise of computing power and the expansion of AI-enabled technologies, e.g., digital assistants such as chatbots are commonplace [7] for customer assistance in many web pages and traditional phone systems.

In this paper, we aim to develop an AI-based interview simulator for helping novice requirements engineers to acquire interview skills. More specifically, our research goals are:

1) Create a chatbot that engages in a natural conversation with requirements engineers.
2) Emphasize good interview practice during the conversation, e.g., using context-free questions [8], [9], making summaries [10], etc.
3) Introduce natural language pitfalls, e.g., ambiguity, incompleteness, etc. in the engineer-chatbot conversation.

4) Elaborate on a design theory for chatbots applied to REI interviews.

This research applies the *Design Science* [11] methodology, due to its requirement of developing a working software product. We have carried out three *Design Science* cycles and the results are promising. The chatbot is now able to answer questions about system goals, tasks, users, and constraints. The current version of the chatbot is publicly available for evaluation purposes (see section IV-C), although further research will be necessary to achieve our research goals.

The remainder of this paper is organized as follows: In section II, we provide the relevant background information. In section III, we introduce the *Design Science* methodology, that we use for the design and implementation of the assistant. Section IV summarizes the *Design Science* cycles carried out so far. In section IV-B, we discuss the simulator architecture and section IV-C provides detail about its implementation with IBM Watson technologies. In section V, we discuss a (very) preliminary evaluation of the simulator. Finally, in sections VI and VII, we report the threats to validity, conclusions and future works, respectively.

II. Background

We provide the relevant background information in the following sections. In section II-A, we review how different sciences have addressed the need that practitioners receive adequate interview training. In section II-B, we discuss the interview training process in RE. In the remaining sections, we discuss chatbot-related issues, i.e., the educational value of chatbots (section II-C), and the IBM's Watson technologies (section II-D).

A. Interview training

The need for adequate interview training has been raised in several disciplines, such as education, security and health.

Yang and Evans [12] discuss the need for training for the use of a newly introduced educational application. A university has introduced a system for managing library resources that helps teachers to generate reading lists and students to access those lists. The problem is that a significant number of users including students, teachers, and library staff do not know how to utilize service properly. They developed a chatbot that acts as a virtual tutor to instruct and train users to solve basic issues.

Kowalski, Pavlovska, and Goldstein [13] explain the training of security personnel in a company. Such training was mostly based on intranet e-learning, but traditional web-based learning processes have not been proved effective. This study used chatbot technologies to enhance security awareness training. The results of case studies conducted on the company suggest that the attitudes of the respondents appear to be more positive to security when chatbots are used instead of the current traditional e-learning security training courses.

Fitzmaurice et al. [6] highlights that young doctors do not have enough chances for interviewing patients to acquire interview skills. In general, students are supposed to interview many patients with acute and severe psychiatric disorders. Unfortunately, patients are reluctant to be interviewed by multiple students. To solve this problem, an online interactive e-learning simulator called 'Virtual Interviews for Students Interacting Online for Psychiatry (VISIOn)' has been developed. Students can use simulator anytime to learn about their profession and apply their learning outcomes in real life.

B. Interview training in RE

The problems associated to interview training suggest the need for some computerized support to the education of novice requirements engineers. Computer-Aided Software Engineering (CASE) tools have been applied to a diversity of software engineering in a variety of ways. Only a few tools have been specifically created for REI [10].

Gilvaz & do Prado Leite [14] presents an interview assistant to help software engineers during the interview process. The assistant uses Wetherbe's indirect questioning approach [15] for interviewing and provides automated support for the software engineers in eliciting necessary information for the information systems.

A business interview simulator has been developed by Bollweg, Shahriar, Stemmermann & Weber [16] as part of their Enterprise resource planning MOOC course. Their primary purpose in developing a simulator is to provide students with interviewing skills so that they can efficiently conduct interviews (face-to-face or online) with managers from real companies later in the technical phase of course. The simulator has been tested on 38 students enrolled in the course. Students were asked to fill questionnaires right after using the simulator. The results indicate that the simulator performed well technically to help students gaining interview skills as compared to traditional e-learning process.

Yamanaka and Komiya [17] illustrates a method to navigate interview-driven software requirements elicitation. The method uses the progress management table to keep users on track. To check the effectiveness of the proposed method an experiment was conducted on SE student with and without navigation method. As a result, they clarify that the user requirements elicitation work depends on the business knowledge and work experience of each individual.

Kato et al. [10] presents a model of navigating interview processes for eliciting requirements to help novice analysts. Their model stores expert analyst's data in the forms of thesaurus, state transition model, and template, then they use that information for the interview navigation process. The navigation process help analysts to specify SRSs in IEEE 830-1993 format. They validated their model by comparing the SRSs created by both novice and expert analyst and results confirm that the navigation method will help both novice and expert analysts as well. However, the development of supporting system based on the method is yet to be done.

Stanica et al. [18] developed a Virtual Reality (VR) Job Interview Simulator to help software engineers increase their job interview performances by practicing their hard (skills related to job) and soft (ocial competencies, communication skills, and personality-related traits) skills. The interview simulation process is performed using VR based environments. However, the product is still under development soon it will be validated with real candidates.

C. The educational value of chatbots

Chatbots have been widely used in different fields but their use in academia is limited despite their potential to support many educational activities. Yang and Evans [12] discuss three chatbots with the purpose: (1) to support the delivery of a Master's course simulation game, (2) to support the training and use of a newly introduced educational application, and (3) to improve the processing of help-desk requests within a university department. The results indicate that AI-based chatbots are quite useful in the field of education but still there are some challenges in the creation of chatbots i.e. chatbot development platforms are not originally designed to support educational activities.

D. IBM Watson technologies

IBM is one of the leading companies actively working to provide AI-based tools for different purposes. Watson is IBM's suite of enterprise-ready AI services, applications, and tooling [19]. IBM has developed many products under the umbrella of Watson technology such as Watson's Assistant, Watson's Discovery Service, Watson Natural Language Understanding, Watson Knowledge Studio, IBM cloud Function, etc.

We have chosen IBM Watson technologies to develop our interview simulator due to its capabilities to adopt AI techniques on different domains with high accuracy [20]. Our interview simulator uses specifically the products: (1) Watson Assistant, (2) Watson Discovery Service, (3) Watson Knowledge Studio, and (4) IBM cloud Functions. We describe them briefly below.

1) Watson Assistant: is IBM's artificial intelligence-based product that lets developers build, train, and deploy smart assistants for businesses. The assistants can be deployed standalone or part of any application. Unlike traditional chatbots that frustrate humans when complexity increases, the Watson Assistant is rather smart and provide external search support [21].

2) Watson Discovery: is an intelligence-based search technology with machine learning and natural language processing capabilities that work accurately to generate meaningful results when applied to huge enterprise data. Moreover, it also provides support to easily create cognitive, cloud-based exploration applications that uncover actionable insights embedded in unstructured data - including own proprietary data, as well as public and third-party data [22].

3) Watson Knowledge Studio: is a cloud-based application that lets developers and domain experts work together to create and train custom machine learning models that could be used to recognize concepts and relationships in unstructured text. Moreover, it also provides support to use the models in IBM Watson Discovery, IBM Watson Natural Language Understanding, and IBM Watson Explorer [23].

4) Cloud Functions: are Functions-as-a-Service platform based on Apache OpenWhisk, that runs application code without servers [24].

III. Methodology

AI-enabled technologies, particularly chatbots, have been used for interview support and training. Some efforts [10], [14], [16], [17] have been made in RE to provide support to requirements engineering during REl, but very little in what regards novice engineers' training and interviewing skills acquisition.

We aim to develop an interview simulator, in line with the works of [16], [6], [18]. As this research's goal is to design a new artifact, we will apply the *Design Science* methodology [11].

Design Science was created in the Information Systems (IS) discipline. IS involve people, organizations, and technology. Developing IS requires proper analysis of all connected entities. Hevner [11] argues that this discipline is characterized by two separate paradigms: behavior science and design science. The first one seeks the creation or justification of theories to explain or predict human behavior. The second one is the engineering approach focused and aims to build inventions that solve people's and organization's problems effectively and efficiently.

While the routine design is the application of existing knowledge to organizational problems, *Design Science* involves finding new solutions to previously unsolved problems or better and more efficient solutions to previously solved problems [25]. Our research matches one of these two situations, depending on each one's interpretation of the current state of the art.

Fig. 1 shows the design science framework for this research. The interview simulator is developed using an iterative approach, creating a sequence of proof-of-concept artifacts. Artifacts are refined through the feedback received from different evaluation mechanisms. Once evaluated, the information flows back both to the environment (reporting about e.g., feasibility, scope limitations, etc.) and to the knowledge base (increasing our knowledge about the research area).

IV. Results

Seven research cycles have been planned, scheduled to finish by July 2020. So far, three cycles have been completed, with promising results. The later cycles are briefly outlined in section VII. In this section, we will report the work carried out in the first cycles, followed by the chatbot architecture and implementation. The progress of the research can be followed "live" at https://www.overleaf.com/read/gcgqmpzzgbzz.

A. Design Science process

1) First research cycle: It was focused on designing a basic chatbot with the following features:

- Accept as input one IEEE-Std-830 [26] requirements specification. We have chosen a short specification available at the nlrpBENCH: The Display Management System.

