

# Asystemt – a Prototype of a Motivating Electronic Assistant

Patrycja Świczowska<sup>1</sup>, Jolanta Bachan<sup>2</sup>, Rafał Rzepka<sup>1</sup>, Kenji Araki<sup>1</sup>

<sup>1</sup>Hokkaido University <sup>2</sup>Adam Mickiewicz University

swiczowska@ist.hokudai.ac.jp

jbachan@amu.edu.pl

rzepka@ist.hokudai.ac.jp

araki@ist.hokudai.ac.jp

## Abstract

This paper presents preliminary research aimed at gaining understanding about user preferences concerning electronic assistants. Our end goal is a system that not only plans the work, but also provides individual motivation for the user, which is a new approach. At this stage, we created a simple dialogue system Asystemt to gather some basic data regarding successful human-computer cooperation. The system was evaluated through a user survey and received relatively high scores on friendliness, naturalness of interaction, helpfulness, and the ability to motivate the user to work. The results show that the implemented techniques, although rather simple, were effective in creating a dialogue system successfully cooperating with the user and motivating them to work. The testers' comments provided valuable insight which will serve as guidelines for creating the actual dialogue system in the future.

## 1 Introduction

Human-to-human communication is a form of cooperative activity, depending on the interlocutors' common beliefs and convictions [Jurafsky and Martin, 2000; Leech and Weisser, 2003], their intentions and aims [Scott and Kamp, 1997], their knowledge, reasoning abilities and the way their utterances are relevant to the topic at hand [Leech and Weisser, 2003]. Similarly, a dialogue between a human and a computer can have a specific purpose. In the case of the system described in this paper, the main goal is to assist the user in completing their everyday tasks while also making sure the user is sufficiently motivated to do the work. The system was meant as a testing tool to verify our views about what constitutes a successful electronic assistant and about users' preferences. We assumed that conducting a friendly conversation with the user, monitoring their work and striving to be generally helpful will positively influence user satisfaction about the cooperation. We also implemented simple motivating techniques, like reminding the user to complete tasks or providing a simple reward for having done the work. The data gathered in this research is meant to be

used later as the starting point for creating the actual program, of which the system described in this paper is a prototype.

The paper is organized as follows. Section 2 presents related work on similar dialogue systems acting as electronic assistants. Section 3 characterizes the Asystemt dialogue system, which was created for the purposes of the research to collect some data about user's preferences. Section 4 presents the method and results of the evaluation. Section 5 discusses the results and includes remarks about further research directions. Section 6 describes our ideas for a motivating module that will be implemented in the final version of the system. Section 7 concludes the paper.

## 2 Related Work

Research in dialogue systems is a vast and quickly developing field. Recently, there have been proposed many innovative systems meant to interact with the user. Some of them are meant for general use and employ complicated algorithms for information retrieval to stay relevant to the topic of the conversation, for example [Yan *et al.*, 2016] or [Higashinaka *et al.*, 2014]. Others rely on spoken dialogue and the information that can be extracted from the speaker's utterances about their intentions, for example [Liu and Lane, 2016]. Smartphone-compatible dialogue systems, such as Siri [Apple, 2017] or Cortana [Microsoft, 2017] are widely used, and chatbots are becoming increasingly popular with businesses [Business Insider, 2016]. Moreover, there is some research into human-computer relationships in the field of dialogue systems [Bickmore and Picard, 2005].

The end goal of the research described in this paper is to create a task-oriented dialogue system that acts as both an assistant and a calendar. Related work in the field includes systems like SelfPlanner described in [Refandis and Alexiadis, 2011]. This system is a web-based calendar application integrated with Google Calendar and Google Maps that introduces a new method of planning and rearranging tasks, especially with respect to meetings. Another program similar to our system is Timeful [Bank *et al.*, 2012]. It is a smartphone application acting as an electronic organizer, also cooperating with Google Calendar. The program tailors itself to the user and over time learns their working habits to better plan their tasks. A notable contribution to the field is the CALO project [DARPA, 2017]

aimed at creating an intelligent electronic assistant for the user. Related research in electronic calendars includes, among others, new methods of managing time between meetings in different geographic locations [Meisels *et al.*, 2014], integrating the calendar bar into the e-mail window [Rosenberg *et al.*, 2014], adding tasks to the calendar via voice commands [Gustafson, 2013], improving visual representation of the calendar interface [Hurst and Morris, 2016; Bisht, 2016] or extracting relevant information about the tasks from an ontology rather than directly from the user to improve interaction with the calendar application [Agnatis *et al.*, 2016].

The survey of the field shows that the current main focus is to improve planning strategies and techniques in the calendar applications. However, providing a workplan for the user does not guarantee that they will complete the tasks. As far as we know, there has been little to no research into actually motivating the user to work. The end goal of our research is therefore creating an electronic assistant that not only would plan the user's tasks and rearrange them accordingly, but also to help the user stay focused and motivated. To achieve this, first we test users' preferences about cooperation with the system and the implemented motivating features. This is done via the dialogue system described in the next section.

