Informatics

A Suggestion-Based User Interface for P2P Payments in Mobile Chat Applications

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Media and Human-Centered Computing

eingereicht von

Felix Walcher, B.Sc
Matrikelnummer 01427555

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Thomas Grechenig

Wien, 24. Jänner 2022

Felix Walcher   Thomas Grechenig

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

A Suggestion-Based User Interface for P2P Payments in Mobile Chat Applications

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media and Human-Centered Computing

by

Felix Walcher, B.Sc
Registration Number 01427555

to the Faculty of Informatics

at the TU Wien

Advisor: Thomas Grechenig

Vienna, 24th January, 2022

Felix Walcher   Thomas Grechenig

Erklärung zur Verfassung der Arbeit

Felix Walcher, B.Sc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 24. Jänner 2022

Felix Walcher

Kurzfassung

Mobile P2P (Person zu Person) Zahlungsmethoden werden immer populärer. Messengerdienste wie *Facebook Messenger* oder *WhatsApp* haben in ausgewählten Regionen P2P Zahlungsmethoden in ihren Benutzeroberflächen integriert. Um Zahlungen in diesen Messenger zu tätigen, sind die Benutzer/Benutzerinnen aufgefordert, den Zahlungsprozess manuell zu initialisieren und eine mehrstufige Konfiguration durchzuführen. Im Zuge dieser Diplomarbeit wurde eine *suggestion-based* Benutzeroberfläche entwickelt, die dem/der Benutzer/Benutzerin Zahlungen anhand des Konversationskontextes und der Tastatureingabe vorschlägt. Dieser Ansatz verspricht schnellere Zahlungen und eine verbesserte Usability. Die neuartige Benutzeroberfläche wurde in einer Usability Studie mit nachimplementierten Benutzeroberflächen bestehender Messenger verglichen. In der Usability Studie wurden die Benutzeroberflächen auf deren Effizienz, Effektivität und Benutzerfreundlichkeit geprüft. Die Ergebnisse der durchgeführten Studie deuten darauf hin, dass eine *suggestion-based* Benutzeroberfläche die Effizienz von P2P Zahlungen teilweise erhöht. Die automatischen Zahlungsvorschläge des neuartigen Prototyps haben die benötigte Zeit, um den Zahlungsprozess zu starten und um den Zahlungsbetrag zu definieren, signifikant verkürzt. Außerdem haben Zahlungsvorschläge, die auf Basis der Tastatureingaben erstellt wurden, die Zeit zur Fertigstellung eines Tasks um 31,2% und die Fehlerrate um 3,5-mal signifikant reduziert. Zahlungsvorschläge, die auf Basis des Konversationkontextes erstellt wurden, haben keine signifikanten Verbesserungen in der Geschwindigkeit von Zahlungsvorgängen oder Fehlerraten erzielt. Die Ergebnisse der durchgeführten Studie zeigen, dass die nachimplementierten Benutzeroberflächen bestehender Messenger bereits über sehr hohe Effektivität und Benutzerfreundlichkeit verfügen. Daher konnte die neuartige Benutzeroberfläche die Effektivität und Benutzerfreundlichkeit nicht signifikant verbessern. Die durchgeführten Post-Test Interviews haben ergeben, dass ein schneller Zahlungsprozess, der von dem *suggestion-based* Prototypen verursacht wird, Vertrauensprobleme zur Folge haben kann, da nur 6 der 20 Teilnehmer/Teilnehmerinnen der *suggestion-based* Benutzeroberfläche vertrauten. Die Post-Test Interviews der Usability Studie haben jedoch gezeigt, dass die neuartige Benutzeroberfläche von 11 der 20 Teilnehmer/Teilnehmerinnen bevorzugt wurde. Zusätzlich haben 15 der 20 Teilnehmer/Teilnehmerinnen Zahlungsvorschläge in Messenger Apps als nützlich wahrgenommen und könnten sich automatisierte Vorschläge für andere Messengerfunktionen vorstellen.

Schlagwörter: Mobile Messenger, Mobile Benutzeroberflächen, P2P Zahlungsmethoden, Usability

Abstract

Mobile Person-to-Person (P2P) payments are increasing in popularity. Popular mobile messengers like *WhatsApp* or *Facebook Messenger* have included P2P payment features directly into their user interface in selected regions. In these existing user interfaces, the users must manually activate the payment process and complete a multi-step configuration to perform a payment. A suggestion-based user interface was developed as a mobile chat application that provides the user payment suggestions based on the conversational context and the user's keyboard input. This novel approach aims to allow for faster transactions and to improve the overall usability. The novel user interface prototype was compared to the reimplemented user interfaces of the existing solutions through a usability study. During the usability study, the user interfaces were evaluated in terms of efficiency, effectiveness, and satisfaction. The results indicate that the implemented suggestion-based user interface partly improves efficiency. The payment suggestions feature of the novel suggestion-based user interface reduced the time to activate the payment process and to define the payment amount significantly. Furthermore, suggestions based on the user's input significantly reduced the task completion time by 31.2% and the error rate by 3.5 times. Suggestions based on the conversational context did not significantly improve the task completion time and the error rate. The study results revealed that the reimplemented user interfaces of the already existing solutions already provided high effectiveness and user satisfaction. For that reason, the novel user interface did not improve the effectiveness and user satisfaction significantly. The post-test interviews indicate that the faster transaction time of the novel user interface may cause trust issues as only 6 out of the 20 participants trusted the novel suggestion-based user interface. The conducted post-test interviews also indicated that 11 out of 20 participants would prefer to use the novel suggestion-based user interface. Furthermore, 15 of the 20 participants perceived payment suggestions in mobile messengers as useful and could imagine suggestions for other tasks besides payments.

Keywords: Mobile Messenger, Mobile User Interfaces, P2P Payments, Usability

Contents

CHAPTER 1

Introduction

1.1 Problem Description

The high level of adoption of smartphones and continuous enhancement of mobile network capabilities have enabled the introduction of new kinds of mobile services like mobile payment [LW20]. Mobile payments allow users to transfer money using mobile devices such as smartphones or tablets. The interest and adoption of mobile payment services is continuously increasing. According to a recent survey[1], 950 million people used mobile payments worldwide in 2019. Currently, the largest platform offering mobile payment services with an active user base of about 1.2 billion users (Q3/2019) worldwide is *Alipay*, followed by *WeChat*, *Apple Pay* and *PayPal* [2].

Mobile P2P (Person-to-Person) (P2P is often referred to as Peer-to-Peer, but in this thesis, P2P stands for Person-To-Person) payments allow private transactions between two individuals and are an alternative to cash and checks [SK12]. This payment method is increasing in popularity, and the global P2P payment market reached a value of 1,994.8 million dollars in 2020[3]. The COVID-19 outbreak had positive impacts on the growth of the P2P payment market, and predictions promise a market value of 4,615.9 million dollars by 2028. Additionally, mobile payment solutions embedded in social media

[1] Raynor de Best (2021): "Number of proximity mobile payment transaction users worldwide from 2018 to 2023", available at: www.statista.com/statistics/557959/global-mobile-proximity-payment-users/ (accessed August 20, 2021)

[2] Merchantsavvy (2020): "Amazing Stats Demonstrating The Unstoppable Rise of Mobile Payments Globally", available at: https://www.merchantsavvy.co.uk/mobile-payment-stats-trends/ (accessed October 31, 2021)

[3] Zionmarketresearch (2021): "P2P Payment Market By Transaction Mode, Type Of Purchase, And Applications: Global Industry Perspective, Comprehensive Analysis And Forecast, 2020 – 2028", available at: https://www.zionmarketresearch.com/report/p2p-payment-market, (accessed August 20, 2021)

platforms that allow P2P transactions between social media users gain in popularity [MHE19]. Messengers like *Facebook Messenger*[4], *WeChat*[5], *WhatsApp*[6] or iMessage[7] have included P2P payment features directly into their chat user interface. In existing solutions, users have to actively select an icon above the keyboard in order to initiate a P2P payment in the chat user interface that is similar to the initial process of sending multimedia content, stickers or a location to the chat partner. Except for *iMessage*[6] and *Facebook Messenger*[4], users leave the current context of the chat user interface while completing the multi-step configuration and confirmation of the P2P payment. For example, in *WhatsApp*[6] , users are able to send money to other social media users by hitting the “Transaction“ icon during a chat conversation. This interaction is followed by defining the amount of the transaction in a new view. In *iMessage*[7] and Facebook Messenger[4], users are not leaving the current context of the chat interface. The transfer can be configured and completed in a view that replaces the keyboard. This approach seems to be promising but it still requires numerous necessary interactions to perform a P2P payment. A reduction of the necessary interactions to perform a payment and a user interface that does not interfere the conversation flow could improve the usability of the P2P payment features in mobile chat applications. Higher usability could lead to an even better acceptance and wider adoption of P2P payments in mobile chat applications.

The aim of this thesis is to design and prototype a novel user interface approach for P2P payments in chat applications that allows faster transaction by reducing the necessary steps to perform a P2P payment and by increasing the overall usability by not interfering the conversation flow (e.g. by leaving the current context). To achieve this, the novel user interface should provide the user with real-time inline-suggestions for transactions. Suggestion-based approaches aim to increase the usability in terms of efficiency, effectiveness and user satisfaction of P2P payments in chat applications by not interfering the conversation flow.

1.2 Expected Results

The goal of this thesis is to improve the usability of user interfaces for P2P payments by providing inline-suggestions based on the conversation context and the user's input. Providing the user with inline-suggestions for transactions aims to reduce the number

[4]Elise Moreau (2020): “How to Pay Facebook Friends With Messenger”, available at: https://www.lifewire.com/pay-facebook-friends-with-messenger-4146438. accessed (August 20, 2021)

[5]WebNotes (2019): “How to Do Money Transfer in WeChat Accounts?”, available at: https://www.webnots.com/how-to-do-money-transfer-in-wechat-accounts/ (accessed August 20, 2021)

[6]Ritesh Bendre (2020): “WhatsApp Payments: How to set up, invite and send money to your friends”, available at: https://www.bgr.in/how-to/whatsapp-payments-how-to-set-up-invite-and-send-money-to-your-friends-573900/ (accessed August 20, 2021)

[7]Support Apple (n.d.): “Send and receive money with Apple Pay ”, available at: https://support.apple.com/de-de/HT207875#sendmoneyinmessages (accessed August 20, 2021)

of required interactions to perform a payment and does not interfere the conversation flow like already existing user interfaces. Usability is a key factor for the success of software [CAGLGC+18]. Bad usability leads to lower performance. A suggestion-based user interface for P2P transactions will be implemented as a prototype and compared to already existing user interface approaches. By continuously analysing the context of the chat conversation and the user's keyboard input, the new approach will offer suggestions about possible P2P payment transactions. This approach is promising because it has already shown significant usability improvements for the use of stickers [LHM+20], images [KGH+20], replies [YFR+18], and messaging-related tasks [CXDH19] in chat applications. Currently, there is a lack of evidence in literature about the use of inline-suggestions for P2P Payments in mobile chat user interfaces.

This thesis should answer the following research questions:

- RQ1: Is a suggestion-based user interface more efficient than already existing solutions for P2P payments in chat applications?
- RQ2: Is the user's effectiveness higher with a suggestion-based user interface for P2P payments in mobile chat applications than already existing solutions?
- RQ3: Is the perceived level of satisfaction higher by using a suggestion-based user interface for P2P payments than already existing solutions?

Efficiency, satisfaction and effectiveness are the leading attributes of usability [Wei20]. Efficiency describes the speed and accuracy a user performs a task. It will be measured by the amount of time spent on each activity, the time needed to complete a specific task and the error rate. For the evaluation of efficiency, the task completion time, the time spent on each activity, and error rate of each task will be collected and analysed during the usability study. RQ1 will determine whether the novel approach or the existing user interfaces are more efficient. Effectiveness is the user's ability to complete a certain task [Wei20]. It will be measured by the number of successfully completed tasks. RQ2 will help to find out if a suggestion based user interface offers higher effectiveness than already existing solutions.
The user's satisfaction is the level of comfort, pleasure and fulfilment of expectations. It will be measured according to the System Usability Scale (SUS) questionnaire [Bro95]. The questionnaire contains 10 questions of the original SUS. Each question needs to be ranked with a value between 1 and 5. The evaluation of this questionnaire offers a score between 0 and 100 that will be used to compare the different prototypes. The third research question will find out which approach provides the user with more satisfaction based on the calculated SUS score.
The results of the usability study will be used to answer the research questions in order to determine if a suggestion-based user interface improves the usability of P2P payments in mobile chat applications.

1.3 Methodological Approach

A literature review will be carried out about the concepts and existing approaches related to suggestion-based user interfaces. The literature review will also focus on theories and concepts of mobile P2P payments.

Research will also be done about existing mobile chat applications that contain a P2P payment feature. Based on related work in the field of mobile payments and context-based user interfaces for inline-suggestions as well as existing solutions, a novel approach for a P2P payment user interface will be designed and implemented using Design Space Analysis [MBS93].

The prototype will be developed as a mobile cross-platform application (*iOS* and *Android*) using *Flutter*[8], *Google's* UI toolkit. The implemented prototype will be evaluated using a comparative usability study with 20 participants to test and compare it with existing solutions. The participants of this study will be recruited based on their experience with mobile chat applications and mobile banking. To compare the prototype to existing solutions in a usability study, their user interfaces will be re-implemented as additional prototypes. Currently, the P2P payment features of *WhatsApp*, *Facebook Messenger*, *WeChat* and *iMessage* are restricted to use in only a small number of countries worldwide (e.g the United States of America) [MHE19]. Therefore, in order to perfectly imitate the existing solutions, their prototypes will be implemented based on documentations, tutorials, and videos that can be found online.

The usability study will consist of 2 tasks the participants have to conduct with the re-implemented prototypes of already existing user interfaces and the novel user interface. During the tasks the participants have to transfer money in different scenarios. The prototypes will record the error rates, task completion times and time spent on each activity as quantitative data. This data will be used to evaluate the different prototypes on their efficiency. Furthermore, the number of successfully completed tasks will be used to evaluate the effectiveness of the different prototypes. The collected data will then be evaluated with descriptive statistics and Analysis of variance (ANOVA) to allow a comparison of the different prototypes in terms of efficiency and effectiveness. After the completion of the tasks of one prototype, data of the perceived usability of the prototypes will be collected through a subjective questionnaire. The questionnaire will be according to the System Usability Scale (SUS) [SZN19]. The evaluated SUS scores of the tested prototypes will be used to compare the prototypes on the user satisfaction. Additional, unstructured post-test interviews will be conducted to get qualitative data about the perceived usability and trust of the tested prototypes. The results of the usability study should find answers to the defined research questions.

[8]Flutter (n. d.), "Design beautiful Apps ", available at: https://flutter.dev (accessed December 5, 2020)

1.4 Structure

Chapter 2 contains a literature review about mobile P2P payments and the different types of Recommender Systems. In this chapter, related work concerning suggestion-based user interfaces in mobile chat applications will be discussed.

In Chapter 3, the design and implementation process of the novel suggestion-based prototype will be documented. At first, this chapter contains detailed documentation of the already existing approaches of P2P payments in mobile chat applications. After this, the design decisions based on a design space analysis will be discussed. This chapter also contains detailed documentation of the three different implemented prototypes of this thesis. The chapter ends with a short overview of the used technology and technical architecture of the implemented prototypes.

Chapter 4 will contain detailed documentation of how the three implemented prototypes were evaluated with a comparative usability study.

Chapter 5 contains the evaluation of the collected data of the usability study. This chapter includes a summary of the background survey and the post-test interview. Furthermore, the results of the statistical analysis of the collected quantitative data will be interpreted.

In Chapter 6, the findings of Chapter 5 will be discussed, and the three defined research questions will be answered.

Chapter 7 contains the summary of this thesis, including its most important insights. This chapter ends with an outlook for future work.

CHAPTER 2

State-of-the-Art

In this chapter, the underlying theories and concepts of a suggestion-based user interface approach for P2P payments will be discussed. At first, this chapter contains an overview of mobile P2P payments. Recommender Systems (RS) are the underlying concept of suggestion-based user interfaces. Because of this, in this chapter RS and in particular Context-Aware Recommender Systems (CARS) and Conversational Recommender Systems (CRS) will be discussed. Furthermore, the existing suggestion-based user interface approaches for mobile messengers found in the literature will be highlighted.

2.1 Theories and Concepts

2.1.1 Mobile P2P (person-to-person) payments

Mobile P2P payments allow private transactions between two individuals and are an alternative to cash and checks [SK12]. The global P2P service market size was estimated to be worth 16.13 Billion USD in 2020[1], and this payment method is increasing in popularity. It is estimated that the mobile P2P payment sector will reach a market size of 43.26 Billion USD by 2026, with an annual growth rate of 17.86%.

Since 1990, payment methods have faced a rapid digital transformation caused by users having access to new technologies and corporations that focused on becoming more customer-centric [LRVRLC21]. The digital transformations changed the strategies of corporations, consumer patterns, and how corporations interact with consumers. Financial corporations were pioneers of the digital revolution by introducing new technologies

[1]Research and Markets (2021): “Person-to-Person Payment Services Market Research Report by Services, by Application, by Region - Global Forecast to 2026 - Cumulative Impact of COVID-19”, available at: https://www.researchandmarkets.com/reports/4896701/person-to-person-payment-services-market-research#tag-pos-1 (accessed August 9, 2021)

like mobile banking, mobile P2P payments, or blockchain technology. The speed of technology adoption depends on the perceived user's needs for specific services and the support of large corporations. In the 1990s, the first attempts to introduce digital P2P payment methods failed like *CyberCash* for example. In 1999, *PayPal* introduced the first successful electronic P2P payment system that operated independently from a traditional financial company. The introduction of *eBay* caused the need for a secure online payment platform, and after the cooperation with *eBay*, *PayPal* became a mass payment service.