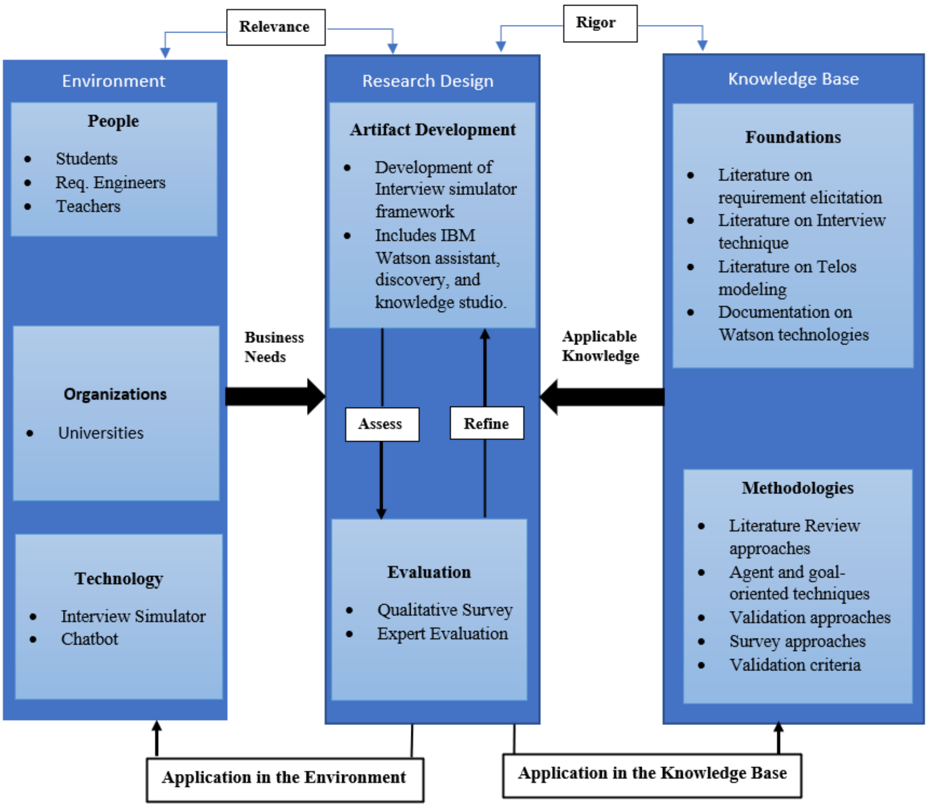


Fig. 1. Research design science framework (based on [11]).

- Answer queries related to **goals**, **tasks**, and **users** of the system described in the requirements specification.

This second point deserves clarification. The design of the interview simulator relies on a RE concept model. There are several RE model proposals, e.g., Sutcliffe and Maiden [27, Fig. 9], Diamantopoulos et al., [28, Fig. 2], the FRISCO Report [29], etc. We have not chosen anyone yet; the most appropriate model will surface along with the research work.

We have defined a basic RE concept model, which will be extended in later research cycles. This model contains three concepts: `goals`, `tasks` and `users`, along with two relationships (`tasks operationalize goals` and `users perfom task`). This RE concept model can be seen as an *L1*-model based on a *L0*-model (meta-model) with just two components: *classes* and *relationships*. Our RE model can be easily implemented in many tools (such as the Knowledge Studio, see next cycle), and formalized using a variety of lenguages, e.g., TELOS [30].

The features were implemented using the Watson Assistant, Watson Discovery, and IBM Cloud functions. Watson Assistant provides support to identify the interviewee's intents and the cloud function runs queries on the discovery service. The discovery service returns the information related to user intent, due to its ability to assign meaning to text documents.

The first cycle's proof of concept was evaluated using *Expert evaluation*. The second author played this role. The first version of the simulator received extensive criticism. The expert raised issues about the definition of almost all intents, the usage of entities (see the technical details in the sections below), and the analysis of the SRS document. The feedback comments were addressed in the next cycle.

2) Second research cycle: This cycle aimed at making the assistant more robust; thus, we didn't introduce new features but improve the current ones. The main lines of action were the introduction of Watson Knowledge Studio and its machine learning model. The model was deployed on Watson Discovery. The Assistant and the Cloud functions were updated to accommodate the changes.

Similarly, the second cycle was evaluated using expert analysis and feedback was recorded to be incorporated in the third cycle. The main issues raised by the expert were: (1) incomplete responses, (2) the appearance of irrelevant text in the responses, (3) issues related to naming conventions, and errors in the definition of entities.

3) Third research cycle: The third cycle is focused on adding more intents and improving the performance of the machine learning model with the use of a bigger training dataset. To speed up the annotation process for model training and reduce the human annotator workload, we created a domain-specific dictionary for each entity type (goals, tasks, etc.). Dictionaries are equivalent to *L2*-models, that is, instances of the *L1*-model (the RE concept model) which contains domain-specific concepts. This third cycle is evaluated in section V.

B. Simulator architecture

Fig. 2 depicts the interview simulator architecture. The user initiates the conversation and interacts with the simulator using the provided web interface (arrow labeled as ”2”). The Watson Assistant collects the inputs (arrow ”3”) and identifies the intent. For instance, when the user enters the question:

`What are the system goals?` (1)

the Watson Assistant recognizes the question due to examples provided previously, and identifies that the requirements engineer wants to find out about system `#goals`.

When an intent is identified, the corresponding information is retrieved. However, the information about the RE concepts is not readily available to the Assistant yet. As indicated before, such information is represented as a regular IEEE-Std-830 requirements specification (”target SRS”), usually in PDF format. The information should be extracted by Watson from this document (the SRS is available in this link). This is possible thanks to the Watson Discovery service’s document understanding technology.

Information extraction is triggered by the Watson Assistant using IBM Cloud functions (arrows ”4” and ”5”). In the case of the question 1, the answer provided by the simulator is:

```
The goal of this project is
developing a system capable
to manage the displayed contents
over the Internet, so that an
administrator may make changes
from anywhere.
```

Typos and syntactic errors (e.g., `The goal of this project is developing` instead of `The goal of this project is to develop`) appear because the ”target SRS” contains such an error. Fixing the specification would fix the answer.

Watson Discovery’s ability to understand complex documents such as SRSs is limited. To help Discovery in this task, we have implemented the RE concept model (*L1*) in Knowledge Studio. The implementation is straightforward because Knowledge Studio’s model is similar to our *L0* model.

Fig. 3 shows how Knowledge Studio works. We have to annotate domain-specific documents (”training data”) according to the RE concept model, and the AI generalizes to new documents. More elaborated operations, like k-fold cross-validations, can be performed in this environment. To ease annotation, we have created a dictionary containing domain-specific concepts (about the ”target SRS” problem domain).

The machine learning model trained in Knowledge Studio was deployed to Watson Discovery (arrow ”1”). The features of the simulator do not change with this trained model. However, Discovery’s abilities to uncover the right information are greatly improved.

The third cycle proof-of-concept is live at https://ibm.co/2MgoGDa. Future versions will be deployed in different URLs to maintain traceability between research cycles and artifacts.

C. Simulator implementation

We provide some further technical details in this section about the simulator implementation.

1) The Watson assistant: The Watson Assistant consists of intents, entities, and dialogues. The purpose of these components is:

- Intents: Intents are purposes or goals that identify which action is required from the assistant.
- Entities: Entities are like nouns that represent information in the user input that is relevant to the user’s purpose.
- Dialogues: The dialogues are used to interact with the user. They provides relevant response to the user based on the identified intent or context from application.

We have used the following intents, entities, and dialogues during the implementation:

- Intents: We have created six intents (so far): `#goal`, `#users`, `#tasks`, `#constraints`, `#benefit`, and `#integration`. Moreover, to identify intents, Watson needs to have some ”user examples”. User examples are domain-specific questions that will be asked from the assistant in different ways. We have provided Watson the examples using the context-free questions using our common sense and guidelines proposed by [32], [8].
- Entities: Similarly, entities have been created for each intent and have been used for annotation of the user examples.Some examples are: `user`, `goal`, `system`, `constraint`, etc.
- Dialogues: The Assistant consists of multiple dialogues for interacting with users. Depending on user input and intent identification, a dialog will be executed. For instance, the `goal` dialog will be executed if the `#goal` intent is identified. Once a dialogue is invoked, a query to Discovery will be sent using a Cloud function, and the results received from Discovery will be shown to the user, as described in section IV-B.

2) Knowledge studio: The training process includes the identification of entities, co-references, and relationships of interest in documents. We have collected the documents describing four systems related to displays (i.e display management systems). The process of creating a machine learning model in the Knowledge Studio is:

1) We define the model (in Watson’s terms, the ”type system”) using the entity and relationship concepts.
2) We upload the training data set and match words with the model concepts. For example, the real-world thing ”writer” is annotated as a ”user” entity. ”Writer creates scenes” would be annotated a two entities (”writer” and ”scene”) and one relationship ”create”). The annotated documents represent the ”ground truth”.
3) The Knowledge Studio uses the ground truth to train the model.