### 3 Dialogue System Asyistent

To create a dialogue system best tailored to the user's needs, several evaluation stages have been envisioned. The system presented in this paper is a prototype using scripted dialogue. The main goal of the system was to test the users' general preferences and indicate possible future developments. As such, the evaluation results serve as guidelines for the next stage of the research. After analyzing the user's opinions, we have successfully determined the direction of our future work.

This section presents the dialogue system. The following subsections describe in turn the technical specification, goals of the system, the program's implementation, and the cooperation taking place between the system and the user.

#### 3.1 Technical Information

Asyistent is a text-in-text-out dialogue system and was designed according to the structural approach to dialogue systems. As such, it relies on finite automata and a regular dialogue structure [Sadek and de Mori, 1998]. The initiative in the conversation is entirely on the system. It asks the user questions about the work and based on the answers gathers the data necessary for further operation. The data is stored in text files. Some additional files are created on the user's hard drive when necessary.

The system's package includes a text manual. In crucial moments like introducing itself or the work beginning, the program offers to open the file for the user to read through.

The user may still open the file manually at any other point, as the system informs him or her of the location of the file.

Since the system was meant to serve as preliminary research and needed to be evaluated, the package distributed to the users contained an additional text file with the user survey.

The package also contains simple sound files, which are utilized to signalize the need to answer a pending question from the program or to remind the user about tasks.

The program was created in Python and uses the GUI window as user interface.

#### 3.2 Goals of the System

The program is aimed at people who need motivation in their everyday work and who would like to have a personal assistant helping with their work. Specifically, the program's main goal is to help users who have a workplan ready, but forget about the tasks or do not feel like completing them. The system serves as a motivating assistant for the user to ensure he or she actually does the work. The features of the program were decided after oral consultations with potential users to better suit their needs.

The main goals of the program are as follows:

- helping the user in creating the workplan;
- reminding the user about upcoming tasks;
- motivating the user to complete the tasks;
- conducting a natural and friendly conversation<sup>1</sup> with the user;
- establishing a satisfying cooperation with the user.

#### 3.3 Implementation

When the user launches the program, the system introduces itself to the user. This includes asking for the user's name – which is then saved as text for later use – and offering to open the manual file. Then the program asks for the workplan for the day. All tasks on the plan need to be input in a way specified in the manual (but not necessarily in a chronological order), which includes name of the task, time when the task begins in HH:MM format, and time allocated for completing the task in minutes. The program sorts the tasks chronologically according to the HH:MM data given by the user, and checks against any overlaps. If there are none, the plan is accepted; if some tasks overlap, the program asks the user to modify the plan and conducts checking for the new plan. All modifications are made through the GUI window. The steps are repeated until the plan gets accepted. The program then displays the workplan in its window, showing the starting time and name of each task. The plan is also stored in a separate file.

After that, the user is asked to fill in information about rewards. A reward is a website or a file on the hard drive – for example an e-book – that the user would like to open in their spare time after completing a task. The user is asked to provide a simple name for the website or file – for example FB for Facebook – for the program to easily recognize what

---

<sup>1</sup> The conversation is task-oriented and restricted to the topics crucial for completing the system's goals.

it needs to open, without the need to type the whole url address or path again. An example is given fore the user to know how to do this. The user inputs the data in the GUI window and the program stores it in a text file. Each line of the text consists of the short name and the full address of the website or file, essentially becoming a two-item list. All later operations on this data are therefore conducted as operations on lists.

When the work actually starts – which may be immediately after creating the workplan or any time later – the program greets the user by name and informs him or her of the current date and time. Then it displays the workplan again and asks for confirmation. This is the last chance for the user to make changes to the plan. The new plan is checked against overlaps in the same fashion as the original plan.

Next, the program asks whether the user would like to turn on the sound reminder functions. The first function signalizes an upcoming task. A few minutes before the task begins, the program reminds the user that there is a task approaching. If the user wants to use this function, he or she is asked to input the number of minutes before the task when he or she wants to be reminded. The second reminder signalizes the time allocated for a task coming to an end. Similarly, the user inputs the number of minutes when the sound should ring before the task ends.

All information concerning time is displayed in HH:MM format. To ensure that the program displays it at appropriate times, a time-matching function is employed. This function matches current time extracted from the OS with the time given on the workplan for each task, and when both pieces of data match, the program displays corresponding information about the task at hand beginning or ending. This function checks for matches every 10 seconds. Similarly, the reminder functions work with the time-matching function, with the selected number of minutes being incorporated into the calculations. For example, if a task begins at 14:10 and the user wants to be reminded about it 5 minutes before, the program recalculates the string “14:10” into minutes, subtracts 5, recalculates this again into HH:MM format and then checks for matches with current time in the OS. If a match is detected, the program displays a reminder and activates the alarm.

