Proceedings of
Umeå's 24th Student Conference in Computing Science

USCCS 2020

S. Bensch, J. Eriksson, K. Främling, T. Hellström (editors)

UMINF 20.01
ISSN-0348-0542

Department of Computing Science
Umeå University

Preface

The Umeå Student Conference in Computing Science (USCCS) is organized annually as part of a course given by the Computing Science department at Umeå University. The objective of the course is to give the students a practical introduction to independent research, scientific writing, and oral presentation.

A student who participates in the course first selects a topic and a research question that he or she is interested in. If the topic is accepted, the student outlines a paper and composes an annotated bibliography to give a survey of the research topic. The main work consists of conducting the actual research that answers the question asked, and convincingly and clearly reporting the results in a scientific paper. Another major part of the course is multiple internal peer review meetings in which groups of students read each others' papers and give feedback to the author. This process gives valuable training in both giving and receiving criticism in a constructive manner. Altogether, the students learn to formulate and develop their own ideas in a scientific manner, in a process involving internal peer reviewing of each other's work and under supervision of the teachers, and incremental development and refinement of a scientific paper.

Each scientific paper is submitted to USCCS through an on-line submission system, and receives reviews written by members of the Computing Science department. Based on the review, the editors of the conference proceedings (the teachers of the course) issue a decision of preliminary acceptance of the paper to each author. If, after final revision, a paper is accepted, the student is given the opportunity to present the work at the conference. The review process and the conference format aims at mimicking realistic settings for publishing and participation at scientific conferences.

USCCS is the highlight of the course, and this year the conference received 8 submissions (out of a possible 11), which were carefully reviewed by the reviewers listed on the following page.

We are very grateful to the reviewers who did an excellent job despite the very tight time frame and busy schedule. As a result of the reviewing process, 4 submissions were accepted for presentation at the conference. We would like to thank and congratulate all authors for their hard work and excellent final results that are presented during the conference.

We wish all participants of USCCS interesting exchange of ideas and stimulating discussions during the conference.

Umeå, 8 January 2020

Suna Bensch
Jerry Eriksson
Kary Främling
Thomas Hellström

Organizing Committee

Suna Bensch
Jerry Eriksson
Kary Främling
Thomas Hellström

With special thanks to the reviewers

Sule Anjomshoae
Suna Bensch
Paolo Bientinesi
Lili Jiang
Monika Jingar
Anna Jonsson
Timotheus Kampik
Julian Alfredo Mendez
Michele Persiani
Esteban Guerrero Rosero

Table of Contents

Gendered robot voices influencing trust - towards a robot recommendation system

Zhongkai Cai

Department of Computing Science
Umeå University, Sweden
mrc18zci@cs.umu.se

Abstract. This paper explores a factor thought to influence user trust towards a robot: gendered robot voices. In particular, we explore whether the gender of robots' voices affects human trust towards to a robot recommendation system. Results from a 2 (female voice vs male voice) × 1 (trust) study (n = 20) show that a user's trust in a robot can be influenced by the recommendation from a robot with a gendered voice. Moreover, we find that users tend to lend more trust to a robot whose voice is of the same gender as their own. These findings have implications for education and health promotion in relation to HRI (human robot interaction) and call for further investigation into the development and maintenance of trust between robot and user.

1 Introduction

The importance of trust has long been recognized in behavioral science, psychology, and management. It is also a key element for effective team work when the team is composed purely of humans, or humans and machines [11, 12].

In human-human interaction, people are more likely to follow the recommendations of a person they trust [7]. Robots are machines and as such do not have a gender. However, many of the gender-related perceptions and expectations formed in human-human interactions can be easily transferred to human-robot interactions [3]. Additionally, trust is viewed as a tool that recommendation systems can use to increase their probabilities of convincing a user to select the recommend choice [14].

Voices contain a rich amount of information. Research has revealed that voices can provide clues on factors pertaining to the speaker's perceived attractiveness [9], personality [20], sexual orientation [17], health [4] and intelligence [16].

For these reasons, we employ a robot recommendation system to investigate trust through decision making. In particular, we explore whether the gender of robot's voices affects human trust towards to robot recommendation system. To answer this research question, participants are asked to complete a task based on instructions provided by a Pepper robot. We hypothesize that:

– H1: Users are more likely to accept the recommendation of a robot with a female voice than of a voice with a male voice.

S. Bensch, J. Eriksson, K. Främling, T. Hellström (Eds.): Umeå's 24th Student Conference in Computing Science USCCS 2020, pp. 1–9, January 2020.

- H2: Users are more likely to accept the recommendation of a robot with a male voice than of a voice with a female voice.
- H3: Users tend to lend more trust to robot whose voice is of the same gender as their own.

In this paper, Section 3 details the experiment. Section 4.1 describes the results of the study, and Section 4.2 discusses these results and future research.

2 Related work

Researchers in human-robot interaction have become increasingly interested in the factors that influence people's trust of robots in a variety of contexts: UAVs (Unmanned Aerial Vehicles) [6], household assistant robots [15], autonomous cars [19], and tour guides [1].

A majority of research into human-robot trust has focused on performance-based trust. Robot performance is considered to be the most influential factor in human-robot trust according to a review on trust in HRI[8]. Researchers have successfully employed models of competence-based trust of robots used in robot decision making [2] and evaluations of human-robot team effectiveness [6].

An area of studies within Human-robot interaction (HRI) explores the overlap between robotics and gender studies to determine how gender bias in human-human interaction carries over to HRI. In [10], the authors explored methods for conducting studies on gendered Voice-User Interfaces (VUI), a speech application interface that enables human-robot communication through Text-To-Speech technology. In [18], the researchers compared a synthesized robotic voice to some natural, gendered human voices and found an initial hesitation by humans when interacting with robotic-sounding robots.

Furthermore, In [13], the researchers suggested that a male voice might produce expectations and responses based on stereotypes about males, whereas a female voice might generate responses based on stereotypes about females. In [5], the authors reported on in-group gender bias for psychological closeness to robots and indicated that people tended to feel more positively towards a same-gendered robot.

3 Method

In this section, we describe a user study that investigates the effects of gendered robot voices on the trust a human towards to robot recommendation system has in a robot within the context of a competitive game. Participants played the game with a robot where they tried to gain points as much as they can by answer questions correctly in 5 minutes.

3.1 Design

In this study, the independent variables are *female voice* and *male voice*. The dependent variable is *trust*.

3.2 Materials

A. Robot

The robot used for this experiment is a pepper robot: a wheel-based humanoid robot. The robot has a pair of 5-degrees of freedom(DOF) arms and 2-DOF hands that can perform human-like gestures. A touch screen build in the front of the robot's chest is used to display the Graphic User Interface(GUI) of the recommendation system in this experiment. The only difference in this experiment is the gender of the robot's voice.

Fig. 1. Pepper robot used for the experiment.

B. Recommender system GUI

In this experiment, Pepper's answers come from a content-based recommendation engine and its recommended results are not 100% accurate. This means that the correctness of Pepper's recommended answers is uncertain for the user.

Fig 2 shows the process flow of our system. The system starts with a $Welcome$ page. This is followed by an $Information$ page which present and read the followed message: *" Hello, I am Pepper. In this experiment, we ask you to answer some questions. There are more than 40 questions in this system and you need to answer as many questions as you can within 5 minutes. Moreover, I will answer these questions together with you and I will recommend an answer to you for each question. You need to hear my answer and decide whether to believe my answer or not. When you try to answer another question, you must click the YES or NO button to jump to another question page. For each question, if your answer is correct you will get 10 points. At last, you will have a total score. The user with the highest total score will win a gift."*

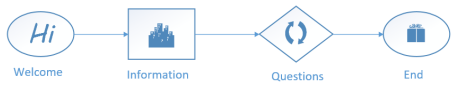

Fig. 2. Process flow of the experimental stages.

Fig 3 shows a sample of a question page. It shows a question and plays a sound with a gendered robot voice: *"Here is my recommendation, I highly encourage*

you to choose [Item Name].“ During this speaking time, it is required no action from the participant. When the speaking is over, you can click the YES or NO button to jump to another question page. If you click YES, it means you trust Pepper's answer and use it as your answer; If you click NO, it means you do not trust Pepper's answer and your answer is different from Pepper's answer.

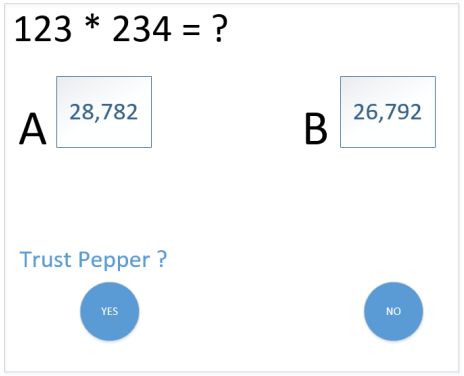

Fig. 3. A sample of question page.

The interaction continues, participants will continue to answer questions in the same way. When the 5-minute answer time is over, the system will jump to the end page. The *end* page will show the total score of the participant.

C. Trust Metrics

Quantifying the factors of trust applicable to HRI can be divided into two categories: robot intentions and robot performance [11]. In this study, we use a trust questionnaire to measure a participant's degree of trust. We normalised the scores according to the questionnaire instructions gathering values between -3 to +3, where +3 means a completely positive perception of the dimension by the participants and -3 means a completely negative perception.

3.3 Participants

We collected data from 20 participants recruited from the Umea University campus (10 male, 10 female). Participants' ages ranged from 18 to 30 ($\mu = 21.65, \sigma = 2.1$). All were former or current undergraduate students from a range of fields of study including computing science, physics and mathematics. 30% of the participants stated that they had a previous interaction with a robot in the past. The experiments took place in a controlled environment (a classroom) with one participant, the robot and the experimenter present.

3.4 Experiment condition

The experiment includes two conditions:

1. The participant interacts with the robot with a female voice and answer some questions in 5 minutes.
2. The participant interacts with the robot with a male voice and answer some questions in 5 minutes.

3.5 Experiment procedure measures

The experiment include two main stages:

1. Have an interaction with the robot in one of the two assigned conditions. Each participant does the experiment twice, once using each voice.
2. Completion of some questionnaires.

In order to assess participants' trust to the robot recommendation system, we analyzed the participant's choices and survey responses from the post-experiment questionnaire.

4 Results and Discussion

4.1 Results

The results acquisition was accomplished by a questionnaire that the participants filled in after the end of the experiment. In the beginning, they were asked about participant's gender, their age and any sort of previous interaction with a robot. Furthermore, a question about their overall trust with the Pepper robot on a scale from -3 to +3 helped the researcher identifying the participant's overall trust towards the robot. Finally, they were asked to rate the two experiments based on their trust degree on a scale from -3 to +3. The raw results is summarised in Table 1, where 'F' corresponds to females and 'M' to males. For better visualization, we can use a Box and Whiskers chart representation the data which is presented in Fig 4.

In order to examine if there are statistically significant differences between these two experiments, the two-tailed paired t-tests were performed when checking for significant differences between the two experiments. For all tests, the significance level was chosen to be 0.05. And we got that the p-value = 0.63 between the first experiment and the second experiment. This result indicate that there is no significant difference between these two experiments.

In contrast, after grouping experimental data by gender of the participant, the two-tailed paired t-tests resulted in significant differences between all paired examined experiments with p-values < 0.05 between 1^{st} and 2^{nd} experiment. This result indicates that there is a significant bias between both groups to preferentially trust robots whose voice is of the same gender as their own.

Finally, the final test results when comparing people who had a previous interaction with a robot or not were somewhat surprising since there was not a significant difference between the two experiments, as shown in Table 2, where 'YES' corresponds to previous interaction and 'NO' to not previous interaction. Apparently, participants regardless of previous interaction with a robot or not, seem to have the same attitude towards it in different gendered voice.

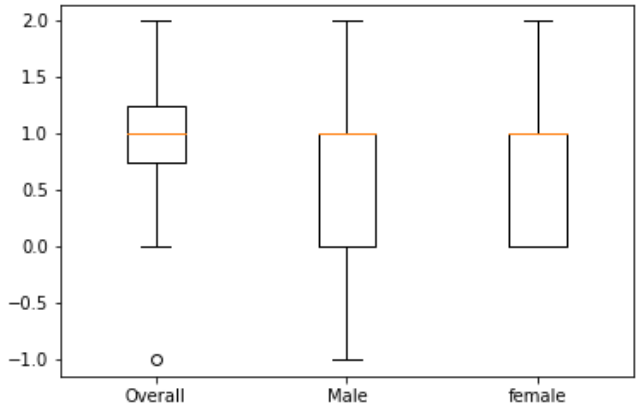

Fig. 4. Box and Whiskers chart of raw results.

Participant	Age	Gender	Previous Interaction	Overall experience	1^{st}Experiment	2^{nd}Experiment
					Female voice	Male voice
1	20	M	YES	2	1	2
2	21	M	NO	1	1	1
3	21	M	YES	0	0	1
4	26	F	NO	-1	1	0
5	20	F	NO	1	1	1
6	24	M	YES	2	1	2
7	22	M	NO	1	0	1
8	21	M	NO	1	0	1
9	19	F	NO	2	2	1
10	22	F	NO	1	1	1
11	19	F	NO	1	1	0
12	23	F	NO	2	2	1
13	22	M	YES	0	-1	0
14	20	F	NO	0	0	0
15	21	F	NO	1	1	0
16	19	M	YES	1	0	1
17	24	M	NO	1	0	1
18	26	F	NO	2	2	1
19	22	M	NO	1	0	1
20	21	F	YES	0	1	0
Mean	21.65			0.95	0.70	0.80
SD	2.1			0.83	0.80	0.62

Table 1. Raw experimental results.

	female voice	**male voice**
F.	p > 0.05	p > 0.05
M.	p > 0.05	p > 0.05

Table 2. p-values when comparing people with a previous and not previous interaction.

4.2 Discussion

From the experimental results, we found full support for one out of three hypotheses mentioned in Section 1. The first and second hypothesises made are not supported because, we ran a t-test, though found no evidence to support a significant difference between these two experiments p=0.63.

Regarding the 3^{rd} hypothesis, after filtering experimental data by gender of the participant, we found that the male participants were less trusting of female voice agents while female participants were more trusting of the female voice agent. The results from our experiment provided strong support for our hypothesis (p¡0.05).

Finally, the comparison based on previous interaction or not did not give evidence that affects the participant's trust degree.

5 Conclusions

In this paper, we explores a factor thought to influence user trust towards a robot: gendered robot voices. Results from a 2(female voice vs male voice) × 1(trust) study (n = 20 participants) shows that there is no proof to say that users are more likely to accept the recommendation of a robot with a female voice than of a voice with a male voice and there is no proof to say users are more likely to accept the recommendation of a robot with a male voice than of a voice with a female voice. Moreover, we find that users tend to lend more trust to robot whose voice is of the same gender as their own. These findings have implications for education and health promotion in relation to human-robot interaction and call for further investigation into the development and maintenance of trust between robot and user.

6 Limitations

As with most user study, there are lots of limitations that need to be acknowledged. Firstly, the interface design of a recommendation system may impact the perceived credibility and overall trust of the system. Furthermore, our experiment was conducted in a controlled environment which may has affected the results that would have been obtained if it took place in the real world. Finally, our findings are limited to a specific age range (Mean = 21.65) and a small number of participants. With a greater amount of participants, more data would have been collected and the results may have been more dominant.

7 Future work

Future research in this area should continue to investigate other factors that necessary for trust in human robot interaction. Moreover, it would be beneficial to investigate how the development and maintenance of trust between a robot and user.

8 Acknowledgement

I would like to thank all the people who helped make this paper into what it is today through their feedback and peer review. I also want to thank all of those who participated in the study.