After that success, many non-financial companies began to offer payment services like *Square Cash*, *clearXchange*, *Google Pay*, and *Apple Pay*. Additionally, mobile payment solutions embedded in social media platforms that allow P2P transactions between social media users gained popularity. Messengers like *Facebook Messenger* or *WeChat* do already support P2P payments [MHE19]. *Facebook* allows its users to make and request payments from other *Facebook* users via *Facebook Messenger*. The users need to link their debit or credit card or *PayPal* account to their *Facebook* profiles to process payments. In group chats, the P2P payment feature can be used to split a payment among the group participants. Another social network that offers P2P payments is *WeChat*. *WeChat* is a social media network that is mainly popular in Mainland China, developed by *Tencent*. The application provides multiple mobile payment modes like NFC or QR Code. Users that link their debit or credit cards to *WeChat* can process P2P payments, buy goods and services, or receive funds from other users.

Traditional financial companies began to fear P2P payment platforms because they were afraid of losing control over the money flow [LRVRLC21]. To encounter this, many banks decided to offer platforms as alternatives to the new platforms of non-financial companies that provide payment services based on their existing infrastructure and market access. For example, *Bizum* is a successful P2P payment service that a Spanish traditional financial company operates.

2.1.2 Recommender Systems (RS)

Recommender Systems (RS) are systems that provide users suggestions for items of any data [RD19]. To do so, RS handle complex information and suggest items based on the user's preference. RS aim to support users with the decision-making process and to improve the user experience. RS are used in e-commerce, e-learning, business services, etc.

The field of RS emerged from Information Retrieval and message filtering to solve recommendation problems in the 1990s. Traditional RS are structured in two dimensions (Users x Items). This structure represents users that may interest in items. With a rating function, a RS provides the user suggestions of items the user might be interested in [BMB17]. Traditional RS can be distinguished into three categories: content-based, collaborative, and hybrid-based RS.

Content-based RS suggest users items depending on past preferred items. For this approach, a user must rank sufficient items until the RS can suggest new items. Because of this, a content-based RS suggests only items based on past preferences.

Collaborative RS provide suggestions based on the preferences of similar users. This approach is challenging for users that have unique preferences. Furthermore, the accuracy of collaborative RS depends on the size of the user base.

Hybrid RS are a combination of content-based and collaborative RS. Such systems provide item suggestions based on the user's and similar users preferences.

Nowadays, traditional RS are rare because of their limitations in handling complex data. In the first decade of the 2000s, researchers began to consider the user's context for recommendations [RD19]. This approach led to the introduction of Context-Aware Recommender System (CARS). In CARS, the two-dimensional structure of RS was extended to a three-dimensional structure (User x Item x Context) by considering the user's contextual information. By considering additional information about the user's context, this approach aims to provide more personalised suggestions. The user's context is, for example, the user's current activity, behaviour, emotional state or location [BMB17].

The high adoption of smartphones leads to an enormous number of considerable context information for CARS. For example, a restaurant recommendation application can suggest the user restaurants depending on the user's taste, location, schedule, etc. CARS needs to collect, model, and incorporate the context and make decisions to provide context-based suggestions.

Contextual information collected and considered by mobile CARS can be separated into six groups [BMB17]:

- **Location-based contextual information:**
  Location-based CARS use different techniques to estimate the user's position. For example, by Global Positioning System (GPS) or Global System for Mobile Communications (GSM). The Travelling and Point of Interest (POI) sector are popular applications for location-based CARS. For example, CARS can suggest users to visit POI based on the user's location, travel patterns, preferences, etc.

- **Social contextual information:**
  Social media platforms collect and process a massive amount of personal information about their users. Social information includes personal interests, friends and similar users, likes, dislikes, etc. The use of social-based information for CARS may raise privacy and trust issues for social media platforms.

- **Temporal contextual information:**
  Temporal context information can include, for example, the time of the day (morning, evening, night), weekday, season, etc. Such data can be helpful for providing suggestions. For example, studies have shown that people prefer some music genres to specific times of the day or particular events, like Christmas songs.

- **Emotional contextual information:**
  Users commonly use smart devices and wearables to monitor their physiological state and well-being. Data collected through these devices can provide knowledge about the user's state of mind like happiness, sadness, fear, anger, etc. Based on this data, CARS can consider the user's emotional context for suggestions.

- **Activity-based information:**
  Smartphones and wearables collect motion data by measuring the X, Y, and Z-axis acceleration. Based on this data, CARS can distinguish between different human activities. For example, between low-level actions, high-level actions, indoor activities, or outdoor activities. Considering activity information, CARS can provide suggestions based on the current activity and the user's activity patterns.

- **Multi-dimensional information:**
  CARS tend to combine multiple contextual information for suggestions because of the dependency between contextual factors. For example, the weather and the user's surrounding may influence the user's mood. CARS that consider multiple contextual information require complex recommender algorithms for the suggestions of items.

2.1.3 Conversational Recommender Systems (CRS)

The aim of Conversational Recommender Systems (CRS) is to provide users suggestions for tasks through dialogues. Through dialogues and feedback, the user provides the system with his current preferences. Additionally, CRS can give the user explanations to suggestions.

Currently, there is no widespread definition of CRS. [JMCC21] defines CRS as followed:

> "A CRS is a software system that supports its users in achieving recommendation-related goals through a multi-turn dialogue." [JMCC21]

One of the first CRS was a computer in a library that provided users reading suggestions in the late 1970s [JMCC21]. These suggestions were based on the user's personality and preferences the system gathered through interactive questions in natural language. At that time, CRS struggled with the limitations of natural language processing. In recent years language processing has advanced. Nowadays, users interact through voice commands with their smartphones or smart home devices. The latest systems detect commands with high recognition accuracy. Chatbots processing natural language are widespread and are used in various applications. For example, chatbots are increasingly used for customer service. However, the latest voice assistants and chatbots are still far from the desired support that could provide CRS. Nowadays, most CRS are based on machine learning instead of gathering preferences through simple dialogues.

CRS operate task-oriented. They provide recommendations to support users with the decision-making process. Furthermore, CRS can help users finding relevant information. Part of CRS is also to gather user preferences and provide explanations to given suggestions. CRS usually interact with users through multi-turn conversations. This characteristic distinguishes CRS from simple question answering systems. An example of a simple question answering system is Apple's Siri. Siri is capable of reacting to recommendation requests but struggles with multi-turn conversations. To support multi-turn conversations, CRS must conduct state management to consider past conversations and the current state.

Inputs and outputs can vary in modalities. Inputs of CRS could be speech, text, button inputs, gestures, etc. Output, for example, can be voice, text, multimedia content, etc. Additionally, for CRS it is not defined who drives the multi-turn conversation. It can be the user, the system, or another user. CRS have similarities with conversational search. CRS and conversational search rank suggestions depending on their relevance, the user's preferences, and search query. Furthermore, both systems have to understand and interpret user intents. However, conversational search systems are limited to speech and text inputs.

2.2 Related Work

2.2.1 MessageOnTap

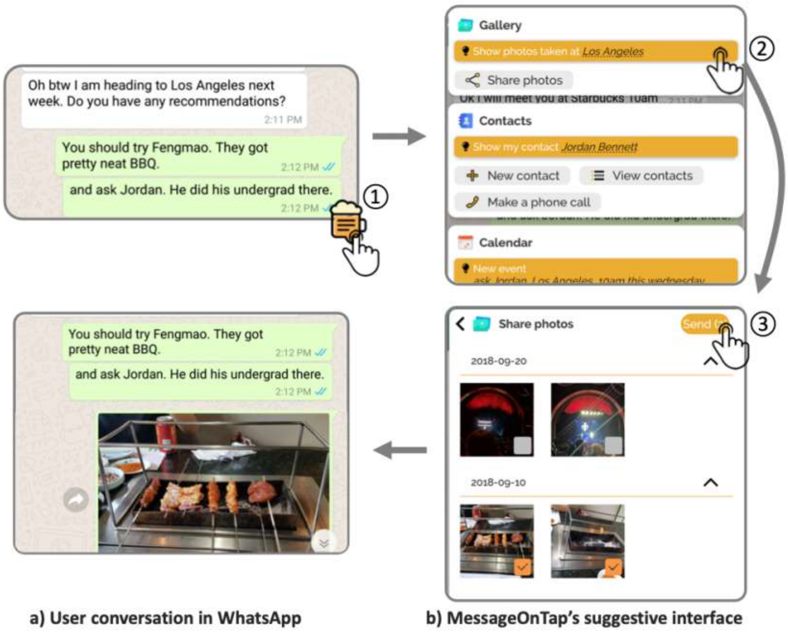


Figure 2.1: Interaction flow of an image suggestion [CXDH19]

Text conversations in mobile chat applications often contain information that leads to tasks in external apps like, for example, checking calendar entries, creating a reminder, or sharing content [CXDH19]. To perform such tasks, the user needs to switch to another application. For example, if the user receives the message "When is the party tonight?" the user may intend to check today's calendar entries. The user needs to leave the messenger application, change to the calendar app, and search in today's calendar entries for the information. After this, the user has to return to the messenger and reply with the information acquired from the calendar app. For this, the user has to formulate the reply manually or copy-paste the information from the calendar entry. As demonstrated in the previous example, tasks across different apps are inefficient and harm the conversation flow.

MessageOnTap is an interface that suggests messaging-related tasks as shortcuts that require access to external apps. The interface was implemented as an overlay for popular messenger apps like *WhatsApp* or *Facebook Messenger* for *Android*. When *MessageOnTap* is activated, the interface continuously analyses the current message conversation for key phrases that are related to temporal information, persons, and events. Based on key-phrases *MessageOnTap* suggests shortcuts for tasks in external apps. *MessageOnTap* supports shortcuts for the *PhotoGallery*, *Documents*, *Calendar*, *Reminders*, and *Contacts*. Figure 2.1 highlights the interaction flow of image suggestions based on detected keywords in the chat conversation of *WhatsApp* (1). Tapping on the *MessageOnTap* icon displays the overlay user interface that provides the user with message-related task suggestions (2). In this example, the system recognised the location "Los Angeles" as the most relevant keyword. By tapping on the button "Show photos taken at Los Angeles", *MessageOnTap* suggests the user images that were taken at this location (3). Now the user can attach suggested images to the chat conversation.

For the keyword extraction *MessageOnTap* makes use of the Google Natural Language Application Programming Interface (API). Furthermore, text processing takes place in the cloud to be more battery efficient. This architecture may cause concerns about privacy because messenger conversations usually contain sensitive data. The suggested shortcuts are ranked based on the relevance to the conversation. The system also supports auto-completion for tasks that require text input, like creating a new contact or reminder.

Suggested shortcuts for tasks in external apps reduce the time required to switch to another app and complete a specific task. An in-lab user study was conducted to evaluate the ability and effectiveness of *MessageOnTap*. Participants were 3.1 times faster performing a message-related task by using *MessageOnTap*. The post-survey and post-interview of this study highlighted that the participants enjoyed using *MessageOnTap* and could imagine similar suggestion features in mobile messenger applications.

2.2.2 MilliCat

Visual communication in mobile messenger is increasing with the use of visual elements like images, emojis, GIFs, and memes [KGH+20]. To search and find the right image during a chat conversation can be cumbersome and distracting. Current mobile messengers require numerous steps to include a public image from the internet into the chat conversation. The user has to switch to another application, find the right image on the internet and store it on his phone. Then the user needs to include the saved image into the chat conversation. Messengers offer shortcuts for importing images to conversations, but these shortcuts still require numerous steps to perform. This procedure may also discourage the use of images during a conversation. A conducted survey indicated that 81.57% of the asked participants would like to receive image suggestions for public images in mobile chat applications depending on the conversational context [KGH+20].

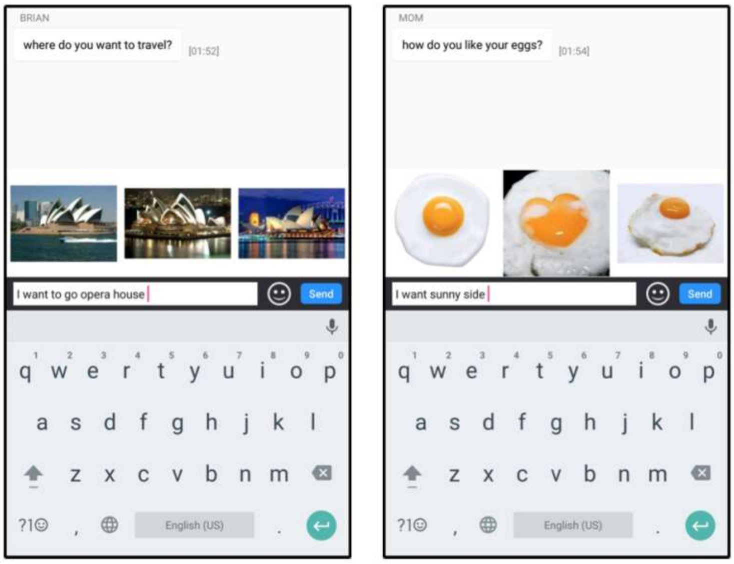


Figure 2.2: Image suggestions of MilliCat [KGH+20]

MilliCat is a real-time image recommender system that recommends public images based on the user's input text in a mobile messenger. This feature can be compared to auto-completion of text. After receiving image suggestions, the user can add the most suitable image to the conversation with a single tap (see Figure 2.2). *MilliCat* extracts keywords and phrases from the user's text input and uses them to query an external image archive. *MilliCat's* keywords are limited to concrete nouns in combination with adjectives.

Personal images are used for social communication, and because of privacy concerns, *MilliCat* focuses only on public images. The developers of *MilliCat* claim that public images could be helpful to share information or to emphasise opinions in chat conversations.

To evaluate *MilliCat* two user studies were conducted with 45 participants. At first, a controlled lab study analysed how potential users would react to image suggestions in a messenger. The second study was a field study in which participants used *MilliCat* for 8 to 10 days. This field study indicated that *MilliCat* increases the use of images during chat conversations by 1.8 times. Furthermore, *MilliCat* reduced the image selection process latency by 3.19 times in comparison to other chat applications. The evaluation of *MilliCat* indicates that images suggestions increase the “fun” factor of the mobile chat experience. Many participants of the studies explained that chat conversations became more fun by using public images. Furthermore, several participants stated that suggested public images can help to describe something that may otherwise require a lot of words and effort. Image suggestions like used in *MilliCat* could reduce usability if the system provides the user with unintended or wrong image suggestions. Overall, the results of the studies indicate that suggestions of public images cause numerous positive effects on the usability of mobile messenger.

2.2.3 PhotoReply

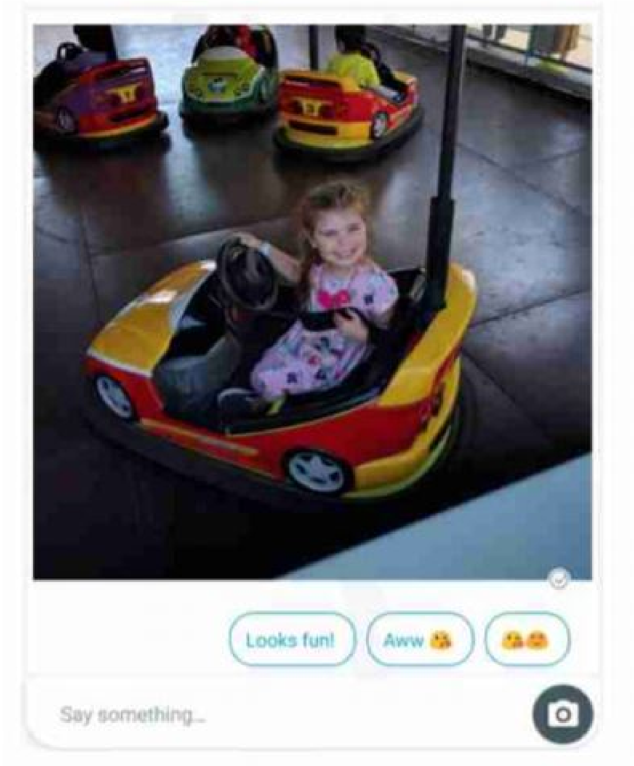


Figure 2.3: Example suggestion of PhotoReply [YFR+18]

PhotoReply is a system that helps users communicating more efficiently by providing suggestions for responses to received photos in mobile messenger[YFR+18]. When a user gets a photo from the other chat participant, *PhotoReply* suggests automatically various fitting responses under the received image (see Figure 2.3). By tapping on one of the suggested responses, the system sends the selected response as a message. *PhotoReply* was part of *Google Allo*[2], a messenger that stopped operating in 2019. This feature was the most frequently used predictive assistance feature in *Allo*.

The input of the system is an image and the output natural text. In comparison to image captioning systems, *PhotoReply's* output text does not describe the input image. *PhotoReply* suggests an output text that corresponds to a human reaction while seeing a photo in the context of a conversation. For example, if the system receives an image of a sleeping baby. The system should suggest a message like "So Cute" instead of an image description.

The creation of suggestions in *PhotoReply* consists of 4 main steps. In the first step, features of the image get extracted that create an embedding of the image. After this, the image gets categorised into concepts based on an image classification model. A response generation model provides the system with 20 responses that fit the image embedding. In the end, the system determines which responses to suggest.

An online evaluation was conducted with participants using *PhotoReply* in the *Allo* messenger. According to the study results, in 25.8% of cases, participants chose to answer with a suggested response when receiving a photo.

SmartReply was another feature in *Google Allo* [Juh16]. It provided the user reply suggestions based on the conversational context and the user's personality. Suggested replies were, for example, text, stickers, emojis, doodles or GIFs. While chatting using *Google Allo*, *SmartReply* tried to adopt the user's personality for the suggestions. In 2018, Google adopted *SmartReply* into the regular *Android Messages* app[3].