To avoid disturbing the user, for both reminder functions only the sound is played, but the program does not maximize the GUI window. As a result of the aforementioned oral user survey, the reminder functions were made optional. The user can use both, one of them or neither. Additionally, some tasks – like, for example, sending an e-mail – may last shorter than the time decided for the second reminder function to ring before the ending of the task. In such cases, the alarm does not ring. The user is informed about this beforehand in the manual.

Next, the program displays the time when the first task begins.

The following steps are then executed for all tasks on the workplan list:

- reminding the user about the upcoming task (if the user opted to use this feature);

- signalizing the task starting and informing the user about the amount of time allocated for the task;
- reminding the user about the task coming to an end (if the user opted to use this feature);
- signalizing the task ending and asking whether it was completed;
- if the task was completed, it is added into the file containing a list of completed tasks; the user then may opt to open a website as a reward;
- if the task was not completed, it is added into the file containing a list of failed tasks;
- notifying the user about the time the next task starts; this step is not executed if the previous task was the last one on the workplan list.

After the work is finished, the program summarizes the day by displaying both the completed and the failed tasks lists. Both lists are stored as text files for the user to access at any time in the future. Next, the program asks for an opinion about the cooperation, thanks the user for working with him and turns itself off after saying goodbye.

The system included a reminder for the user to fill out the survey just before the program turned off. This feature was only present for the purposes of evaluation and will be removed once the research is completed and the final version of the program is created.

Figure 1 shows a screenshot of the program’s interface containing some interaction with the user. The first picture presents the interface in its original language, which was Polish, although the program could easily be adapted into other languages as well. The second picture is the English translation of the first picture’s content.

### 3.4 Cooperation with the User

Asystent was created to test users’ preferences about cooperation with an electronic assistant. As the research is preliminary, the system implements simple techniques merely to test whether this behavior would be accepted by users and whether it should be implemented in the actual program. We tried to make the system as friendly, motivating and helpful for the user as possible and then evaluated implemented techniques.

The system saves all the user’s tasks, indicates issues with the workplan, stores information about both completed and failed tasks, reminds the user about upcoming tasks and summarizes the work done throughout the day. This is reminiscent of the work of a human assistant or secretary. Moreover, to avoid disturbing the user, the program works in the background; whenever there is a message from the system, only the corresponding sound is played, but the GUI window is not maximized. The system also keeps the conversation to the minimum. This makes it similar to an electronic organizer.

```

Python 2.7.6 Shell
File Edit Shell Debug Options Windows Help

Czy zadanie zostało wykonane?
nie
Rozumiem, w takim razie wpiszę to zadanie na listę zadań
niewykonanych.

Następne zadanie, nakarmić koty, zaczynamy o 14:03.

Następne zadanie zacznie się za 5 minut.
Dalej na liście jest: "nakarmić koty". Planowany czas na wykonanie
tego zadania to 5 min. Oby tylko zdążyć! :)

Zostały Ci 3 minuty :)

Czy udało Ci się ukończyć zadanie?
tak
Gratuluję! Wpisuję więc to zadanie na listę wykonanych zadań i życzę
powodzenia przy następnym :)

Jeśli masz wolną chwilę, sugeruję skorzystać z funkcji nagradzania :)
Jeśli nie chcesz niczego otwierać, wpisz "nie".
chcę
Który plik lub stronę internetową chcesz otworzyć?
fejs
Proszę bardzo.
Przypominam, że następne zadanie to zadzwonić do koleżanki i zaczyna
się o 14:15.

Jeszcze tylko 5 minut do kolejnego zadania!
Ostatnie zadanie to zadzwonić do koleżanki. Masz na nie 10 min.
Powodzenia!

Zostały Ci 3 minuty do planowanego zakończenia zadania.

Jak poszło? W porządku?
tak
Gratuluję! To było Twoje ostatnie zadanie na dziś!

Szanowna Użytkowniczko! Teraz nastąpi podsumowanie Twojej dzisiejszej
pracy.
Ln: 194 Col: 4

```

```

Python 2.7.6 Shell
File Edit Shell Debug Options Windows Help

Was the task completed?
no
I see, I'll put it on the list
of unfinished tasks then.

We begin the next task, feeding the cats, at 14:03.

The next task will start in 5 minutes.
Next on the list is: "feeding the cats." You have 5 minutes
allocated for this task. Make it in time! :)

You have 3 minutes left :)

Were you able to complete the task?
yes
Congratulations! I'll put this on the list of completed tasks
and I wish you luck with the next one :)

If you have a moment to spare, I suggest you use the reward feature :)
If you don't want to open anything, type "no."
I want
Which file or website do you want me to open?
fejs
Here you go.
I'd like to remind you that the next task is calling a friend and
it starts at 14:15.

There are only 5 minutes left before the next task!
Your last task is calling a friend. You have 10 minutes to do this.
Good luck!

You have 3 minutes left until the end of the task.

How was it? OK?
yes
Congratulations! This was your last task for today!

Dear User! Now I shall summarize all the work
you completed today.
Ln: 194 Col: 4

```

Figure 1: Asyistent's interface.

