TI2806: Context Project

Final Report

Chatbot - Embedded Systems

Delft University of Technology

Group CB-03

Y. Runhaar, 4703235
Q. Fantazia, 4693833
M. Khattat, 4588231
N. Posner, 4735382
S. Tariq, 4620097
V. Wernet, 4554582
Supervisor: W-P Brinkman

Tuesday 7th April, 2020

Contents

Chapter 1 Introduction

The most crucial choices a student will make is about which college and major they decide to join. According to a statistical analysis performed by Koenig (2018) in the U.S. News World Report, majors such as Computer and information science, Engineering and Engineering technology yield the highest employment rates and salaries compared to other majors. In an article they wrote about the factors that influence youths career choices, Akosah-Twumasi (2018) argued that the knowledge of issues related to 'job security' and 'salaries' may pressure youth to choose a career path based on the benefits associated with a particular profession. This causes an influence in the decision making of a student who will not necessarily apply for a major they would enjoy doing, but instead their choice is going to shift to a more reliable major. Thus, many students will apply for studies such as Computer Science even though it might not be well-suited for them. Our team has been asked by the Delft University of Technology's communication department to develop a Chatbot in order to help students with their decision making, and specifically students interested in the master program Embedded Systems. The communication department gave our team a set of requirements that needed to be fulfilled. The final product needed to be a chatbot with which it is possible to have a conversation on the Embedded Systems study program. It should coach the student into making a decision as well as be able to answer frequently asked questions. The chatbot needed to be accompanied by a content management system which should allow the communication department to modify some of the content of the chatbot as well as provide them with useful statistics about the interactions with the chatbot. Our team was also required to use the Rasa (2019) open source machine learning tool for conversational artificial intelligence as back-end of our chatbot system in order to provide feedback about this framework which might be used in future projects at TU Delft.

The purpose of this report is to present the main functionalities behind our chatbot as well as discussing to which extent they satisfy the needs of the end-user.

The structure of this report is as follows. An overview of our final developed and implemented software product will be provided in Chapter 2. Chapter 3 will be presenting a reflection on the final product and process from a software engineering perspective. Chapter 4 will be describing the developed functionalities the chatbot as well as its content management system. Chapter 5 will focus on human-computer interaction by describing the HCI module that was achieved by user interaction with the developed chatbot. An evaluation of the functional modules and the final product in its entirety will be given in Chapter 6. This also includes a failure analysis which will evaluate where the product does not perform as needed. Finally, in Chapter 7, an outlook will be given regarding the prospective future for our chatbot system, the possible improvements and the strategy to achieve these improvements.

Chapter 2 Software Product

As a team we were responsible for creating and developing a 3 part component system. These 3 components are the following: chatbot UI, Rasa NLU Core and the maintenance desktop application. These components are developed for the communication department of TU Delft and they are the client of this application.

The first components relates to the creation of a chatbot application that is to be used by future students that are considering to choose TU Delft as their university to study at, this includes the chatbot UI and the Rasa Core. With this comes many questions and concerns not only for the students but also for the communication department. To automate this process, the chabot is designed to coach and answers questions. The coaching is the core functionality. It allows an automated way of not only answering questions but also taking initiative to push students in the right direction. The chatbot is not biased, and only serves as a helping hand for students that are not certain of how to proceed and where to proceed. Many tweaks have been added to the Rasa Core to make it as accurate as possible. This entail hours of experimenting, testing, and user interaction to correct any errors that occurs during live demos. After a long trail, as a team we have managed to create a robust natural language AI that is able to converse with users in a positive and enlighting manner. To make the AI accessible, we have made the front-end of the chatbot modular. That is, it can be embedded in another site with ease. The rasa core is also fully modular and can be setup on an other server.

The desktop maintenance application is the main means of interaction to the data that is being generated by the users communicating with the chatbot. It gives insight on what topics are discussed, country the student is currently in, student education level, session count, and the ability to adjust reponsones of the chatbot with a couple of clicks of a button. The desktop application has 5 important functionalities. It is capable of generating graphs based on real-time data from the server, it is secured with a login session, it is able to adjust the responses in the database, add new users that are able to have access to the dashboard, and export the graphs to a PDF. As a team we made sure that the desktop application is cross-platform (Linux/-MacOS/Windows) and also accessible through the browser if needed. It also comes with installers for each operating system. The desktop application as a whole helps enable the communication department to have a clearer insight in what types of topics tend to appear frequently, and to modify the chatbot as needed as time passes as previously mentioned.