2.2.4 Sticker Suggestions in Hike Messenger

A sticker recommendation system was embedded into the *Hike* messenger [LHM+20]. This feature suggests stickers based on the conversational context (see Figure 2.4).

In messenger applications, emojis, stickers, and GIFs are used in conversations to express thoughts and emotions [LHM+20]. For example, emojis are usually used in combination with text. In comparison to emojis, stickers are graphical alternatives for messages. Stickers offered by the *Hike* messenger are cartoon characters or objects containing stylised text for the representation of common textual phrases. Finding the most suitable sticker out of thousands of available stickers can be cumbersome for the user. Therefore, the developers of the *Hike* messenger implemented a sticker recommendation system that suggests stickers based on the conversational context to counter this problem. Whenever the user receives a message from the other chat participant, the sticker recommendation system suggests the most suitable sticker to reply. This system also suggests stickers

[2] Google Allo (2019): "Allo has signed off.", available at: https://allo.google.com (accessed July 16, 2021)

[3] Chaim Gartenberg (2018): "Google is bringing Allo's Smart Reply feature to the Android Messages app", available at: https://www.theverge.com/2018/1/24/16929038/google-android-messages-smart-replies-allo-sms-ai-assistant-project-fi (accessed November 10, 2021)

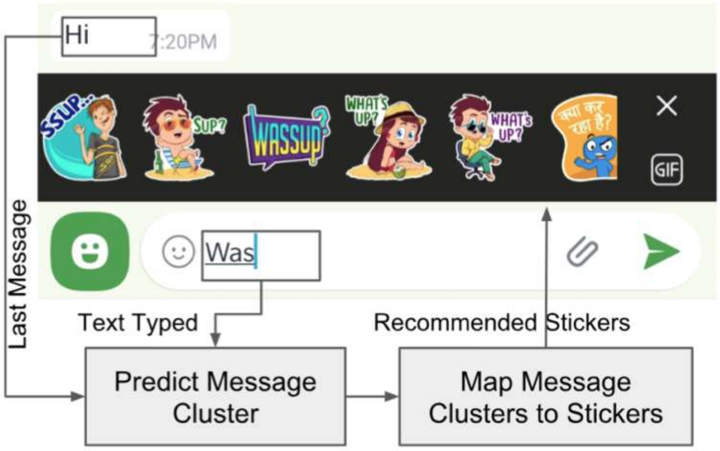


Figure 2.4: *Hike's* Sticker Recommendation User Interface and Flow Diagram [LHM+20]

depending on the user's input while typing a message. The suggestions must change each time the user presses a key. To counter delays, new sticker suggestions must be available in milliseconds. For that reason, the recommendation system runs on the user's mobile device without a network connection.

The sticker recommendation system operates in two main steps. At first, the system tries to predict what the user intends to write depending on the previous received messages and the user's input. This intent prediction underlies a language model that was trained on past chat conversations. After this, the system suggests the user a sticker that is mapped to the predicted intention.

Before the introduction of this sticker recommendation system, A/B testing was conducted with thousands of participants. Compared to older implementation, participants who used the version with the sticker recommendation system increased the use of stickers by 8%.

2.2.5 meChat

meChat is a personal assistant for sharing photos depending on the conversational context in mobile messenger applications [KRPL19]. The system searches for images saved on the user's smartphone that are relevant to the current chat conversation.

Around 4.5 billion photos are shared by users of *WhatsApp* daily. This enormous number indicates that photos are an essential communication medium in mobile messengers. Cloud-based mobile assistants like *Google Assistant* already suggest the user's private photos based on the current context. Such assistants operate on the server-side, and because of this, such services may harm the user's privacy while collecting and analysing private photos. Furthermore, image processing on the server side may cause higher latency for suggestions.

meChat's intelligence operates on the device, and because of this, it protects the user's privacy by not sending any personal content into the cloud. The system continuously interprets the conversational context and analysis images stored on the user's device for its semantic. During the analyses, the system categorises conversations and images that are stored on the smartphone. If available, GPS information embedded in images will be also considered. After the categorisation, the photos are getting ranked based on their relevance to the chat conversation.

A conducted evaluation proved that *meChat* operates with less latency than existing cloud-based text or image classification solutions.

2.2.6 Suggestions in mobile keyboards

Most mobile keyboards are equipped with a suggestion bar that provides auto-completion and correction based on the user's input [PA19]. The user can select the most appropriate word from a list of words in the suggestion bar during typing (see Figure 2.5). Auto-completion and correction suggestions improve the typing speed and accuracy.

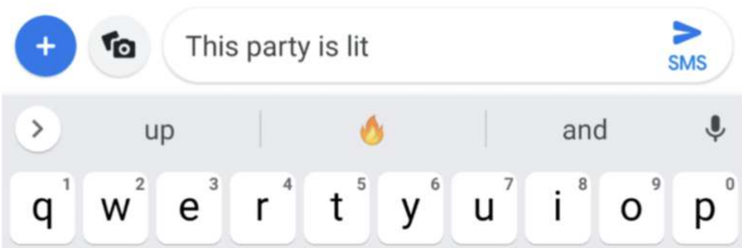


Figure 2.5: *Gboard's* emoji suggestion[4]

For example, *Google's Gboard* keyboard provides suggestions for text, emojis, GIFs or translations based on the previously typed text in the suggestion bar[4]. Furthermore, the user can start a google search directly from the keyboard and share search results with the chat participant [PA19].

2.2.7 Siri Suggestions

With the release of *iOS 9* in 2015, *Apple* adopted *Siri Suggestions* to their mobile operating system[5]. *Siri* is Apple's virtual assistant, and it suggests what the user might

[4]Swaroop Ramaswamy (2019): "Federated Learning for Emoji Prediction in a Mobile Keyboard", available at: https://arxiv.org/abs/1906.04329 (accessed December 13, 2021)

[5]Jason Holte (2018): "iOS 12: How Location Intelligence Can Help Your App Take Advantage of Siri Suggestions", available at: https://www.digmap.com/blog/location-intelligence-siri-suggestions/ (accessed July 14, 2021)

want to do next[6](see Figure 2.6). These suggestions are based on the user's device usage patterns and locations[5,7].

Figure 2.6: *Siri* Suggestion examples[6,7]

Siri Suggestions examples:

- While creating emails or calendar events, *Siri* suggests adding contacts that had been included in previous emails and events[5].
- *Siri* suggests who might be to the caller for unknown numbers based on phone numbers found in recent emails.
- If a calendar event contains a location, *Siri* notifies the user when to leave, considering the current traffic condition.
- For users having a valid boarding pass in *Apple Wallet* , *Siri* suggests viewing the flight status in the Maps application.
- While writing, *Siri* places suggestions above the keyboard containing topics the user has recently used, accessed or consumed, for example movie names, places, or general phrases.
- While typing in the search bar of Safari, *Siri* suggests websites and information related to the user's preferences.

[6] Apple (2021): "About Siri Suggestions on iPhone", available at: https://support.apple.com/en-gb/guide/iphone/iph6f94af287/ios (accessed July 14, 2021)

[7] TYRONE (2018): "IOS 12 SIRI SUGGESTIONS, SHORTCUTS AND OTHER ENHANCEMENTS", available at: https://www.iphonetricks.org/ios-12-siri-suggestions-shortcuts-and-other-enha (accessed July 14, 2021)

- While booking flights or creating appointments online, *Siri* suggests adding them to the Calendar application.

- *Siri* learns about which kind of news the user is interested in and suggests them in *Apple's News* app.

2.2.8 M Suggestions

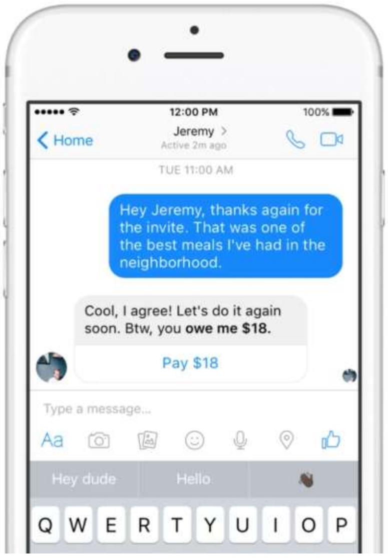


Figure 2.7: Payment Suggestion in *Facebook Messenger*[9]

In 2017, Facebook implemented context-based suggestions for text conversations in *Facebook Messenger* that pop up in the chat user interface for interactions like sending stickers, making payments, sharing locations, or making plans[8]. *Facebook Messenger* marked detected phrases in past messages like, for example, "you owe me $10" which may lead to a payment[9]. Furthermore, an action button appeared under the message that contained the detected phrase (see Figure 2.7). Clicking on the "Pay" button initiated the payment process. The suggestion feature had an intrusive interaction interface [CXDH19]. For example, if a time or date value could be detected within a message, the messenger suggested to create a new calendar entry. A too high rate of false-positive suggestions led

[8]Facebook (2017): "M Now Offers Suggestions to Make Your Messenger Experience More Useful, Seamless and Delightful", available at: https://about.fb.com/news/2017/04/m-now-offers-suggestions-to-make-your-messenger-experience-more-useful/seamless-and-delightful/ (accessed October 11, 2020)

[9]Tim Schiesser (2016): "Facebook Messenger adds polls and payment suggestions", available at: https://www.techspot.com/news/66433-facebook-messenger-adds-polls-payment-suggestions.html (accessed August 9, 2021)

to negative user reviews that described the feature as irrelevant and annoying. Facebook discontinued the suggestion feature for *Facebook Messenger* in 2018[10].

2.3 Implications for the Design and Implementation

The mobile P2P market is expected to grow in the next decade. The latest trend is the support of P2P payments in mobile messenger. The most popular messenger worldwide is *WhatsApp*, with 2 billion monthly active users in 2021[11].

A suggestion-based user interface for P2P payments in mobile messenger aims to improve usability. Suggestion-based approaches in mobile messenger have already shown significant usability improvements for the suggestion of stickers [LHM+20], images [KGH+20], auto-replies [YFR+18], and message-related tasks [CXDH19]. The underlying concept of suggestion-based user interfaces are CARS. CARS are systems that provide suggestions for items of any data considering the user's context [RD19]. Suggestions for payments based on the conversational context and the user's input have characteristics of CRS [JMCC21]. Conversations in mobile messenger consist of multi-term dialogues driven by the user and the other chat participant. CRS provide user suggestions based on the user's preferences that are identified through multi-term dialogues.

Most of the discussed existing solutions (see Section 2.2) provide the user suggestions based on the conversational context and the user's input. For that reason, suggestions in the novel suggestion-based user interface for P2P payments will also be based on the conversational context and the user's input. Suggestions in *MessageOnTap* and *MilliCat* are triggered through keywords extracted from the current chat conversation. This technique will also be used for the implementation of the novel prototype.

Facebook's discontinued suggestions feature provided the user suggestions that were based on the conversational context and not on the user's keyboard input. Compared to the novel prototype that will support only payment suggestions, *Facebook's M Suggestion* feature provided suggestions for a wide range of interactions (e.g. sending stickers, sharing locations, etc.). A high degree of false-positive suggestions caused negative user reviews. False-positive suggestions harm the user experience of RS [KGH+20]. Because of this, the keyword extraction process will be done by using advanced language analysis to avoid false-positives.

Latency is also a factor that needs to be considered during the implementation process [LHM+20]. Especially suggestions based on the user's input must appear with low latency to ensure high usability. If not, users writing with a high wpm (words per minute) may get disturbed or irritated by too late suggestions.

[10]Facebook (2017): "Facebook's Perfect, Impossible Chatbot", available at: https://www.technologyreview.com/2017/04/14/152563/facebooks-perfect-impossible-chatbot/ (accessed October 11, 2020)

[11]Statista Research Department (2021): "Most popular global mobile messenger apps as of July 2021, based on number of monthly active users", available at: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/ (accessed September 23, 2021)

CHAPTER 3

Design and Implementation of a Suggestion-Based User Interface

In this chapter, the design and implementation process of the novel suggestion-based prototype will be highlighted. At first, this chapter contains the documentation of the already existing approaches of P2P payments in mobile chat applications. After this, the design decisions made during the design process of the novel implemented prototype will be explained using *Design Space Analysis*. Furthermore, this chapter contains detailed documentation of the three different implemented prototypes of this thesis including their differences, benefits and limitations. This chapter ends with a short overview of the used technology and the technical architecture of the implemented prototypes.

3.1 Existing User Interfaces for P2P Payments in Mobile Chat Applications

Currently, popular messengers like *Facebook Messenger*, *WeChat*, *WhatsApp*, and *iMessage* have included an implementation of a P2P payment feature directly into their chat user interface. The current solutions are similar to sending multimedia content, stickers, or a location to the other chat partner.

The necessary steps to perform a payment in the user interfaces of *Facebook Messenger*, *WeChat*, *WhatsApp*, and *iMessage* have been documented according to videos, documentation, and blog entries found online because the P2P payment features of the analyzed messengers are restricted to use in only a small number of countries worldwide. For example, the P2P payment feature of *Facebook Messenger* is only available in the United States of America [MHE19]. This documentation was created to guide as a template for the re-implementation of existing solutions as additional prototypes. Furthermore, it was used to get ideas for the novel suggestion-based approach and to ensure that the novel implemented prototype is compatible with the designs of the the already existing solutions. For this documentation, only the *iOS* versions of the existing messenger were considered. The interfaces between the *iOS* and *Android* implementations differ because of the different guidelines and standards of the corresponding platforms.

In General *iMessage*, *Facebook Messenger*, *WeChat*, and *WhatsApp* have implemented a similar user interface for P2P transactions. The main difference in *iMessage* and *Facebook Messenger* is, that the interaction does not take place in a new window but rather in a window that replaces the keyboard.

Analyzing the user interfaces of these four messengers showed that users require to conduct the following four activities to perform a P2P payment:

1. **Activation of the payment process:**
   In all analysed messengers, the icon to perform a payment is not visible in the default conversation window. To get access to the payment icon the user needs to press the "+" (in *Facebook Messenger*, *WeChat*, and *WhatsApp*) or the "Application" (in *iMessage*) icon that is placed in the message composer. After this, the user gets access to the different functions, like sending multimedia content, stickers, locations, or making payments.

2. **Specification of the amount:**
   After pressing the "payment" icon in *WeChat* and *WhatsApp* a new window appears. In *iMessage* and *Facebook Messenger*, the keyboard disappears and a new view appears instead of the keyboard. In all cases, the user is asked to define the amount he wants to transfer. In *WhatsApp* and *WeChat* the user has to enter the amount of the transaction with a number keypad. In *iMessage* and *Facebook Messenger*, the user has to define the amount by increasing and decreasing the amount using the "+" and "-" button. Optionally, the user is able to swipe up a virtual keypad to change the amount of the transaction.

3. **Adding of an optional payment note:**
   In all analysed messengers the user can attach an additional payment note to the payment. In *WeChat* and *WhatsApp* the optional payment note can be set in the same window the amount is entered. In *iMessage* and *Facebook Messenger*, a message can be attached to the transaction after the definition of the amount as a separate step.

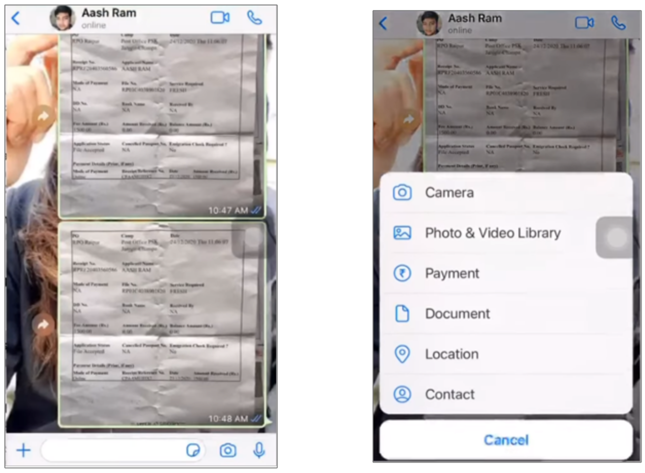
4. **Confirmation of the payment:**
   To confirm a payment, the user needs to verify the payment through the successful completion of the security process. Depending on the user's device and settings the security process can be confirmed by providing a passcode, fingerprint, or by facial recognition.

In the next subsections, the required steps to perform a payment in the analysed mobile messengers are documented.

3.1.1 WhatsApp[1]

The P2P payment feature of *WhatsApp* is currently available in India and Brazil[2].

1. Activation of the payment process (see Figure 3.1)



(a) Tap on the "+" icon

(b) Tap on the "Payment" button

Figure 3.1: Activation of the payment process in WhatsApp[1]

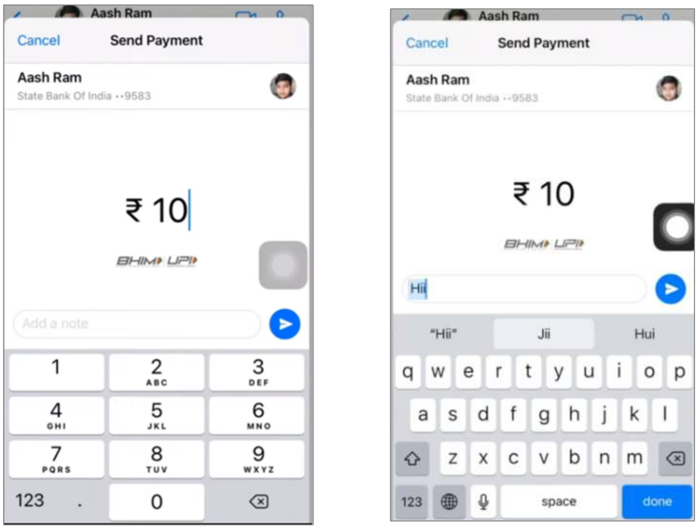
To activate the P2P payment process in *WhatsApp*, the user has to tap the "+" icon, which is located in the message composer (see Figure 3.1(a)). After this, an overlay appears that contains buttons for sending content like images, locations, payments, documents, etc. (see Figure 3.1(b)). Tapping on the "Payment" button activates the payment process, and the user leaves the conversation screen to the payment configuration screen.