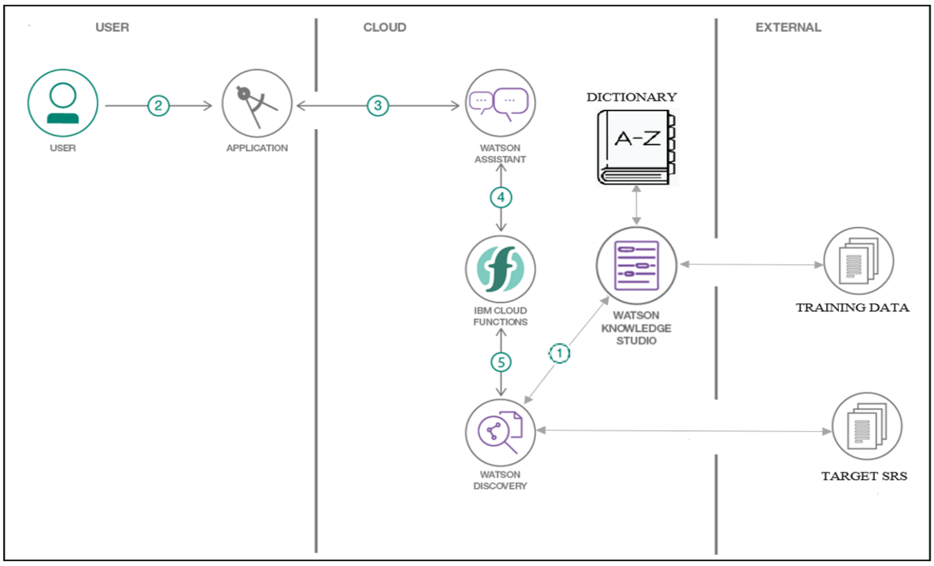


Fig. 2. Chatbot Architecture (following [31]).

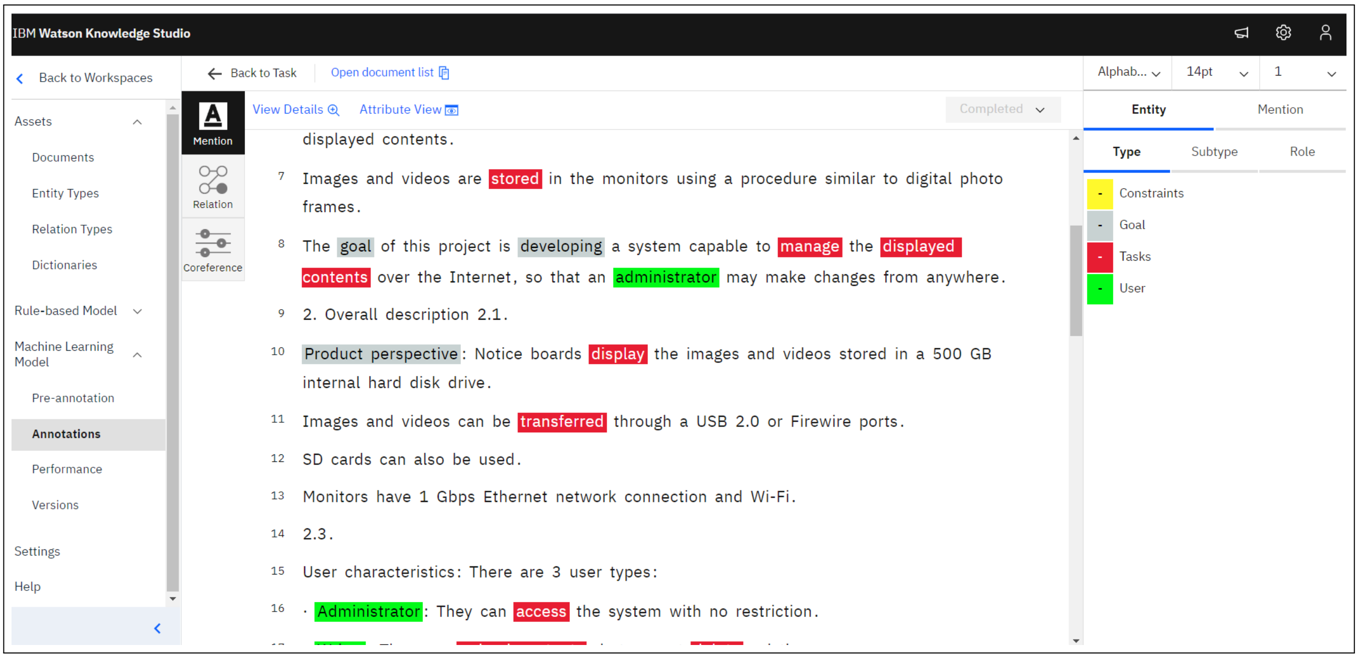


Fig. 3. Knowledge Studio's annotation screen.

3) Discovery service: The discovery service contains the "target SRS". The "target SRS" is also annotated, but annotation does not require much effort. Once the document is uploaded, it is only necessary to identify the document headings. The document is thus broken down into sub-parts that can be automatically processed by Discovery's AI. The model trained in Knowledge Studio was deployed into Discovery (it is a built-in feature) for better text recognition.

4) Cloud functions: Our prototype uses node.js -based IBM cloud functions. Cloud functions behave as an API to send/receive requests. In our case, Watson Assistant makes requests to the API, and the API responds after querying on Watson Discovery.

V. Chatbot Evaluation

In the first and second research cycles, we have used *Expert evaluation* as recommended by Kitchenham et al. [33]. In this section, we will describe the evaluation conducted during the third cycle, where the expert has been replaced by a student. This represents an intermediate step towards the application of more formal evaluation procedures. The open character of the feedback provided by the expert has been limited in the case of the student evaluation. Instead of submitting his/her opinion, the student fills out two surveys:

- Conversation survey: https://bit.ly/3gcQOVt
- Opinion survey: https://bit.ly/3e7LgcR

The student was selected by convenience. He is a first-year MSc student with the Universidad Politécnica de Madrid. He has taken a Requirements Engineering course in the semester previous to this evaluation, including (short) interview training.

To facilitate the process of evaluation, context-free questions were given to the student. The simulator-student conversation lasted 5 minutes approximately. After interacting with the simulator, the student was asked to fill the surveys.

The first survey aims to assess the degree to which the student has acquired knowledge about the future software system during the interaction with the simulator. This survey contains sentences, e.g., *The system shall allow the upload of new content*, that are either *true* or *false*. The student was able to answer almost all (9 out of 10) questions correctly. This result shall be taken with obvious caution due that (1) just one student participated in the evaluation and (2) the Display Management System is a rather simple system.

The second survey collects the student's impressions regarding the interaction with the simulator (not the knowledge acquired). The responses are given on a 5-point Likert scale. Quite not surprisingly, the simulator has received a very good rating (most of the responses are 4's and 5s'). Further evaluation efforts are necessary.

Both surveys include one open-ended question. The student suggested that the simulator should answer questions more precisely; in the current version, all information related to one user intent is displayed at once. The simulator should also provide support for clarification questions. Student suggestions have been recorded and will be incorporated in the next cycles. Most of the suggestions match our future work; hence the research project will not experience changes in the short term.

VI. Threats to Validity

The threats to validity are described according to Runeson et al. [34, p. 71 – 73], whom in turn are based on Yin's seminal work [35]. There are four types of validity: (1) Construct validity, (2) internal validity, (3) external validity, and (4) reliability.

A. Construct validity

Construct validity refers to the degree of concordance between the researchers' perspective and the objective reality of the phenomenon under study. This threat mainly impacts the instruments used to measure the effectiveness of the interview simulator, such as the surveys described in section V. The mitigation strategy is provided by the cyclical arrangement of the *Design Science* methodology. The instruments are examined after each cycle using open interviews with study participants, and improvements are introduced before the next cycle.

B. Internal validity

Internal validity refers to incorrectly assign correlations to the category of causal relationships. This threat operates in the achievement of goals 1-3, i.e., pretending that the engineer-simulator conversation is a "natural" conversation when it is not. The mitigation strategy rests on increasing the involvement of RE students and instructors in the evaluations of the proofs of concept. Cycles 1-2 use expert evaluation, cycles 3-4 student surveys and cycles 5-6 small-size experiments with students and instructors. The simulator will be used during one semester in a master-level RE course. The number and diversity of checks increase our confidence in the simulator's capabilities.

C. External validity

External validity aims that the simulator will work in contexts different from the one where it has been initially designed. This threat operates in two directions:

- The interview simulator applies to different problem domains. The modular structure of the simulator (Knowledge studio, Discovery) supports defining requirements problems in different domains. One research cycle has been specifically devoted to testing the simulator in different domains.
- The design theory (goal 4) enables the creation of new simulators, probably using different elicitation techniques. Unfortunately, this research will not be able to address this threat. We plan to open the simulator code to the community as soon as the 1-semester live test has finished (March 2021) with satisfactory results. External replication will be possible since then.

D. Reliability

Reliability seeks that our research is described with enough detail to enable replication by external researchers. In this regard, the minute detail of the research is publicly available (see footnote 1). The simulator code will be publicly available in March 2021.

VII. Conclusion and Future Research

The research goals that we have stated at the beginning of this paper have not been achieved yet; this paper reports work in progress. However, we have obtained some significant results that will be extended before the project end:

- The interview simulator understands context-free questions, retrieving the right information. It also has some basic social abilities, and the capacity to recognize the context, e.g., provide information about goals without the questions explicitly mention them.
- The simulator's knowledge base has been defined at 3 levels: (1) *L0*, i.e., classes and relationships, (2) *L1*, containing RE-related concepts, and (3) *L2*, including domain-specific concepts. This knowledge base is easy to implement in Watson using the Knowledge Studio and Assistant services.

- Likewise, the knowledge base is easy to formalize using RE languages, e.g., TELOS [30]. This is the starting point to elaborate on a design theory for the construction of REI assistants, e.g., for different elicitation techniques such as surveys.
- The modular structure of the knowledge base leverages the simulator's ability to maintain conversations in different domains. Code changes will not be likely required.