References

[1] Sean Andrist, Erin Spannan, and Bilge Mutlu. Rhetorical robots: making robots more effective speakers using linguistic cues of expertise. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pages 341–348. IEEE Press, 2013.

[2] Min Chen, Stefanos Nikolaidis, Harold Soh, David Hsu, and Siddhartha Srinivasa. Planning with trust for human-robot collaboration. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 307–315. ACM, 2018.

[3] Meia Chita-Tegmark, Monika Lohani, and Matthias Scheutz. Gender effects in perceptions of robots and humans with varying emotional intelligence. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 230–238. IEEE, 2019.

[4] Brian R Duffy. Anthropomorphism and the social robot. *Robotics and autonomous systems*, 42(3-4):177–190, 2003.

[5] Friederike Eyssel, Laura De Ruiter, Dieta Kuchenbrandt, Simon Bobinger, and Frank Hegel. If you sound like me, you must be more human: On the interplay of robot and user features on human-robot acceptance and anthropomorphism. In *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 125–126. IEEE, 2012.

[6] Amos Freedy, Ewart DeVisser, Gershon Weltman, and Nicole Coeyman. Measurement of trust in human-robot collaboration. In *2007 International Symposium on Collaborative Technologies and Systems*, pages 106–114. IEEE, 2007.

[7] David Gefen. E-commerce: the role of familiarity and trust. *Omega*, 28(6):725–737, 2000.

[8] Peter A Hancock, Deborah R Billings, Kristin E Schaefer, Jessie YC Chen, Ewart J De Visser, and Raja Parasuraman. A meta-analysis of factors affecting trust in human-robot interaction. *Human factors*, 53(5):517–527, 2011.

[9] Susan M Hughes, Franco Dispenza, and Gordon G Gallup Jr. Ratings of voice attractiveness predict sexual behavior and body configuration. *Evolution and Human Behavior*, 25(5):295–304, 2004.

[10] Jaeyeol Jeong and Dong-Hee Shin. It's not what it speaks, but it's how it speaks: A study into smartphone voice-user interfaces (vui). In *International Conference on Human-Computer Interaction*, pages 284–291. Springer, 2015.

[11] John D Lee and Katrina A See. Trust in automation: Designing for appropriate reliance. *Human factors*, 46(1):50–80, 2004.

[12] D Harrison McKnight and Norman L Chervany. The meanings of trust. 1996.

[13] Clifford Nass, Youngme Moon, and Nancy Green. Are machines gender neutral? gender-stereotypic responses to computers with voices. *Journal of applied social psychology*, 27(10):864–876, 1997.

[14] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[15] Maha Salem, Gabriella Lakatos, Farshid Amirabdollahian, and Kerstin Dautenhahn. Would you trust a (faulty) robot?: Effects of error, task type and personality on human-robot cooperation and trust. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 141–148. ACM, 2015.

[16] Harriet MJ Smith, Andrew K Dunn, Thom Baguley, and Paula C Stacey. Concordant cues in faces and voices: Testing the backup signal hypothesis. *Evolutionary Psychology*, 14(1):1474704916630317, 2016.

[17] Jaroslava Varella Valentova and Jan Havlíček. Perceived sexual orientation based on vocal and facial stimuli is linked to self-rated sexual orientation in czech men. *PloS one*, 8(12):e82417, 2013.

[18] Michael L Walters, Dag Sverre Syrdal, Kheng Lee Koay, Kerstin Dautenhahn, and R Te Boekhorst. Human approach distances to a mechanical-looking robot with different robot voice styles. In *RO-MAN 2008-The 17th IEEE International Symposium on Robot and Human Interactive Communication*, pages 707–712. IEEE, 2008.

[19] Adam Waytz, Joy Heafner, and Nicholas Epley. The mind in the machine: Anthropomorphism increases trust in an autonomous vehicle. *Journal of Experimental Social Psychology*, 52:113–117, 2014.

[20] Miron Zuckerman and Robert E Driver. What sounds beautiful is good: The vocal attractiveness stereotype. *Journal of Nonverbal Behavior*, 13(2):67–82, 1989.

RTA in user studies, equal result with less work?

Mattias D.K. Hagberg

Department of Computing Science
Umeå University, Sweden
id14mhg@cs.umu.se, swenee.hagberg@gmail.com

Abstract. Retrospective think-aloud (RTA) protocol is a method widely used in user tests. A number of attempts to improve on the current protocol has already been made. Performing user tests can sometimes be hard to justify because the result of a user test is unknown before hand but the cost very much not. It's not surprising that approving an unknown investment is hard. Therefor, anything to make user tests easier is a win for everybody.
This paper follows an experiment that asks: *Does removing the test leader from RTAs test phase effect the amount of user problems found or the users time to complete the test?* In practical terms, is it possible to cut costs in RTA by not observing the user during the user test? And is this an improvement?
By comparing the result of a user test that includes observations during the test phase, with one that excludes observations the research question can be investigated.
The paper could not conclude whether the new RTA protocol gives better, worse or an equally good result. No statistical difference was found for time to completion nor amount of problems. The new RTA protocol might still be useful, future studies are advised to explore how it differs from RTA with other definitions of user problems, different test objects and different kinds of tasks.

1 Introduction

User tests are a great way to get insight into how users use e.g. a website, what they think about it and how they navigate through it. The user tests can, however, be an expensive and time consuming part of development.

The overarching goal of the paper is to find a cheaper way to perform user tests. This would benefit many parties and it can be done by investigating the question: *Does removing the test leader from the test phase in RTA effect the amount of user problems found or the users time to complete the test?* Dedicating resources to user testing can sometimes be difficult to justify because it's not clear what that benefit will be before hand, or there might not be enough resources to begin with. Anything to make user testing more easy is a win for all parties.

This paper focuses on a type of user test called Retrospective think-aloud protocol (RTA). Why specifically RTA was chosen is because there seems to be less research on it compared to another talk aloud test called CTA. Another

S. Bensch, J. Eriksson, K. Främling, T. Hellström (Eds.): Umeå's 24th Student Conference in Computing Science USCCS 2020, pp. 11–23, January 2020.

reason is that there have only been a few improvements to RTA as explained in 2.1. The paper follows an experiment that investigates whether or not the test leader can be removed completely from the test phase (see section 2 for an explanation of the different phases). The experiment compares two groups where Group 1 performs a RTA test and Group 2 performs the same test but without a test leader in the test phase. By performing this experiment we can evaluate how the presence of the test leader affects the user. If removing the test leader either improves or has no effect on the result there is an opportunity to cut costs on user tests and encourage more user testing from both companies and researchers. If no observations are needed the test leader is freed up to e.g. start up another user test or do other work.

In section 2 what RTA is and how it is used is explained followed by a bit of history for context. In section 3 the experiment that the paper follows is outlined together with how the result will be measured. There is also a list of tasks that the test contains together with the answers. Section 4 includes 3 tables with raw data and statistics. Finally, in section 5 the result is discussed, problems is presented and some future work is discussed together with an outline of a possible future experiment.

2 RTA

RTA is a type of user test in the family of talk aloud tests. As illustrated in Fig.1 there are more talk aloud protocols that are very similar to RTA but this paper will only focus on RTA. Similar to many user tests RTA is used to explore how a *user* does something. It is used to get a *users* perspective and to find out what and where problems occur for the *user*. The paper uses the term *user* as the person that is performing the test and is talking aloud, essentially our users is equivalent to test persons. The basic commonality between the talk aloud tests is that they all perform a test with a user that talks aloud about what the user is thinking during the test. What sets RTA aside from the other talk aloud tests is that RTA does not make the user talk about the test until the test is complete. In other words, RTA is when the user talks about the test after it is performed, hence the word retrospective in retrospective think-aloud.

The term *test leader* will be used to describe the person that interviews the user and starts up the test.

RTA can be divided into three phases:

1. **Intro.** The test is described, consent is given and tasks are explained to the user.
   E.g. Test leader: "You will be given a scenario and one task to perform, after that we will talk about the result. The screen will be recorded during the test. Do you consent?"
2. **Test phase.** The user performs the tasks while the test leader takes notes and observes the user.

E.g. User starts at the home page of the website and gets the task "You want to find information about X article". During this time the test leader observes that the user looks confused and writes this on a notepad.

3. **Verbalisation phase.** The user is interviewed. During the interview the user watches a video of the performed test to help remember what exactly was done during the test.
   E.g. Test leader: "Please explain what you did and why, you may play and pause the video of the recording". The video starts and as the user is explaining the test leader could ask "What happened here? You looked confused at this part." and note down the answer.

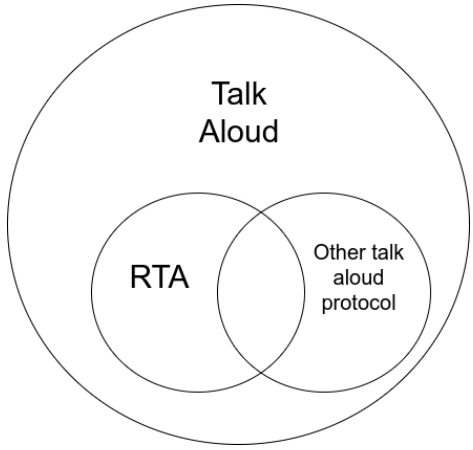

Fig. 1. RTA is a one of all the talk aloud protocols. There are other talk aloud protocols that are similar to RTA

As everything does, RTA has its pros and cons. A pro is that RTA is not affected by task complexity, unlike other talk-aloud tests, because the user is never disturbed while the tasks are performed. This gives us a result with one less source of bias. The cons is that the test as a whole will take longer, which makes it more expensive to perform. It also makes both the user and the test leader more tired which needs to be considered when designing a user test as found by Willis and McDonald [8].

2.1 History of RTA

The more general talk-aloud protocol began in psychology and has since then found itself adopted by, among others, the computer science field. Jørgensen with his paper in 1990 [5] is often cited as the paper that took the talk-aloud protocol from psychology into the computer science field as he concluded that the talk-aloud protocol is a valid way of gathering user data. Since 1990 the RTA protocol's validity as a method of extracting information has been confirmed by Haak, De Jong, Schellens [2] and Guan, Cuddihy, Ramey [1].

At first there was no video recording during RTA but one of the few improvements made to the RTA protocol is the addition of video as concluded by Guan, Cuddihy, Ramey [1] and validated by van Merriernboer, van Gog, Witte [4]. The video records the test and is played back during the verbalisation phase. The video acts as a cognitive cue and in general a cue is something to help the user remember what thoughts and actions the user took during the test phase. In theory it does not have to be limited to a video but until this day the only other variation has been video & eye tracking as tested by Guan, Cuddihy, Ramey [1]. It should be noted that in the same paper the participants was never showed the eye movements in the video, however, the result shows interesting information is hidden in the movements of the eye that can show what strategies the users use to complete the tasks.

Willis and McDonald [8] tried to improve the RTA protocol by switching order in the protocol. Instead of having the user completing the test and then continuing to the verbalisation, Willis and McDonald [8] compared the standard RTA with a version that had a verbalisation phase after each task. This resulted in more user data and same task completion but longer task times, more errors and more clicks. This provides evidence that a verbalisation for each task does in fact not improve the RTA protocol but more research is needed to conclude if this is anything general.

3 Method

The experiment was a navigation test where the users were tasked to find specific pages in a website. The test was performed on a computer, the screen was recorded and after the test the video of the recording was used as cue in the verbalisation phase. The experiment used a between subjects design, as written by Oeldorf-Hirsch [7], to test if time to completion or the amount of user problems changed when the test leader was removed during the test phase. The method "between subjects design" is when one group acts as a control group and another group performs the same procedure but with one variable changed. This method was used to avoid a carryover effect and to prevent that the user learned things between the different tests. Each user got a randomly assigned group to ensure a non biased result. The downside to this method is that more test users is needed since each user only performs either the first or the second test, never both.

Considering the comparison of the amount of problems found in Group 1 (RTA test) and Group 2 (RTA without test leader), the study has two possible outcomes:

If 1. Problems in Group 1 ¿ Problems in Group 2:
Removing the test leader does not improve RTA because that version finds less problems than the standard RTA.

If 2. Problems in Group 1 $\leq$ Problems in Group 2:
Removing the test leader is a valid option in RTA because that version finds more or equally many problems than the standard RTA.

When considering the time to completion found in Group 1 (RTA test) and Group 2 (RTA without test leader), the study has two possible outcomes:

If 1. Time in Group 1 ¡ Time in Group 2:
Removing the test leader does not improve RTA because that version is slower than the standard RTA.
If 2. Time in Group 1 $\geq$ Time in Group 2:
Removing the test leader is a valid option in RTA because that version is equally fast or faster than the standard RTA.

3.1 The experiment

The experiment consisted of two groups of users, Group 1 and Group 2. What separated the groups was whether or not the test leader was present to do observations during the test phase (see section 2 for an explanation of the different phases). In Group 1 the test leader was present and observing the user. In Group 2 the test leader gave instructions and when the user was ready to start the test leader left the user alone and waited outside the room. When the user was done with the tasks the user went and retrieved the test leader.

Below is an explanation of how the experiment was implemented in the different phases.

1. **Intro.** Introduce the test subject to the test, answer questions explain the verbalisation phase. The test subject gives verbal consent.
2. **Test phase.** The user starts the usability test at which point the test leader either 1. starts observations or 2. leaves the room.
3. **Verbalisation phase.** The user is interviewed. Test leader introduces the verbalisation phase again and starts the video. Both the test leader and the user may play/pause the video at will.

3.2 Test object and users

All users where students from Umeå University that spoke Swedish.

The test was conducted on this website: `http://www.ub.umu.se/`, Umeå University's Library home page, see Fig. 2. The test specifically used the Swedish version because there are differences between the Swedish version and the English version that affect the test. It should be noted here that soon after the experiment was performed the website changed a lot. The experiment accessed the website for the last time at 2019-Nov-22.

3.3 Metric of result

In order to make the experiment reproducible the metric used to define a user problem will be: "*Any clicks on links that do not take the users one step closer to the goal is considered a problem*".

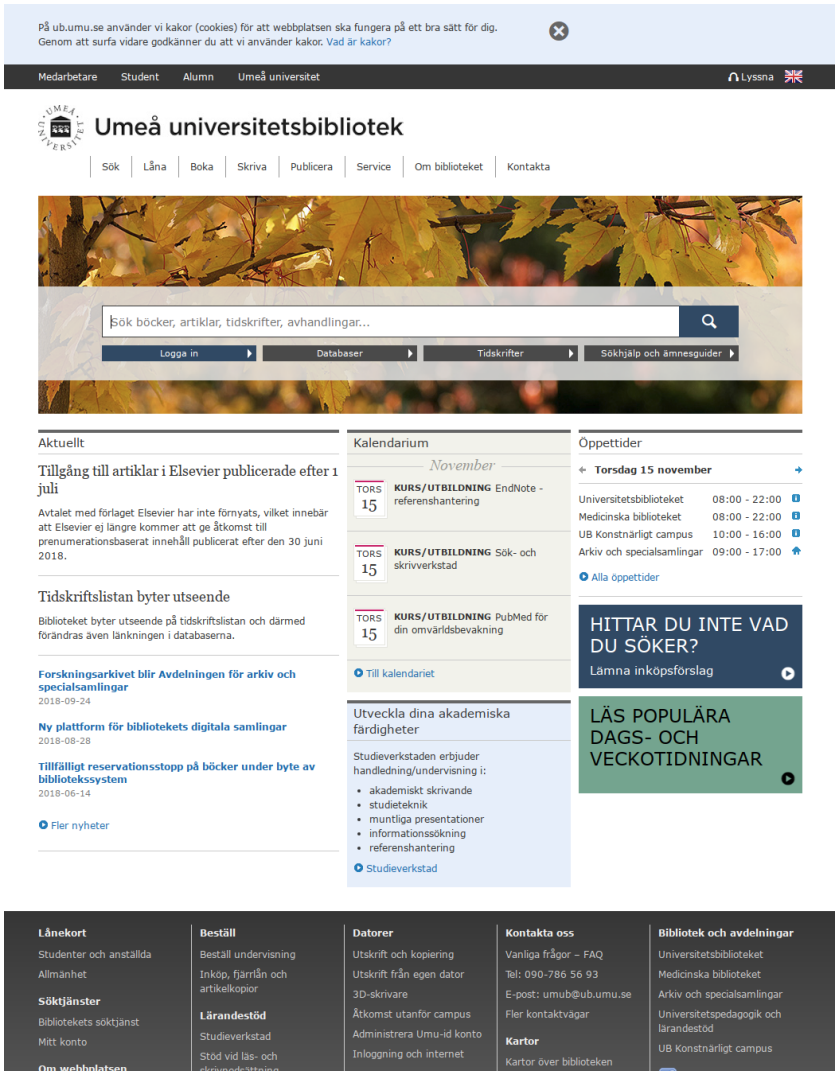

Fig. 2. `http://www.ub.umu.se/`. Umeå University Library, home page.