The components need to be incorporated in the current environment of the communication department. Seeing that an infrastructure and an environment already exists, it is of high importance that the components are modular. This enables each component to be incorporated in TU Delft's infrastructure. We designed the system in such a way that the components are loosely coupled. We have decided since the beginning of the first sprint to develop the application in a real environment with real settings. We wanted to tackle real world problems and learn how to deal with them with best practices. None of our components are run locally, rather they are run on the live server. Due to all of this we are very certain we are able to reproduce this project to any other company/department.

Chapter 3 Product and Process Reflection

The team has been developing a product for around 2 months. How did everything go?

Agile methodology: All the teams in the context project have been instructed to use the agile methodology which relies on demonstrations, showing work to the customer and discussing the product rather than trying to document everything extensively which takes a lot of time. The Agile methodology has had a lot of advantages and one of the most important ones is being able to adapt to feedback. "Agile methods are based on adaptive software development methods, while traditional Software Development Life Cyclemodels (waterfall model, for example) are based on a predictive approach"(Stoica, Mircea, & Ghilic-Micu, 2013)

Progress of the project: The project sometimes went just as the team desired, but there are some points where we had to slow down. At the start of the project, we had to start coding very slowly because making a chatbot was a new topic and there was a lot of literature that we had to read, for example, about natural language processing, reading the document which was offered to us by the context ta and reading and learning rasa. Later on, the team started with delivering code at a quicker pace and before the end, the team started to deliver with greater quality by focusing on writing tests.

Features: At the start of the project, the team tried to set the bar high and implement some cool features. To start off with, the team put a lot of effort to ensure that the product is secure against SQL injection and encrypting data with https. This has been indicated by the book "Software security: building security in", the book demonstrates that "the dollars spent on network security and other perimeter solutions are not solving the security problem and we must build better software"(McGraw, 2006) In addition, The team tried to expand the desktop application like being able to get the statistics in a pdf, installers for any operating system and being able to run the desktop application on the browser. Furthermore, the application has an intuitive UI and represents the data to the maintainer in a simple yet effective manner.

Although the team wanted to do some features, due to lack of time we had to leave these features out. The first feature is using Grammarly to correct the user input in case there was a typo. That would have been a nice feature because having typos could confuse the natural language processing. In addition, we wanted to implement voice recognition which takes a lot of time and can be quite tricky to implement and a source of bugs.

RASA: Privacy is a key issue that users consider when using a chatbot (Følstad, Nordheim, & Bjørkli, 2018) and that RASA solves since it is an open source framework that can be hosted in-house. This can be a tricky issue with chatbots particularly due to the fact that it is not possible to know beforehand what the user will say and ask. RASA does very well at FAQ and task based chats, being able to pick up on a user intent and giving a standard response. It is also possible to write automated tests for this. When it comes to a software engineering standpoint it is not clear how to write tests for stories that can be automated as opposed to being done manually.

Chapter 4 Developed Functionalities

As mentioned in chapter 2 the software product consists of 3 components namely ChatBot UI (User Interface), desktop maintenance application and Rasa server. These components communicate with each other through a central server.

The Chatbot UI is the front end which runs on the TU Delft website. It's main functionality is to send the messages of the user to the server. This communication is made possible by Rasa server which listens on the incoming messages on a specific port, and send the result back to the Chatbot UI. The processing of the user message is done by Rasa server which itself consists of 2 components: Rasa NLU (Natural-language understanding) (Inc, 2019) and Rasa Core (Inc, 1999). These 2 components have different functionalities. Rasa NLU uses AI techniques to understand and extract information from the text and Rasa core uses an opensource machine learning library like *tensorflow* to guide the flow of conversation.

The other component is the desktop maintenance application which has 3 main functionalities: Login page, the database page and the dashboard. The login page provides a secure way to login and use the desktop application. The maintainer is required to login with a username and a password. The password should be at least 7 characters and contains both letters and numbers. The password will then be converted to SHA-256 and will be sent to the server for comparison with the one already saved into the database. Finally, the server will approve or disapprove the authentication.

The Database page allows the maintainer to easily see and modify values related to the topics in the database. These are responses and variables like links to external websites for more information.