The forthcoming research cycles will address the following requirements:

- Cycle 4: Provide partial instead of complete information to the requirements engineer.
- Cycle 5: Improve the conversation abilities, allowing the interview simulator to understand engineers' meta-questions, such as "is there anything else", "am I correct", or summaries.
- Cycle 6: Introduce natural language pitfalls, e.g., ambiguity, incompleteness, etc.
- Cycle 7: Apply the interview simulator to different problem domains.

The interview simulator will be tested in a real RE course in the academic year 2020-2021. Once it proves effective in the classroom, it will be opened to the RE community for free use and improvement.

ACKNOWLEDGMENTS

This research has been funded by the FEDER/Spanish Ministry of Science and Innovation Research State Agency, Project PGC2018-097265-B-I00. It has also received support from the Universidad Politécnica de Madrid, Project IE1819.1002.

REFERENCES

[1] I. Sommerville and P. Sawyer, *Requirements engineering: a good practice guide*. John Wiley & Sons, Inc., 1997.
[2] C. Wohlin *et al.*, *Engineering and managing software requirements*. Springer Science & Business Media, 2005.
[3] F. Anwar, R. Razali, and K. Ahmad, "Achieving effective communication during requirements elicitation-a conceptual framework," in *International Conference on Software Engineering and Computer Systems*. Springer, 2011, pp. 600–610.
[4] D. Zowghi and C. Coulin, "Requirements elicitation: A survey of techniques, approaches, and tools," in *Engineering and managing software requirements*. Springer, 2005, pp. 19–46.
[5] A. Davis, O. Dieste, A. Hickey, N. Juristo, and A. M. Moreno, "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review," in *14th IEEE International Requirements Engineering Conference (RE'06)*. IEEE, 2006, pp. 179–188.
[6] B. Fitzmaurice, K. Armstrong, V. Carroll, D. Dagger, and M. Gill, "Virtual interviews for students interacting online for psychiatry (vision): a novel resource for learning clinical interview skills," *Psychiatric Bulletin*, vol. 31, no. 6, pp. 218–220, 2007.
[7] A. Maedche, C. Legner, A. Benlian, B. Berger, H. Gimpel, T. Hess, O. Hinz, S. Morana, and M. Söllner, "Ai-based digital assistants," *Business & Information Systems Engineering*, vol. 61, no. 4, pp. 535–544, 2019.
[8] D. Gause and G. Weinberg, *Exploring Requirements: Quality Before Design*. Dorset House Pub., 1989.
[9] K. Wiegers and J. Beatty, *Software Requirements*, ser. Developer Best Practices. Pearson Education, 2013.
[10] J. Kato, S. Komiya, M. Saeki, A. Ohnishi, M. Nagata, S. Yamamoto, and H. Horai, "A model for navigating interview processes in requirements elicitation," in *Proceedings Eighth Asia-Pacific Software Engineering Conference*. IEEE, 2001, pp. 141–148.
[11] A. Hevner and S. Chatterjee, "Design science research in information systems," in *Design research in information systems*. Springer, 2010, pp. 9–22.
[12] S. Yang and C. Evans, "Opportunities and challenges in using ai chatbots in higher education," in *Proceedings of the 2019 3rd International Conference on Education and E-Learning*, 2019, pp. 79–83.
[13] S. Kowalski, K. Pavlovska, and M. Goldstein, "Two case studies in using chatbots for security training," in *IFIP World Conference on Information Security Education*. Springer, 2009, pp. 265–272.
[14] A. P. P. Gilvaz and J. S. do Prado Leite, "Faes: A case tool for information acquisition," in *Proceedings Seventh International Workshop on Computer-Aided Software Engineering*. IEEE, 1995, pp. 260–269.
[15] J. C. Wetherbe, "Executive information requirements: getting it right," *Mis Quarterly*, pp. 51–65, 1991.
[16] L. Bollweg, A. Shahriar, R. Stemmermann, and P. Weber, "Who am i, and if so how many? a multi-entity chatbot business-interview simulator for individualized practical assignments in moocs," in *EdMedia+ Innovate Learning*. Association for the Advancement of Computing in Education (AACE), 2019, pp. 928–936.
[17] T. Yamanaka and S. Komiya, "A method for navigating interview-driven software requirements elicitation work: Effectiveness evaluation of the method from the viewpoint of efficiency," *International Journal Of Systems Applications, Engineering & Development*, vol. 5, 2011.
[18] I. Stanica, M.-I. Dascalu, C. N. Bodea, and A. D. B. Moldoveanu, "Vr job interview simulator: where virtual reality meets artificial intelligence for education," in *2018 Zooming innovation in consumer technologies conference (ZINC)*. IEEE, 2018, pp. 9–12.
[19] About watson. [Online]. Available: https://www.ibm.com/watson/about
[20] N. Therkelsen-Terry. Why watson? more accuracy. less training time. insanely better cx results. [Online]. Available: https://www.ibm.com/blogs/watson/2017/09/more-accuracy-less-training-time-better-cx-results-with-watson/
[21] M. Miller. About watson assitant. [Online]. Available: https://cloud.ibm.com/docs/assistant?topic=assistant-index
[22] N. Bleiel. About discovery. [Online]. Available: https://cloud.ibm.com/docs/discovery?topic=discovery-about
[23] Watson knowledge studio. [Online]. Available: https://www.ibm.com/cloud/watson-knowledge-studio
[24] Cloud functions. [Online]. Available: https://www.ibm.com/cloud/functions
[25] S. Karmokar and H. Singh, "Improving the website design process for smes: A design science perspective." in *CONF-IRM*, 2012, p. 26.
[26] "Ieee recommended practice for software requirements specifications," *IEEE Std 830-1998*, pp. 1–40, 1998.
[27] A. Sutcliffe and N. Maiden, "The domain theory for requirements engineering," *IEEE Transactions on Software Engineering*, vol. 24, no. 3, pp. 174–196, 1998.
[28] T. Diamantopoulos, M. Roth, A. Symeonidis, and E. Klein, "Software requirements as an application domain for natural language processing," *Language Resources and Evaluation*, vol. 51, no. 2, pp. 495–524, 2017.
[29] W. Hesse and A. Verrijn-Stuart, "Towards a theory of information systems: The frisco approach," *FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS*, vol. 67, p. 81, 2001.
[30] M. Koubarakis, A. Borgida, P. Constantopoulos, M. Doerr, M. Jarke, M. A. Jeusfeld, J. Mylopoulos, and D. Plexousakis, "A retrospective on telos as a metamodeling language for requirements engineering," *Requirements Engineering*, pp. 1–23, 2020.
[31] R. Hagarty. Combine watson knowledge studio and watson discovery. [Online]. Available: https://github.com/IBM/watson-discovery-food-reviews
[32] G. Kotonya and I. Sommerville, *Requirements engineering: processes and techniques*, ser. Worldwide series in computer science. J. Wiley, 1998.
[33] B. A. Kitchenham and S. G. Linkman, "Desmet : A method for evaluating software engineering methods and tools," 2000.
[34] P. Runeson, M. Höst, A. Rainer, and B. Regnell, *Case Study Research in Software Engineering: Guidelines and Examples*. John Wiley and Sons Ltd., April 2012.
[35] R. Yin, "Case study research: Design and methods [kindle dx version]," Retrieved from http://www.amazon.com, 2009.

E Opinion Survey for third cycle

The following survey form has been used during the evaluation of the third cycle; below we show the results of the opinion survey.

Interview Simulator Evaluation

This form is used for evaluation of AI based Chatbot that simulates the interview for Novice Requirements Engineers.

The Chatbot offered convincing and natural interaction. *

1	2	3	4	5
○	○	○	◉	○

The Chatbot was able to answer questions correctly. *

1	2	3	4	5
○	○	○	○	◉

Learning about interview skills using Chatbot was fun. *

1	2	3	4	5
○	○	○	◉	○

It was easy to use and understand. *

1	2	3	4	5
○	○	◉	○	○

I think Chatbot will be helpful in gaining interview skills. *

1	2	3	4	5
○	○	○	◉	○

What's your opinion about the interaction with the simulator.

It was a good experience. but it needs improvement like short answers, summaries, maybe conversation tone requires changes like natural conversation.

F Conversation Survey for third cycle

In addition to the opinion survey the following survey form also has been used during the evaluation of the third cycle; below we show the results of the conversational survey.

Interview Simulator Evaluation

This is the second evaluation form used for AI based simulator.
In order to fill the form candidates are requested to use and understand the simulator answers carefully. Questions are related to the concepts learned with the assistant. For instance, goals, tasks, users, and constraints of the system.

Based on your understanding of display management system specification after using simulator, decide whether the given statements are True or False.

The system shall allow the upload of new content. *

- (•) True
- ( ) False

The system shall not reveal the status of each panel. *

- (•) True
- ( ) False

Default language of system should be "Spanish" *

- (•) True
- ( ) False

System should provides support for "Italian "language as well. *

- ( ) True
- (•) False

The goal of this project is to develop a system capable to manage the displayed contents over the Internet. *

- (•) True
- ( ) False

The system shall allow the preview of playlists. *

- (•) Ture
- ( ) False

The uploaded content may be PDF. *

- ( ) Ture
- (•) False

Writer can upload contents, but cannot delete existing ones. *

- (x) True
- ( ) False

One of the goal of system is to help students to manage their workload.