Asyistent implements several different techniques to make the conversation natural. One of them is using colloquial<sup>2</sup> language in the dialogue. Another one is introducing variety to the conversation script. Even though all the dialogue is predetermined in the program code, for most utterances there exist a few (usually 3) varieties carrying the same meaning; at appropriate points in the conversation, the system chooses one at random. All the options are roughly the same in terms of semantics and register, but this lets the program avoid repetition. Most of all, the program's utterances come with a few second's delay after the user's answers; we decided to set the delay to two seconds in most places. This serves to imitate an actual human interlocutor and was inspired by online chatting applications. Usually when using a communicator, the other person responds after a few moments that it takes them to think about the answer and type it into the application. Asyistent is technically capable of responding immediately, but the delays have been included into the code so that the conversation would feel more natural.

To minimize the number of errors occurring due to invalid input of the data and to give the user some freedom in their word choices, Asyistent usually gives the user possible answer options for the question, but will recognize some other answers with similar meaning, which we input into the code. For example, even if the program expects the answer *ready*, it will still recognize the answer *ok*, which might seem more natural for the user.<sup>3</sup> Furthermore, to recognize the answer, Asyistent does not differentiate between upper and lower case letters, and only needs the first few characters of the word. This was implemented to avoid errors arising from spelling mistakes, which tend to occur with longer words. The only strictly restricted input format applies to writing down the tasks and the website url addresses; this was necessary for the program to be able to process them, which was unavoidable for this system.

The users' response to all the above features, whether positive or negative, was meant to give us some general idea about the direction in which we should develop the actual system.

Figure 2 illustrates the flow of conversation between the user and the system. The utterances in italics imitate the system's utterances. They convey the meaning of the actual utterances that were too long to include in the figure. Dotted arrows mark transitions between stages performed for every task on the workplan except for the last one.

The number of turns in the conversation varied depending on the number of tasks input by the user, but generally stayed within the 45-60 range.

<sup>2</sup> Although the level of formality in communication may greatly differ for a human assistant depending on their relationship with their boss, we opted for colloquial utterances to create the feeling of familiarity with the user.

<sup>3</sup> All the options (three on average for every possible utterance) were written manually into the code by the authors.