An example of this is illustrated in Fig. 3 where the user starts on Page 1. The user can either go to Page 2 or Page 3, in both cases the user will be able to reach the goal. Through page 3 it will take 4 steps to the goal but through Page 2 just 2. If the user proceeds to Page 3, one problem is added since it would be faster to go to Page 2. This is represented with a X over the line. Now however when the user stands on Page 3, the closest path is to go to Page $4 \rightarrow 5 \rightarrow$ goal. This illustrates that even if the user gets side tracked the amounts of problems counted is dependent on if the clicked link increased or decreased the distance to the goal. There is also an exception to this rule; Backing up is not considered a problem, if the closest path to the goal is towards the previous page then this is disregarded because it is not crucial to take the optimal path as long as the user reaches the goal.

The experiment tests navigation and one could argue that in a navigation experiment the user makes a problem not only by clicking links but also by looking in the wrong area. This is a real problem in the experiment, e.g. if the user searches for an unprecedented amount of time on one page this experiment would not find this as a problem. There is a quite a few of problems if this unlimited searching problem were to be tackled. First, the experiment would have to use an eye tracker to find exactly when the user starts looking in a specific area. Second, the amount of time for a search would be different between users. Third, how would you define each enclosed area. The experiment limits itself to not handle the unlimited search problem but instead measuring the time to complete the entire test. Time to completion will give a general indication of where the unlimited search problem might occur.

Hassenzahl [3] suggests that assessing a problems severity is hard even for industry professionals. The experiment limits itself to avoid severity assessing.

The experiment is limited to use the current definition of user problem in order to 1. not make the experiment expand way beyond the capabilities of the current one, 2. make the experiment reproducible and 3. avoid subjective biases.

3.4 Software and facilities

The tests are performed on a computer and the screen capture software used was gtk-RecordMyDesctop[1]. In the software the options "encode on the fly" was ticked and video quality = 30. "encode on the fly" will give a viewable video ready as soon as the recording is stopped but at a price of processor power. This will ensure that neither test leader nor user will have to wait to view the video.

The tests were executed in a quiet room alone.

3.5 Tasks

There were 3 main tasks where the correct answer is a link to the correct page. There were also 2 minor tasks where the subject needed to navigate back to the start page. Below is the list of tasks that users were presented with.

[1] http://recordmydesktop.sourceforge.net/about.php

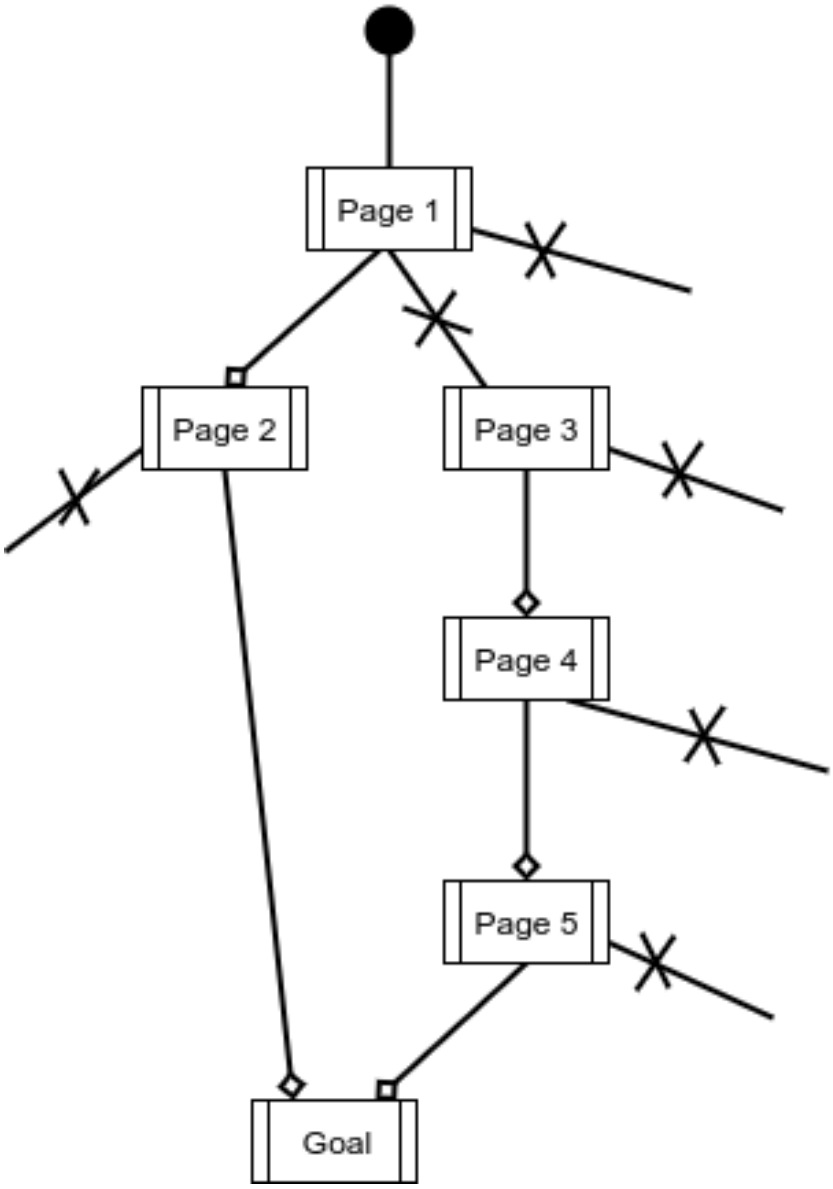

Fig. 3. A graph of an example website. Each page is represented by a box and they are connected together with links, here represented by lines. Any line with a X represents a link that takes the user further away from the goal. Any line with a diamond at the end points to the page that is closer to the goal. This shows that even if the user picked the non-optimal path, i.e. $1 \rightarrow 3$, the penalty for this is only one problem.

1. You can print papers from UB. In this assignment you want to find out how much it costs to print one piece of paper. Link the page you found the answer on here: .....
2. Go to "http://www.ub.umu.se/" NOT: "http://www.ub.umu.se/en"
3. Umeå students have access to eduroam. In this assignment you want to find out how to install it on your Iphone. Link the page you found the answer on here: .....
4. Go to "http://www.ub.umu.se/" NOT: "http://www.ub.umu.se/en"
5. How many physical copies of a book authored by Dante (you may pick any of his work) does UB have? Answer with link, book title and amount here: .....

Optimal path: Here are the paths with minimal amounts of steps to reach the answer. UB is the notation for the start page `http://www.ub.umu.se/`. Each "→ " shows what link needs to be pressed to get to the next page. For example UB → service → utskrift, scanning och kopiering. This means that in the start page there is a link somewhere that is called "service". After pressing that link the user needs to find and understand that "utskrift, scanning och kopiering" is the next link towards the answer.

1. UB → service → utskrift, scanning och kopiering → priser
2. UB → service → Datorer och internet → inloggning och internetåtkomst → trådlöst nät vid Umeå universitetet → relaterade artiklar: ställ in eduroam för Iphone/iPad (Umu wifi)
3. Search → avancerad sökning → -författare/upphovsman- -Dante- → Dantes gudomliga komedi + 2 kopior

Correct links:

1. `https://www.ub.umu.se/service/utskrift-skanning-kopiering`
2. `https://manual.its.umu.se/stalla-in-eduroam-for-iphoneipad-ios-umu-wifi/`
3. E.g:
   Dantes gudomliga komedi,
   2,
   `https://umu-primo.hosted.exlibrisgroup.com/primo-explore/fulldisplay?docid=UMUB_ALMA21139540820004996&context=L&vid=UmUB&lang=en_US&search_scope=de_scope&adaptor=Local%20Search%20Engine&tab=default_tab_umub&query=creator,contains,dante,AND&sortby=rank&mode=advanced&offset=0`

4 Result

The result consists of 14 tests divided into the two groups. 7 in Group 1 (with test leader observations) and 7 in Group 2 (without test leader observations). 2 additional subjects result was discarded because a non-allowed page was used (`http://ub.umu.se/en`). In Table 1 the raw data gathered is presented.

User	Problems	Time (min)	Group
1	9	9.23	1
2	8	8.10	1
3	8	9.10	1
4	13	14.06	1
5	14	13.45	1
6	7	7.50	1
7	14	13.50	1
8	7	8.18	2
9	8	9.53	2
10	10	10.49	2
11	13	13.53	2
12	13	8.35	2
13	7	9.54	2
14	6	8.02	2

Table 1. Raw data gathered from the experiment. It shows the amount of problems discovered, the time to complete the whole test and what group the user did the test in.

	Group 1	Group 2	Metric
Mean	10.41	9.14	Problems
SD	3.10	2.91	Problems
SEM	1.17	1.10	Problems
N	7	7	Problems
Mean	642.00	577.71	Time (min)
SD	173.09	109.58	Time (min)
SEM	65.42	41.42	Time (min)
N	7	7	Time (min)

Table 2. Showing the mean, standard deviation, standard estimated error for both amounts of problems and time to completion when comparing Group 1 and Group 2.

	P-value	$\mu_1 - \mu_2$	95% CI
Problems	0.4394	1.29	[-2.22, 4.79]
Time (sec)	0.4226	64.29	[-104.42, 232.99]

Table 3. Showing the p-values, the mean difference and the 95% conference interval for the t-test between amount of problems and time to completion.

For the comparison between Groups 1 and 2: two independent sampled two tailed t-tests with significance level 0.05 was performed to test h_0 = "Group 1 and 2 is differrent". When comparing the amounts of problems, the t-test gives a p-value: 0.4394. This can be found in Table 3. p is p ¿ 0.05 which means that there is no statistically significant difference between the amount of problems discovered in Group 1 compared to Group 2. When comparing time to completion, the t-test gives a p-value: 0.4226. This can be found in Table 3. p is p ¿ 0.05 which means that there is no statistically significant difference between time to completion discovered when comparing Group 1 to Group 2.

In Table 2 the mean values, standard deviations, errors and number of participants are presented. Both the problems and the time to completion are very close to each other and this is enforced by the big p-values of almost to 0.4.

5 Conclusion

In the beginning of this paper we asked *Does removing the test leader from the test phase in RTA effect the amount of user problems found or the users time to complete the test?* and we found the answer to be: We still do not know. This experiment can not conclude whether the methods gives better, worse or equally good result.

We found that there is no significant difference between RTA with or without a test leader present during the test phase. In other words, because we could not refute h_0 = "Group 1 and 2 is differrent" we do no know if the new method is better, worse or equally good compared to RTA.

In the experiment there were two user results that had to be discarded because they used the disallowed page `http://www.ub.umu.se/en`. Both of these performed the test in Group 2, which means that if there had been a test leader present the two discarded result could maybe have been avoided. Realistically there should be less discarded results in RTA then in the new RTA protocol because there is a human present that can correct any unexpected problems, this is exactly what was observed. The discarded results can not be used in the statistical analysis because they do not represent the amount of problems on the Swedish website. If we take this at face value the experiment finds that ca. 20% of the new RTA protocols result is discarded. It should be noted that this number is highly dependent on the design of the experiment and the tasks, still it is definitely a not a positive number. It could have been designed around by making it impossible to use anything but the allowed website and this should be possible in most browsers.

With some closing words we can say that this paper failed at making an improved version of the RTA protocol by removing the test leader during the test phase. However, there is still a possibility that the new RTA protocol could be useful see 5.2 for more discussions on this.

5.1 Problems

It was interesting that the results between the two groups were so incredibly similar. This might be a coincidence followed from variance but it might be a built in bias from the objective definition of user problem that was used. To eliminate this potential bias either a recreation of the experiment with bigger sample size is needed or a new definition of the user problem.

One problem that occurred after the experiment was performed was that the website `http://www.ub.umu.se` was updated. This means that the closest reproduction of this experiment that can be made is one that uses the new interface of the website.

5.2 Future work

The new RTA protocol might still be useful, future studies are advised to continue with other definitions of user problems, different test objects and different kinds of tasks. Using a new definition of user problem eliminates the potential bias found in the current experiment. New tasks might effect the how the new RTA protocol differs from RTA, this could be explored. A new test object should be obvious since it does not exist anymore.

For reproducing the experiment a number of advise follows. The website in the current experiment has been updated and now does not have the same interface, note that any complete recreation will be difficult. Reproduction studies are advised to use the "encode on the fly" or a corresponding feature option to avoid unnecessary waiting time. Consider also having some mechanism to avoid making the user come get the test leader, maybe a button that indicates when the user is finished. This would be useful if the test leader is busy doing some other work. Another quality-of-life advise is to make sure there is no way for the user to deviate into disallowed websites, one could e.g. block the page right out or redirect the user to the correct website. This should be possible in most browsers.

One idea that tests if the new RTA protocol works in a more complex experiment could be: use professional test leaders to test if it is usable "in real life". Below there is an outline of an experiment that can be used to test a complex experiment with the new RTA protocol.

New RTA with professionals. How does the new RTA protocol behave when it is used by real professionals?

The point of this experiment is to test how the new protocol behaves when exposed to the variance of test leaders, since that is where it would be used "in real life". The experiment uses experienced or semi-experienced test leaders while everything else in the test stays the same. It uses an actor as user to provide test results as similar as possible. The experiment would first need to teach the test leaders the new RTA protocol and then to have them perform it. One consideration is to decide what user problem definition to use. Should the test leaders be allowed to use whatever definition they want? Also consider if the

table of user problem classifications that Khajouei, Peute, Hasman and Jaspers [6] created is a good option for this. It is also recommended that a *within group design* is used, because that would reduce the impact of participants variance. i.e. have each test leader use both RTA and the new RTA protocol.

6 Acknowledgement

I would like to thank Assoc. Prof. Suna Bensch for her leadership and Prof. Kary Främling for his support and reviews during the study. I would also like to extend my thanks to Ida Wiklund, Emil Söderlind and Folke Hagberg for reviewing my paper.

References

[1] Zhiwei Guan, Shirley Lee, Elisabeth Cuddihy, and Judith Ramey. The validity of the stimulated retrospective think-aloud method as measured by eye tracking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1253–1262, New York, NY, USA, 2006. ACM.

[2] Maaike Haak, Menno De Jong, and Peter Schellens. Retrospective vs. concurrent think-aloud protocols: Testing the usability of an online library catalogue. *Behaviour & IT*, 22:339–351, 09 2003.

[3] Marc Hassenzahl. Prioritizing usability problems: Data-driven and judgement-driven severity estimates. *Behaviour & Information Technology*, 19(1):29–42, 2000.