- ( ) True
- (x) False

"Supervisor" can not review the notice boards content. *

- ( ) True
- (x) False

What's your opinion about the information that the simulators gave.

The simulator response was very good. Simulator gave answers to all questions somehow relevant but requirements listed were too long for each question. The simulator should provide answers to question with precise answers. Moreover, it should also provide support to questions related to clarification.

G Opinion Survey for fourth cycle

The following survey form has been used during the evaluation of the fourth cycle; below we show the results of the opinion survey.

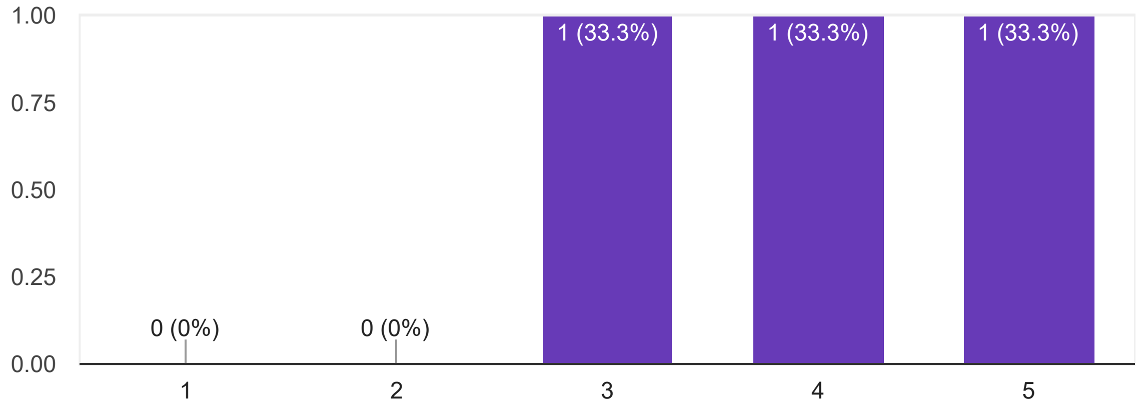
Interview Simulator Evaluation

3 responses

Publish analytics

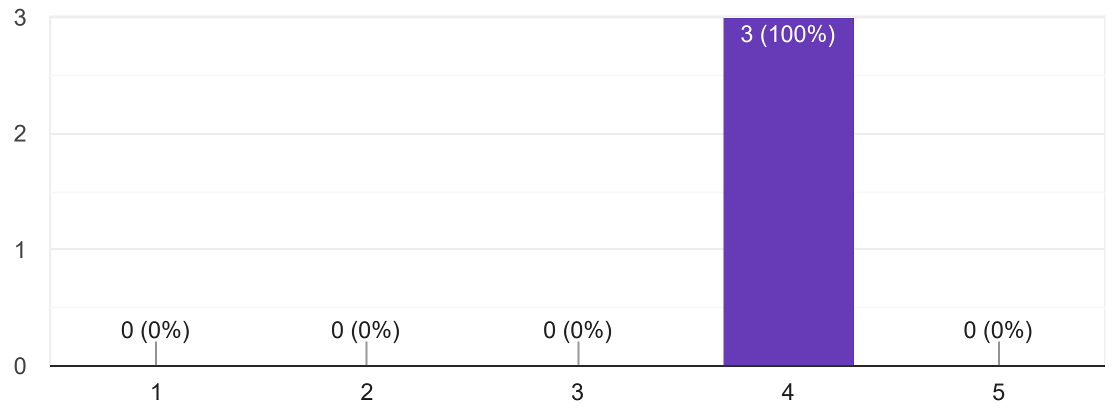
The Chatbot offered convincing and natural interaction.

3 responses



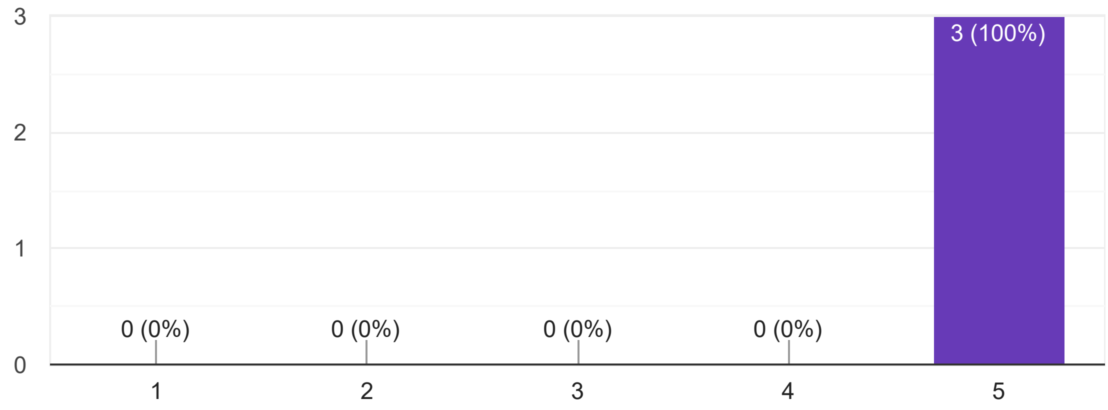
The Chatbot was able to answer questions correctly.

3 responses



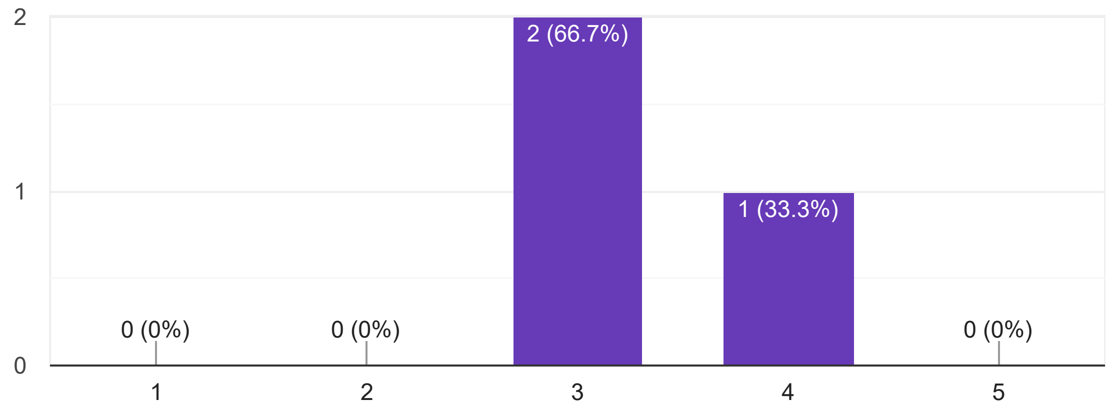
Learning about interview skills using Chatbot was fun.

3 responses



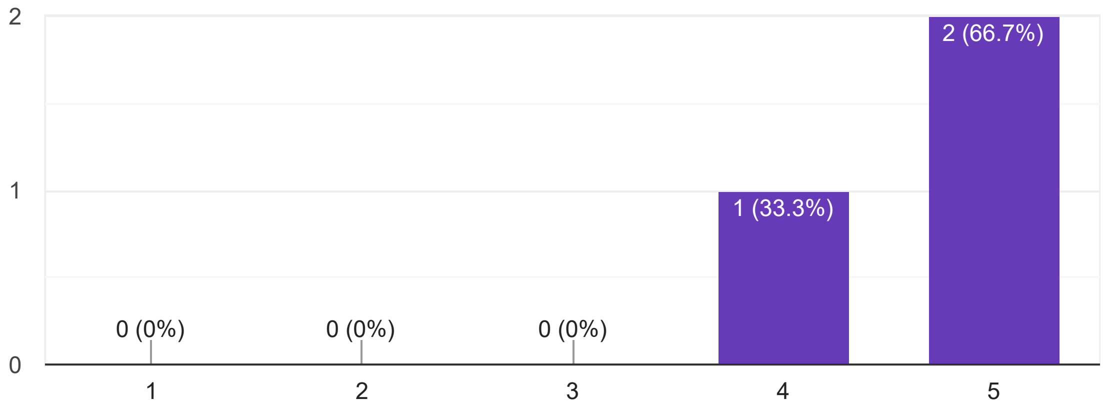
It was easy to use and understand.

3 responses



I think Chatbot will be helpful in gaining interview skills.

3 responses



What's your opinion about the interaction with the simulator.

3 responses

interaction was limited it should have support to ask question such as more info about specific task, and maybe summary at the end of conversation, so that user can see it at one place rather scrolling.

It is my second time with this tool and i have seen many improvements like (simulator highlighted the important words and this time the answers were not so long). But still, it needs improvement like simulator including irrelevant words in answers.

The simulator is user friendly and provides all the answers in an integrated way. The question was defined in precise manner. Satisfied in using this simulator.

H Conversation Survey for fourth cycle

In addition to the opinion survey the following survey form also has been used during the evaluation of the fourth cycle; below we show the results of the conversational survey.