2./3. Specification of the amount and optional payment note (see Figure 3.2)

To define the payment amount in *WhatsApp* the user has to type in the preferred amount using the keypad in the payment configuration screen (see Figure 3.2(a)). To attach an optional payment note, the user has to write the preferred note into the text input field located under the payment amount (see Figure 3.2(b)). Tapping on the blue icon beside the text input field for the payment note activates the payment confirmation process.

[1]techAsh (2021): "How to set up and Use WhatsApp payment feature on iPhone | WhatsApp se paise kaise transfer kare", available at: https://www.youtube.com/watch?v=m8DbrmkoSAM (accessed March 10, 2021)

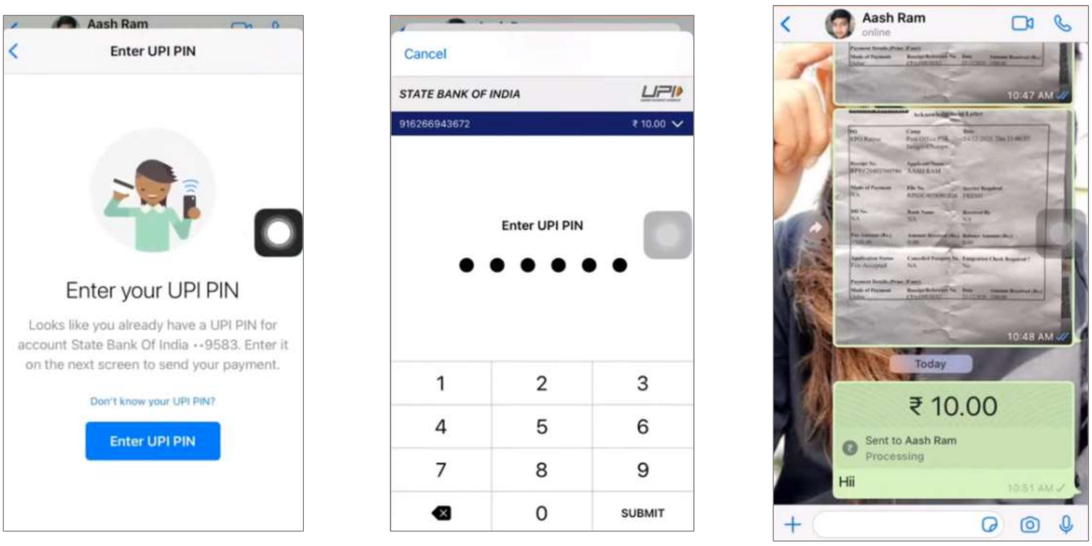
[2]WhatsApp (n.d.): "Learn more about participating countries", available at: https://faq.whatsapp.com/general/payments/learn-more-about-participating-countries/?lang=en&fbclid=IwAR2ckFTC8QZi6AyVozoNLqsSAeiHP7Y6oIDTopSLYaQIT_UQl7uG1c42lDc (accessed July 27, 2021)



(a) Specification of the amount (b) Specification of the note

Figure 3.2: Specification of the amount and optional payment note in WhatsApp[1]

4. Confirmation of the payment (see Figure 3.3)



(a) Tap on the blue button (b) Enter the pin (c) Transaction completed

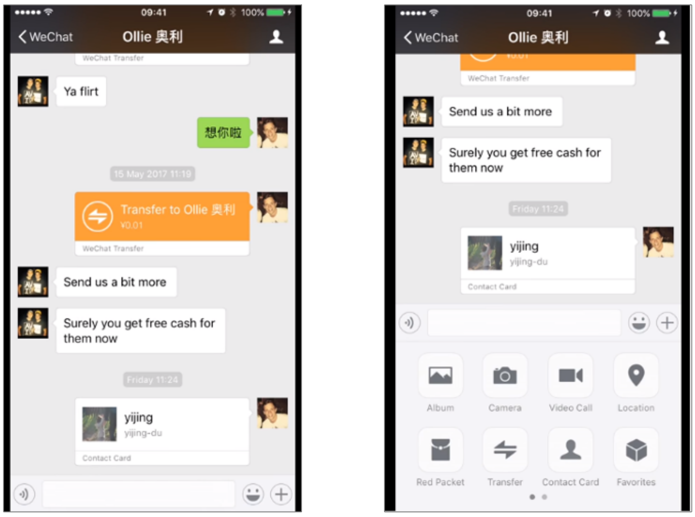
Figure 3.3: Confirmation of the payment in WhatsApp[1]

In *WhatsApp*, the multi-step confirmation process takes place in a screen that overlays the payment configuration screen (see Figure 3.3(a)). Tapping on the "Enter UPI PIN" button initiates the pin code inquiry (see Figure 3.3(b)). After entering the correct pin code, the user returns to the conversation screen, and the transaction is now visible in the chat (see Figure 3.2(c)).

3.1.2 WeChat[3]

The P2P payment feature of *WeChat* is currently available in China (Mainland), Hong Kong, South Africa and Malaysia[4].

1. Activation of the payment process (see Figure 3.4)



(a) Tap on the "+" icon (b) Tap on the "Transfer" icon

Figure 3.4: Activation of the payment process in WeChat[3]

To activate the P2P payment process in *WeChat*, the user has to tap the "+" icon, which is located in the message composer (see Figure 3.4(a)). After this, the keyboard disappears and gets replaced by a menu that contains buttons for tasks like sending an image, starting a video call, sending a location, sending money, etc. (see Figure 3.4(b)). Tapping the "Transfer" button activates the payment process, and the user leaves the conversation screen to the payment configuration screen.

2./3. Specification of the amount and optional payment note (see Figure 3.5)

To define the payment amount in *WeChat* the user has to type in the preferred amount using the keypad in the payment configuration screen (see Figure 3.5). To attach an optional payment note, the user has to tap on "Add Note" and write the preferred note

[3]Edward Lindeman (2017): "WeChat fund transfer", available at: https://www.youtube.com/watch?v=a6pRfRBqiuY (accessed October 17, 2020)

[4]WeChat (n.d.): "Why I don't have the WeChat Pay feature?", available at: https://help.wechat.com/cgi-bin/micromsg-bin/oshelpcenter?t=help_center/topic_detail&opcode=2&plat=1&lang=en&id=150821aqi3y2150821bmjfqb&Channel=helpcenter (accessed July 27, 2021)

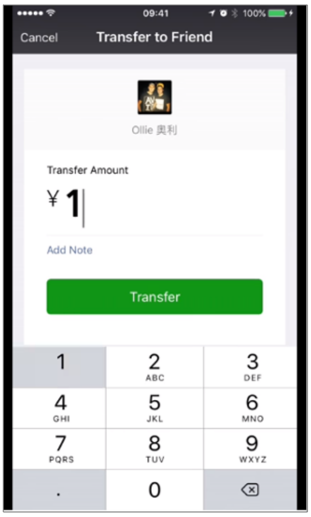
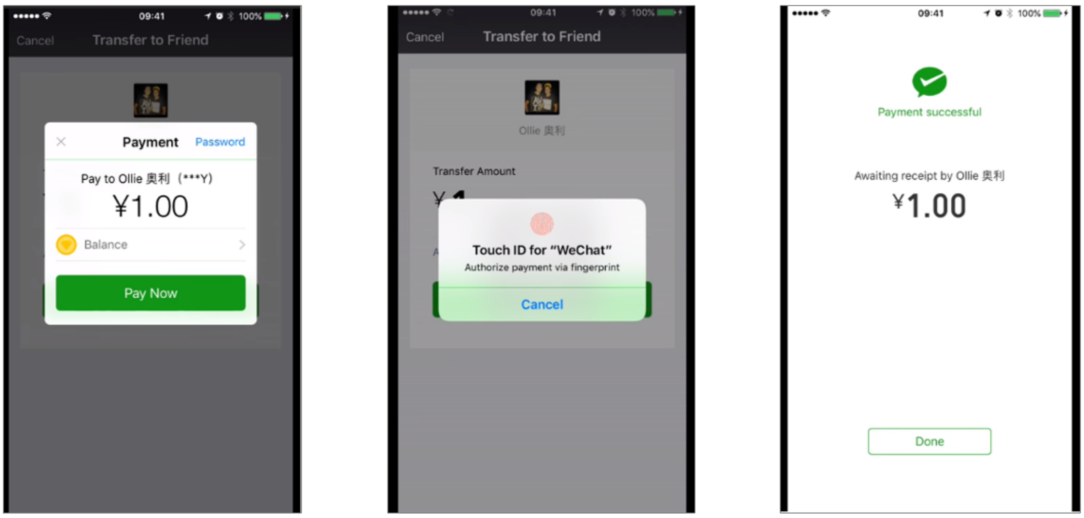


Figure 3.5: Specification of the amount and optional payment note in WeChat[3]

into a text input field. Tapping on the green "Transfer" button activates the payment confirmation process.

4. Confirmation of the payment (see Figure 3.6)



(a) Tap on the green button (b) Scan fingerprint (c) Tap on the "Done" button

Figure 3.6: Confirmation of the payment in WeChat[3]

The multi-step confirmation process in *WeChat* takes place in a screen that overlays the payment configuration screen (see Figure 3.6(a)). Tapping on the "Pay Now" button initiates a security inquiry (see Figure 3.6(b)). After the completion of the security inquiry, the transaction is completed (see Figure 3.6(c)). By tapping on the "Done" button, the user returns to the conversation screen, and the transaction is now visible in the chat.

3.1.3 Facebook Messenger[5]

The P2P payment feature of *Facebook Messenger* is only available in the US[6].

1. Activation of the payment process (see Figure 3.7)

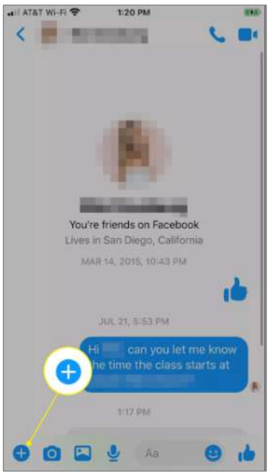


Figure 3.7: Activation of the payment process in Facebook Messenger[5]

To initiate a payment in *Facebook Messenger*, the user has to tap the "+" icon in the message composer in the conversation window (see Figure 3.7). After this, the keyboard disappears and gets replaced by the transaction amount definition view.

2./3. Specification of the amount and optional payment note (see Figure 3.8)

In the amount specification view, the user can increase and decrease the amount of the transaction by tapping on the "+" or "-" button (see Figure 3.8). By tapping on the "Edit Amount" button or by sliding up, the amount definition view extends, and a keypad appears. With the help of the keypad, the amount of the transaction can be set. After tapping the "Pay" button, the optional payment note can be set with a text input field.

[5] Elise Moreau (2020): "How to Pay Facebook Friends With Messenger", available at: https://www.lifewire.com/pay-facebook-friends-with-messenger-4146438 (accessed March 3, 2021)

[6] Facebook (n.d.): "How do I send or receive money in Messenger?", available at: https://m.facebook.com/help/messenger-app/1386234371667067/iphone-app-help/?helpref=platform_switcher&cms_platform=iphone-app (accessed July 27, 2021)

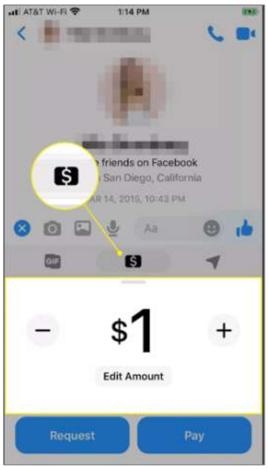
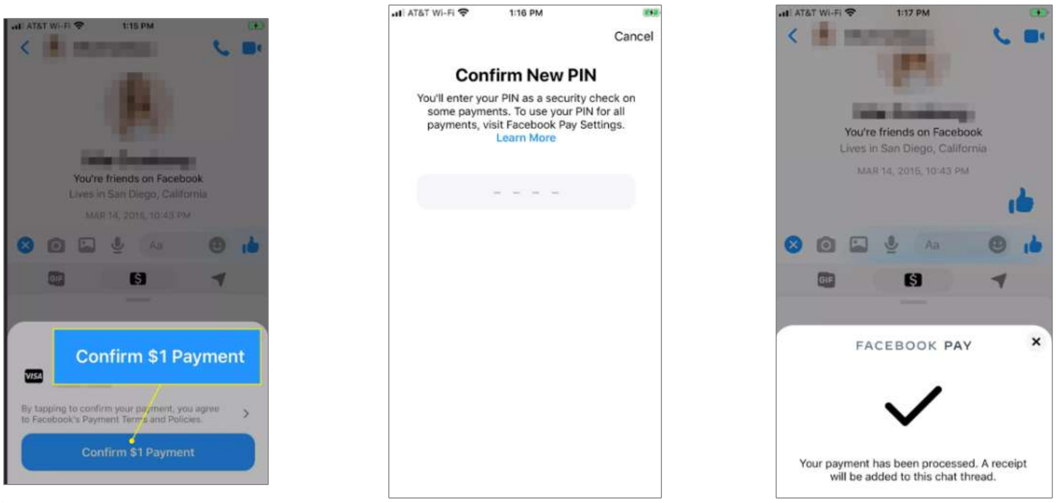


Figure 3.8: Specification of the amount and payment note in Facebook Messenger[5]

4. Confirmation of the payment (see Figure 3.9)



(a) Tap on the blue button

(b) Enter the pin

(c) Tap on the “x“ button

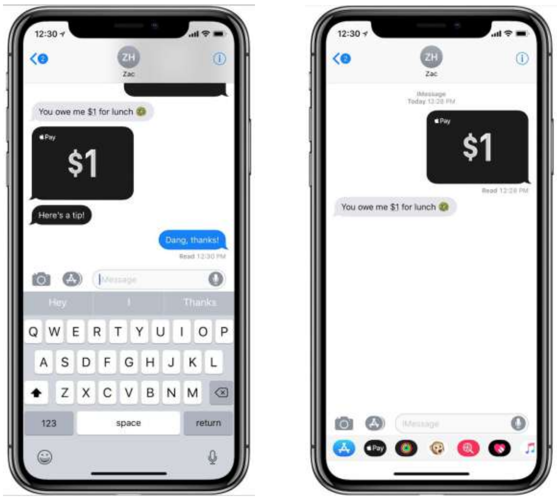
Figure 3.9: Confirmation of the payment in Facebook Messenger[5]

In *Facebook Messenger*, the multi-step confirmation process takes place in a view that overlays the payment configuration screen (see Figure 3.9(a)). Tapping on the “Confirm $1 Payment“ button initiates the pin code inquiry (see Figure 3.9(b)). After entering the correct pin code, the transaction is completed (see Figure 3.9(c)). By tapping on the “x“ button, the user returns to the conversation screen, and the transaction is now visible in the chat.

3.1.4 iMesssage[7]

The P2P payment feature of *iMessage* is only available in the US[8].

1. Activation of the payment process (see Figure 3.10)



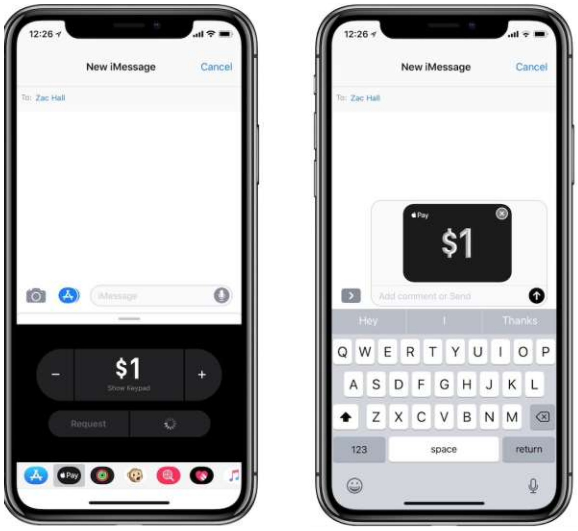
(a) Tap on the "App" icon (b) Tap on the "Pay" icon

Figure 3.10: Activation of the payment process in iMessage[7]

To initiate a payment in *iMessage* the user has to tap the "App" icon located in the message composer (see Figure 3.10(a)). After this, the predictive text bar of the keyboard disappears, the message composer gets extended under the message text input field, and a icon bar appears (see Figure 3.10(b)). By tapping on the "Apple Pay" icon, the keyboard disappears and gets replaced by the transaction amount definition view.