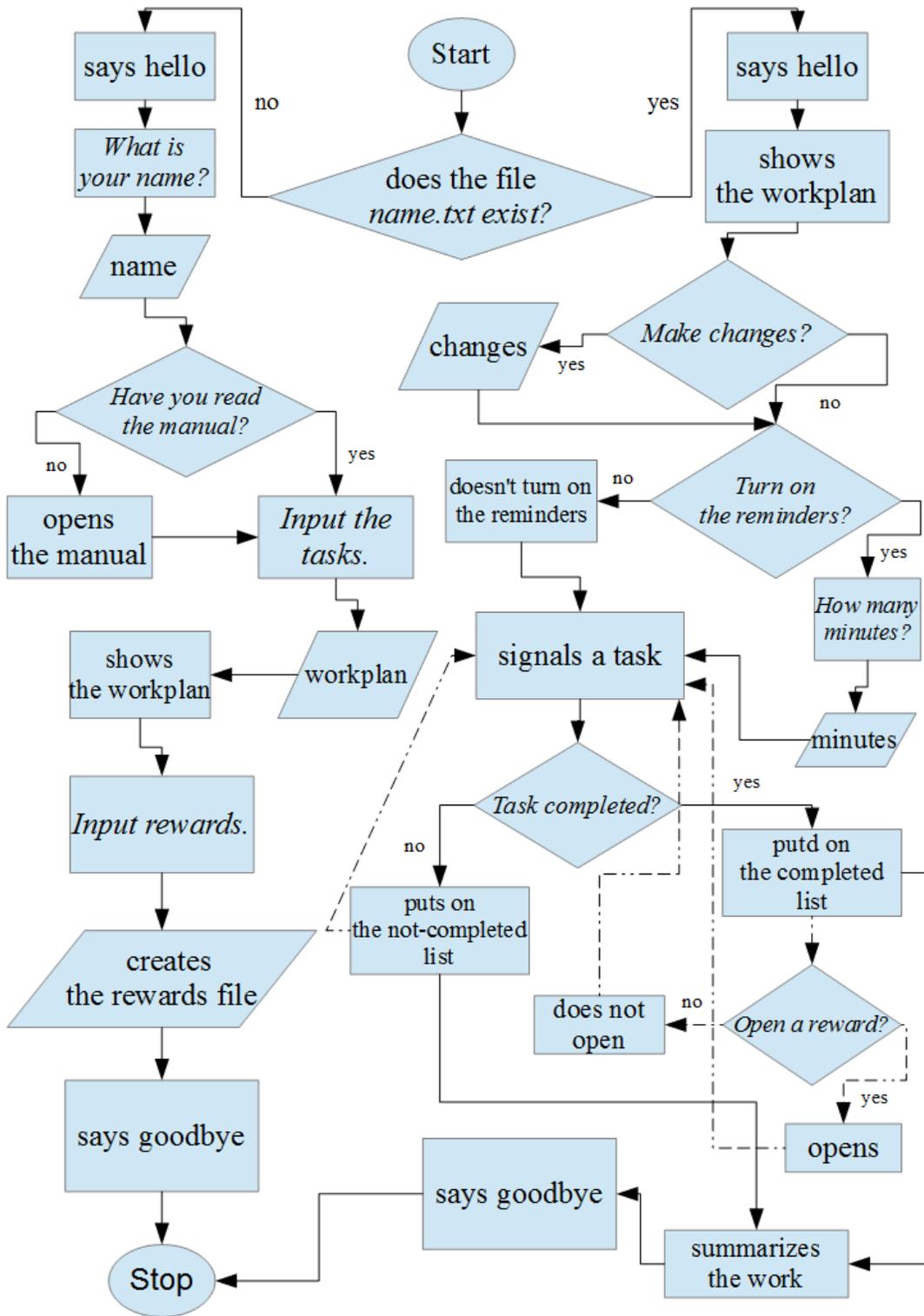


Figure 2: Cooperation with Asystem.

## 4 Evaluation

The program was evaluated through a user survey that came with the system's package. The users were allowed a few days to test the program and then were asked to complete the survey. Whenever there were issues with the program's functioning, the users were allowed to contact the author for help. However, the testing itself was not supervised; the users installed Python and launched the program on their own personal computers, and interacted with it without any assistance from the author.

There were seven users (5 females and 2 males) aged 21-30 who completed the evaluation survey. Six of them had previous experience with Python, which may have helped them handle cooperation with Asystemt. The participants were not paid for their efforts. Moreover, while we recognize the need of a bigger-scale evaluation, no other participants were available at the time of conducting the tests.

Asystemt was evaluated on a 5-point scale, with 1 being the lowest and 5 being the highest score, on the following criteria:

- naturalness of interaction;
- helpfulness;
- flexibility of dialogue;
- friendliness;
- ability to motivate the user to work;
- quality of the cooperation.

All the criteria were explained in detail in the survey to avoid confusion. The following paragraphs include a rather faithful paraphrase of the wording used in the user survey for the explanations of each criterion.

Naturalness of interaction meant being able to imitate a human interlocutor. Most users gave Asystemt rather high ratings in this regard. Four users gave it 4 points and one user gave it 5 points. The other two people rated it at 3 points (mean value: 3,9 points).

The program was regarded as helpful if the users felt that it had useful features making their work easier and if it did not disturb them. This aspect received mixed ratings. Three people gave the program 5 points, two 4 points and the other two 3 points and 1 point respectively (mean value: 3,9 points).

Flexibility of the conversation, defined as the dialogue not giving the impression of being strictly pre-defined, was the lowest rated aspect. Three people gave the system 2 points, one person 3 points, two 4 points and only one person gave it 5 points (mean value: 3,1 points).

Friendliness, on the other hand, was rated best. The program was regarded as friendly if it was able to create a nice working atmosphere and assisted the user to the best of its abilities. Four people gave it 5 points and the other three 3 points (mean value: 4,1 points).

Asystemt was also rather motivating. This was measured by the user's personal impression that they completed more tasks compared to when working alone. Three people gave the program 5 points, two 4 points and the other two 3 points and 1 point respectively (mean value: 3,9 points).

The last criterion – the quality of the cooperation – was measured by whether the program was a good assistant and whether the cooperation went smoothly. This is another aspect that received mixed ratings. Three people rated it at 5 points, two 3 points and the other two 2 points and 1 point respectively (mean value: 3,4 points).

The survey also contained a question whether the users would like to work with Asystemt in the future. Four people answered positively. According to them, Asystemt has an easy user interface, provides motivation and is useful. The three people who would not work with Asystemt again claimed that the user interface was unattractive graphically and that there occurred many errors during the work. Some errors were caused by software incompatibility and will be discussed in more detail in the next section. Other problems occurred when the user input a utterance the system did not recognize or if the workplan was not input in the format specified previously in the manual.