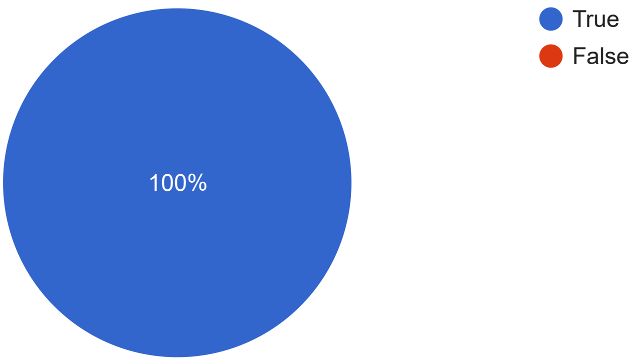
Interview Simulator Evaluation

3 responses

Publish analytics

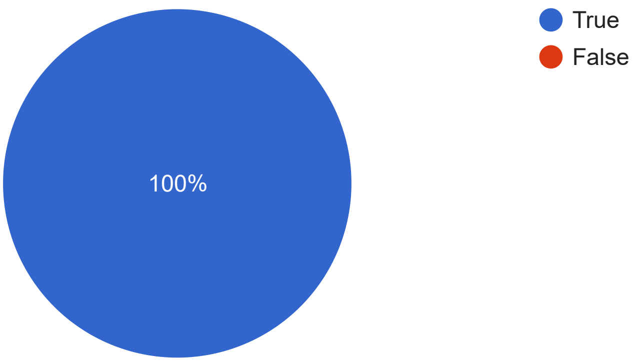
Notice boards are used for displaying information such as ads, institutional information, etc.

3 responses



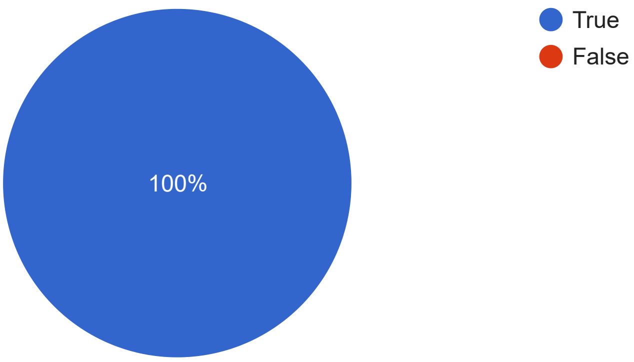
Access to system is possible using e-mail address and password .

3 responses



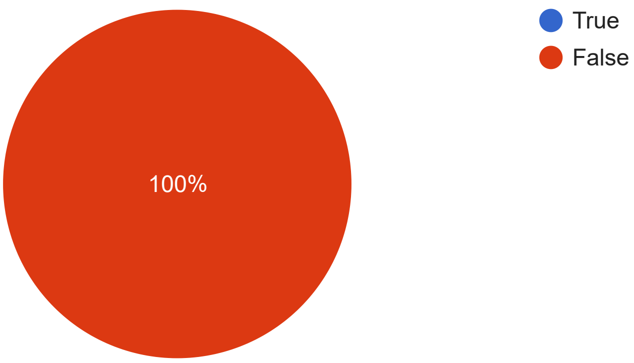
Notice boards display the images and videos stored in a GB internal hard disk drive.

3 responses



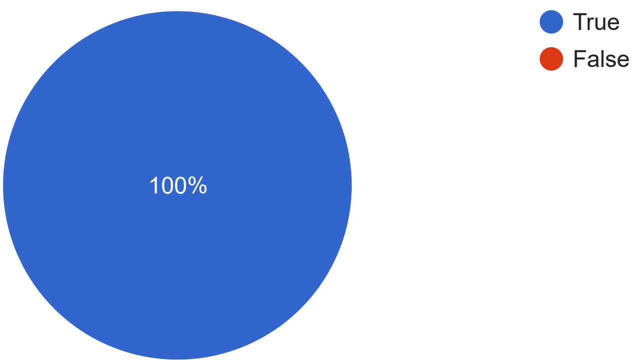
The system shall not reveal the status of each panel.

3 responses



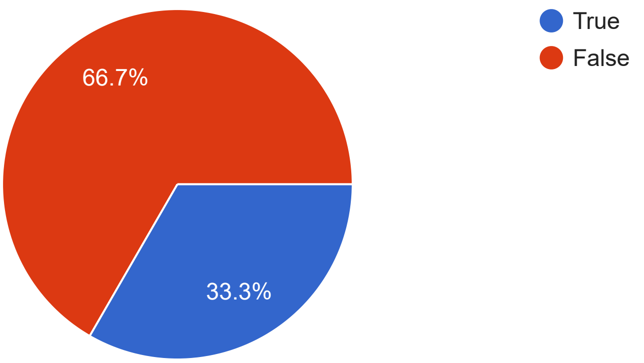
Default language of system should be "Spanish"

3 responses



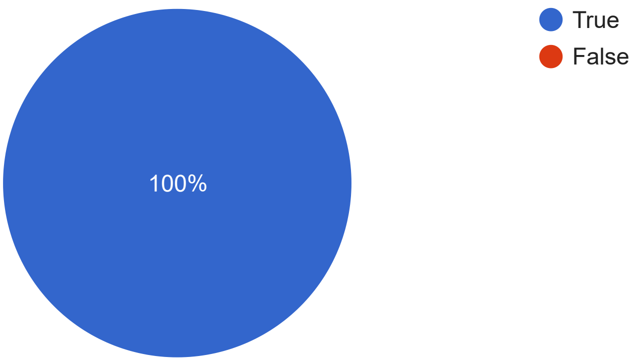
System should provides support for "Italian "language as well.

3 responses



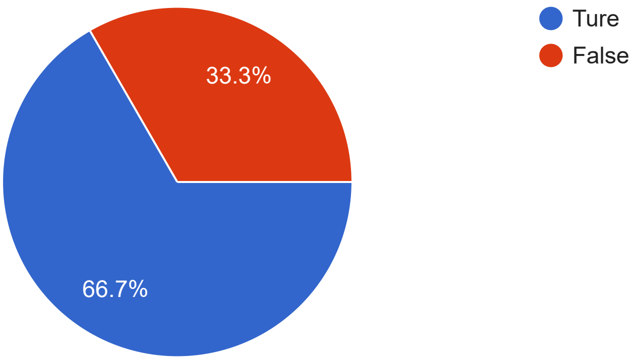
The goal of this project is to develop a system capable to manage the displayed contents over the Internet.

3 responses



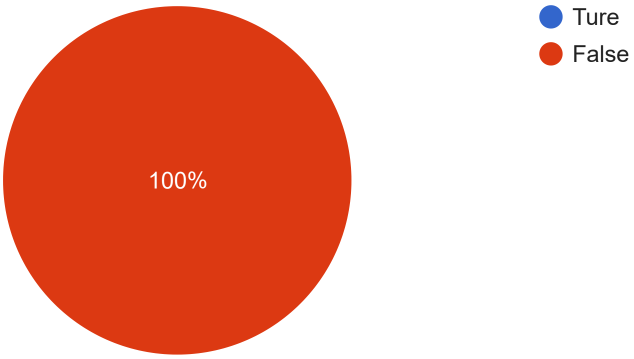
The system shall allow the preview of playlists.

3 responses



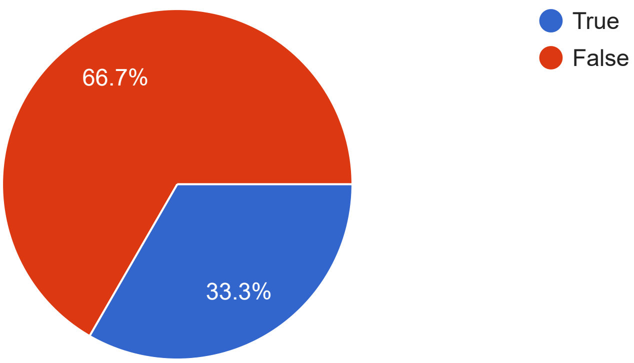
The uploaded content may be PDF.

3 responses



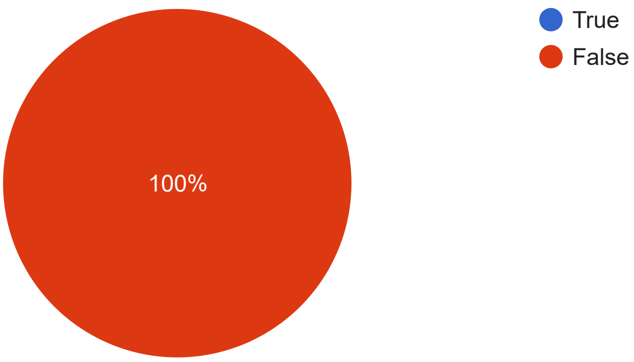
Writer can upload contents, but cannot delete existing ones.

3 responses



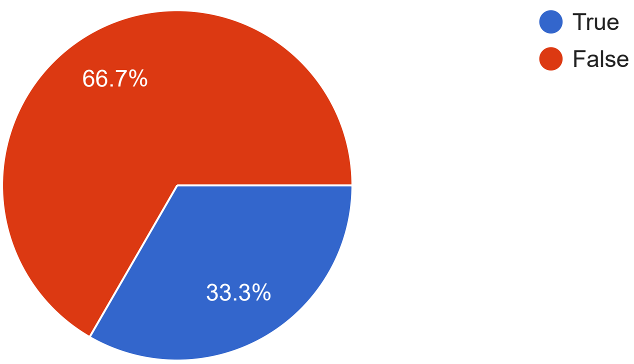
One of the goal of system is to help students to manage their workload.

3 responses



"Supervisor" can not review the notice boards content.

3 responses



What's your opinion about the information, simulator provides related to system requirements.