[7] Michael Potuck (2017): "How to Pay Facebook Friends With Messenger", available at: HowtosetupanduseApplePayCash (accessed July 19, 2021)
[8] Apple (n.d.): "Set up Apple Cash", available at: https://support.apple.com/en-us/HT207886?fbclid=IwAR0BBfPvw60pDM0HVNQTRK2hv8PPXS8Lsd5kyx4bGjvYOM9EBZ-WUoDqt_k (accessed July 27, 2021)

2./3. Specification of the amount and optional payment note (see Figure 3.11)

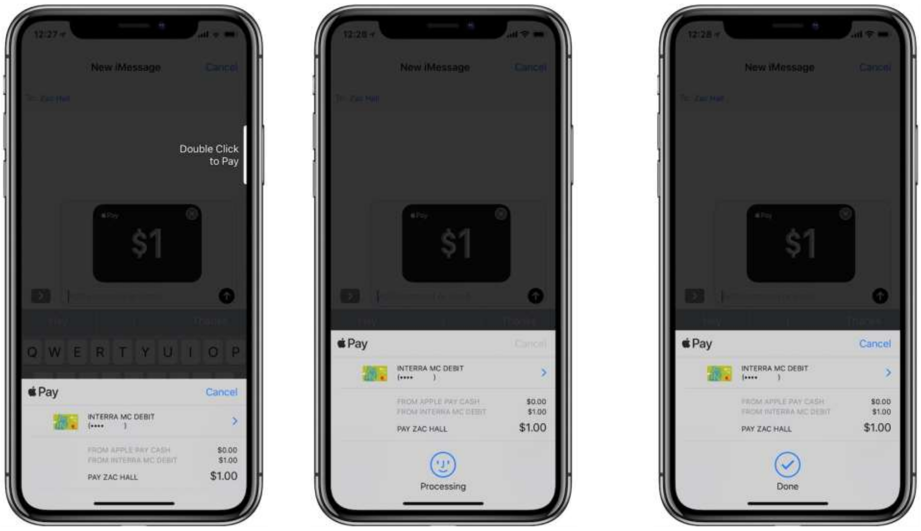


(a) Specify the amount (b) Attach a payment note

Figure 3.11: Specification of the amount and optional payment note in iMessage[7]

In the amount specification view, the user can increase and decrease the amount of the transaction by tapping on the "+" or "-" button (see Figure 3.11(a)). By tapping on "Show Keyboard" or by sliding up, the amount definition view gets extended, and a keypad appears. With the help of the keypad, the amount of the transaction can be set. After tapping the "Pay" button, the optional payment note can be set with the main text input field in the payment preview view (see Figure 3.11(b)). Tapping on the black icon in the message composer initiates the payment configuration process.

4. Confirmation of the payment (see Figure 3.12)
The payment confirmation process in *iMessage* takes place in a view that overlays the payment configuration screen (see Figure3.12(a)). After the completion of the security inquiry (see Figure3.10(b)), the transaction is completed (see Figure 3.12(c)). By tapping on the "Cancel" button, the user returns to the conversation screen, and the transaction is now visible in the chat.



(a) Activate Face ID (b) Proceed Face ID (c) Transaction completed

Figure 3.12: Confirmation of the payment in iMessage[7]

3.2 Design Approach

At the beginning of the design process, the user interfaces of the already existing messengers that support P2P payments have been analysed and documented (see Section 3.1). After this, three different prototypes have been designed as high-fidelity mockups in *Adobe XD*[9]: A prototype that represents *WhatsApp* and *WeChat*, another protoype that represents *iMessage* and *Facebook Messenger*, and the novel suggestion-based prototype. The approach of *WhatsApp* and *WeChat* were represented as one prototype because of their identical interaction flow. For the same reason, *iMessage* and *Facebook Messenger* were combined into one prototype.

For the design of the novel suggestion-based user interfaces for P2P payment in mobile chat applications, a design space analysis was conducted. With the help of this method, the most important design decisions were made to provide a comprehensible design outcome.

After this, the designed mockups were implemented as a mobile application using *Flutter*.

3.2.1 Design Space Analysis

Design space analysis is an argumentation-based approach to design artefacts [MBS93]. It helps software designers to think about how to design and it produces a comprehensible outcome. The outcome of the design space analysis is a design space instead of an artefact. It contains the decisions and reasons that influenced the design process. The semi-formal

[9]Adobe (n.d): "Design like you always imagined.", available at: https://www.adobe.com/products/xd.html (accessed August 24, 2021)

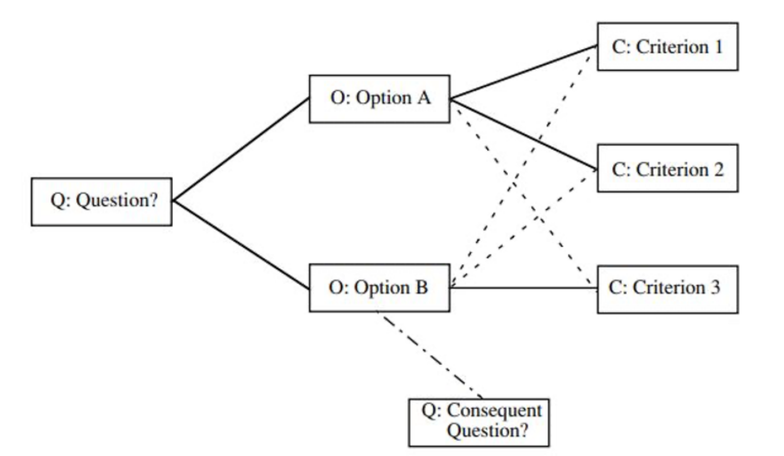


Figure 3.13: Components of a Design Space Tree [MBS93]

notation Question Option Criteria (QOC) is used to represent the design space of the design outcome of an artefact. The design space contains all covered design options and the reasons for a design outcome.

Questions are the main components of a design space and represent the issues of a design process. *Options* are the different possible answers to *Questions*. *Criteria* are arguments that support or deny an *Option*. A design space can be represented as a tree (See Figure 3.13). A solid line between an *Option* and a *Criteria* highlights that a *Criteria* fits for an *Option* and a dotted line that a *Criteria* does not fit for an *Option*. This representation is beneficial for the communication between team members and people with different backgrounds. It is also useful to understand the reasons for design issues.

3.2.2 How to produce a Design Space

The process of design space analysis consists of following five phases [MBS93]:

- **Phase 1: Identify relevant information**
  In this phase information about the main issues and relevant *Questions*, *Options* and *Criteria* must be gathered.

- **Phase 2: Structure material into a rough *QOC***
  In this phase, *Questions* must be defined through finding new design ideas and information about *Options* and *Criteria* must be structured

- **Phase 3: Flesh out a design space**
  In this phase, new ideas for a design space must be found. Furthermore, the *Options* and *Criteria* must be defined.

- **Phase 4: Reformulate the design space components**
  In this phase *Questions*, *Options* and *Criteria* must be reformulated to improve decomposition.

- **Phase 5: Make design decisions**
  In this phase, the *Options* must be evaluated using the chosen *Criteria*. After this, the *Option* that is supported by the most *Criteria* will be chosen for the design.

The 5 phases of producing a design space are represented as a sequential model but the real design process is not carried out in a sequential manner. Because of this, this order does not have to be interpreted as a strict sequential model.

3.2.3 Considered Design Questions

The following eight design questions were formulated during the design process of the novel suggestion-based user interface using design space analysis.

Question 1 (Q1): Should a payment suggestion automatically pre-define a detected amount for the transaction?

Option 1 (O1): Yes

Option 2 (O2): No

Criterion 1 (C1): Requires fewer steps to perform a payment

Criterion 2 (C2): Does not confuse the user

Criterion 3 (C3): Is compatible with the interaction flow of current messengers

Discussion:

Design Question 1 is defined to determine if a suggestion should automatically pre-define a detected amount for a transaction.

The first considered *Option (O1)* is that a suggestion should automatically pre-define a detected amount for a transaction. An already pre-defined amount would implicate fewer steps for the user to perform a payment (C1) because the user does not need to set the payment amount manually. Furthermore, this approach is compatible with the interaction flow of current messengers (C3). Messengers currently do not support payments suggestions, but *iMessage* underlines detected amounts in the chat conversation. In *iMessage*, the view to set up a transaction appears with the detected amount already set for the payment by clicking on a detected underlined amount. Automatically defined amounts could confuse or annoy users if the system pre-defines unintended payment amounts (C2).

Another considered *Option (O2)* is that the system does not automatically pre-define detected amounts for a payment. This solution would require more steps to perform a payment (C1) because the user must define the transaction amount manually. Because the user sets the amount manually, *Option 2* is resistant against false-positive detected amounts that could confuse the user (C2). As *iMessage's* amount detection demonstrates, this solution would not be compatible with the interaction flow of current messengers (C3). *iMessage* automatically defines detected payment amounts.

	Criterion 1	Criterion 2	Criterion 3
Option 1	+	-	+
Option 2	-	+	-

Table 3.1: Design Question 1 Evaluation Matrix

The evaluation of these two *Options* with the defined *Criteria* benefits *Option 1* (predefine a detected amount for a transaction) for the design of the novel prototype (see Table 3.1).

Question 2 (Q2): How payment suggestions should get activated?

Option 1 (O1): By the user

Option 2 (O2): By the system

Criterion 1 (C1): Requires fewer steps to perform a payment

Criterion 2 (C2): Resistant to false-positive suggestions

Criterion 3 (C3): Makes the user aware of possible payments

Discussion:

During the design process, how and who activates payment suggestions in the novel prototype was a key question (Q1). The first defined *Option (O1)* is that the user has to activate payment suggestions manually in the user interface. In this case, whenever the user activates the payment process, the system searches automatically in the conversation for payment intents. If a payment intent gets detected, the system automatically suggests the user to perform a payment with a detected payment amount already set. *Option 1* is more resistant to false-positive suggestions (C3) because the user receives only payment suggestions if he wants to. However, *Option 1* requires more steps to perform a payment (C1) and does not make the user aware of possible payments (C3).

The other defined *Option (O2)* is that the system continuously scans the conversation for payment intents and automatically provides the user with payment suggestions. This *Option (O2)* is less resistant to false positives (C2) because the user could get suggestions for payments he does not want to perform. *Option 2* requires fewer steps to perform a payment (C1) because the user does not need to activate the payment process. Furthermore, automatic suggestions by the system could make the user aware of possible payment intents in the chat conversation or let the user explore the payment feature in the messenger (C3).

	Criterion 1	Criterion 2	Criterion 3
Option 1	-	+	-
Option 2	+	-	+

Table 3.2: Design Question 2 Evaluation Matrix

The evaluation of the three defined *Criteria* led to the decision to use automatic suggestions for detected payment intents by the system (O2) (see Table 3.2). This design choice implies that the system must be resistant to false-positive suggestions.

Question 3 (Q3): How to present a payment suggestion?

Option 1 (O1): As a pop-up

Option 2 (O2): As a dialog that appears in the conversation view

Option 3 (O3): As an extension of the message composer

Criterion 1 (C1): Does not interfere with the conversational context

Criterion 2 (C2): Does not annoy the user

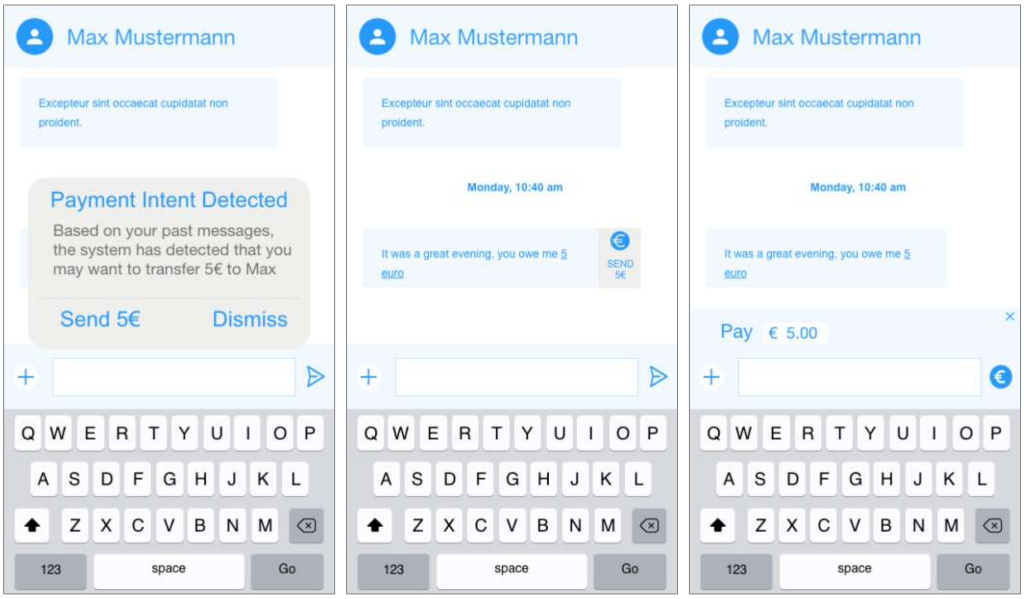
Criterion 3 (C3): Is compatible with the design of current messengers

Criterion 4 (C4): Requires fewer steps to perform a payment

Discussion:

The third design question that appeared during the design process is how to represent a payment suggestion. The first defined *Option (O1)* is to represent a payment suggestion as a pop-up that is shown when a payment intent gets detected (see Figure 3.14(a)). *Option 1* may interfere with the conversational context (C1) and could annoy the user while chatting (C2). Furthermore, this *Option (O1)* is not compatible with the design conventions of messengers (C3) because the analysed messenger user interfaces do not contain pop-ups for the P2P payment feature. *Criterion 1* supports *Option 1* because it could require fewer steps to perform a payment if the payment process starts with the appearance of the pop-up.

The second *Option (O2)* defined to this *Design Question* is that a suggestion could be represented as a dialog view that appears in the chat conversation next to the message that contains the detected payment intent (see Figure 3.14(b)). Clicking on the dialog view next to the detected payment intent would activate the payment process. *Option 1* does not interfere with the conversational context (C1) because the user can ignore the dialog if he does not intent to conduct a payment. Such a dialog may annoy the user during a chat conversation (C2). Furthermore, this *Option* is not compatible with



(a) Option 1 (O1) (b) Option 2 (O2) (c) Option 3 (O3)

Figure 3.14: Question 3 (Q3) Mockups

the design conventions of messengers (C3) because the user interfaces of the analysed messengers do not contain dialogues or interaction menus in a chat conversation view. Additionally, this *Option* requires an extra step by tapping on the dialog view to start the payment process (C4).

The third *Option (O3)* was to represent a payment suggestion as an extension of the message composer (see Figure 3.14(c)). *Option 3* would not interfere with the conversational context (C1) because the user can easily ignore a payment suggestion if he does not intend to send money. On the other hand, an extension of the message composer could annoy the user while receiving false-positive suggestions (C2). *Option 3* would be compatible with the design conventions of mobile messengers (C3). The analysed user interfaces of messengers like *WhatsApp* or *iMessage* have included a lot of functionality into the message composer like sending images, GIFs, locations, etc. This *Design Option (O3)* also can reduce the number of necessary steps to perform a payment (C4) if the user can set up the payment in the message composer.

	Criterion 1	Criterion 2	Criterion 3	Criterion 4
Option 1	-	-	-	+
Option 2	+	-	-	-
Option 3	+	-	+	+

Table 3.3: Design Question 3 Evaluation Matrix

Three of the four defined *Criteria* support the *Option 3*, and because of this, the payment suggestions were placed into the message composer (see Table 3.3).

Question 4 (Q4): How to mark detected keywords that cause a payment suggestions?

Option 1 (O1): Detected keywords get underlined

Option 2 (O2): The colour of detected keywords changes

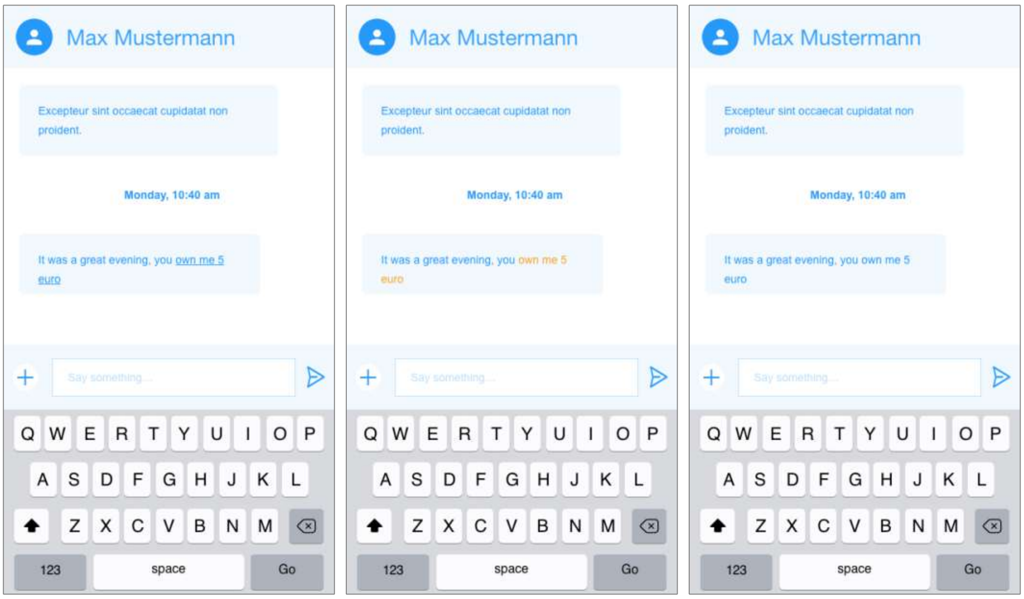
Option 3 (O3): Detected keywords do not get marked

Criterion 1 (C1): Enables replicability of suggestions

Criterion 2 (C2): Is compatible with the design of current messengers

Criterion 3 (C3): Does not confuse users

Discussion:



(a) Option 1 (O1) (b) Option 2 (O2) (c) Option 3 (O3)

Figure 3.15: Question 4 (Q4) Mockups

Another question that occurred during the design process was about how to mark detected keywords that cause a payment suggestion. The first *Option (O1)* is to underline detected keywords. This method is already used by messengers for detected locations, dates, or phone numbers (see Figure 3.15(a)). Option 1 would enable replicability of payment suggestions (C1). Furthermore, it would be compatible with the design of current messengers (C2), because as already mentioned, messengers like *iMessage* or

WhatsApp already make use of it. This *Option* would also not confuse the user (C3) because it is already a common way to represent hyperlinks.

Another *Option (O2)* that came up is to change the colour of detected keywords (see Figure 3.15(b)). This *Option (O2)* would also help to provide replicability for payment suggestions (C1). But changing the colour of detected keywords would not be compatible with the designs of current messengers (C2), because none of the analysed messengers changes the colours of keywords in the conversation view. Because of this, this design approach could confuse user during a conversation (C3).