The dashboard is the place where the maintainer can see the most up to date statistics about the Chatbot. The statistics are about topics discussed in chat, countries from which people interacting with the chat, a set of unknown keywords which could possibly correspond to new topics that are not covered by the chat and finally, some statistics about the number of users who have used the chat in the past. To enable the application to use these statistics, first they need to be stored. When the user is interacting with the Chatbot we extract useful information, e.g. topic currently being discussed, and send them to the database. This data is then retrieved from the database by the desktop application and will be shown in different interactive graphs. It means that the user can click on a graph to get more detailed information about dates. They can also be adjusted, filtered and sorted.

The desktop application also provides the ability to export data of the dashboard page to external documents like PDF. It gives the maintainer the ability to easily use the graphs in future without the need of the application.

There are also security components which are invisible to the end user but ensure a secure end-to-end connection between all components:

- Hypertext Transfer Protocol Secure (HTTPS). It is used for a secure communication over the network and prevents man-in-the-middle attacks.
- BCrypt (Sha-256 + random salt). All the passwords stored in the database are hashed and salted using this method. This ensures that the passwords are protected against rainbow table attacks, dictionary attacks, and brute-force attacks. (Provos, 2019).
- JSON Web Tokens (JWT). This ensures that only users that have authenticated themselves are the only ones allowed to access end-points (API).

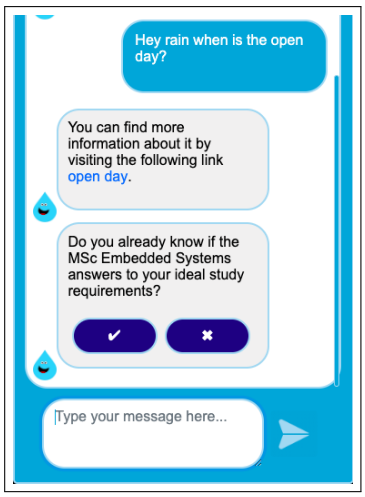
- Cross-origin resource sharing (CORC). When visiting a web page, it makes requests to different places on internet to load assets like fonts or images. If these requests are not being checked, the security of the web browser can be at risk. The CORC is a standard security policy which allows the server who can access its assets, and how the assets can be accessed. It allows us to deny some requests like DELETE to avoid malicious behavior.
- Helmet.js. It is a module that helps us secure HTTP headers returned by the server. HTTP headers are metadata about HTTP requests or responses which give the client (browser) and server the ability to send additional information in a transaction.
- Input Validation and Sanitization. It is a defense mechanism to provide extra security. Validation checks if the input originating from the user meets a set of criteria (for example when creating a password for a new admin, it should be at least 5 characters and contains both letters and numbers). Sanitization modifies the input to ensure that it is a valid input (for example converting text messages to lower case before sending them from chatbot UI to the Rasa server).
- SQL Prepared Statements. The idea of using SQL Prepared statements is to prevent SQL injection. It is based on the idea that an input coming from a user is not trustworthy. So it separates the data coming from user and the query that is going to be sent to the database. As a result it prevents SQL intjection which could cause to leak data or even to modify data on the server by a malicious user.

Chapter 5 Interaction Design

The objective of this chapter is to describe the user and usage situation of the chatbot. A typical use case scenario and corresponding design decisions shall be outlined along with usage assumptions. Building on the methods presented by Brinkman (2019a) in his lecture series at the Delft University of Technology(TU Delft) this chapter shall act as a usability evaluation compiled from the results of countless contextual inquiry interviews. Taking place over consecutive weeks these interviews were structured as ten to fifteen minute interactions with the chatbot whereby a developer observed the user in the course of the user's normal activities and discussed how easy and helpful those activities with the user were after the chat had ended. Following user centered design principles a description of the product design and the user interaction derived changes or outcomes of the usability evaluations of the product shall be given.

The primary target users of the software are prospective students of TU Delft Embedded Software masters study program. Within the primary target user group there are graduating bachelors students from TU Delft, local from the Netherlands and International students. All requiring individual assistance when making an informed decision about their study choice. When designing the usability evaluation great care has been taken to ensure that each subgroup has been represented and their feedback incorporated into the final product. (Sharp & Preece, 2007) As mentioned the typical use case scenario is one of a student applying to a university program. This student primarily needs more information about the study and vitally coaching to help enable him/her to make the best decision about their future. One of the main discoveries of the usability evaluation is that users often do not want to receive coaching and have very specific questions on their mind. This led the developers to incorporate an accelerator into the user interface(UI) as shown below in figure 5.1 To allow the user to perform common tasks fast (Nielsen, 1994) and ensure that the user receives coaching.