3 responses

The simulator provides brief and straight related to question. It focuses on Images and videos that are stored in the monitors also tell the procedure of storage.

well, overall info was fine but there should be support to ask question like more information about tasks or goals..and maybe detail about specific feature

Some time simulator highlights the irrelevant words e.g. (Contain and simulator just highlighted "on" inside this contain). Moreover, In some answers simulator include irrelevant words which are not appropriate for a particular question

I Students SRSs

The following SRSs have been created by students during the evaluation of a simulator in the fifth cycle.

Chatbot-Interview Simulator

This form is used to create Software Requirements Specification (SRS) for Display Management System using AI-based interview simulator. This form is aimed at novice requirements engineers, SRS is based on IEEE-830 format. Data will be used for the evaluation of system. Please provide as much detail as possible.

Purpose of System *

Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames. The goal of this project is developing a system capable to manage the displayed contents over the Internet, so that an administrator may make changes from anywhere.

User Characteristics *

User characteristics: There are user types: • Administrator: They can access the system with no restriction. Writer: They can upload contents, but cannot delete existing ones. • Supervisor: They can review the notice boards' contents, but cannot manage them (they cannot add, modify or remove images or videos)

Functional Requirements of System *

SRS: Display management system Introduction Purpose: Define the requirements of the Display Management System (DMS). This system will operate the electronic notice boards installed at the School of Computer Science.

The system shall allow the creation, edition and deletion of playlists, so that downloaded videos can be included in certain playlists. It will also be possible to change the order of videos in a playlist.

It also will show whether the panels have correctly received the contents that have been sent to them. The system shall report the disk capacity used and left, both in the panel's disks and the application server's disk.

Offline playlists contain videos that are stored on the system but not displayed on the panels. The system shall allow the preview of playlists. The system shall allow the removal of material. Content will be removed from both notice boards' disks and the application server's disk.

The system shall allow the recording and management of users who can access the application via a login and different levels of privileges. Access will be through e-mail address and password.

Access will be through e-mail address and password. The system shall allow the upload of new content. The contents may be images, videos and Flash animations. Uploaded new content must be associated with a title, description, some tags, duration, date of activation and expiration date.

Non-Functional Requirements of System *

The system shall meet W Web design standards. The system shall comply with the corporate image of the School of Computer Science. The application shall be available in English and Spanish. The default language is Spanish.

The system shall communicate with the panels through the Internet using each panel's IP or their broadcast IP. The system shall be accessible and usable. The system shall meet W Web design standards.

Any integration with other systems

Product perspective: Notice boards display the images and videos stored in a GB internal hard disk drive. Images and videos can be transferred through a USB or Firewire ports. SD cards can also be used. Monitors have Gbps Ethernet network connection and Wi-Fi.

Any benefits provided by system

Notice boards are ' LCD monitors displaying information (ads, institutional information, etc.) using photo or video formats. Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames.

Anything else

Feedback: (1) Any issues you noticed while interacting with the system. (2) Any suggestions. *

I am using the system for the third time this time system is much improved and this time I didn't notice any issue.e.

Based on your interaction with system , how will you rate the system from 1 to 5 (5 being best)? *

1	2	3	4	5
○	○	○	○	◉

Will you recommend your colleagues to use our system for acquiring requirements elicitation skills? *

- ◉ Yes
- ○ No
- ○ Maybe

Chatbot-Interview Simulator

This form is used to create Software Requirements Specification (SRS) for Display Management System using AI-based interview simulator. This form is aimed at novice requirements engineers, SRS is based on IEEE-830 format. Data will be used for the evaluation of system. Please provide as much detail as possible.

Purpose of System *

The goal of this project is developing a system capable to manage the displayed contents over the Internet, so that an administrator may make changes from anywhere. Images and videos are stored in the monitors using a procedure similar to digital photo frames.

User Characteristics *

User characteristics: There are user types: • Administrator: They can access the system with no restriction. Writer: They can upload contents, but cannot delete existing ones. • Supervisor: They can review the notice boards' contents, but cannot manage them (they cannot add, modify or remove images or videos).

Functional Requirements of System *

FN-1: The system shall allow the creation, edition and deletion of playlists, so that downloaded videos can be included in certain playlists. It will also be possible to change the order of videos in a playlist.
FN-2: Access will be through e-mail address and password. The system shall allow the upload of new content. The contents may be images, videos and Flash animations. Uploaded new content must be associated with a title, description, some tags, duration, date of activation and expiration date.
FN-3: The system shall allow the edition of the characteristics of a newly uploaded content, that is, you can modify the title, description, tags, activation date and expiration. The system shall allow the creation, edition and deletion of playlists, so that downloaded videos can be included in certain playlists.
FN-4: It will also be possible to change the order of videos in a playlist. The system shall manage the state of playlists. A playlist can be "online" or "offline." Online playlists are sent to monitors to display videos. Offline playlists contain videos that are stored on the system but not displayed on the panels.
FN-5: It also will show whether the panels have correctly received the contents that have been sent to them. The system shall report the disk capacity used and left, both in the panel's disks and the application server's disk.

Non-Functional Requirements of System *

NFR-1: The system shall communicate with the panels through the Internet using each panel's IP or their broadcast IP. The system shall be accessible and usable. The system shall meet W Web design standards.
NFR-2: The system shall meet W Web design standards. The system shall comply with the corporate image of the School of Computer Science. The application shall be available in English and Spanish. The default language is Spanish.
NFR-3: The system shall allow the recording and management of users who can access the application via a login and different levels of privileges. Access will be through e-mail address and password. The system shall allow the upload of new content.

Any integration with other systems

Product perspective: Notice boards display the images and videos stored in a GB internal hard disk drive. Images and videos can be transferred through a USB or Firewire ports. SD cards can also be used. Monitors have Gbps Ethernet network connection and Wi-Fi.

Any benefits provided by system

Notice boards are ' LCD monitors displaying information (ads, institutional information, etc.) using photo or video formats. Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames.

Anything else

This system will operate the electronic notice boards installed at the School of Computer Science. Notice boards are ' LCD monitors displaying information (ads, institutional information, etc.) using photo or video formats. Currently, notice boards do not have any management software to administer the displayed contents.

Feedback: (1) Any issues you noticed while interacting with the system. (2) Any suggestions. *

IBM Watson Assistant is an interactive AI simulator and provide relevant answers in various scenarios.

Based on your interaction with system , how will you rate the system from 1 to 5 (5 being best)? *

1	2	3	4	5
○	○	○	◉	○

Will you recommend your colleagues to use our system for acquiring requirements elicitation skills? *

- ◉ Yes
- ○ No
- ○ Maybe

Chatbot-Interview Simulator

This form is used to create Software Requirements Specification (SRS) for Display Management System using AI-based interview simulator. This form is aimed at novice requirements engineers, SRS is based on IEEE-830 format. Data will be used for the evaluation of system. Please provide as much detail as possible.

Purpose of System *

Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames. The goal of this project is developing a system capable to manage the displayed contents over the Internet, so that an administrator may make changes from anywhere.

User Characteristics *

User characteristics: There are user types: • Administrator: They can access the system with no restriction. Writer: They can upload contents, but cannot delete existing ones. • Supervisor: They can review the notice boards' contents, but cannot manage them (they cannot add, modify or remove images or videos).

Functional Requirements of System *

It also will show whether the panels have correctly received the contents that have been sent to them. The system shall report the disk capacity used and left, both in the panel's disks and the application server's disk.

Access will be through e-mail address and password. The system shall allow the upload of new content. The contents may be images, videos and Flash animations. Uploaded new content must be associated with a title, description, some tags, duration, date of activation and expiration date.

Content will be removed from both notice boards' disks and the application server's disk. The system shall reveal the status of each panel, ie: if it is running, online, or if for any reason it is down or unreachable. It also will show whether the panels have correctly received the contents that have been sent to them.

Offline playlists contain videos that are stored on the system but not displayed on the panels. The system shall allow the preview of playlists. The system shall allow the removal of material. Content will be removed from both notice boards' disks and the application server's disk.

Non-Functional Requirements of System *

The system shall meet W Web design standards. The system shall comply with the corporate image of the School of Computer Science. The application shall be available in English and Spanish. The default language is Spanish.

The system shall communicate with the panels through the Internet using each panel's IP or their broadcast IP. The system shall be accessible and usable. The system shall meet W Web design standards.

Any integration with other systems

Product perspective: Notice boards display the images and videos stored in a GB internal hard disk drive. Images and videos can be transferred through a USB or Firewire ports. SD cards can also be used. Monitors have Gbps Ethernet network connection and Wi-Fi.

Any benefits provided by system

Notice boards are ' LCD monitors displaying information (ads, institutional information, etc.) using photo or video formats. Currently, notice boards do not have any management software to administer the displayed contents. Images and videos are stored in the monitors using a procedure similar to digital photo frames.

Anything else

Feedback: (1) Any issues you noticed while interacting with the system. (2) Any suggestions. *

Seems to be no issue with system , in my opinion system should have some sort ambiguity in answers in real life interviews answers are always ambiguous somewhat.It will make interviewee to ask same question again and verify it.

Based on your interaction with system , how will you rate the system from 1 to 5 (5 being best)? *

1	2	3	4	5
○	○	○	○	◉

Will you recommend your colleagues to use our system for acquiring requirements elicitation skills? *

- ◉ Yes
- ○ No
- ○ Maybe

J Opinion Survey for sixth cycle

The following survey form has been used during the evaluation of the final cycle; below we show the results of the opinion survey.