[4] Fred Paas Jeroen J. G. van Merriernboer, Tamara van Gog and Puk Witte. Uncovering the problem-solving process: cued retrospective reporting versus concurrent and retrospective reporting. *Journal of Experimental Psychology: Applied*, 11(4):237–244, 2005.

[5] Anker Helms Jørgensen. Thinking-aloud in user interface design: A method promoting cognitive ergonomics. *Ergonomics*, 33, 04 1990.

[6] R. Khajouei, L.W.P. Peute, A. Hasman, and M.W.M. Jaspers. Classification and prioritization of usability problems using an augmented classification scheme. *Journal of Biomedical Informatics*, 44(6):948 – 957, 2011.

[7] Anne Oeldorf-Hirsch. Between-subjects design. In Mike Allen, editor, *The SAGE Encyclopedia of Communication Research Methods*, chapter Between-Subjects Design, pages 91–92. SAGE Publications, Inc, 2018.

[8] Leanne M. Willis and Sharon McDonald. Retrospective protocols in usability testing: a comparison of post-session rta versus post-task rta reports. *Behaviour & Information Technology*, 35(8):628–643, 2016.

Dynamic load management system for local-network based game

Game using shared big screen and smartphones as interactive controllers

Andreas Wendel

Department of Computing Science
Umeå University, Sweden
c15awl@cs.umu.se

Abstract. The use of mobile devices and smart phones has increased to the point that it is unlikely one will find a person that does not currently own one. Given that many people now have these small computer with themselves it would be easy for them to bring it up anywhere and use it as an interactive controller. By making a game system that allows locally connected users to interact with a big screen such as a TV, a game could be made where the number of users where not limited by the number of controllers available. This could then pave the way for local games where a lot more users are connected at once compared to the classic 4 users per console. To allow as many users as possible to be connected at once when using data intensive real time interaction between the mobile devices and not to stress a single server a test system that share the network load between the mobile devices based on the character location is made. This is done by allowing the mobile clients to host different regions of the game world and allowing user to swap host depending on agent position in the game. This should hopefully allow large number of players to play data intense "mobile to shared big screen" games together in a local network together, which the resulting data points to being possible.

1 Introduction

Recently several platforms and games have been released where the smartphone is not the main platform running the game but is instead acting as the input device. The game itself is displayed on a larger screen shared with other users. Two examples are the quiz game platform Kahoot[1] and the virtual console Airconsole[2]. These are however limited to only a few devices or games with no real time data transmission. Current systems does not allow data to be sent back to the mobile in real time making it possible for example to steer a character on the smartphone and see the position on the large screen and your mobile device. Is this because the extra data transfer might cause high latency in the network?

[1] http://www.kahoot.com, Kahoot webpage, accessed 2019-09-10.
[2] http://www.airconsole.com, Airconsole homepage, accessed 2019-09-10

S. Bensch, J. Eriksson, K. Främling, T. Hellström (Eds.): Umeå's 24th Student Conference in Computing Science USCCS 2020, pp. 25–43, January 2020.

The game system that is built and referred to in this paper is similar to the platform and games previously mentioned. The test game system was built for the purpose of testing the latency and hardware load on a mobile device and on the main server for two different network architectures. First a hybrid client server architecture to share the network load between the mobile devices and the other a classic client server architecture using only one main server. The tests are conducted with up to 10 clients at a time to see if a given mobile device can handle the higher load.

Fig. 1 shows the test game with 12 clients connected located in the green game space. The clients are able to move freely between the zones and will automatically change server to a sub-host if they enter a new region. This region will then be assigned to a new client in that region or to the first one to enter it. The main server is here to manage the sub servers and to receive and display data on the big screen. The agent in this case is a circular rendered character which the user can control using their mobile devices.

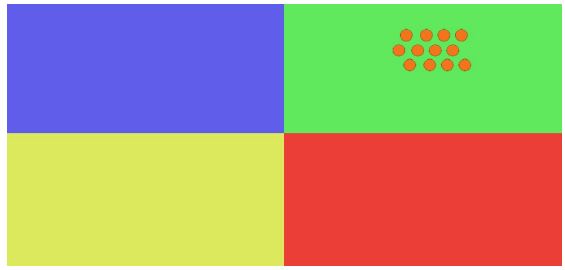

Fig. 1. Test game system shown on the big screen showing the total game space subdivided. Twelve character agents of the clients can be seen in the green zone as orange circles.

For the mobile device another similar version of the test game system was made with the difference being a more zoomed in view, a more detailed agent and with the ability to see the actions made by all the agents in view inside the mobile display.

The goal with this study is to answer the research questions stated below by running different tests. The results for the different tests shows the latency for up to 10 users at a given time and shows results for the latency in different scenarios such as:

- All 10 users in one game region.
- Users changing game region.
- The entire game space is hosted by a single main server.

1.1 Research Questions

Below are the research questions that are stated for this research paper.

- Is there significant gain in network latency and CPU performance in the clients and hosts when dividing the work load in the previously mentioned manner for the test game created for this paper.
- Can this dynamic load management system based on agent location be used using mobile devices as sub servers or will the overhead and network latency cost be to high.

2 Theory

In this section the theoretical background for this paper will be presented.

2.1 Load management system based on position

A paper by Chen et al. [1] explains a load management system for massive multiplayer games (MMG). Chen shows that by using more servers to host for each region of the game world, the load could be managed and dynamically divided between each host. This paper will try an adaptation of this method. Clients will in this paper be able to act as one of these regional hosts, sometimes called sub hosts or sub servers. This by allowing the mobile device client to also host a region while the main host act as a server list. This results in a reduced load in a client server hybrid architecture. This can potentially allow more users to be connected at the same time in one game and opening up a system for a massive multiplayer local game (MMLG) system.

2.2 Hierarchical Networking Architecture

The need of clients connecting to each other eliminates the use of a standard client server based architecture. However a full peer to peer architecture is not actually used either. This results in a server client hybrid referenced in this paper as the Hierarchical client server architecture. Fig. 2 shows that there is an hierarchical structure where the clients are at the bottom and the sub host clients in the middle tier and the main/master server at the top of the hierarchical pyramid.

This is similar to the distributed architecture from the study "A Distributed Architecture for Multiplayer Interactive Applications on the Internet" [2] where the main server delegates the work to the region servers and are only in use when a new client joins the game. However the difference here is that the main server will also receive some data at low feed rate to display some parts of the game on the shared big screen. More specifically, the data will be the agent's location for all the clients and will be sent via the sub servers every 0,1 second. Another similar solution that will contribute to a network architecture solution used is the publisher/subscribe model[3]. Fiedler et al. goes into detail of an architecture for a massive multiplayer game that subscribes to the different sub servers the client is close to.

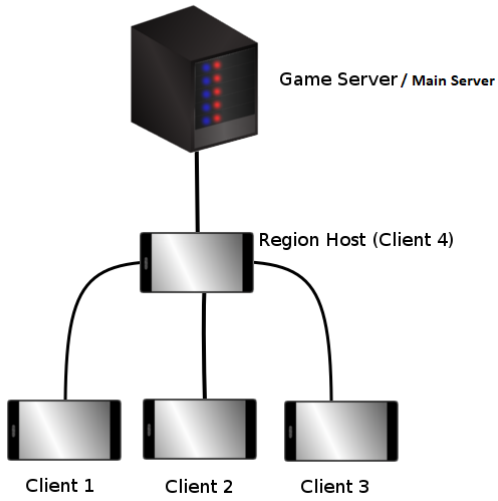

Fig. 2. Image of the hybrid network structure showing the clients at the bottom, connected to a sub-server which is also a client to the main server.

2.3 Accepted latency level

Several studies have been conducted to determine an acceptable level of latency. One of these found that users starts losing concentration around 580 ms latency from touch devices. However it was deemed that the test user were fooled by the high frame rate on the devices and that the same test on a device with a lower frame rate would have probably lowered threshold for the accepted latency level. Another study [4] done for the game Everquest, a massive multiplayer online game (MMORPG), showed that a latency between 500 to 1000ms was acceptable.

3 Method

To test the latency difference between a single server architecture and a hierarchical server architecture a test game is used made with the game engine Unity[3]. The network programming is done using the C# programming language that is supported by the Unity game engine. The test game is then used for the different scenarios in order to get results such as latency between the client and the sub server.

A main server running a big screen such as a TV screen that has an overview of the game and is shared with all the users to watch their agent on screen and using smartphone as the individual controller[5] for the users. The test game is built to send data back to the user mobile device in order for the game state

[3] https://unity.com/, Unity homepage, accessed 2019-11-12.

to be displayed in part on the mobile device itself. A test is made in order to find out if frequent data communication between the mobile and the host will cause high latency as well as a test that checks if a load management system based on the agent position reduces the latency and if the performance of the smartphones is effected by letting them sub-host regions of the game space.

3.1 Setting up the test game

In order to test the different scenarios and record the latency as well as the CPU load a test game was made with the Unity Game engine. The game is a 2d game with a top down view with one big field (the game world). The field is split into four regions that represent the different regions that are to be managed by four different sub-servers. The orange circular shapes that can be seen in Fig. 1 are representing the players characters. While the view on the mobile devices are zoomed in on the users character agents the big screen have a overview of all the game world with a view of all the player locations. The main-server will receive all the location data of all users in a less frequent rate in order to update all players characters simulating the fact that the character location does not need to be as smooth on the shared big screen compared to the mobile devices of the users. No more data will be handled by the main-server running the big screen but the sub-server on the other hand will handle the collisions between player characters and the transferring of clients to other game regions.

If a player character enters a new region that player will be transferred to the sub-server that is in control of that region. If no sub-server has been assigned to that region the client will be assigned as the new sub-server for that region of the game world.

For the smartphone screen a more zoomed in version of the game can be seen. The screen follows the player with the player agent located at the center of the screen at all times. In Fig. 3 the view on the mobile device can be seen where the user of the smartphone has its player agent highlighted in light purple and another player agent can be seen as a gray circle.

The same game state in Fig. 3 can be seen with the big screen in Fig. 4.

3.2 Packages used

There are several package types that are to be sent in order to communicate between the clients, sub-servers and main-server. The different package types used are the following: *ClientJoin*, *ClientList*, *Position*, *ServerInfo*, *RegionAssign*, *RegionJoin* and *Ping*. These types are given a different byte value that are placed at the first byte in all the data packages in order for the receiving device to determine what the correct response are.

Client join The client join package is shaped in the following manner: *[TypeCode][ClientId]*.

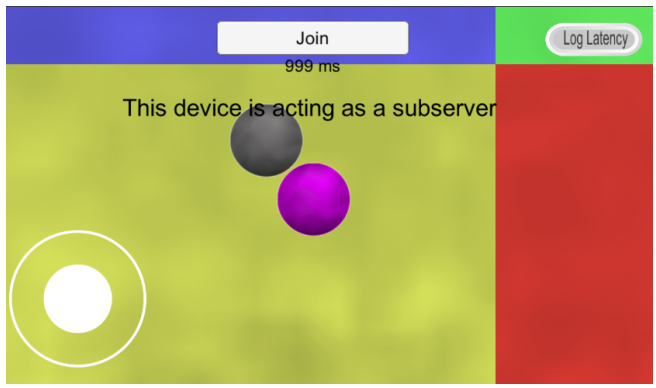

Fig. 3. The view on a smartphone display where the user of that smartphones is in a light purple color and colliding with another player. The user is also assigned to be the sub-host for the yellow region.

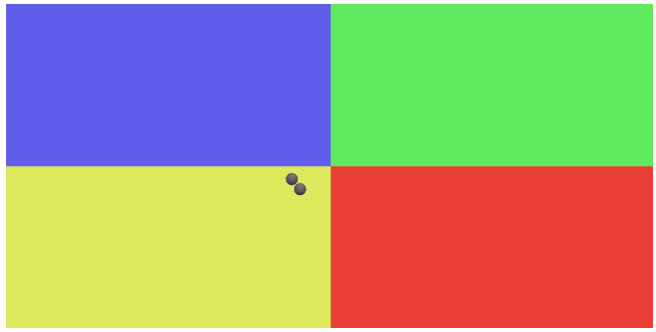

Fig. 4. View of the big screen displaying 2 characters in the yellow region

Each element within the square brackets "[ ]" being a byte in size giving a total data size of two byte for the client join package. The *ClientJoin* package is sent by a client when it joins a new sub-server. These are also sent by the sub-servers to tell other clients that a new client has joined the sub-server so that a character can be drawn representing the newly connected client. In order to see the sequence of communication with these different types of packages between the client and sub-server aswell as the main server sequence diagrams are used. The sequence diagrams also show the communication between the C# thread and the Unity game thread.

Client list The client list package is shaped in the following manner: *[TypeCode][ClientIdofSender][ClientNr1][ClientNr2][ClientNr3]...[StopByte]*. This list is sent by the sub-server to a newly connected client so that it may draw the other clients already connected on the sub-server joined. The stop byte is used so the client knows when to stop reading from the network steam.

Position The position package is shaped in the following manner: *[TypeCode][ClientId][XPosHigh][XPosLow][YPosHigh][YPosLow]*. The position is sent by the client to the sub-server the client is connected to when the client is moving. This will be broadcast to all other clients connected to the same sub-server aswell as a copy to the main-server for it to draw the character. The sub-server may also send a position to a client with the clients own id in order to correct the clients characters position after a collision has been detected making a sort of penalty-based correction method.

Region assign The region assign package is shaped in the following manner: *[TypeCode][RegionId]*. The first user that joins the main-server will be assigned a region to be in charge of where the region id can be between 1 and 4 representing the different regions of the game world.

Server info the server info package is shaped in the following manner: *[TypeCode][ClientId][IP1][IP2][IP3][IP4][portHigh][portLow]*. When a client has received a region assign package the client will then need to send a server info package to the main-server in order for the server to have the information such as local ip address and port number needed for the clients that enters the sub-server controlled region to make a TCP connection.

Region join The region join package is shaped in the following manner: *[TypeCode][IP1][IP2][IP3][IP4][portHigh][portLow]*. When a client enters a new region already controlled by another client (sub-server) a Region join is sent with the information necessary in order to make a TCP connection directly to the sub-server.

Ping The ping package is shaped in the following manner: *[TypeCode][ClientId][PingIdHigh][PingIdLow]*. The ping package are sent from the client to the sub-server to measure the latency.

3.3 Measuring latency

In order to measure latency the package type *Ping* is used. As shown in Fig. 5 the ping package is sent from the client via the sub server to the unity thread, the game thread, where it is handled like any other package. It ends up in the same queue as all the other packages sent between the sub server and client. In order to get the data from the mobile device an analytic tool called log cat is used to log the data via the USB cable to a connected computer to be displayed in a terminal.

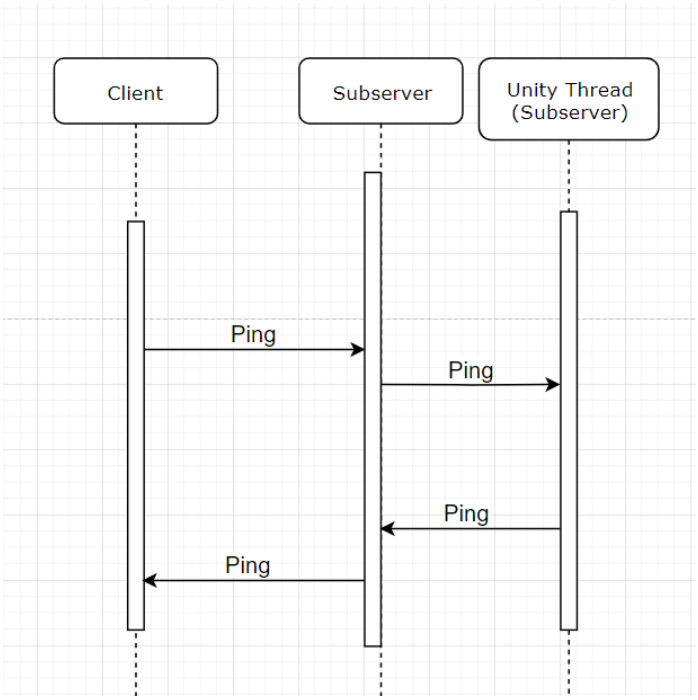

Fig. 5. Ping package sequence from client to sub-server and back

3.4 Measuring CPU load

Using Unity's built in tool *Profiler* the CPU performance can be measured. The Profiler tool is connected with the Android device over USB and logs the data back via the USB to be displayed. In this tool it is shown how many ms it takes to run the C# script at a given frame rate as seen in the result section.