Table 1 presents the overview of the user's ratings.

The survey also included two open questions regarding the user's opinion about the cooperation and about the issues with the program. Most often, the users described errors arising from the incompatibility between Asystemt and their version of the text editor, which resulted in the data not getting written into files and thus being unusable. Moreover, an error meant that the program needed to be relaunched, which was rather tiresome. One user indicated that they found some instructions of the program unclear. Furthermore, on occasion the users' answers were unreadable for the program and as such generated errors or misunderstandings. For example, the program recognized a positive answer as a negative one or did not recognize the answer at all, so it remained idle until a proper answer was typed in. Yet another user indicated that, despite clear instructions, they turned off the program during work and left the house, hoping to be able to resume the cooperation after returning. This was not possible for this version of the program, so the user's satisfaction dropped significantly.

Positive comments praised Asystemt for being easy to use, helpful with organizing work and motivating. One user complimented the reward system of opening a website of choice, because it imitated this user's usual habit of only switching to relaxing activities after having completed a task.

The users' comments included suggestions about adapting the program for various operating systems and mobile devices such as smartphones. Other proposed improvements regarded the graphical form of the user interface and the possibility to plan more than one day ahead, which was not included in this version of the system. One of the users also advised to reduce the number of conversation turns and to include the feature of turning off the delays, because the dialogue becomes tiresome during consecutive launches of the program.

## 5 Discussion and Further Developments

The results of the survey indicate that the system was rated rather well. All the aspects received a mean score above 3 points, which was the middle of the scale, and four of six

No	Age	Sex	Experience with Python	Naturalness of interaction	Helpfulness	Flexibility of dialogue	Friendliness	Motivating to work	Quality of cooperation	Would use again
1	27	F	yes	5	5	2	5	5	5	yes
2	29	M	yes	3	3	2	3	3	3	no
3	30	M	yes	4	4	5	3	4	2	no
4	28	F	yes	4	5	4	5	5	5	yes
5	28	F	yes	4	1	3	3	1	1	no
6	21	F	yes	4	5	4	5	4	5	yes
7	27	F	no	3	4	2	5	5	3	yes
				Mean	3,9	3,9	3,1	4,1	3,9	3,4
				Median	4	4	3	5	4	3
				Mode	4	5	2	5	5	5

Table 1: Results of Asystem’s evaluation.

criteria had a mode value of 5, which is the highest score. Especially high ratings were given for helpfulness, friendliness, being motivating and the quality of the cooperation. Naturalness of interaction also scored high. This means that, while some improvements are necessary, some features of the system can be implemented in the actual program to increase user satisfaction. Moreover, most users would like to work with Asystem again in the future. Therefore, it can be concluded that the program did well in its role as an assistant. We also saw that users responded well to a more interactive calendar application, which is an important finding for our future work.

However, not all users were satisfied with the program. The users who gave it the lowest ratings were also the ones who encountered most errors. The errors were mainly caused by the incompatibility of the program with some versions of text editors – which Asystem used to store the data – and with Mac OS, as it was written for Windows.<sup>4</sup> As such, their dissatisfaction most likely stems from technical issues rather than from the poor performance of the program as an assistant and consequently has no bearing on the main topic of the research, which was testing users’ preferences about an electronic assistant. Moreover, User 5 in particular stood out as the most dissatisfied with the program, and the ratings given by this user do not correspond to ratings given by other users regarding the respective criteria. It is possible that this user’s ratings were particularly low because they encountered especially many errors with the program and as such can be seen as unusual. Other users gave the system rather different ratings, as is reflected in the median and mode values given in Table 1. Removing User 5’s ratings from statistics raises mean scores on helpfulness, friendliness and quality of cooperation by as much as 0,4 points each. This paints a different picture of the users’ general satisfaction with the

program. However, we still recognize the need to eliminate the described issues.

The lowest scores were given for flexibility of the conversation. Since the system was rather simple, this was a necessary cost. The comments about the structure of the conversation – like there being too many turns or the initiative being entirely on the system – indicate possible directions of developing the actual system. To increase the flexibility of the conversation, the program could, for example, use an annotated corpus of utterances and choose its questions and answers based on the flow of the conversation at any given moment. Another possible solution would be to make the program gather and store user data like name, previous workplans, completed and failed tasks, user preferences regarding work, and so on, and use this information in all further conversations. This would also mean modifying the structure of the dialogue accordingly.

Other potentially useful features would be displaying the workplan on command at any point during the work, turning the reminder functions on and off whenever desirable, and providing a graphic representation of the user’s progress. The last feature is especially important for adapting the program to smartphone operating systems. While the system described in this paper required working with a computer, it seems reasonable to assume that most users would like a more portable program, perhaps in the form of a smartphone application. This need came especially clear through the comments of the user who terminated the program mid-work to pursue outdoor activities. Putting the program on a smartphone would enable users to report any progress with work on the spot without having to wait to get home and turn on the computer to connect to the program’s database.