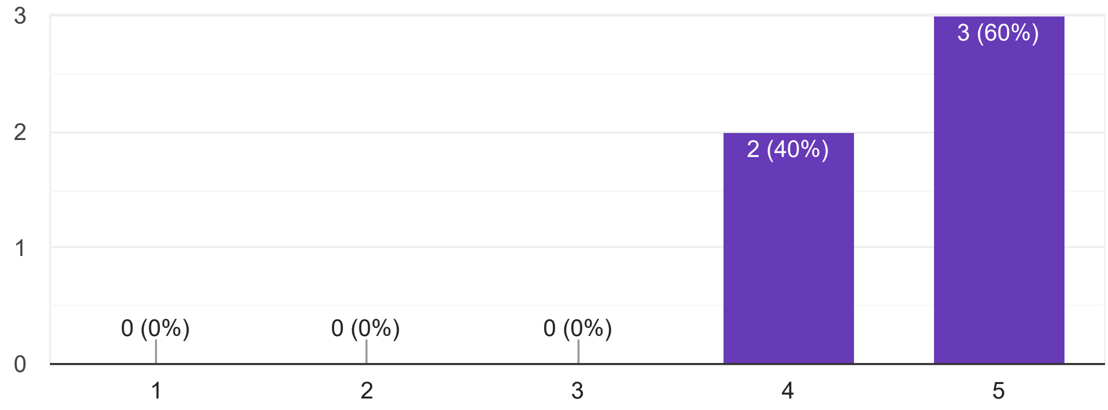
Interview Simulator Evaluation

5 responses

Publish analytics

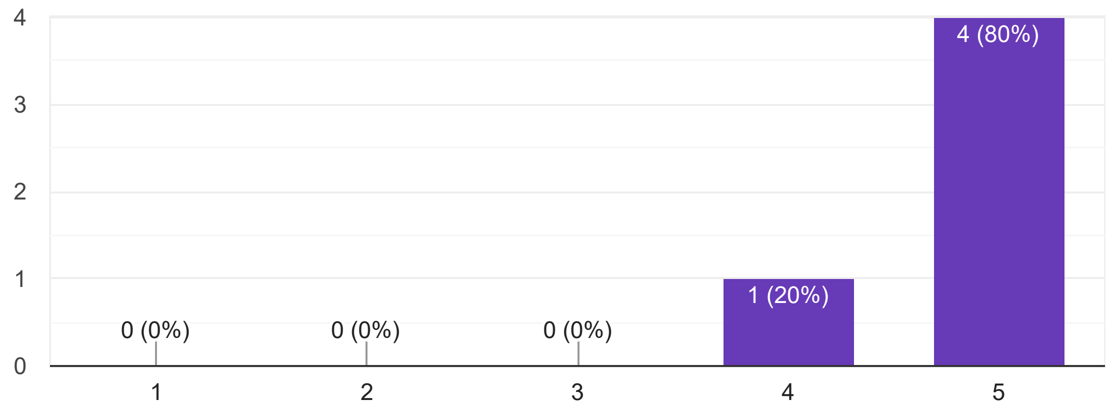
The Chatbot offered convincing and natural interaction.

5 responses



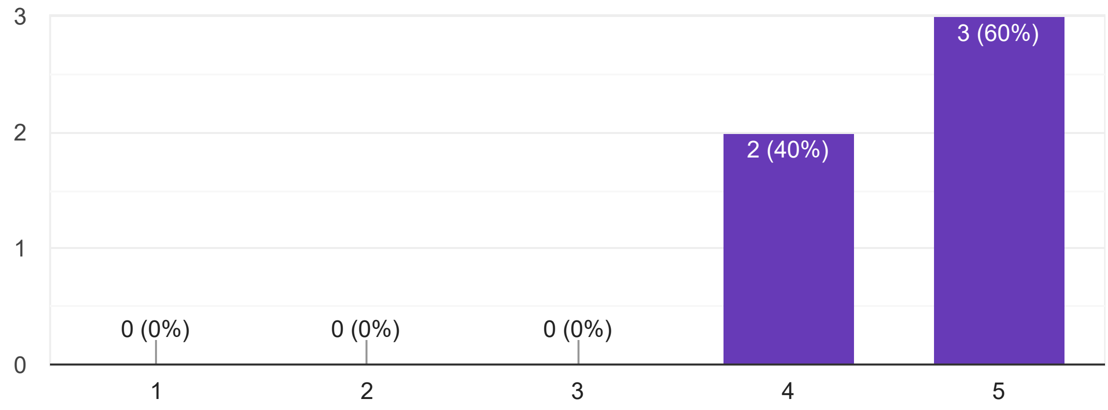
The Chatbot was able to answer questions correctly.

5 responses



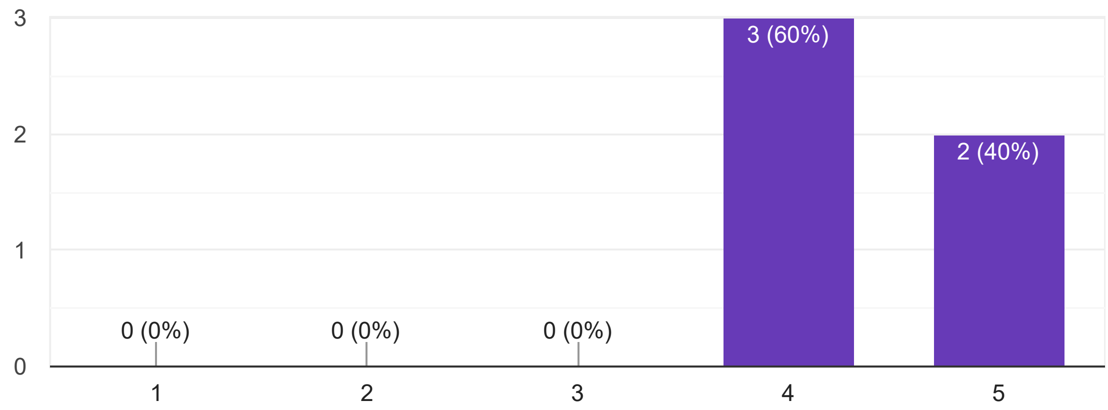
Learning about interview skills using Chatbot was fun.

5 responses



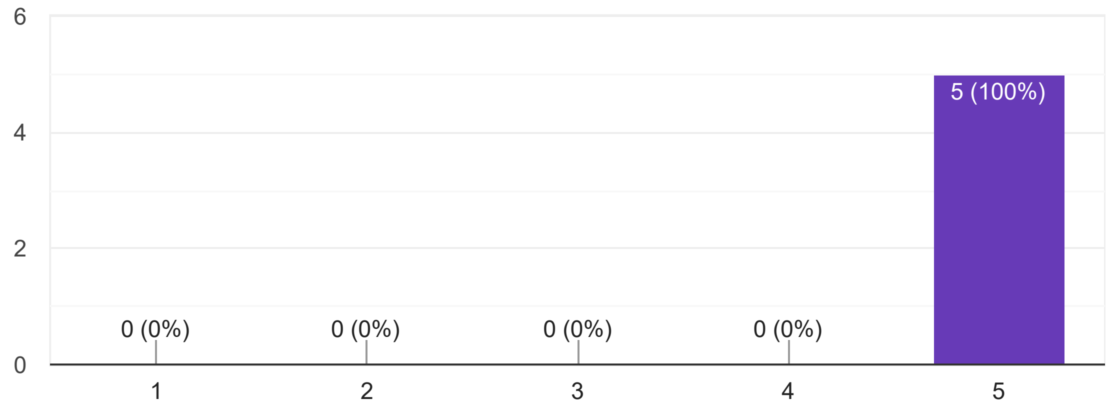
It was easy to use and understand.

5 responses



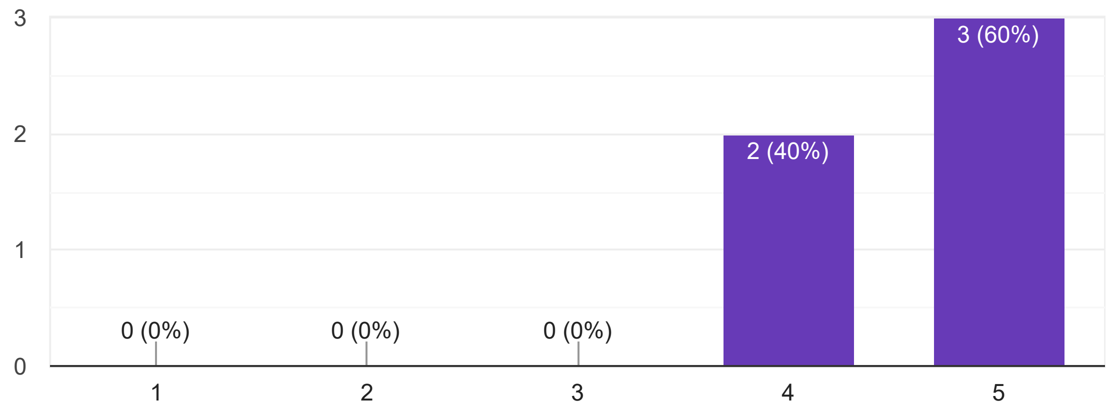
I think Chatbot will be helpful in gaining interview skills.

5 responses



Based on your interaction with system , how will you rate the system from 1 to 5 (5 being best)?

5 responses



Will you recommend your colleagues to use our system for acquiring requirements elicitation skills?

5 responses