3.5 Communication between clients and sub-servers

The communication between the clients and the sub-host will be done using the TCP protocol both for position data and action data.

Joining sub-server Joining a sub-server is done by a client after a region join package has been received by the main server as seen in Fig. 6 sequence diagram.

Sending position data Position package will be sent frequently when a client is moving at a rate of 1 position package every 0.05 seconds as shown in the sequence diagram Fig. 6 after the dotted line between the client and the sub-host. Position packages will also be received from the sub-server with the position from all the other clients sent one package at the time at the same rate as the client is sending its own position. A position package can also be sent by the sub-server to the client in order to correct its position if a collision is detected as seen in Fig. 7.

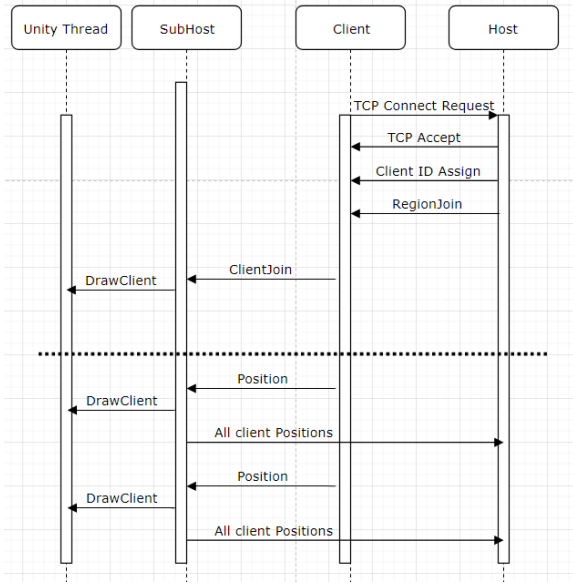

Fig. 6. Client joining the game and being assigned to join a sub-server. Position package being sent from client to sub-server and the sequence below the dotted line are repeated.

Client leaving sub-server region When a client agent leaves a sub-hosted game region it will be transferred to a new sub-server in the newly entered region. If however that region is empty then the region are to be transferred to the client entering and the client becomes the sub-host for that region. The sub-server will detect if the client leaves the region and will then remove it as a client and send

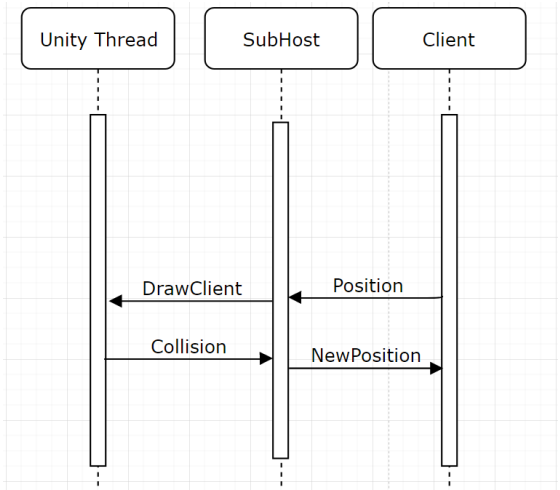

Fig. 7. Position being sent back to the client to correct itself after a collision.

a region assign package to the main-server in order to get the information of the sub-server in the region that the client enters. If the region is assigned to a sub-host client then the client receives a region join otherwise a region assigned is sent to the client.

3.6 Communication from sub-servers to main-server

A TCP connection will be made between the sub-servers and the main-server. The data being transferred are positional data from all the clients characters including the sub-servers characters aswell. These will be sent by the clients via the sub-servers and then finally to the main-server as seen in Fig. 6. If the sub-server detect a client leaving the region the sub-server will send a region assign package to the main-server in order to get the information of a client in the region that the client enter.

3.7 Communication between the main-server and the clients

Only when a new client joins the game for the first time will the main-server and client be in direct communication. The client will receive a message with its id and then what sub-server to join. If a sub-server has not been assigned a *RegionAssign* will be received from the main-server. This can be seen with the first 4 packages sent in the Fig. 6.

3.8 Communication between the server thread and Unity game engine thread

Unity is the game engine used for the test game in this paper. Unity uses co-routines instead of threads because it is not thread safe. In order to decouple the

network programming from the game engine the game system is created using standard threads in the programming language C# and is then communicating with the Unity game thread via a common data queue. Safe communication with the Unity game thread was established by locking the data queue of actions and positions before filling it with data from the standard C# thread and then letting the Unity main thread read from it when the queue is unlocked.

3.9 Devices used

For the results a few different devices where used. For the main-server a custom built PC using a Intel(R) Core(TM) I7-9600k CPU with 3.60 GHz clock speed with a Windows 10 64-bit operating system was used. It was connected with an Ethernet cable to the router. A Samsung Galaxy S7 is used as a client and an ASUS Zenfone 4 Max ZC554KL is used for one of the sub servers where both the mobile devices were connected via the local 2.4GHz Wi-fi network located around 0.5 meter from the router during the tests.

4 Results

The CPU and the latency are not measured at the same time during the tests. This means that the results between the CPU load and the Latency are not in sync with one another. Nevertheless the test circumstances are identical in both cases. Below the results can be seen for different scenarios with different amount of clients and sub servers for different devices.

4.1 Position Updates

In Fig. 8 one can see the latency result from a test where a single user is moving and sending positional data to the sub server. The same test was executed again to measure the CPU load on the sub server device as seen in Fig. 9.

In Fig. 10 a similar test was preformed but this time with 10 clients connected and moving simultaneously on a single sub server, where one of the clients being a Samsung Galaxy S7 device. The result for the CPU load on the sub server device can be seen in Fig. 11 where the same test was executed a second time.

A final test for positional updates was preformed where the stationary computer as mentioned in Section 3.9 were acting as the single sub server device while having 10 clients connected and moving simultaneously. The CPU load result can be seen in Fig. 12.

4.2 Collision Handling

These test check the load on the network and on the CPU performance when collisions between player agents are made.

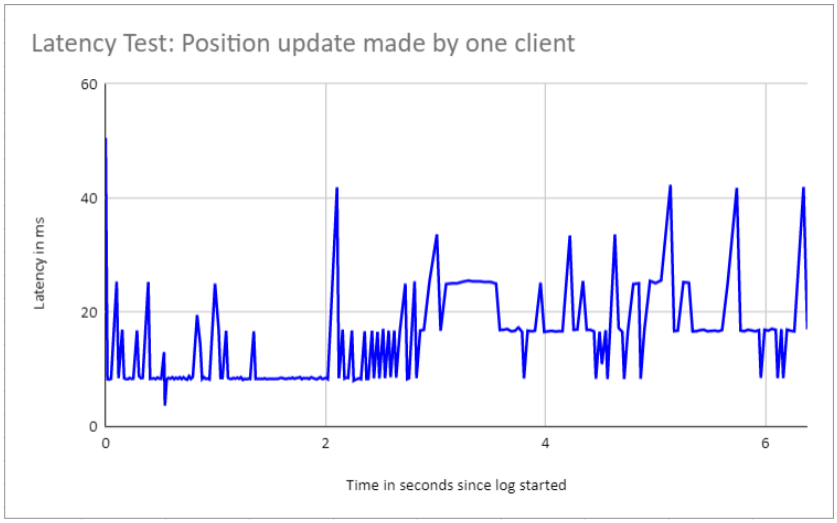

Fig. 8. Latency in ms between a Samsung Galaxy S7 device as the moving client and a ASUS Zenfone 4 Max ZC554KL as the sub server.

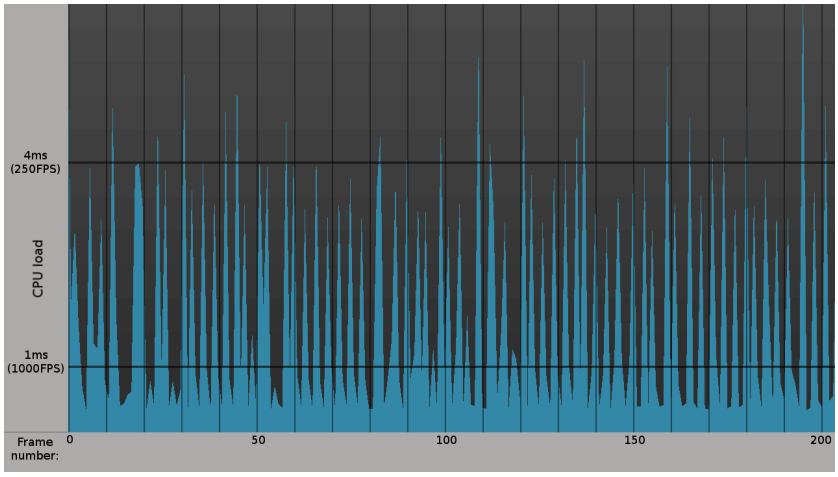

Fig. 9. CPU load in ms between the Samsung Galaxy S7 device as a moving Client and ASUS Zenfone 4 Max ZC554KL as the sub Server

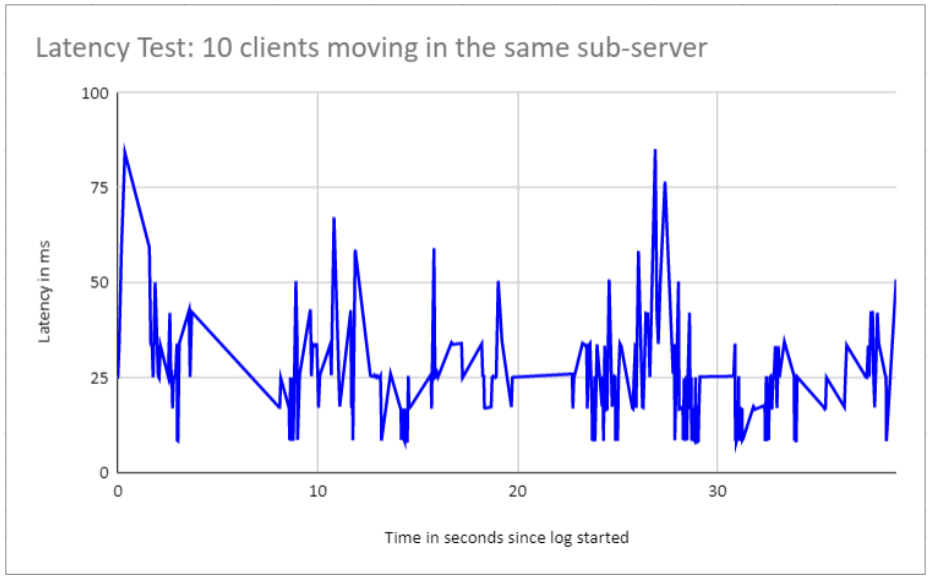

Fig. 10. Latency in ms between the Samsung Galaxy S7 device as one client and ASUS Zenfone 4 Max ZC554KL as sub server when 10 clients are moving around simultaneously

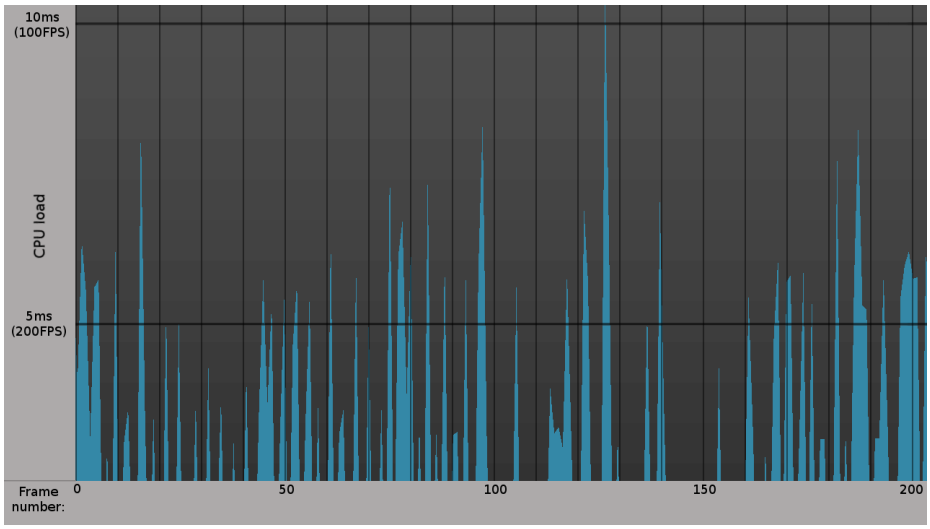

Fig. 11. CPU load for the device ASUS Zenfone 4 Max ZC554KL as sub server when 10 clients from the stationary PC are moving around simultaneously

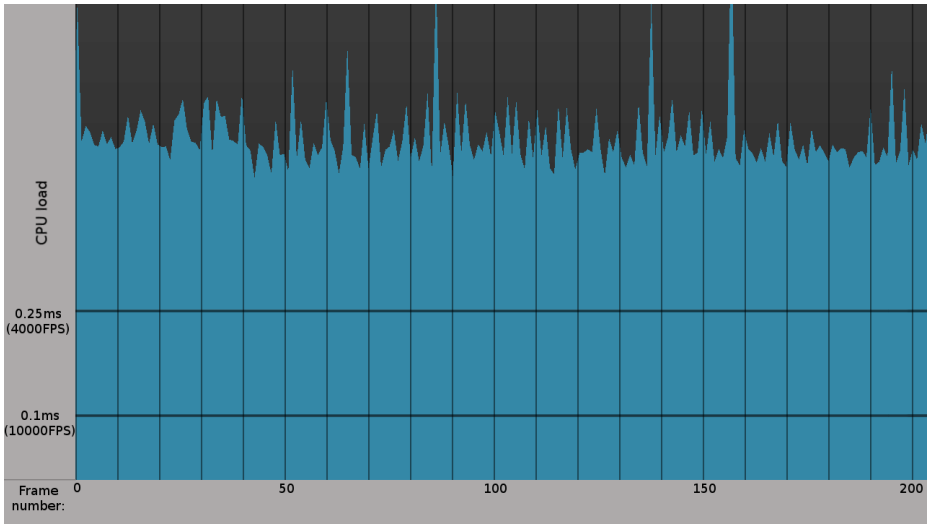

Fig. 12. CPU load for the stationary PC when acting as the only host for 10 clients moving around simultaneously

Collision between two clients The following results in Fig. 13 and Fig. 14 shows the latency and CPU load when two clients are colliding. The sub server running on a ASUS Zenfone 4 Max ZC554KL and the client running on Samsung Galaxy S7.

4.3 Emigration and Immigration of clients

Below are the results from when clients immigrate to a new sub server. The results was used when the main server was running on the stationary PC, the sub server device that the clients where immigrating to was the ASUS Zenfone 4 Max ZC554KL device and the client device logging data was a Samsung Galaxy s7 that were already connected to the sub server.

The latency from one of the clients to the sub server during the time 10 clients immigrated can be seen in Fig. 16. Take note that this results where not measured during the same test. These 10 clients immigrated one by one during 6 seconds.