The information gathered from the user survey helped decide main directions of development for the future system. Those include increasing flexibility of the conversation and

<sup>4</sup> There was only one user who attempted to launch Asystem on Mac OS. After they notified the authors, the program’s code was

adapted in their version of the package, but as it took several trials to work properly, the user’s satisfaction dropped.

of the input data (for example by utilizing a large corpus), reducing the number of turns in the dialogue, devising a graphically attractive user interface and adapting the program to various operating systems, especially those meant for smartphones. Another crucial development will be allowing the user to plan for more than one day, for example for a week or for a month, which was not possible in the simple testing program used for this research. Moreover, the system should have a feature similar to Timeful's ability to plan the tasks itself instead of putting most of the planning work on the user. Our experiment, while simple and small-scale, helped us identify and understand all the users' needs described above.

## 6 Remarks on Motivating Techniques

One of the main goals of our future system is to be able to motivate the user to complete the planned tasks. Although the ability to do this was one of the highest rated aspects of the current version of the system, this feature can and must still be improved. One of the possible directions of improving this aspect of the program could be including a motivating module that adapts to the user's individual needs. Motivation is a complex psychological concept and as such different people are motivated in different ways. For example, some people may be motivated by the friendly approach Asystent had, but others may need a more strict assistant. We understand this as a difference between the system playing the part of a supportive friend and a strict boss, and implementing the aforementioned difference as utilizing different utterances taken from the corpus. Perhaps for the former group the system could praise them for completing a task while the latter group would prefer a system that rebukes them a lot for being late with their work (although those two are not mutually exclusive, what we mean here is the different percentage of "praises" and "rebukes" in the system's utterances for both groups). Another potential discrepancy could arise between users who work slowly but steadily throughout a prolonged period of time and those who are motivated by time pressure and only complete all the work right before the deadline. For those users the system would plan their tasks on the timeline accordingly. Finally, some users might prefer an assistant who quietly records their progress, while others might be more motivated by a program that reminds them about the tasks multiple times a day. For the latter group, the system's reminder functions would be much more active than for the former group. All these illustrate the motivating factors that we would like to take into consideration for our future program, although this list is by no means complete.

To understand each user's preferences, the system would need to track information like: what percentage of the tasks was completed? Does the user work every day or do they skip some days? Does the user use the reminder functions? Does the user catch up with overdue work or does it keep piling up? Is the user's progress stable for a longer period of time or is it different every day? Does the user respond well (i.e. do they make more progress) to the current verbal strategy (for example "praises" versus "rebukes")? What is the time

distance between the completion of a task and its deadline?, and so on.

It is crucial that the system tailors its motivating methods to the individual user. This will be achieved by creating a general list of motivating factors and allowing the system to adjust their weights based on observing the user's activity and productiveness. While this might be difficult without relying on self-reports from the users, our system could calculate the user's effectiveness as a function of the percentage of completed tasks and time that it took to do the work based on the input from the user, thus providing a more stable tool of evaluating the progress. The system could also periodically ask the user to rate the level of their motivation and, while comparing the self-reported motivation level to the observed effectiveness level, the system could adjust its motivating techniques until the results are optimal.

To be able to compare the two, we will need to obtain a general idea about how the self-reported motivation level relates to the observed effectiveness level. Perhaps a good solution would be to first release a version of the system that lacks any motivating techniques. We could observe the correlation rates between motivation and effectiveness levels in users not influenced by our motivational module and establish an equation based on the gathered data. Then we could use this version of the program as the baseline for comparison to the end-goal program that will implement motivating techniques. In this way we could observe whether the motivating module made a difference in the user's performance.

## 7 Conclusions

The research described in this paper served as a testing phase for different ideas for the development of an electronic assistant. We presented a simple dialogue system acting as an assistant for the user in their everyday work. The program served as a testing tool for building an actual electronic assistant and was evaluated in detail through a user survey. The results clearly show that it was able to accomplish its main goals, which were conducting a friendly and natural conversation with the user and being helpful and motivating in completing the pre-planned tasks. The system was rated well as an assistant, which was the primary goal of the research. It scored lower as a dialogue system, especially on the flexibility of the conversation. However, this was unavoidable with such a simple system. The evaluation helped indicate numerous potential developments, which will be implemented in the actual program in the future.