Figure 5.1: User Interface to control discussion flow



The developers have used interpretive usability evaluation assessments throughout the development of the product. To assess the usability of the Chatbot according to the following criteria: How effective is the chatbot at fulfilling its purpose of activating the users study choice(see product vision document for details). Furthermore the chatbot should be efficient at achieving its goals. The developers have designed and conducted user testing to ensure that the number of clicks is minimal when a user tries to achieve a task. The developers have also focused on minimising the learning curve for a new user. From user testing the developers identified that a proportion of users could not identify the chatbot on the Embedded Systems home page. After adding a textbox as shown below in figure 5.3 this problem has dissipated.(Sharp & Preece, 2007)

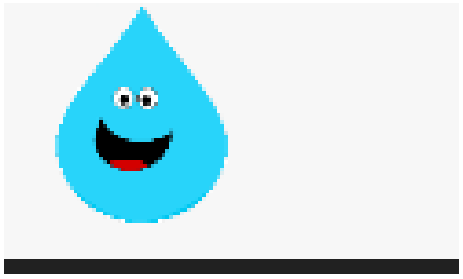


Figure 5.2: Before learnability evaluation

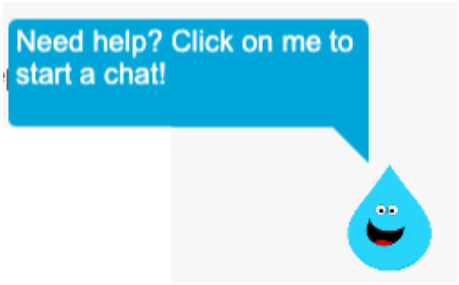


Figure 5.3: After learnability evaluation

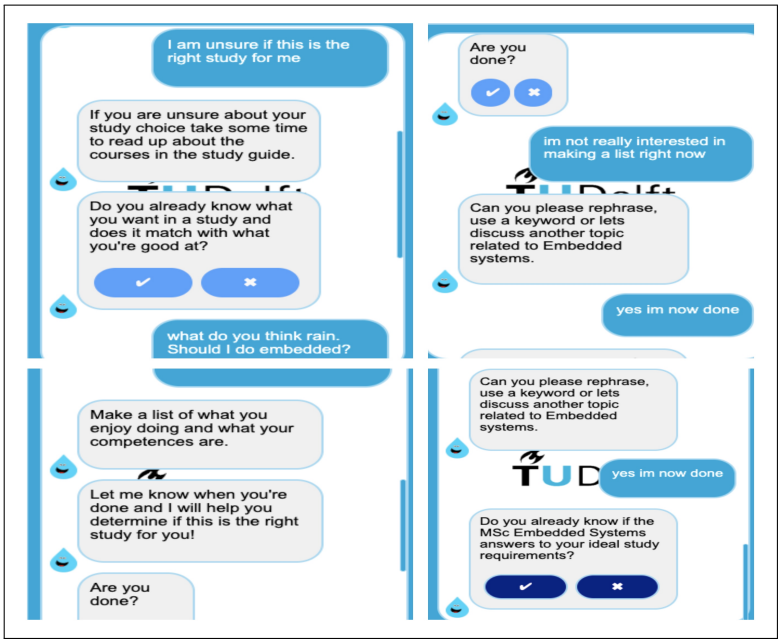
Following empirical usability evaluation methods the developers conducted several interviews throughout the design process with real users of the product. Since many of these target users were international these interviews were mostly over social media networks such as WhatsApp and Facebook. This benefit stems from our system design: having a totally portable website that can be accessed on mobile, laptop and tablet - Anywhere in the world with a click of a link. Consequently our usability field study has been broad in its scope - reaching potential users in Aruba, England, France, Germany, India, Netherlands, Pakistan, South Africa and USA all in their own comfort as if they were not participating in a usability evaluation. By conducting formative evaluations over multiple weeks key usability obstacles such as typing speed and colors were identified and improved upon. (Brinkman, 2019b)

Conducting supervised experiments with expert target users over multiple weeks at the University, critiques of the usability of the prototype were discussed and then taken into consideration during the next sprint. (Sharp & Preece, 2007) These experts can be described as five Computer Science and Engineering students who have followed the Systems variant track and have or are about to graduate from TU Delft. They have an aptitude for technology, TU Delft, the Embedded Systems program contents and are deciding whether to follow the Embedded Systems masters degree. Guided by their own personal experience when applying to a University program these experts role-played target users and identified areas for product improvement such as the handling of non binary user responses.