The third considered *Design Option (O3)* was not to mark keywords of payment intents (see Figure 3.15(c)). *Option 3* would not provide replicability of suggestions to the user (C1) and would be incompatible with the designs of mobile messengers (C2). Additionally, getting a payment suggestion without knowing why they occurred could confuse the user (C3).

	Criterion 1	Criterion 2	Criterion 3
Option 1	+	+	+
Option 2	+	-	-
Option 3	-	-	-

Table 3.4: Design Question 4 Evaluation Matrix

The evaluation of the *Criteria* caused that *Option 1* (underline keywords that cause payment suggestions) was used for the design of the novel prototype (see Table 3.4).

Question 5 (Q5): How to confirm a payment?

Option 1 (O1): A payment confirmation is not necessary

Option 2 (O2): In an external view

Option 3 (O3): In a view that overlays the chat conversation

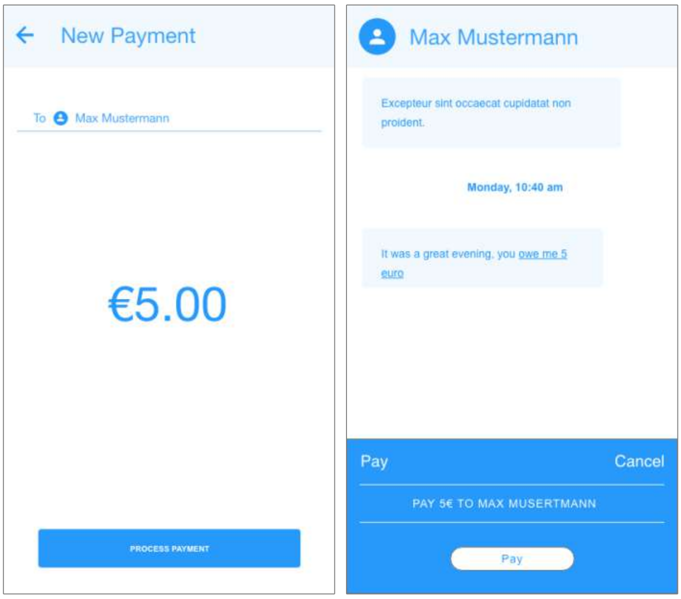
Criterion 1 (C1): Requires fewer steps to perform a payment

Criterion 2 (C2): Does not cause unintended payments

Criterion 3 (C3): Is compatible with the interaction model of current messengers

Discussion:

The first *Option (O1)* that was considered for Questions 5 is that the user does not need to confirm a payment. This *Option (O1)* has the benefit that it requires fewer steps to perform a payment (C1). Besides, having a passcode inquiry but not having a payment confirmation may cause unintended payments (C2). Furthermore, this *Option (O3)* does not follow the interaction model of the analysed messengers (C3) because all analysed messengers provide a payment confirmation before the password inquiry.



(a) Option 2 (O2) (b) Option 3 (O3)

Figure 3.16: Question 5 (Q5) Mockups

Another considered *Option (O2)* is to provide the user a payment confirmation view in an external view (see Figure 3.16(a)). *WhatsApp* and *WeChat* navigate the user to an external view to confirm the payment. Because of this, this *Option (O2)* would follow the same interaction model as other messengers (C3). Furthermore, confirming the payment in an external view would prevent unintended payments (C2). But this solution requires more steps to perform a payment (C1).

The third *Option (O3)* is to display a payment confirmation view that overlays the chat conversation (see Figure 3.16(b)). *Option 3* may be less vulnerable for unintended payments (C2), and it follows a similar interaction model like *iMessage* (C3). But this *Option* requires more steps to perform a payment (C1). However, this approach does not interfere with the conversational context.

	Criterion 1	Criterion 2	Criterion 3
Option 1	+	-	-
Option 2	-	+	+
Option 3	-	+	+

Table 3.5: Design Question 5 Evaluation Matrix

Two of the three *Design Criteria* support *Option 2* and *Option 3*. To reduce the number of necessary steps to perform a payment was one of the may goal of the novel prototype and because of this, the payment confirmation was placed in a view that overlays the conversational screen (O3) (see Table 3.5).

Question 6 (Q6): Where to change the payment method?

Option 1 (O1): In the payment suggestion view

Option 2 (O2): In the payment confirmation view

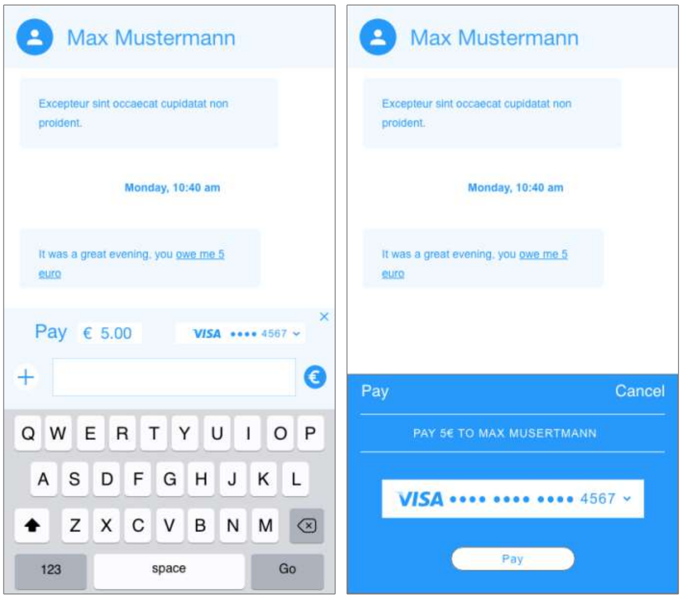
Option 3 (O3): The user cannot change the payment method

Criteria 1 (C1): Requires fewer steps to perform a payment

Criteria 2 (C2): Is compatible with the functionality of current messengers

Criteria 3 (C3): Does not overload the user interface

Discussion:



(a) Option 1 (O1) (b) Option 2 (O2)

Figure 3.17: Question 6 (Q6) Mockups

Design Question 6 focuses on where to change the payment method used for the payment. The analysis of the already existing messengers that support P2P payments showed that all of these messengers provide the user the possibility to change the payment method during the transaction process. The first *Option (O1)* that occurred was that the user can change the payment method in the message composer (see Figure 3.17(a)). *Option 1* would require fewer steps to perform a payment (C1) instead of placing it into an external view. Furthermore, this solution is compatible with the design of current messengers (C2). The only defined *Criteria* that does not support *Option 1* is that this solution may overload the message composer (C3).

Option 2 is to place the selection of the payment method into the payment confirmation view (see Figure 3.17(b)). This *Option (O2)* may require more steps to perform a payment (C1), but it is also compatible with the design of current messengers (C2). *Option 2* has the the benefit that it does not overload the user interface.

Option 3 is that the user cannot change the payment method during the payment process. This *Option (O3)* would reduce the steps necessary to perform a payment (C1) and it does not overload the user interface (C3). However, *Option 3* would be incompatible with the functionality of existing messengers (C2). All of the analysed messengers let users changing the payment method during the payment process.

	Criterion 1	Criterion 2	Criterion 3
Option 1	+	+	-
Option 2	-	+	+
Option 3	+	-	+

Table 3.6: Design Question 6 Evaluation Matrix

All three defined *Design Options* get supported by two of the three defined *Design Criteria*. In the end, *Option 1* was chosen for the design of the prototype with the implication to avoid an overloaded user interface (see Table 3.6).

Question 7 (Q7): How to add an optional payment note?

Option 1 (O1): With a second text input field in the extended message composer

Option 2 (O2): With the main text input field of the message composer

Option 3 (O2): With a text input field that appears in the payment confirmation view

Criterion 1 (C1): Does not overload the message composer

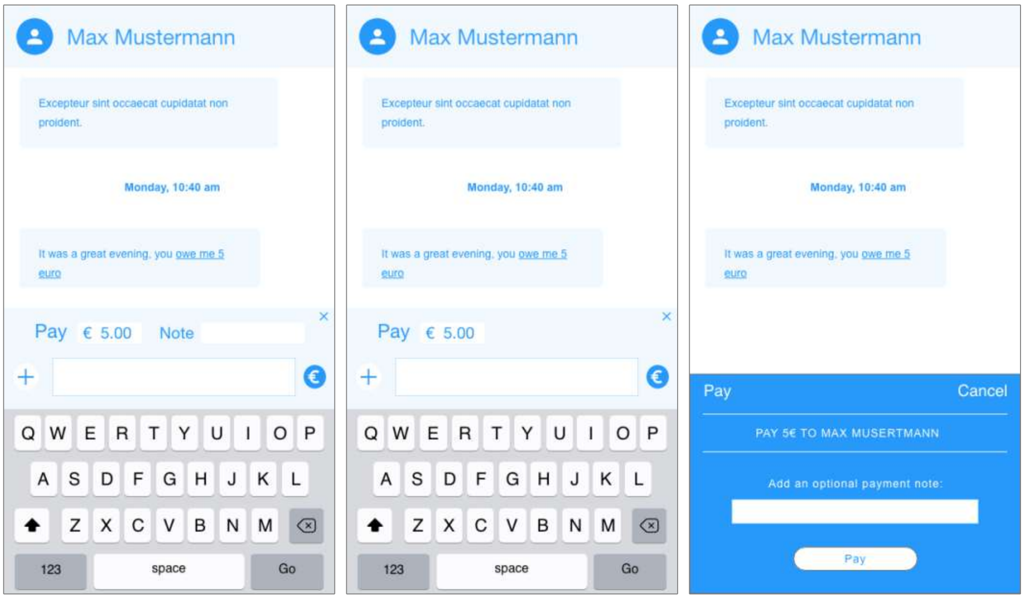
Criterion 2 (C2): Promotes users to add a payment note

Criterion 3 (C3): Does not interfere with the conversational context

Discussion:

Question 7 focuses on how the user can add an optional note to a payment. The first *Option (O1)* is through a second text input field in the message composer (see Figure 3.18(a)). *Option 1* could overload the message composer (C1). However, this *Option (O1)* may encourage the user to add a payment note (C2) because the text input field would be placed next to the input field for the amount. Furthermore, this *Option* would not interfere with the conversational context (C3).

The second considered *Option (O2)* is that the user can set the additional payment note with the main text input field in the message composer that is used to compose text messages (see Figure 3.18(b)). *Option 2* would not overload the message composer (C1) and could motivate the users to add a payment note (C2). Furthermore, this solution does not interfere with the conversational context (C3).



(a) Option 1 (O1) (b) Option 2 (O2) (c) Option 3 (O3)

Figure 3.18: Question 7 (Q7) Mockups

Another *Design Option (O3)* is to place a text input field in the payment confirmation view (see Figure 3.18(c)). *Option 3* would not cause an overload of the message composer (C1). But it may not encourage the user to add a payment note (C2) because it could be easily skipped during the payment process (C2). Furthermore, this *Option (03)* would interfere with the conversational context (C3) because composing the payment note in another view may disturb the conversational flow.

	Criterion 1	Criterion 2	Criterion 3
Option 1	-	+	+
Option 2	+	+	+
Option 3	+	-	-

Table 3.7: Design Question 7 Evaluation Matrix

All defined *Design Criteria* supported *Option 2*, and because of this, this *Option 2* (to add the optional payment note into the main text input field of the message composer) was chosen for the design of the novel prototype (see Table 3.7).

Question 8 (Q8): How to define the amount of a transaction?

Option 1 (O1): With a text input field

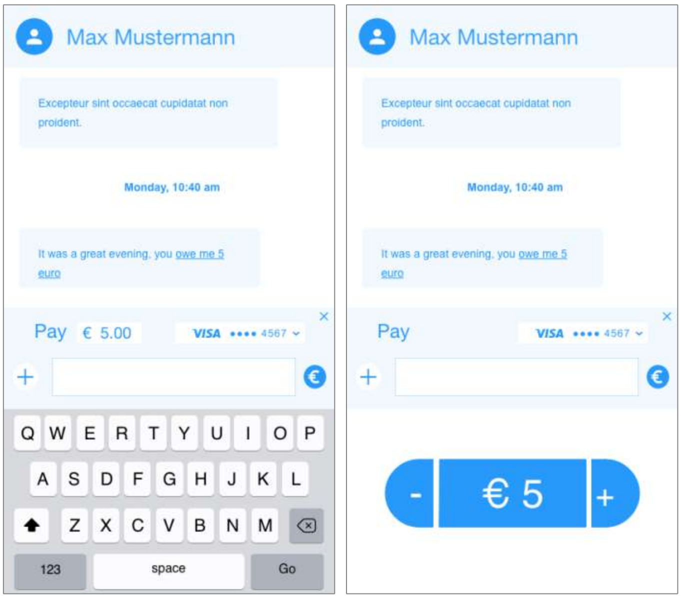
Option 2 (O2): With "+", "-" buttons

Criterion 1 (C1): Requires fewer steps to perform a payment

Criterion 2 (C2): Does not confuse the user

Criterion 3 (C3): Is compatible with comma amounts

Discussion:



(a) Option 1 (O1) (b) Option 2 (O2)

Figure 3.19: Question 8 (Q8) Mockups

The analyzed messengers that currently support P2P payments have two different approaches to define the payment amount. In *WhatsApp* and *WeChat* users define the payment amount with a text input field and in *iMessage* and *Facebook Messenger* with "+", "-" buttons.

Question 8 was defined to find out which of these two approaches should be used for the novel prototype.

Option 1 is that the user can define the amount with a text input field (see Figure 3.19(a)). This *Option (O1)* requires fewer steps to define the desired amount (C1). Furthermore, *Option 1* would not confuse the user (C2) because a text input field is a well-known input method in mobile applications. Additional, *Option 1* is compatible with comma amounts (C3).

Option 2 is to use a "+" and a "-" button to define the amount for a transaction (see Figure 3.19(b)). This input method requires more steps to define higher amounts (C1). For example, to define 10 euros, the user has to press nine times the "+" button. Furthermore, this input method is not compatible with comma amounts (C3) and may confuse the user (C2).

	Criterion 1	Criterion 2	Criterion 3
Option 1	+	+	+
Option 2	-	-	-

Table 3.8: Design Question 8 Evaluation Matrix

Evaluating the *Options* with the defined *Criteria* lead to use text input field to define the amount of a transaction (O2) for the novel prototype (see Table 3.8).

3.2.4 Summary of the Design Decisions

Design Question:	**Chosen Design Option**
Q1: Should a payment suggestion automatically pre-define a detected amount for the transaction?	O1: Yes
Q2: How payment suggestions should get activated?	O2: By the system
Q3: How to present a payment suggestion?	O3: As an extension of the message composer
Q4: How to mark detected keywords that cause a payment suggestions?	O1: Detected keywords get underlined
Q5: How to confirm a payment?	O3: In a view that overlays the chat conversation
Q6: Where to change the payment method?	O1: In the payment suggestion view
Q7: How to add an optional payment note?	O2: With the main text input field of the message composer
Q8: How to define the amount of a transaction?	O1: With a text input field

Table 3.9: Design Questions and chosen Design Options

In this subsection, the results of the conducted design space analysis will be summarised (see Table 3.9) and compared to the reimplemented prototypes of already existing solutions (see Table 3.10).

In the novel suggestion-based user interface, detected payment amounts found in the conversational context will be automatically pre-defined to reduce the number of necessary steps to perform a P2P payment. In the reimplemented user interfaces of *WhatsApp/WeChat* and *iMessage/Facebook Messenger*, payment amounts found in the conversational context are not automatically pre-defined.

Furthermore, the conducted design space analysis revealed that the novel user interface will activate payment suggestions automatically. In comparison, the user interfaces of *WhatsApp/WeChat* and *iMessage/Facebook Messenger* do not automatically activate the payment process. In these user interfaces, the user has to manually activate the payment process.

	Suggestion-Based User Interface	**User Interface of WhatsApp/WeChat**	**User Interface of iMessage/Facebook Messenger**
Who defines the payment amount?	the system automatically	the user manually	the user manually
Who activates the payment process?	the system automatically	the user manually	the user manually
Where to define the payment amount?	in the message composer	in an external view	in a view that replaces the keyboard
Where to change the payment method?	in the message composer	in an external view	in the payment confirmation view
Where to define a payment note?	in the message composer	in an external view	in the payment preview view
Detects payment intents?	yes	no	partly (underlines payment amounts)
How to confirm a payment?	in a view that overlays the chat	in an external view	in view that overlays the chat
How to define the payment amount?	with a text input field	with a text input field	with “+“,“-“ buttons

Table 3.10: Comparison between the novel user interface and the existing solutions

In the novel suggestion-based user interface, payment suggestions will be placed into the extension of the message composer. The user interfaces of *WhatsApp/WeChat* and *iMessage/Facebook Messenger* do not provide payment suggestions. In the novel user interface, in the extended message composer, the user will be able to change the payment method and define the payment amount with a text input field. In the reimplemented user interface of *WhatsApp/WeChat*, the user is able to define the payment amount and change the payment method in an external view. In the reimplemented user interface of *iMessage/Facebook Messenger*, the user is able to change the payment amount in a view that replaces the keyboard and change the payment method in the payment confirmation view.

The main text input field of the message composer will be used to attach an optional payment note to a transaction in the suggestion-based user interface. In the reimplemented user interface of *WhatsApp/WeChat*, the user is able to attach an optional payment note in the external view where to change the payment amount. In the reimplementation of *iMessage/Facebook Messenger*, the user is able to attach an optional payment note in the payment preview view with the main text input field.