4.4 Assigning Sub server

The first user that enters a area will receive the control of a region to host, assigning the client and making it a sub server. In Fig. 17 one can see the CPU load result on a client that are to become a sub server.

The two spikes that can be seen in Fig. 17 happens because the main server does not send the assign server package immediately to a client that just joined the main server. However if a client already had an established connection with the main-server and is assigned to host a sub region then the assign region

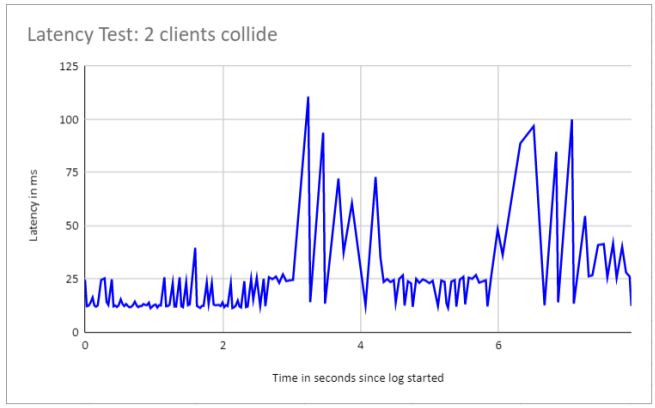

Fig. 13. Latency results in ms between a client running on a Samsung Galaxy S7 and the sub-server device ASUS Zenfone 4 Max ZC554KL while a collision between two player agents are made.

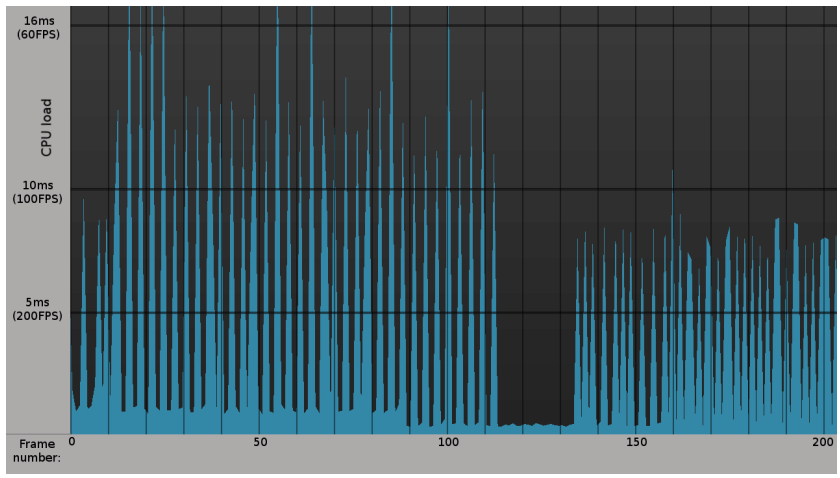

Fig. 14. Script load on the CPU for the sub-server when a player object colliding with another player object. The device hosting the sub-server being ASUS Zenfone 4 Max ZC554KL.

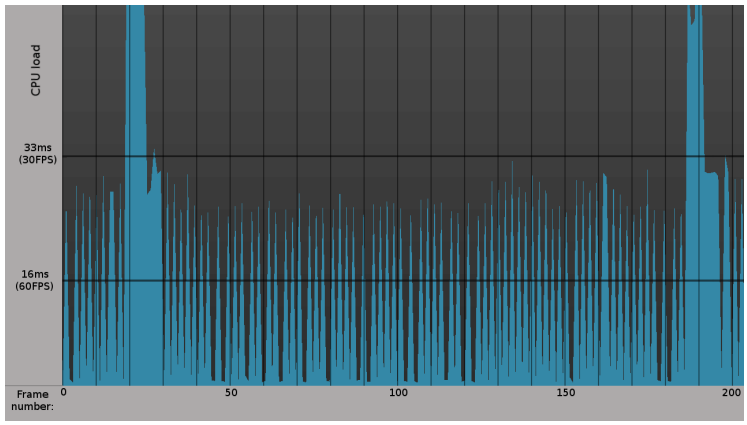

Fig. 15. CPU load on the sub server device receiving 10 immigrating clients in 2 bursts.

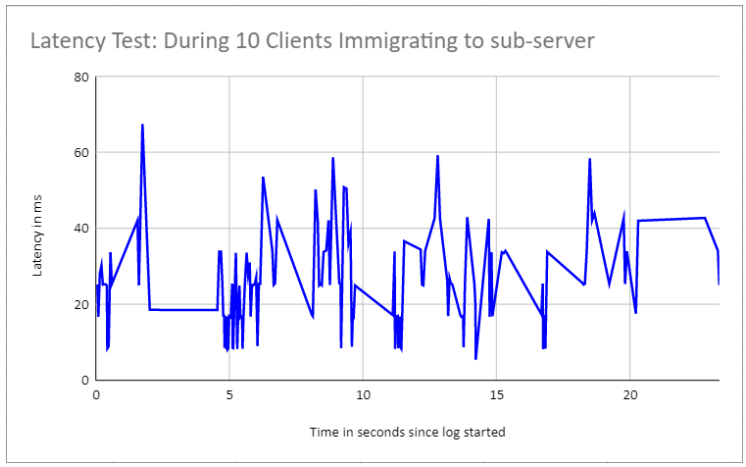

Fig. 16. Latency between one client and a sub-server receiving 10 new clients in a immigration.

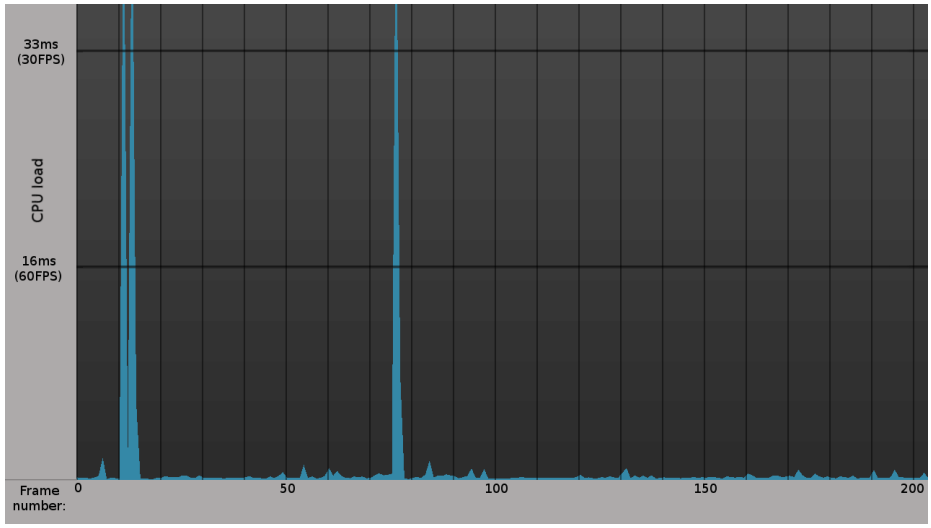

Fig. 17. The scripts load on the CPU on the client device ASUS Zenfone 4 Max. ZC554KL when joining the main server on the Stationary PC and is assigned to host the sub region it is in.

package will be sent immediately. So the first spike is when the client joins the main server and the second is the handling of becoming a sub server. In the result we see a increase in CPU load up to and above 33ms but only for around 3 to 5 frames.

In the Fig. 18 the CPU load result for the main server can be seen when a new client joins and later is assigned to be a sub server for a region of the game.

5 Discussion

In order to get the best performance for a system of this type a great deal of care should be taken into account when designing the game. The game design has a lot to do with the results and how the load will be handled. A game that has its world divided in many rooms or regions may benefit from a location based load management system.

A problem with the results is that when more than two clients were used the remaining clients had to be run from the stationary PC that were also acting as the main server. This does not accurately represent 10 clients being run using only smartphone devices since the mobile devices would probably bottle neck the system more then the locally hosted clients. This however were not possible due to lack of smartphone devices being available for testing. Another problem with the results are that only up to 10 clients were used during the tests. Because of the few number of clients tested, the number of clients where this game system outperforms a single host client server system is unknown. An automated test could have helped solve this by reduced the time taken to setup the different

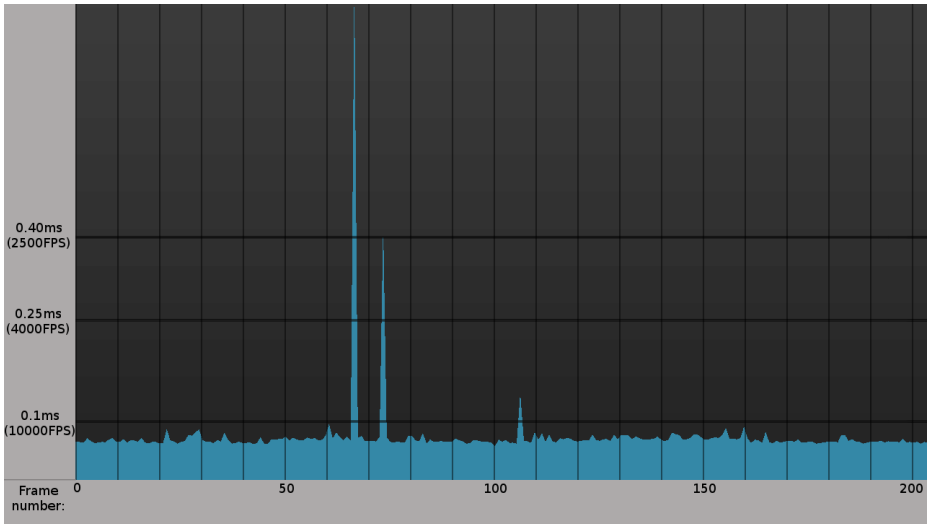

Fig. 18. A client joins the main server and is assigned to host a region it is in. The device running the test game as the main server is the stationary PC device.

tests and increased the quality of the results by allowing more tests with more clients.

Even though the latency limit of 500 ms was achieved my own experience tells me that a latency of up to 500 ms is way above a comfortable gaming experience. Personally a more comfortable latency would be a maximum of 100 ms and this is also archived.

5.1 Future work

In order to make sure a system like this can handle many more clients future testings and development should be done. The limit for this system is still unknown given that only tests with no more than 10 clients were preformed. This could give answers to when a game system using a hybrid server client architecture would have less CPU load and latency compared to a traditional server client architecture.

The separation of the responsibilities between the client sub servers and main server could be mixed in various ways in order to gain the most of a game system like this. Another approach to the position update sequence and collision handling are to use the UDP protocol instead on TCP for sending position packages. It would most likely reduce the load on both latency and CPU further. There is also interest in how the game world is subdivided. If a implementation were to be subdivision based on population density the load for a sub server might be reduced because the region it hosted could be divided further dynamically.

To improve the mobility of this system future work could be done to remove the need of a main server or removing the need of a external device hosting the

main server. Instead a adaptation where the main server is hosted by a smart device could be used allowing large game spaces to be hosted anywhere by a number of smart devices completely free from external stationary devices.

6 Conclusion

With the given results there is no reason why a game of this sort could not be implemented even with larger amount of clients being connected at once. Given that the system is modular with the smartphones hosting the game regions there should be no close limit to how many clients could be connected simultaneously, though future tests is needed to make sure what the number of clients limit is.

The study shows that it is possible for at least 10 clients to be connected using this setup, however because the tests only used up to 10 clients it is to soon to say that it is feasible for this system to replace a single server architecture solution.

For the few number of clients tested here a single server can handle the CPU load and latency well without problem. This makes the overhead and complexity of the game system solution unnecessary given that a simpler solution could be used instead.

References

[1] Jin Chen, Baohua Wu, Margaret Delap, Björn Knutsson, Honghui Lu, and Cristiana Amza. Locality aware dynamic load management for massively multiplayer games. New York, NY, USA, 2005. Association for Computing Machinery.

[2] C. Diot and L. Gautier. A distributed architecture for multiplayer interactive applications on the internet. *IEEE Network*, 13(4):6–15, July 1999.

[3] Stefan Fiedler, Michael Wallner, and Michael Weber. A communication architecture for massive multiplayer games. pages 14–22, 01 2002.

[4] Tobias Fritsch, Hartmut Ritter, and Jochen Schiller. The effect of latency and network limitations on mmorpgs: A field study of everquest2. New York, NY, USA, 2005. Association for Computing Machinery.

[5] Rikard Johansson Marcus Löwegren. Using your smartphone as a game controller to your pc. Master's thesis, Blekinge Institute of Technology, 2013.

Heuristics for Mobile Applications, Intended for Young Children

Ida Wiklund

Department of Computing Science
Umeå University, Sweden
id16iwd@cs.umu.se

Abstract. Conducting a heuristic evaluation has become an accepted method of usability evaluation of user interfaces. But there are no heuristics adapted to touchscreen applications for very young children. As young children may behave differently than older users they should need a specified set of heuristics. Thus, the task of this paper is to *adapt and rank Nielsen's heuristics to touchscreen interfaces intended for children aged 3-5 years.* There are already design principles adapted to young children, but no set of heuristics. Therefore, this paper introduces "Touchscreen Usability Heuristics for Young Children" (TUHYC). The new set of heuristics was created by conducting two interviews and handing out surveys over internet. A prototype was also created according to the heuristics, to demonstrate them. As child behavior is unpredictable some testing might still be needed along with the use of the heuristics, but these heuristics may still reduce the scope of testing.

1 Introduction

Mobile applications for children are becoming more and more popular, partly because of their availability and ease of use. Some applications are more preferable and easier to use, both for parents and children. It would be desirable if the child could consume the product by himself, but some applications for children aged 3-5 years require surveillance by adults. Young children are unpredictable and different from older users [4]. They can quickly lose interest and focus on what they are doing, they cannot read, and they are more impatient.

That is why a set of usability principles specially designed for applications intended for young children may be helpful to application developers. In this paper the ages of the children are specified to 3-5 years, and the platforms to mobiles and tablets. *Nielsen's heuristics* [6] were reconstructed and adapted to younger children and touchscreen interfaces, and ranked according to importance in application development. Thus, the task is to *adapt and rank Nielsen's heuristics to mobile touchscreen interfaces intended for children aged 3-5 years.* Heuristics are principles for finding problems in an interface and design principles are basic principles for designing an interface, in this case heuristics were designed.

There has been earlier studies about children, touchscreens and design principles. A study is about heuristics for child e-learning applications [1]. That paper

S. Bensch, J. Eriksson, K. Främling, T. Hellström (Eds.): Umeå's 24th Student Conference in Computing Science USCCS 2020, pp. 45–61, January 2020.

explains Nielsen's heuristics in a child perspective and adds new heuristics about children use and e-learning for children. Another research paper evaluates how design principles should be used when developing an application for children [4]. The paper focuses on some general design principles, including Donald Norman's principles [9]. A third paper focuses on developing playability heuristics [5], designed for evaluating mobile games. But there are no specific heuristics about children aged 3-5 years and mobile touchscreen interfaces.

The research task was investigated by interviewing and distributing surveys to experts with experience in interaction design, children and application development. The results of the study are a set of "Touchscreen Usability Heuristics for Young Children" (TUHYC) and a prototype to demonstrate the heuristics. In section 2 there is an explanation of Nielsen's heuristics and a presentation of earlier work. Section 3 contains a discussion about the methodology and the results are presented in section 4. The paper concludes with section 5.

2 Related work

Jakob Nielsen has developed ten heuristics [6], which are general principles for interaction design, to have in mind when designing user interfaces. See table 1 for all the usability heuristics of Nielsen. The heuristics may also be used for a heuristic evaluation [7], a usability engineering method, to review the user interface and find usability problems. The user interface gets examined by a few evaluators with regard to the heuristics. Research has shown that the heuristic evaluation is an efficient usability engineering method [3]. The method also reduces costs as it is a quick method and does not require money for e.g. laboratories and test persons [7].