Starting in week four the team began to conduct a "user evaluation hour". At each Monday lab session held at the Drebbelweg a weekly hour sessions was scheduled for some of the developers to walk around and ask other students of the context project to chat with the chatbot. These moments were incredibly helpful to gather missing questions and their variants that the chatbot could not answer along with identifying new conversation topics that were not thought of before hand. Taking place at the beginning of the week, the developers had enough time to make product adjustments before Friday meetings with clients and supervisors. Furthermore the team has had three one hour sessions in the later weeks of the project with a student born in India that is currently enrolled as a second year student in the masters study embedded systems at TU Delft and works for the Embedded Systems communications team. These sessions helped to identify information needs of international students.

Asking the client to use the prototype chatbot during a weekly meeting produced a high level usability evaluation. The client of this particular product has domain specific knowledge to act as an expert. The setup was as follows with one student acting as a guide and the others as observers. Following this moment a discussion between the client and developers took place which led to product changes. During the midterm evaluation an observation was made by Dr A, Panichella who is the teacher of the context project that some messages from the chatbot were too long and should be split. Along with the chabot not being able to handle non relevant questions such as those pertaining to Game of Thrones. Following Benyon, Turner, and Turner (2005) error message design guidelines attention has been paid to how the chatbots error messages are worded and presented. The standard error message does not use double negatives and gives specific instructions to get the chat back on track. Furthermore during user testing there was special focus to try ensure that the chatbot does not influence user behavior. An illustration of how these two thorny issues were dealt with is shown below in figure 5.4. Caveat lector - The bottom and top right images should not be misunderstood as repetitions instead they show the chat follow through.

Figure 5.4: Error recovery and coaching style



To sum up Interaction Designs techniques and methodologies based on empiricism has been an efficacious method for producing software of the highest user centered quality.

Chapter 6 Functional Modules and Product Evaluation

During this project, the team worked hard to deliver a system that could adequately fulfill the needs of the client. Doing so meant that certain trade-offs had to be made to stay within the available time frame. Some features had to be abandoned or compromised on. In this chapter, these decisions will be discussed and an evaluation of the final product will be given, including its possible shortcomings.

As mentioned in previous chapters, the system consists of three, mostly independent components: the web interface, the chatbot engine and the maintenance application. In designing these components, the team has tried to keep them as separate as possible, so that they could be easily adapted or switched out. For instance, the possibility of integrating the chatbot engine on a different platform is something that was kept in mind. This was done successfully for the most part: the logic concerning the dialog is only present in the Rasa backend, while the web interface only concerned itself with presenting the UI to the user and handling interaction with the chatbot engine. However, there is one aspect in which this design goal isn't reflected; for the user statistics, the system relies on the web server to store relevant information in the database. If the chatbot were to be integrated on another platform, this extra layer between the user and the backend wouldn't be present, so this method of handling data would have to be adjusted. The team would have liked to improve this but decided it was more important to get the rest of the functionalities up to standard.

Another issue that came up was whether to aim for more open-ended conversations on broad topics or instead to use closed questions on specific topics. The trade-off here is that with open questions it becomes harder to gauge the user's needs, where they are in the decision-making process and what resources to point them to. With closed questions, it becomes much easier to handle these use cases, but the chatbot will also seem more like a robot. Additionally, the user might also be turned away by what could feel like filling in a questionnaire. Despite these points, the team decided on the closed questions as the better option, because it seemed to give a greater chance at actually making the system usable and able to be of help to students.

Even though the team would have liked to improve on these and other points, the final product is still able to address the client's wishes. It can provide prospective students with basic coaching to make them reconsider their decision process, as well as giving the Communications Department insight into students' needs. Therefore, we can say that the goals set at the start of the project were successfully met.

Chapter 7 Outlook

The purpose of this section is to present the vision our team has regarding the potential future of our chatbot. The chapter will describe concepts that are not yet implemented in our system, the strategy to achieve these concepts and evaluate their possible contribution if they were to be put into effect. Furthermore, this chapter is split into two subsections. The first subsection will report the concepts concerning the chatbot web application, and the second subsection will describe the concepts related to the administrator web and desktop application.

7.0.1 Chatbot application

In this first subsection, the concept related to the chatbot web application will be discussed. According to TU-Delft (2019a), 20% of their students are international. Additionally, the majority of these international students do not originate from an English speaking country (TU-Delft, 2019b). As the other 80% of students have Dutch as their mother tongue, we can safely assume that the vast majority of TU Delft students do not have English as their native language. This motivated our first potential concept which would be to support multiple languages in our chatbot. In order to implement this, we would have multiple versions of the chatbot, each in a different language. The user would then be able to select the language he wishes to convers in. Such an implementation would prompt more users to interact with the chatbot as well as exhibit the diversity at TU Delft.