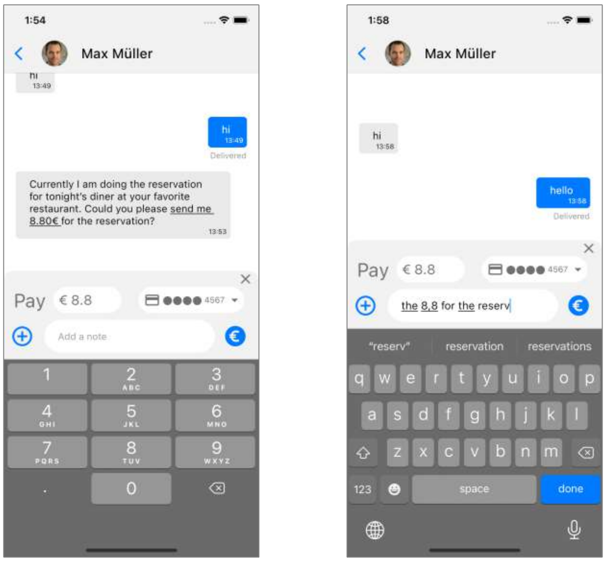
To ensure replicability in the suggestions-based user interface, keywords that cause payment suggestions will get underlined in the chat conversation. The reimplemented user interfaces of *WhatsApp/WeChat* and *iMessage/Facebook Messenger* do not underline keywords for payment suggestions. However, the reimplemented user interface of *iMessage/Facebook Messenger* underlines payment amounts found in past chat messages.

Additionally, in the suggestion-based user interface, the user has to confirm payments in a view that overlays the chat conversation to prevent unintended payments. In the reimplemented user interface of *WhatsApp/WeChat* the payment confirmation takes place in the external view where to define the payment amount. As in the suggestion-based user interface, the payment confirmation in the reimplemented user interface of *iMessage/Facebook Messenger* takes place in a view that overlays the chat conversation.

3.3 Documentation of the implemented Prototypes

For this thesis, three different prototypes were developed. One is the implemented novel chat user interface that provides users in-line payment suggestions. The other two prototypes are reimplemented user interfaces of existing mobile chat applications that support P2P payments. These two additional prototypes were implemented as baseline conditions to compare them with the novel implemented prototype during the usability study.

3.3.1 “Suggestion” Prototype



(a) based on conversation (b) based on user input

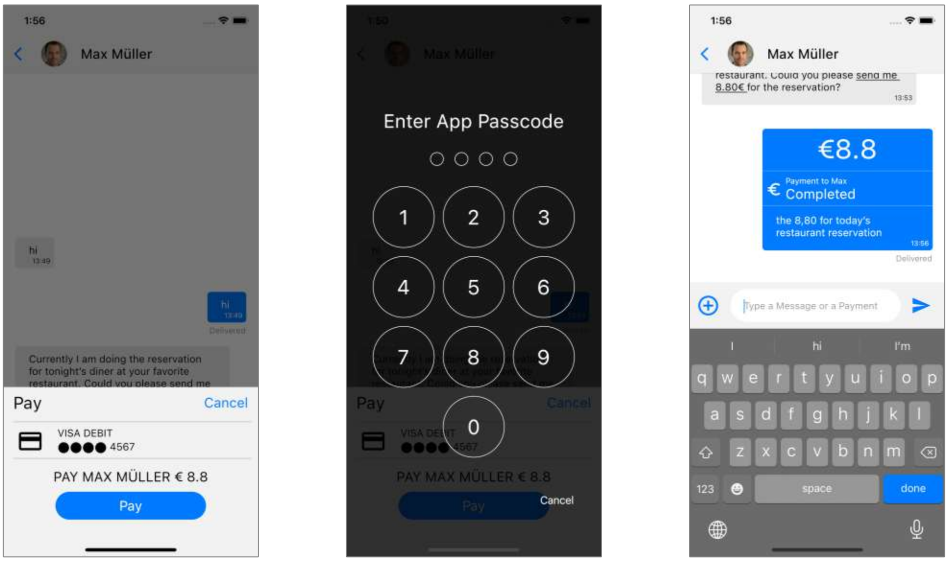
Figure 3.20: Payment suggestions in the “Suggestion“ prototype

The “Suggestion” solution is the novel implemented user interface to allow faster transactions by the reduction of necessary steps to perform a P2P payment and by not interfering with the conversational flow. This user interface provides users with inline payment suggestions based on the conversation context and the user’s input.

This prototype provides the user with payment suggestions based on detected payment intents in the past messages (see Figure 3.20(a)). If a payment intent gets detected, the keywords that caused a payment suggestion get underlined in the message. At the same time, the payment configuration view appears and suggests the user to perform a payment with the detected amount automatically set for the payment.

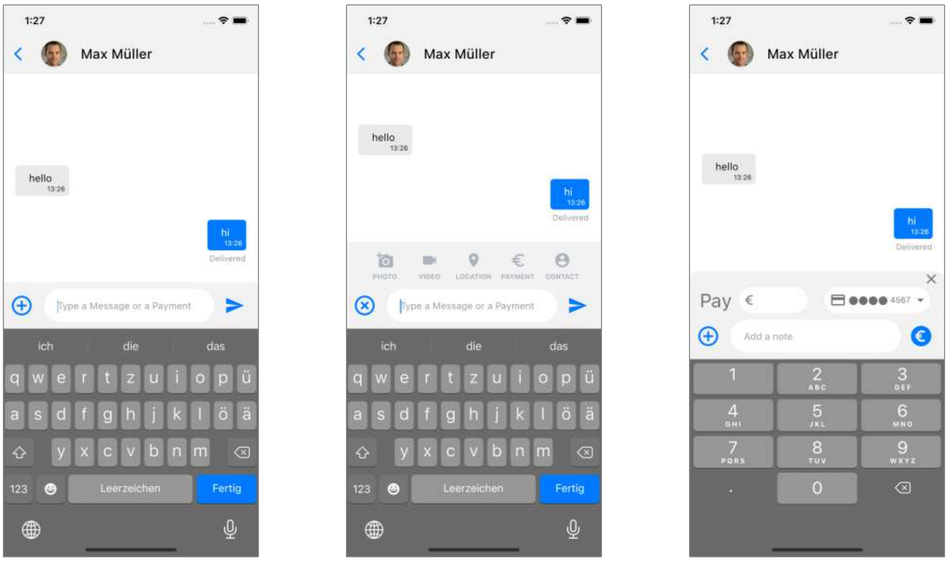
Furthermore, this prototype detects payment intents in real-time on the text the user types into the message text input field (see Figure 3.20(b)). The keywords that caused a suggestion get underlined in the message text input field and the payment configuration view appears.

The automatically detected amount of a transaction can be simply changed by tapping into the amount input field in the payment configuration view (see Figure 3.20(a)). An optional payment note can be added by writing into the main text input field of the message composer.



(a) Tap on the blue button (b) Enter Passcode (c) Transaction Completed

Figure 3.21: Confirmation of the payment in the “Suggestion“ prototype



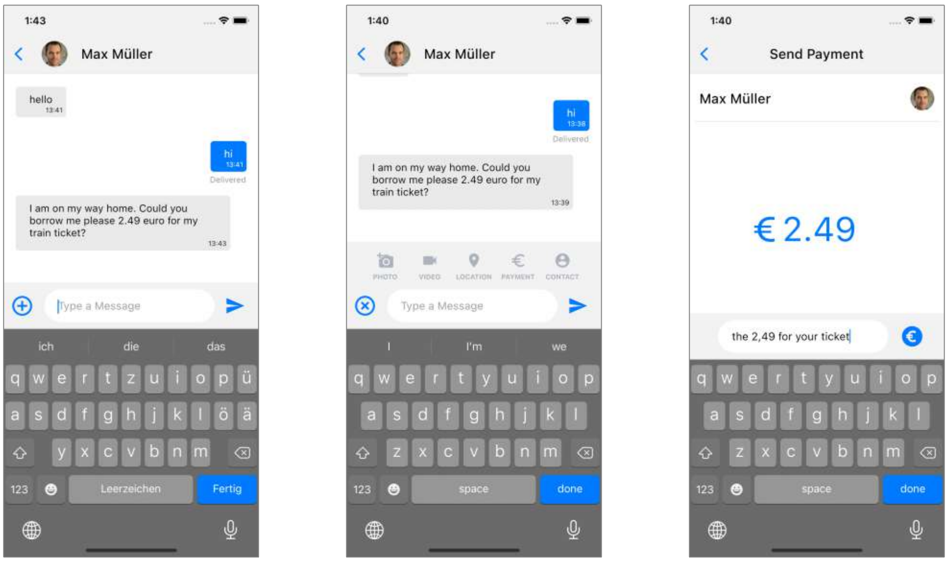
(a) Tap on the “+“ icon (b) Tap on the “€“ icon (c) Transaction activated

Figure 3.22: Manual transaction activation in the “Suggestion“ prototype

To verify a payment, the user needs to press the “€” icon in the message composer while in the payment configuration view. After this, the payment confirmation view appears (see Figure 3.21(a)). Tapping on the “Pay” button causes the appearance of the passcode input query (see Figure 3.21(b)). By entering the right passcode, the user returns back to the conversation screen (see Figure 3.21(c)).

Additionally, in the "Suggestion" prototype, the payment process can be activated the same way as in the "Wizard" prototype (see Subsection 3.3.2)(see Figure 3.22(a)). Instead of leaving the conversation screen after tapping on the "€" icon, the icon bar disappears and the payment configuration appears in the message composer above the message text input field (see Figure 3.22(b)-(c)).

3.3.2 "Wizard" Prototype

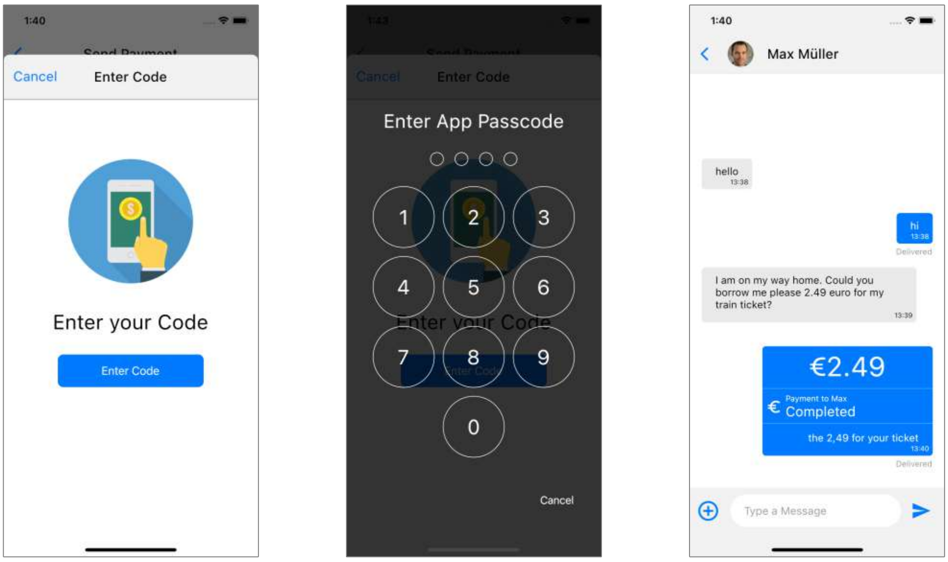


(a) Tap on the "+" icon (b) Tap on the "€" icon (c) Tap on the "€" icon

Figure 3.23: Activation of the payment process in the "Wizard" prototype

WhatsApp and *WeChat* have a similar user interface and follow the same interaction model to perform a P2P payment. Because of this, these two user interfaces were combined into one prototype that represents both solutions. This prototype is referred to as "Wizard". To perform a payment with the "Wizard" prototype, the user needs to tap on the "+" icon in the message composer to initiate a P2P payment in the chat user interface (see Figure 3.23(a)). After this, the message composer extends and an icon bar appears above the text input field (see Figure 3.23(b)). The icons in the extended message composer represent features like sending multimedia content, the current location, funds, etc. Tapping on the "€" icon activates the payment process and the user leaves the conversation screen to the payment configuration screen (see Figure 3.23(c)).

Now the user has to define the amount of the transaction and an optional payment note can be set (see Figure 3.23(c)). To change the amount of the transaction the user must type in the preferred amount using the keypad. Tapping on the "€" icon in the configuration screen initiates the multi-step confirmation procedure of the transaction

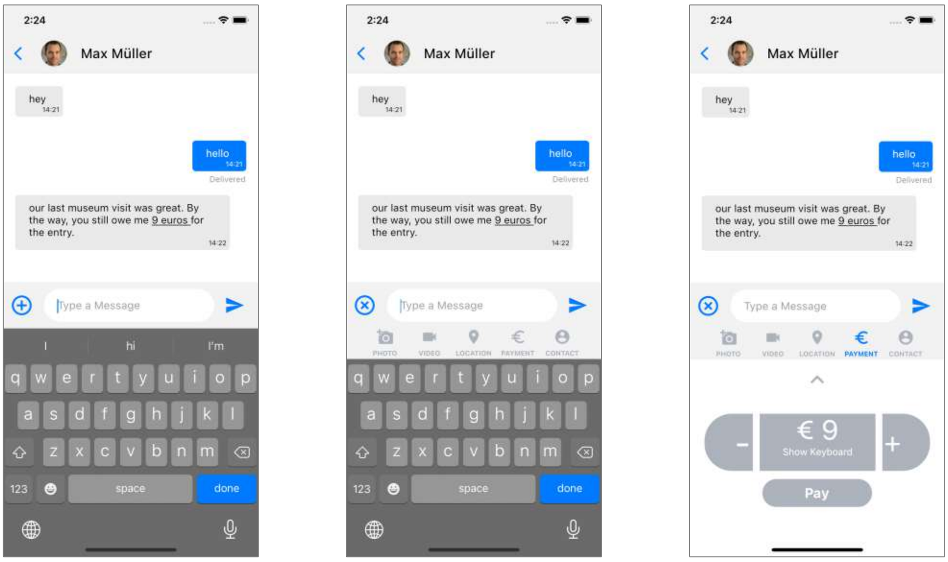


(a) Tap on the blue button (b) Enter Passcode (c) Transaction completed

Figure 3.24: Confirmation of the payment in the “Wizard“ prototype

(see Figure 3.24(a)). After entering the passcode (see Figure 3.24(b)) the user returns back to the conversation screen and the transaction is now visible (see Figure 3.24(c)).

3.3.3 “Inline” Prototype

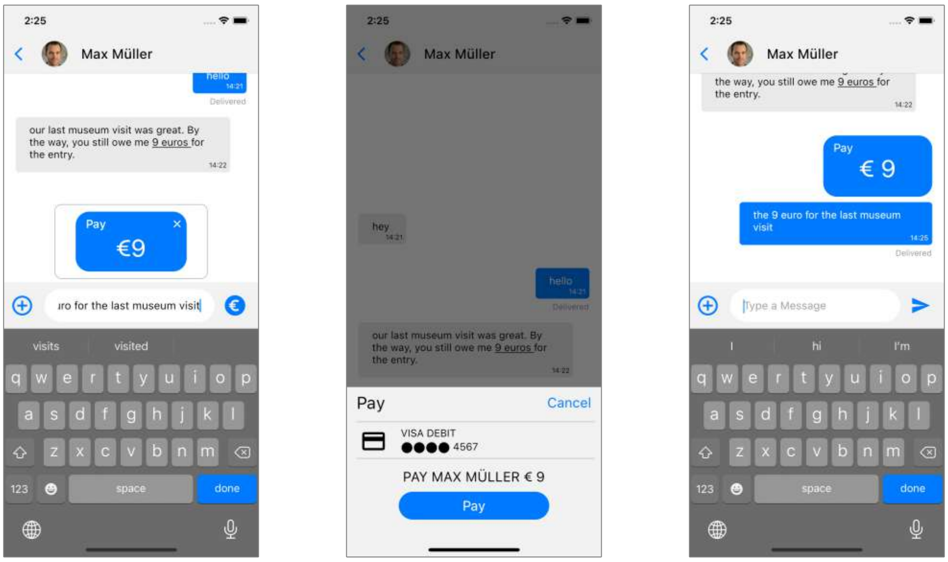


(a) Tap on the “+“ icon (b) Tap on the “€“ icon (c) Tap on the “Pay“ button

Figure 3.25: Activation of the payment process in the “Inline“ prototype

iMessage and *Facebook Messenger* follow a different interaction flow. Unlike *WhatsApp* and *WeChat*, users are not leaving the current context of the chat interface during a transaction. The transaction can be configured and completed in a view that replaces the keyboard. Because of this, the user interfaces of *iMessage* and *Facebook Messenger* have been combined and implemented as the third prototype that is referred to as the "Inline" prototype.

To initiate a payment in this prototype, the user has to tap the "+" icon as in the "Wizard" prototype (see Figure 3.25(a)). After this, the predictive text bar of the keyboard disappears, the message composer gets extended under the message text input field and the icon bar like in the "Wizard" prototype appears (see Figure 3.25(b)). By tapping on the "€" icon, the keyboard disappears and gets replaced by the transaction amount definition view (see Figure 3.25(c)). Additionally, the "Inline" prototype detects amounts in past messages and underlines them in the conversation view. Tapping on an underlined amount in a message triggers the payment process and the amount definition view appears with the detected amount already set.



(a) Tap on the "€" icon (b) Tap on the "Pay" button (c) Transaction completed

Figure 3.26: Confirmation of the payment in the "Inline" prototype

In the "Inline" prototype the user can increase and decrease the amount of the transaction by tapping on the "+" or "-" button in the amount definition view (see Figure 3.25(c)). By tapping on the button with the label "Show Keyboard" or by sliding up, the amount definition view extends and a keypad appears. With the help of the keypad, the amount of the transaction can be set.

By tapping the "Pay" button, the amount definition view disappears. Now the user can type in an optional payment note in the message composer under the payment preview bubble (see Figure 3.26(a)). To complete a transaction in this prototype the user has to tap the “€“ icon located in the message composer. By tapping on this icon a payment confirmation view appears on the bottom of the screen (see Figure 3.26(b)). After the user taps on the “Pay” button, the code input query appears. If the user enters the right passcode the user returns back to the conversation screen and the transaction is visible as a bubble in the chat conversation (see Figure 3.26(c)).