## References

- [Agnatis *et al.*, 2016] Konstantinos Agnatis, Anastasios Alexiadis, and Ioannis Refandis. Intelligent calendar applications: A holistic framework based on ontologies. *International Journal of Artificial Intelligence*, 14(2): 1-22, 2016.
- [Androutsopoulos and Aretoulaki, 2003] Ion Androutsopoulos and Maria Aretoulaki. Natural language interaction. In Ruslan Mitkov, editor, *The Oxford handbook of computational linguistics*, pages 629-649. Oxford University Press, New York, 2003.
- [Apple, 2017] Siri. Retrieved May 2017 from <https://www.apple.com/ios/siri/>
- [Bank *et al.*, 2012] Jacob Bank, Dan Ariely, and Yoav Shoham. Timeful. Retrieved September 2015 from [support.timeful.com](http://support.timeful.com)
- [Bickmore and Picard, 2005] Timothy W. Bickmore and Rosalind W. Picard. Establishing and Maintaining Long-Term Human-Computer Relationships. *ACM Transactions on Computer Human Interaction (ToCHI)*, 59(1): 21-30, 2005.
- [Bisht, 2016] Jitender Bisht. Techniques for visual integration of meeting space in calendar systems. U.S. Patent No. 9,436,934, 6th September 2016.
- [Business Insider, 2016] 80% of businesses want chatbots by 2020. Retrieved May 2017 from <http://www.businessinsider.com/80-of-businesses-want-chatbots-by-2020-2016-12/?r=AU&IR=T>
- [DARPA, 2017] CALO. Cognitive Assistant that Learns and Organizes. Retrieved February 2017 from <http://www.ai.sri.com/project/CALO>
- [Gustafson, 2013] Pamela K. Gustafson. System and method for recording an audio to-do list and adding to-do list items to a calendar application. U.S. Patent No. 8,369,493, 5th February 2013.
- [Higashinaka *et al.*, 2014] Ryuichiro Higashinaka, Kenji Imamura, Toyomi Meguro, Chiaki Miyazaki, Nozomi Kobayashi, Hiroaki Sugiyama, Toru Hirano, Toshiro Makino, and Yoshihiro Matsuo. Towards an open-domain conversational system fully based on natural language processing. In *Proceedings of COLING 2014, the 25<sup>th</sup> International Conference on Computational Linguistics: Technical Papers*, pages 928-939, Dublin, Ireland, August 2014. Association for Computational Linguistics.
- [Hurst and Morris, 2016] Evan H. Hurst and Aires Alisa Morris. Electronic device display screen with graphical user interface for a calendar application. U.S. Patent No. D754,692, 26th April 2016.
- [Jurafsky and Martin, 2000] Daniel Jurafsky and James H. Martin. *Speech and language processing: An introduction to natural language processing, computational linguistics and speech recognition*. Prentice Hall: New Jersey, 2000.
- [Leech and Weisser, 2003] Geoffrey Leech and Martin Weisser. Pragmatics and dialogue. In Ruslan Mitkov, editor, *The Oxford handbook of computational linguistics*, pages 136-156. Oxford University Press, New York, 2003.
- [Liu and Lane, 2016] Bing Liu and Ian Lane. Joint online spoken language understanding and language modelling with recurrent neural networks. arXiv preprint arXiv:1609.01462, 2016.
- [Meisels *et al.*, 2014] Adrienne C. Meisels, Stephen J. Coughlin, and W. Karl Renner. Reserving a time block in a calendar application to account for a travel time between geographic locations of appointments. U.S. Patent No. 8,712,810, 29th April 2014.
- [Microsoft, 2017] Cortana – Meet your personal assistant. Retrieved May 2017 from <https://www.microsoft.com/en-us/mobile/experiences/cortana/>
- [Refandis and Alexiadis, 2011] Ioannis Refandis and Anastasios Alexiadis. Deployment and evaluation of SelfPlanner, an automated individual task management system. *Computational Intelligence*, 27(1): 41-59, 2011.
- [Rosenberg *et al.*, 2014] Gregory A. Rosenberg, Gregory J. Armanini, Ethan H. Diamond, and Andrew T. Garcia. Calendar event, notification and alert bar embedded within mail. U.S. Patent No. 8,745,141, 3rd June 2014.
- [Sadek and de Mori, 1998] David Sadek and Renato de Mori. Dialogue systems. In Renato de Mori, editor, *Spoken dialogues with computers*, pages 523-561. Academic Press, London, 1998.
- [Scott and Kamp, 1997] Donia Scott and Hans Kamp. Discourse modelling. In Ronald Cole, Joseph Mariani, Hans Uszkoreit, Annie Zaenen, and Victor Zue, editors, *Survey of the state of the art in human language technology*, pages 201-204. Cambridge University Press, Cambridge, 1997.
- [Yan *et al.*, 2016] Rui Yan, Yiping Song, Xiangyang Zhou, and Hua Wu. Shall I be your chat companion?: Towards an online human-computer conversation system. In *Proceedings of the 25<sup>th</sup> ACM International Conference on Information and Knowledge Management*, pages 649-658, Indianapolis, USA, October 2016. ACM.