In order to make the chatbot more accessible, our team considered integrating it into other messaging platforms such as but not limited to Facebook Messenger, Slack and Telegram. Since this implementation was a "could have", our time was not invested into this issue. However, by using the already existing chatbot API's for each of these messaging platforms, this concept could be implemented which would make the chatbot highly more accessible.

"Fifty-eight percent of website visits came from a mobile device in 2018, surpassing desktop usage for the fourth consecutive year" (Ciligot, 2019). Since the scale of mobile usage is such an important factor to consider, the idea of having the chatbot as a mobile application was certainly considered. This issue was also not implemented as our team focused on matters with a higher priority. Nevertheless, the chatbot web application could be converted into a mobile application by converting it into a react native application. This would further increase the accessibility of our chatbot.

7.0.2 Administrator application

In the same manner as for the chatbot application, the magnitude of mobile usage can't be ignored. Thus, our team had also considered creating a mobile application for the administrator application. Since the administrator web and desktop application is created using react native, it can easily later on be converted to a mobile application. This would make it easier for the communication department to access the administrator application.

Currently, the administrator application allows only for content to be modified. A prospective change to this our team had in mind would be to allow the administrator to delete and add new content as well. This would require not only require modifying the administrator application but also some changes in how the database is structured.

Finally, our team had the idea that scheduling an update after the administrator modified the content would be a good idea. Currently, the administrator has to click on a button that will immediately update the chatbot application with the new content but doing so will render the application unusable for a couple of minutes. Thus, it would be better if the administrator could schedule such a change during quiet hours.

References

Akosah-Twumasi. (2018). A systematic review of factors that influence youths career choices–the role of culture. *Frontiers in Education*, *3*, 58. Retrieved from https://www.frontiersin.org/article/10.3389/feduc.2018.00058 doi: 10.3389/feduc.2018.00058

Benyon, D., Turner, P., & Turner, S. (2005). *Designing interactive systems: People, activities, contexts, technologies*. Pearson Education.

Brinkman, W.-P. (2019a, May). *Design methodologies and techniques.*

Brinkman, W.-P. (2019b, May). *Usability evaluation.*

Ciligot, C. (2019, May 2). *Mobile app vs. mobile website: A ux comparison.* Clearbridge Mobile. Retrieved from https://clearbridgemobile.com/mobile-app-vs-mobile-website-which-is-the-better-option/

Følstad, A., Nordheim, C. B., & Bjørkli, C. A. (2018). What makes users trust a chatbot for customer service? an exploratory interview study. In *International conference on internet science* (pp. 194–208).

Inc, Â. R. T. (1999). *The rasa core dialogue engine.* Retrieved from https://rasa.com/docs/rasa/core/about/

Inc, Â. R. T. (2019). *Rasa nlu: Language understanding for chatbots and ai assistants.* Retrieved from https://rasa.com/docs/rasa/nlu/about/

Koenig, R. (2018). Your college major does not define your career. *U.S. News and World Report*. Retrieved from https://money.usnews.com/careers/applying-for-a-job/articles/2018-09-24/your-college-major-does-not-define-your-career

McGraw, G. (2006). *Software security: building security in* (Vol. 1). Addison-Wesley Professional.

Nielsen, J. (1994). *Usability engineering*. Elsevier.

Provos, D. T. J. S., Niels; MaziÃĺres. (2019). *A future-adaptable password scheme.* Retrieved from https://www.usenix.org/legacy/events/usenix99/provos/provos_html/node1.html

Rasa. (2019, June). *Open source conversational ai.* Retrieved from https://rasa.com/

Sharp, R. Y., H., & Preece, J. (2007). *Interaction design beyond human-computer interaction*. Chichester, England: John Wiley Sons.

Stoica, M., Mircea, M., & Ghilic-Micu, B. (2013). Software development: Agile vs. traditional. *Informatica Economica*, *17*(4).

TU-Delft. (2019a, June). *Facts and figures.* Retrieved from https://www.tudelft.nl/en/about-tu-delft/facts-and-figures/

TU-Delft. (2019b, June). *Student population.* Retrieved from https://www.tudelft.nl/en/about-tu-delft/facts-and-figures/education/student-population/