There is a research paper on the creation of heuristics intended for child e-learning applications [1]. The research paper explains the heuristics of Nielsen in a child perspective (NUH - Nielsen Usability Heuristics), adds new heuristics about children use (CUH - Child Usability Heuristics) and new heuristics about e-learning for children (EUH - E-learning Usability Heuristics). Usability tests and a heuristic evaluation were conducted on two e-learning applications to test the heuristics. They are evaluating children aged 5-13 years. They found that these new heuristics are very helpful but as child behavior is unpredictable user testing still needs to be conducted.

Another research paper evaluates how design principles should be used when developing an application for children [4]. It focuses on Donald Norman's design principles, feedback, mapping, constrains and affordance [9]. The paper [4] also discusses other principles such as metaphors, common components, how to use the touchscreen, gestures and audio. They conducted semi structured interviews with application developers. They were evaluating children aged 2-11 years. The result was that their design principles are useful when developing applications, but the most important is to have the child in focus, have accurate feedback, a carefully planned interface and informational structure, and to have big buttons.

Usability Heuristics for User Interface Design
Visibility of system status - The user should know what is happening on the screen in time with good feedback.
Match between system and the real world - The system should speak user language, not system language, and display information in a logical and natural order for the user.
User control and freedom - The system should support redo and undo. The user should be able to control the application, and undo or redo something if the user regrets his actions.
Consistency and standards - Platform conventions should be used. Situations, actions and words should mean the same thing, on each page of the application and like on other applications.
Error prevention - The application should eliminate the error from occurring or present a conformation box before committing the action.
Recognition rather than recall - Important information and instructions should be visible or easily retrieved on the application, when needed. The user should not have to remember objects, actions or options, the application should hold information for the user.
Flexibility and efficiency of use - The system is efficient to both experienced and inexperienced users. It should be easy to use for novice user and there should be accelerators on the application, speeding up the interactions for the expert user.
Aesthetic and minimalist design - Irrelevant and rarely needed information should be removed from the application.
Helping the user to recognize, diagnose, and recover from errors - The application should offer easily expressed error messages containing a problem description and suggesting a solution.
Help and documentation - Even if the system should work without documentation, it may be needed. The information text should not be too large, be easily retrieved, user task focused and contain a solution with steps.

Table 1. A table with all heuristics of Nielsen [6].

The general heuristics made by Nielsen are not adapted to specifically fit the smaller mobile touchscreen interface. Therefore, other researchers have developed heuristics adapted to the mobile touchscreen interface, out of Nielsen's heuristics [14]. These should work as a checklist and evaluation method for mobile interfaces. Another paper proposed other heuristics for the mobile interface [10], that were not directly derived from Nielsen's original heuristics. Their heuristics are more adapted to the related software's visualization layer. Thus, they extended the heuristics of Nielsen to get new heuristics specifically for the mobile user interface. There is also a set of usability heuristics specifically made for touchscreen-based devices [12].

There is a framework of touchscreen interaction design recommendations for children [13] created by reviewing relevant literature. This paper with the framework gathered recommendations for application development for children, from other studies, and compared these with practice and found a gap between theory and practice. There have also been three usability studies that tested applications intended for children on 125 children aged 3-12 years [8]. They created

usability guidelines adapted to children and compared these with guidelines for adults. They also explained the importance of adapting the application towards a specific age, not just all children. There is also a paper containing collected, and analysed research about children and technology [2]. That paper presents a catalogue of design principles for technology intended for children.

Developing playability heuristics designed for evaluating mobile games [5] is conducted in another research paper. The paper presents a model consisting of game usability, mobility and gameplay heuristics. The heuristics have been designed through an iterative design process of a mobile game and by evaluating five mobile games, to validate the resulting heuristics. The developed heuristics were considered useful to identify playability problems in mobile games. Other usability heuristics about multiplayer games have also been set up [11], to be used when designing and evaluating network multiplayer games. These heuristics are called Network Game Heuristics (NGH).

Some of the previous research papers have made up heuristics, some specifically for the mobile interface [14, 10, 12], some specifically for games and playability [5, 11], and one for e-learning for children aged 5-13 years [1]. Other papers cover general design principles for mobile applications for young and older children [4, 8, 13, 2], but they have not made up some heuristics. Thus, some general heuristics for evaluation and to support creation of mobile touchscreen interfaces intended for children aged 3-5 are created to overcome these shortcomings. The heuristics of Nielsen therefore were ranked and reconstructed, some were removed, and new heuristics were added, to fit the criteria. This is done by conducting semi structured interviews and handing out surveys to people with experience of interaction design, children and application development.

3 Method

To conduct the task and create heuristics for mobile applications intended for young children, semi structured interviews were conducted and surveys were distributed to experts. Both methods were used because of the value of both quality and quantity. The interviews and survey revolved around Nielsen's heuristics [6], children behavior and mobile applications for children. The experts consisted of people with experience of application use by children aged 3-5 years and interaction designers. The surveys were conducted on internet, with Google's survey tool[1]. The only mandatory questions were the two first ones. The survey and interviews consisted of the same questions, but as the interviews were semi-structured, some spontaneous questions were made up during the interviews. All surveys and interviews were conducted in Swedish, therefore a translation of the questionnaire is presented in subsection 3.1.

The experts answering the survey consisted of 6 people with experience of application use by children and 12 interaction designers. The experts answering the interview consisted of one person with experience of application use by chil-

[1] Google forms: `https://www.google.se/intl/sv/forms/about/`

dren and one interaction designer. Most participants were around 18-30 years old, two were 31-40 years old, three were 41-55 years old, and one was older.

Some tips for new heuristics were taken from the papers about design principles for applications intended for children [1, 4, 8, 13, 2] and some were taken from the papers about game playability [5, 11] to make the applications more interesting. To adapt the heuristics to the mobile interface, some tips for heuristics were taken from the papers about touchscreens and mobile devices [14, 12, 10].

3.1 Presentation and questions for survey and interviews

This is a survey about development of mobile applications for children aged 3-5 years. The task is to create rules (heuristics) for mobile application development intended for young children to facilitate for developers. These heuristics will be developed by adapting some existing heuristics from Nielsen. It is possible that you will not be able to answer all questions. The questions are in the list below:

1. What area do you study, work or have experience with? Application development for children, seen children use mobile applications repeatedly or interaction design?
2. How old are you?
3. When developing applications for children aged 3-5 years, how important on a scale from 1 to 5 do you think each of these heuristics [6] are? 1 is unimportant and 5 is extremely important. The survey participant then gets to see all heuristics in table 1 and grade them separately.
4. How user-friendly do you think mobile applications are generally for children aged 3-5 years? Very useful, easy to use, quite easy to use, hard to use or useless.
5. What problems have you experienced with applications for children aged 3-5 years?
   (a) The child loses interest or concentration due to something boring.
   (b) The child does not understand what happens and gets confused.
   (c) The child presses the wrong button and ends up in an unwanted place.
   (d) The child ends up on another website or advertising site and cannot get back.
   (e) The child needs help from an adult to use the application.
   (f) The child cannot do what is desired and gets annoyed.
   (g) Something else?
6. What do you think caused these problems?
7. Can you give an example of a popular application used by children aged 3-5 years?
8. Can you briefly describe something about how a funny and user-friendly mobile application for children aged 3-5 years should be?
9. Can you mention anything else that is important to consider when developing mobile applications for children aged 3-5 years? For example, in terms of usability, opportunities/limitations with the mobile platform, entertainment and privacy.

3.2 The prototype

Finally, a prototype was developed with regard to the resulting heuristics. The prototype represents a mobile application with some small and simple games to choose between. One of the games is "Memory" and another involves stacking blocks. The prototype contains one start page, one page with links to all small games, and one page for each game. The tools used for this prototype are Adobe Animate, Adobe Illustrator and Figma. Some components and objects were created in Illustrator, the animations were created in Animate and the final prototype was put together in Figma.

4 Results

The result is divided into three subsections. Subsection 4.1 contains the survey and interview answers, which are put together. 4.2 contains the resulting heuristics. The third subsection 4.3 contains the prototype made with regard to the created heuristics.

4.1 The interviews and survey results

The heuristics of Nielsen were graded from 1 to 5 where 1 is unimportant and 5 is very important to consider when creating applications intended for young users. See table 2 for the average scores from the survey and interviews. The average assessment of how user-friendly applications intended for children are, was somewhere between quite easy to easy to use. See table 3 to see how many people have experienced the various problems when monitoring children using mobile applications. All text answers are translated if they were written in Swedish, and removed if they were irrelevant or not containing any information.

Usability Heuristics of Nielsen	**Score**
Visibility of system status	4
Match between system and the real world	4.55
User control and freedom	3.6
Consistency and standards	3.9
Error prevention	3.8
Recognition rather than recall	4.3
Flexibility and efficiency of use	3.9
Aesthetic and minimalist design	4
Help the user to recognize, diagnose, and recover from errors	3.75
Help and documentation	3.2

Table 2. A table with the average score for each heuristic of Nielsen [6], from survey and interview answers.

The text question "What do you think caused these problems?" gave these answers: "A little too difficult for children in that age", "The user interface" and "Too much advertisement that all of a sudden pops up".

Problem	**Nr. people**
The child loses interest or concentration due to something boring in the application.	1
The child does not understand what happens in the application and gets confused.	4
The child presses the wrong button and ends up in an unwanted place.	6
The child ends up on another website or advertising site and cannot get back.	6
The child needs help from an adult to use the application.	6
The child cannot do what is desired and gets annoyed.	3
Added: Too repetitively, the difficulty level does not increase in a legitimate way.	1

Table 3. A table showing the number of people who experienced the various problems, from survey and interview answers. Ten participants had enough experience to answer the question.

The text question "Can you give an example of a popular application used by young children?" gave answers like: "Sago mini game apps" and "Youtube".

The eighth question in subsection 3.1 gave answers such as: "Very simple layout with few choices and clear affordances", "Colorful, big interaction surfaces, clear, simple, not in a lot of steps for one task", "User-friendly, easy to understand, few alternatives, funny, sound, color, animation, environment exchange..", "It should have a clear and simple user interface with images and sound and one subject not many different like watching movie, puzzle, draw etc." and "Colorful, simple, without advertisement and purchases".

Finally, the ninth question in subsection 3.1 gave answers such as: "No popups, no advertisements, no text (they can't read), pictures, animations, colors, sounds (as feedback)", "Important that adults are not needed all the time.", "More important to think of the need of the child than making an application like any other", "Not requiring camera, not requiring information about the user, no advertisement, challenging and developing, simple so no one presses the wrong button." and "Less commercials".

4.2 The heuristics

After conducting the interviews and getting the survey answers some heuristics were created based on the answers and information from the earlier studies. The resulting heuristics are called "Touchscreen Usability Heuristics for Young Children" (TUHYC), and can be found in table 4.

Touchscreen Usability Heuristics for Young Children
Match between system and real world - The system should speak a language the child understands, with metaphors and natural interaction styles. The child should recognize the objects on the screen. The system should display information in a logical and natural order.
Recognition rather than recall - The child should not have to remember objects, actions or options, the application should hold information for the child. Important information and guidance should be visible or easily retrieved when needed. The child should be guided to find functionality with audio, animation and highlighted objects. No tutorials should be used, they may destroy the user experience.
Aesthetic and minimalist design - Remove irrelevant and rarely needed information. Use simple metaphors and avoid using long text elements. Keep the application simple with a simple, effective and visually attractive layout, and carefully adapted to one age category. The interface should be uncluttered and easy to use. Present few options, use a very simple navigation structure with step-by-step structure and keep the child close to the start page.
Visibility of system status - The child should know what happens on the screen in time with good feedback. Visual, sound based and animated feedback immediately for comprehension. Combining different kinds of feedback makes it more effective. Children presses everywhere so immediate feedback may reduce errors. Avoid long loading times, and use clear loading feedback or alternative entertainment if impossible.
Flexibility and efficiency of use - The system should be efficient to both experienced and inexperienced users. Challenge the child by varying the activities and difficulty, the application should be easy to use, but hard to master. There should be clear goals and the child should be rewarded when achieving these. There should be enough information when turning on the application.
Consistency and standards - Platform conventions may be used, if possible. Situations, actions, icons and metaphors should mean the same thing in the whole application and correspond to reality.
Error prevention - Eliminate errors from occurring, but do not use to much restrictions, or present a confirmation box before committing the action. Use constraints when needed and no in-app-purchases. Make clickable items look touchable and use error prevention for touch. Accept half-finished taps, tap times up to five seconds, both single and multi-touch input and many different tapping styles. Make interactive objects bigger than others, not smaller than an average fingertip, and far apart to avoid children tapping the wrong object.
Support playability - Evoke children imaginary and support curiosity to make the application more interesting. Use surprises, rewards, humor and interesting subjects. Keep the child in control and make the system support redo and undo.
Help and documentation - Even if the system should work without written documentation, it may be needed. The information should not be too large, be easily retrieved and user task focused. The help should also contain guidance with audio, animation or highlighted objects.
Compensate child ergonomics - Children often lay their mobile device in their lap so put no important objects at the lower edge, to make them see all objects. Put important objects to the left, to avoid right-handed children clicking on them by mistake. Avoid putting object close to the edge, children often hold their fingers far into the screen.

Table 4. A table with all resulting heuristics for mobile touchscreen applications intended for children aged 3-5. The motivation for each heuristic is in subsection 5.1. The heuristics are ranked according to importance in application development.

4.3 The prototype

The clickable prototype was created considering the newly created "Touchscreen Usability Heuristics for Young Children"[2]. See figure 1, 2, 3, 4, 5, 6, 7, 8 and 9, for images of the prototype. All pages in the prototype have natural transitions and interaction styles with tapping and dragging (heuristic 1). The transitions between the start page, the game page and the pages for each game are sliders to give the impression of moving sideways, and a simple navigation structure (heuristic 3). The child is always only one step away from the game page.

Guidance is available in all short games in the prototype, in form of text, animations and highlighting (heuristic 2). Clickable items have a shadow below to make them appear clickable. The layout of each page is simple with big interactive objects, far apart (heuristic 7). Feedback in different forms is used, including giving rewards when completing a game and changing color on the paintbrush when selecting a color (heuristic 4). All short games challenge the child by varying the difficulty, with i.e. more Memory cards (heuristic 5). Objects, metaphors and actions are recognizable from the real world (heuristic 1). The application supports playability with rewards, interesting subjects and by giving the user freedom to paint and use the blocks as he wants (heuristic 8). Important objects are placed to the left on the screen and there are no important objects in the lower edge, or really close to the other edges, to support child ergonomics (heuristic 10).

Fig. 1. The start page of the application, containing two animations and a play button. The animations support playability and makes the application funnier to use (heuristic 8).

[2] The prototype is accessible online at `https://www.figma.com/proto/Wnf3fCdkVYEsxB9PwtfREQ/Student-Conf?node-id=1\%3A2&scaling=scale-down`

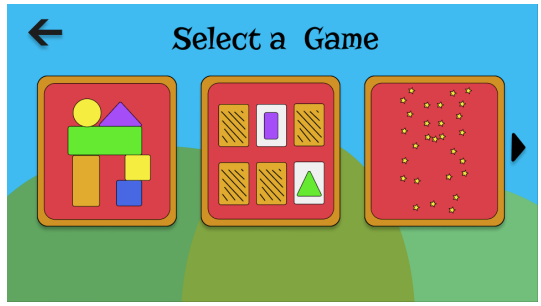

Fig. 2. The game page contains links to all short games. This page shows more games when sliding left. The interactive areas are big, look touchable and are informative, with their describing images (heuristic 7).

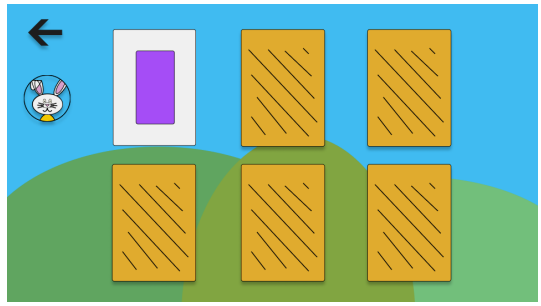

Fig. 3. The memory game contains a set of cards that look touchable when the user can click on them. The rabbit button to the left provides help, when the user clicks on it (heuristic 9).

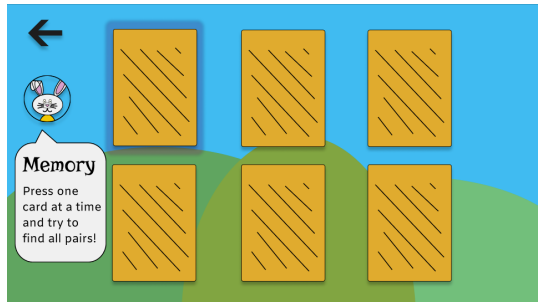

Fig. 4. The memory game, when the rabbit button is pressed. Help is displayed and the highlighted card flashes to get attention (heuristic 9).

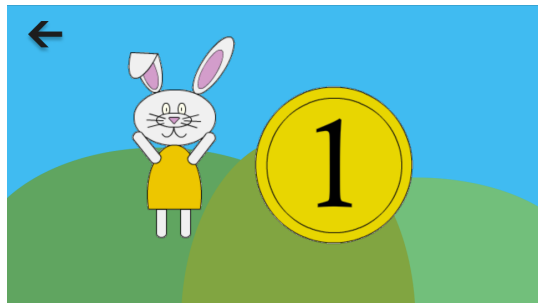

Fig. 5. The reward given when the user finishes a game, in form of an animated happy rabbit and an animated coin (heuristic 4).

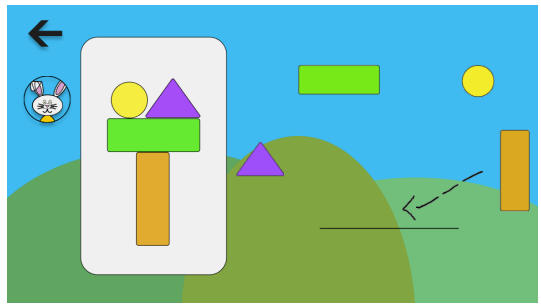

Fig. 6. The stacking blocks game, giving guidance in form of an animation after a certain time (heuristic 2). The animation spins the orange rectangle and draws an arrow.

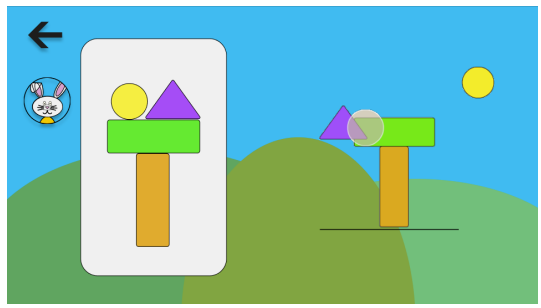

Fig. 7. The stacking blocks game, showing how the child finishes the game after the given guidance.

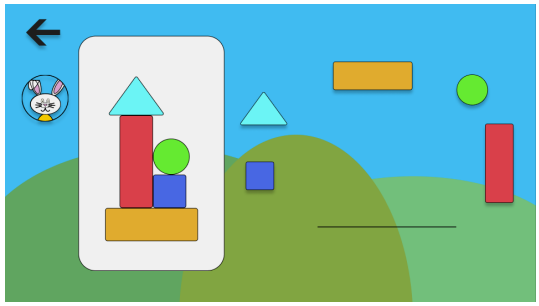

Fig. 8. The stacking blocks game, the next level adds one more block to challenge the child (heuristic 5).

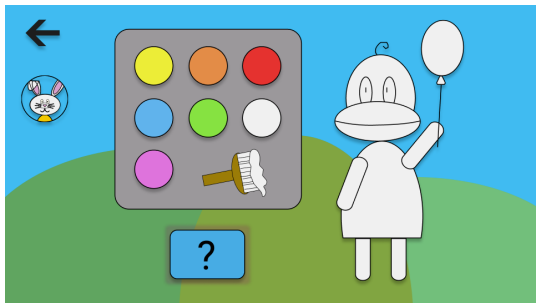

Fig. 9. The painting game, where the child should color the animal according to the information the "?"-button holds. The child is given guidance to press and hold the "?"-button, the button flashes until the child presses it (heuristic 2).

5 Discussion

After reading all previous work in the cited papers, conducting the interviews and getting the answers from the survey, the resulting information was evaluated into suitable heuristics, which were demonstrated in a prototype. All survey answers were not completed since most interaction designers did not have experience with children in these ages. But the interviews and the completed surveys gave good input. The reason why some survey questions were not mandatory is that we did not want guesses from people without experience, and we still wanted them to submit what they had filled in. But the survey and interviews produced usable input in the end.

5.1 Motivation for the created heuristics

Heuristic: Match between system and real world. The answers from the survey suggested that the heuristic named "Match between system and real world" is the most important one. To let the child recognize the real world through the system and thus facilitate the comprehension. This may make the application easier to use, which almost everyone answering the text questions

mentioned as desirable. One paper also discussed about having interesting and real subjects in the application, for the child to recognize [1]. Other papers mentioned the importance of making the application and interaction natural [1, 4, 5, 14, 13]. All this led to a reformulation of the heuristic.

Heuristic: Recognition rather than recall. This heuristic was also very important according to the survey answers. Children needing help from their parents when using an application was one of the biggest problems with applications for young children. Children getting confused also seemed like a recurrent problem. These problems indicate that some kind of guidance may be needed. One of the interviewees and one of the few answering the text questions mentioned animations, images, colors and audio, as important. They suggested this because very young children cannot read. Other papers were also recommending some kind of guidance with audio, animation and highlighted objects, and do not recommend tutorials as they may destroy the user experience [4, 13]. Both papers and our research pointed towards easily retrieved guidance, and no tutorials so the heuristic description was changed according to that.

Heuristic: Aesthetic and minimalist design. Almost everyone answering the two last text questions wanted to keep the application simple with a clean layout and with few choices. Some also mentioned the navigation structure, among them one interviewee who talked about the importance of a simple navigation structure and to keep the user close to the start page. Problems that were recurrent were children getting confused, children needing a parent to explain and children pressing the wrong button and getting lost. These problems indicate a need of a simpler layout and navigation structure to facilitate for the child. As children aged 3-5 years often cannot read, long text elements should be avoided. There are also papers mentioning a simple layout as important [1, 4, 14], and another paper explaining the importance of adapting the application towards one age category [8]. Other studies also emphasize the importance of a simple navigation structure [1, 4]. All these conclusions about simplicity were put into the heuristic about "Aesthetic and minimalist design".

Heuristic: Visibility of system status. Visibility of system status was considered important by the survey answers. One survey answer wanted visual, sound based and animated feedback, to keep the child informed about what is happening. Others also mentioned audio, funny visual elements and animations as important parts of the application. The majority of those who answered the text questions are not interaction designers, thus they do not know the importance of feedback and how it is used in applications. That may have caused few feedback answers. But there were papers discussing feedback [1, 4, 14, 13, 2] and how to combine different kinds of feedback to make it more powerful [4]. Some papers also mentioned that long loading times may ruin the user experience for the child, and how to deal with it [4, 14]. All this information resulted in the heuristic about "Visibility of system status".

Heuristic: Flexibility and efficiency of use. Both interviewees pointed out that the application should challenge the child, to make room for improvement and learning. One pointed out that it is a big problem when applications

are too easy, when they do not challenge the child and get boring. The heuristic was not graded very high, thus its description is changed to focus more on the flexibility of challenging the child, and not on accelerators. Thus, this heuristic will focus on the importance of challenging the child enough, as other papers also mentioned [1, 2, 11], and adapting the application to a specific age category, as another article concluded [8].

Heuristic: Consistency and standards. One survey participant wrote that it is more important to make a usable application for children than making an application like any other. This may be interpreted as if conventions are not important in applications for young children, as one interviewee also thought. Some papers mentioned standards [1], but papers about younger children does not [4, 13] and one paper even mentioned that conventional interfaces are inappropriate for such young children [2]. But still conventions are good to follow in order to reduce cluttered interfaces and to give tips, which another article concluded [8]. Therefore the created heuristic says, "conventions may be used" instead of "conventions should be used". The heuristic was still quite highly ranked by the survey participants, so they might at least think it is important with consistency. As children cannot read, it is important to use recognizable icons and metaphors that mean the same thing everywhere on the application, instead of text. Other research papers also mentioned metaphors as important for understanding and navigation for young children [4, 2]. Metaphors and icons were therefore added to the heuristic description, instead of words.

Heuristic: Error prevention. Error prevention were considered quite important according to the survey. An interviewee thought that too much restrictions would ruin the user experience. Thus, eliminating errors seemed important, depending on the error and its consequences, if it does not constraint the child too much. Some text answers were against advertisement, like other research [4, 13, 8], and the problem about advertisement seemed recurrent. An advice about no advertisement were therefore added to the heuristic. Two recurrent problems from the survey arose though children tapping the wrong button, thus it is important to accept many input tapping styles, as another paper also concluded [13]. A study and two survey answers also wanted to keep the interactive spaces big and far apart to avoid children pressing the wrong button [1]. Some tips about tapping error prevention were therefore added to the heuristic.

Heuristic: Support playability. The heuristic about user control and freedom was graded low in the survey so this heuristic will be adapted and renamed to "Support playability". The system should still support redo and undo, and keep the user in control, but more in a sense of playability where imaginary and curiosity should be supported to make the application more interesting. To let children control the application to get an outlet for their creativity. This idea is supported in some of the text answers, as they mention funny events, and funny elements with colors and animations. Only one person has experienced the problem of children getting bored, but more have experienced the problem of frustrated children, not being able to do what they want. This indicates the importance of supporting playability and keeping the child in control. There is

also a paper about supporting curiosity and imaginary [1], and papers wanting to keep the user in control [1, 5, 8].

Heuristic: Help and documentation. The heuristic about helping users to recognize, diagnose, and recover from errors was assembled with the heuristic about help and documentation because children might not be interested in diagnosing errors, as one interviewee also mentioned. They want to play the game and just get rid of errors, and they cannot read so written help may be unhelpful. Therefore some other guidance may be added to the text. Both heuristics were considered less important to the survey participants, but some written help should always be there in case anyone needs it, according to Nielsen [6], so the heuristic is still left.

Heuristic: Compensate child ergonomics. One paper discussed how children use the mobile device in a different way, compared to how adults use it [13]. They often lay it in their lap, hold their fingers far into the screen and tap in a different way. Our study also indicated a problem with children often tapping the wrong object and getting lost. This problem may have arisen through ignorance that children are using their device in a different way. Thus, a heuristic about child ergonomics to teach how children use the mobile device is added to the list. Our survey and interviews gave no answers about the child's use and ergonomics, therefore this heuristic is given low priority.

5.2 Limitations, Consequences and Future work

The newly created heuristics encompass very young children, in a very narrow age category. They may be usable on children at the age of two but very young children without experience might still have a hard time using the application. The heuristics may also be usable on older children, but not as useful since they can read some and therefore have developed other needs. Thus, these heuristics have a quite limited target group. The heuristics are furthermore only applicable on touchscreens, but some hints may be taken from these heuristics even if another platform is used. The prototype also has its limitations, it is only a prototype, far from a product, and it is not tested.

The consequence of creating these heuristics is that they make it easier to create mobile applications intended for young children without much testing. Usability tests may still be needed because child behavior is unpredictable and testing may reveal other kinds of problems. The heuristics summarize tips concluded from several studies and present them in an easy way. Other studies sometimes present their design principles or heuristics in a cluttered way, but these created heuristics are clear, short and concise. Thus, they may be easier to use and therefore facilitate for application developers.

There is some future work that may be done in this area. Heuristics are easy to use when designing mobile applications, specific heuristics for other age categories, other platforms or children with disabilities should therefore be appreciated. But there is still some work left to do with our heuristics. The "Touchscreen Usability Heuristics for Young Children" are not tested, nor the prototype cre-

ated with the heuristics. Thus, some future work with these created heuristics is still needed.

References

[1] Asmaa Alsumait and Asma Al-Osaimi. Usability heuristics evaluation for child e-learning applications. *Journal of Software*, 5(6):654–661, 6 June 2010. http://www.jsoftware.us/vol5/jsw0506-14.pdf, accessed 2019-09-07.

[2] Sonia Chiasson and Carl Gutwin. Design principles for children's technology. January 2005. http://hci.usask.ca/publications/2005/HCI_TR_2005_02_Design.pdf, accessed 2019-11-06.

[3] Robin Jeffries, James R. Miller, Cathleen Wharton, and Kathy M. Uyeda. User interface evaluation in the real world: A comparison of four techniques. In *Proceedings ACM CHI'91 Conference*, pages 119–124, 28 April 1991. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.330.1188&rep=rep1&type=pdf, accessed 2019-11-22.

[4] Ola Johansson and Morgan Karlsson. App development for children. Computer science, Malmö Högskola, 20 September 2012. http://muep.mau.se/bitstream/handle/2043/17350/C_Uppsats_Ola_Johansson_och_Morgan_Karlsson.pdf?sequence=2&isAllowed=y, accessed 2019-09-07.

[5] Hannu Korhonen and Elina M.I Koivisto. Playability heuristics for mobile games. In *Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services, Helsinki, Finland*, pages 9–16, January 2006.

[6] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. In *CHI '94 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 152–158, 24 April 1994.

[7] Jakob Nielsen. How to conduct a heuristic evaluation, 1 November 1994. https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/, accessed 2019-09-23.

[8] Jakob Nielsen and Katie Sherwin. Children's ux: Usability issues in designing for young people, 13 January 2019. https://www.nngroup.com/articles/childrens-websites-usability-issues/, accessed 2019-11-06.

[9] Donald A. Norman. *The Design of Everyday Things*. The Perseus Books Group, New York; United States, 2 edition, November 2013.

[10] Maria de Graca Pimentel and Olibário Machado Neto. Heuristics for the assessment of interfaces of mobile devices. In *Proceedings of the 19th Brazilian symposium on Multimedia and the web*, pages 93–96, November 2013.

[11] David Pinelle, Nelson Wong, Tadeusz Stach, and Carl Gutwin. Usability heuristics for networked multiplayer games. In *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, pages 169–178, May 2009.

[12] Virginica Rusu, Cristhy Jiménez, Silvana Roncagliolo, Cristian Rusu, and Rodolfo Inostroza. Usability heuristics for touchscreen-based mobile devices. In *2012 Ninth International Conference on Information Technology - New Generations*, pages 662–667, April 2012.

[13] Nikita Soni, Aishat Aloba, Kristen S. Morga, Pamela J. Wisniewski, and Lisa Anthony. A framework of touchscreen interaction design recommendations for children (tidrc): Characterizing the gap between research evidence and design practice. In *Conference: the Interaction Design and Children*, June 2019.

[14] Rosa Yáñez Gómez, Daniel Cascado Caballero, and José-Luis Sevillano. Heuristic evaluation on mobile interfaces: A new checklist. *The Scientific World Journal*, 2014, 11 September 2014. `http://downloads.hindawi.com/journals/tswj/2014/434326.pdf`, accessed 2019-09-24.

Author Index