3.3.4 Differences between the Prototypes

The payment suggestions feature is the main difference between the “Suggestion“ prototype and the two reimplementations of the already existing solutions. Depending on the conversational context and the user’s input, the “Suggestion“ prototype provides payment suggestions. The “Wizard“ prototype and the “Inline“ prototype do not provide active payment suggestions. However, the “Inline“ prototype underlines detected amounts in the current chat conversation. By tapping on a detected amount, the “Inline“ prototype initialises the payment process with the detected amount already predefined. The “Wizard“ prototype does not detect payment amounts in previous messages.

Another difference is the interaction flow of the three prototypes. The user needs to initiate the payment process manually in the “Wizard“ and the “Inline“ prototype by tapping the “+“ icon in the message composer. After this, an icon bar appears above the keyboard, and the user has to tap the “€“ icon. Without a payment suggestion, the activation of the payment process in the “Suggestion“ prototype is identical to the “Wizard“ prototype. In the “Wizard“ prototype, the user leaves the conversation screen and navigates to the payment configuration screen. The user must define the payment amount and attach an additional note at the payment configuration screen. The payment process does not take place in a different screen in the “Suggestion“ prototype and the “Wizard“ prototype. Instead, the payment configuration appears in the conversation screen. In the “Suggestion“ prototype, the payment configuration is placed in the extended message composer. In the “Inline“ prototype, the user has to define the payment amount in a view that appears instead of the keyboard.

In the “Suggestion“ prototype and the “Wizard“ prototype, the user defines the payment amount with an ordinary text input field. In the “Inline“ prototype, the user has to use “+“,“-“ buttons to define the payment amount. For decimals, the user can use an optional keypad.

In the “Wizard“ prototype and the “Suggestion“ prototype, the optional payment note can be attached to a payment where to define the payment amount. In the “Inline“ prototype, the optional payment note can be set in the payment preview view, which appears after attaching the payment amount. The payment confirmation view is not available in the “Wizard“ and the “Suggestion“ prototypes.

In the “Wizard“ prototype, the multi-step payment confirmation occurs in an external screen. The payment confirmation is displayed as an overlay in the conversation screen in the “Inline“ prototype and the “Suggestion“ prototype.

3.3.5 Benefits of the "Suggestion" Prototype

The "Suggestion" prototype aims to increase the overall usability of transactions in mobile messengers. If a payment intent is detected, the prototype automatically suggests the user to perform a payment and this reduces the steps the user needs to activate the payment process. If the detected payment intent contains an amount, the prototype automatically sets the amount for the transaction. In comparison, the "Inline" prototype detects only amounts in messages and does not actively suggest the user to perform a payment like the "Suggestion" prototype.

Like in the "Inline" prototype, in the "Suggestion" prototype users are not leaving the current context of the chat interface. Instead, the transaction configuration process takes place in the chat conversation screen. This approach promises to improve the perceived user satisfaction. In the "Suggestion" solution, the user can set the desired amount and the optional payment note in the same view. This approach reduces also the number of steps required to perform a payment.

3.3.6 Limitations of the Implemented Prototypes

Currently, the P2P payment features of *WhatsApp*, *Facebook Messenger*, *WeChat*, and *iMessage* are restricted to use in only a selected number of countries worldwide. It was therefore not possible to test the existing applications in Austria. Furthermore, these messengers do not offer detailed documentation about their user interfaces. Therefore, the "Wizard" and "Inline" prototypes were implemented based on tutorials and videos found online. They may vary in some details and may not represent the latest version of the messenger at the time of the implementation.

The design of the reimplemented prototype varies from the original messengers. All three prototypes were implemented using the same design theme to prevent design preferences by the participants. Design preferences were not part of the research questions because individual preferences could cause unreliable results.

The transaction process of existing messenger applications that already support P2P payments contain delays due to e.g. network communication and therefore show loading progress indicators. Such delays or loading screens were not included in the reimplementation. However, the implemented prototypes contain a loading sequence of 3 seconds before the passcode input query to simulate network communication in order to provide a more realistic user experience.

The implemented prototypes are not fully functional mobile messengers. The functions are limited to writing messages and performing payments. Other functions that were not part of the research like sending multimedia content are not supported by these prototypes.

3.4 Technical Implementation

In the following subsections, the used technology and technical architecture of the three implemented prototypes will be discussed.

3.4.1 Flutter

The implemented prototypes of this thesis were developed using *Flutter 1.22*.

Flutter is *Google's* open-source framework for the development of native cross-platform applications for *iOS*, *Android*, *Windows*, *Linux*, and the Web from a single codebase [Nap19]. According to *SlashData's* Mobile Developer Population Forecast 2021, *Flutter* is the most popular cross-platform framework with a market share of 45%[10].

Dart is the programming language used for *Flutter*. It is an object-oriented programming language that compiles to native ARM machine code. The compilation of *Dart* code to native code is ahead of time. Furthermore, *Dart* is able to compile just-in-time (JIT) that allows displaying code changes instantly. The user interface for *Flutter* applications is written in *Dart* and because of this there is no need for external languages or visual designers. *Flutter* applications run with at least 60 frames per second (fps) which provides smooth animations and transitions.

The three different prototypes were combined into one application that runs on *iOS* and *Android*. However, the application was optimized to run on the test device of the usability study (an *iPhone X* running *iOS 14*).

3.4.2 Cloud Firestore

Cloud Firestore[11] databases were used to store and exchange the messages and transactions between the users of the implemented prototypes. An additional database stores log events of the prototypes for the evaluation of the usability study.

Cloud Firestore[12] is a part of *Google's Firebase* platform and is a cloud-hosted NoSQL database that allows to store and sync data between client applications. In a *Firestore* database, documents of different data types are stored as collections. Real-time listeners ensure that the data on connected devices stay up to date.

3.4.3 Dialogflow

Dialogflow[12] is *Google's* natural language understanding platform. It allows the integration of conversational user interfaces into apps, devices, etc. *Dialogflow* is capable of handling

[10] Tim Sneath (2021): "Announcing Flutter 2.2 at Google I/O 2021", available at: `https://medium.com/flutter/announcing-flutter-2-2-at-google-i-o-2021-92f0fcbd7ef9` (accessed July 29, 2021)

[11] Google (n.d.): "Cloud Firestore", available at: `https://firebase.google.com/docs/firestore` (accessed July 28, 2021)

[12] Google (n.d.): "Dialogflow ES documentation", available at: `https://cloud.google.com/dialogflow/es/docs` (accessed July 28, 2021)

audio and text as inputs and synthetic speech and text as outputs.

For the novel suggestion-based prototype (the "Suggestion" prototype), two *Dialogflow* agents were defined to detect transaction intents. One agent detects payment suggestions based on the conversational context (the "Passive Suggestions" agent), and the second agent detects payment suggestions based on the user's keyboard input ("Active Suggestions" agent). Both agents were trained with keywords that may lead to transactions in mobile messengers (see Figure 3.27). These keywords were collected through a short prestudy with three participants that were asked to formulate payment intents in messenger conversations. Additionally, the collected keywords were extended with their synonyms. The "Passive Suggestions" agent detects payment suggestions if a sentence

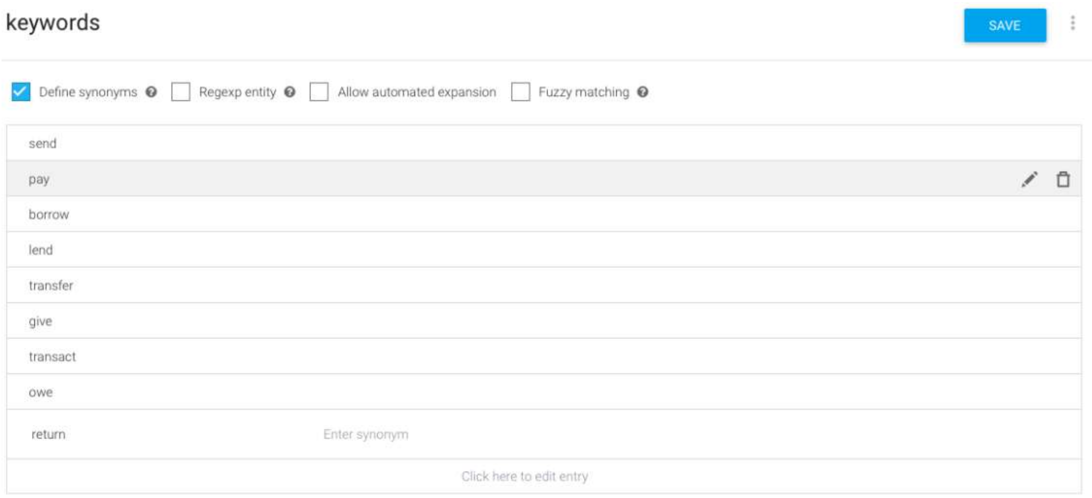


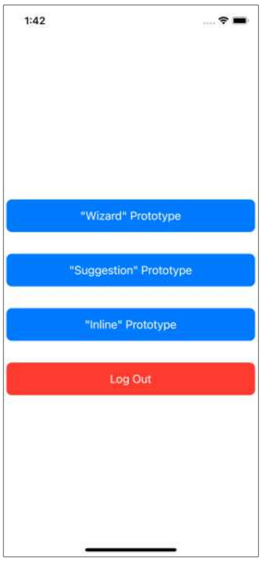
Figure 3.27: Keywords of the Dialogflow agents

in the previously received message contains a keyword that is followed by a pronoun (e.g "borrow me"). Furthermore, the "Passive Suggestions" agent detects amounts in messages (e.g "Please borrow me 10 euros"). Compared to the "Passive Suggestions" agent, the "Active Suggestions" agent detects payment suggestions if the text input contains a keyword and a payment amount (e.g "Send 10 euro").

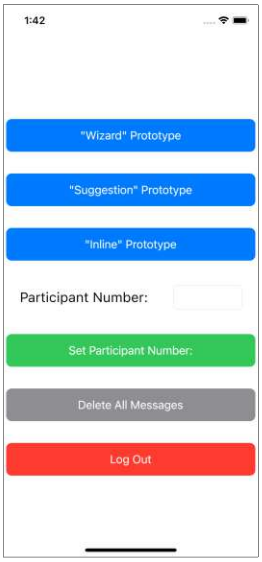
For the "Inline" prototype, one *Dialogflow* agent was defined for the detection of payment amounts in past text messages. The chat conversations of the usability study were scripted to ensure that the suggestions work accurately for the defined tasks.

3.4.4 Technical Architecture of the implemented Prototypes

The three prototypes were implemented in one chat application. The implemented chat application supports two log-in modes. One mode for the participant (see Figure 3.28(a)) and one for the facilitator (see Figure 3.28(b)) of the usability study. The "Facilitator Mode" provides more functionality than the "Participant Mode". For example, in the



(a) “Participant Mode“



(b) “Facilitator Mode“

Figure 3.28: Log-in modes of the application

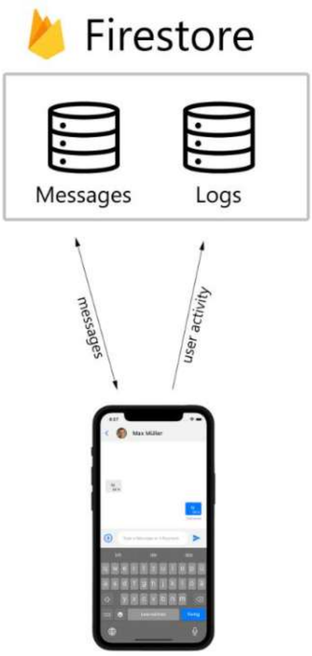


Figure 3.29: Technical architecture of the “Wizard“ prototype

“Facilitator Mode”, the user can set the participant number and delete past messages. Furthermore, the user’s activity gets only logged in the “Participant Mode”. The application supports two-way chat communication between one participant and one facilitator.

After the user logs in with one of the supported modes, the user can select between the three different implemented chat user interfaces. The three implemented prototypes have a similar technical architecture. The user interfaces communicate with a *Firestore* database to exchange text messages and transactions between the two connected users (see Figure 3.29). In the "Participant Mode", the prototypes log each user activity and send it with an additional time-stamp to a second *Firestore* database. The "Inline" prototype is additionally communicating with a *Dialogflow* agent that detects payment amounts in past messages (see Figure 3.30). Each text message the user receives gets forwarded to the "Passive Suggestions" agent. If the agent detects a payment amount in a message, it returns the detected amount to the prototype. The prototype then underlines the detected amount in the message.

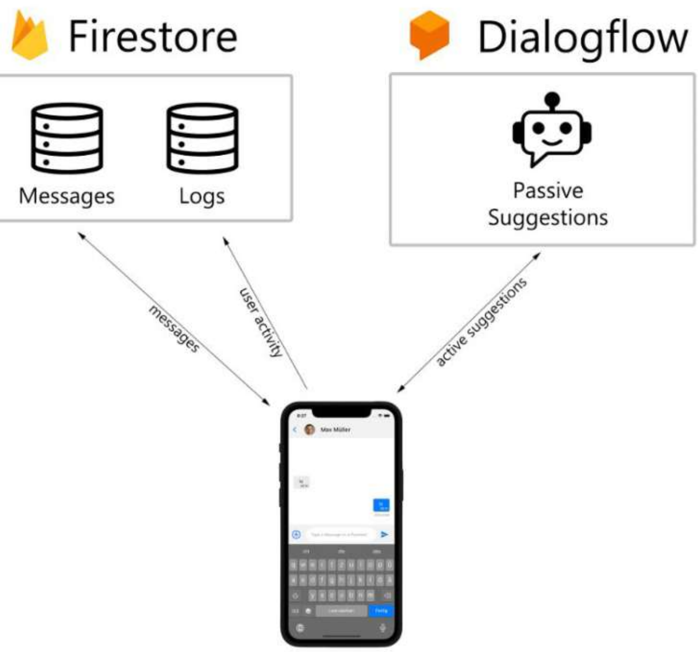


Figure 3.30: Technical architecture of the "Inline" prototype

For the novel suggestion-based prototype (the "Suggestion" prototype), two *Dialogflow* agents were defined (see Figure 3.31). The "Passive Suggestions" agent was created to detect transaction intents in the conversational context. Each message the user receives in the "Suggestion" prototype gets forwarded to the "Passive Suggestions" agent. If the "Passive Suggestions" agent detects a payment intent in the message, the agent sends the keywords that caused a detection back to the prototype. After this, the prototype underlines the detected keywords in the message and triggers a payment suggestion. If the "Passive Suggestions" agent additionally detects a payment amount, the detected amount is set as the payment amount for the payment suggestion. For the detection of transaction intents in the user's active input, the "Active Suggestions" agent was defined. Each time the user changes the input of the text input field in the message composer, the text input is sent to the "Active Suggestions" agent. If the "Active Suggestions" agent detects keywords that may indicate a payment intent, the prototype sends the detected keywords back to the prototype. The prototype then underlines the detected keywords in the input text of the text input field and triggers a payment suggestion. Like the "Passive Suggestions" agent, the "Active Suggestions" agent also detects payments amounts.

Figure 3.31: Technical architecture of the "Suggestion" prototype

CHAPTER 4

Evaluation of the Prototype

The implemented prototype of a novel user interface approach for P2P payments in mobile chat applications was evaluated using a comparative usability study. According to the International Organization for Standardization (ISO) efficiency, satisfaction, and effectiveness are the leading attributes of usability (ISO 9241-11[1])Therefore, the novel implemented prototype was compared with the reimplemented prototypes of the already existing solutions on these three attributes [Wei20]. The results of the usability study were used to answer the research questions of this thesis. This study determined if a suggestion-based user interface improves the general usability of P2P payments in mobile chat applications. In this chapter, the conducted usability study will be described in detail.

4.1 Participants of the Usability Study

The implemented prototypes were evaluated with 20 participants. Each participant used the prototype of the novel approach and the prototypes of the reimplemented existing solutions. Participants of this study were recruited based on their experience with mobile chat applications. The recruited participants were required to have frequently communicated with mobile messengers and to be familiar with mobile banking because the participants had to transfer funds to another chat participant in different scenarios with the implemented prototypes. Furthermore, the participants had to be *iOS* users, because the study had been conducted using an *iOS* device. This ensured that the participants were already familiar with the used operating system. Because of this, only participants that met these criteria were recruited. Furthermore, a background survey was conducted to ensure that the recruited participants met all criteria and to get general information about the participants (see Chapter 5.1). The usability study took place in

[1]ISO (2018): “ISO 9241-11:2018(en)”, available at: `https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:en` (accessed September 13, 2021)

Austria and all participants were Austrian citizens. This study was conducted in English, therefore the recruited participants had to be able to communicate in English.

4.2 Study Setting

The usability study took place in a laboratory setting and each of the 20 participants was tested individually. During the usability study, the participants sat at a table and performed tasks using different user interface prototypes. Using a within-subject design for the study, all participants were tested on each variation of the implemented prototypes. The testing device of the usability study was an *iPhone X* running on *iOS 14* with a screen resolution of 2436 × 1125 pixels.

The prototyped user interfaces were developed in one *iOS* application and therefore the users did not have to switch applications during the usability study. In two different tasks, the participants had to chat and transfer funds to their chat partner using the different prototypes. The chat partner that should receive the funds was simulated by the facilitator using the same prototypes running on another smartphone.

The usability study was guided by one facilitator that provided instructions, answered questions, and observed the tasks. Because of the *COVID-19* pandemic, the usability study did comply with all Austrian *COVID-19* regulations at the time of the study. The facilitator took a minimum distance of two meters to the participant and both were asked to wear *FFP2* face masks.

4.3 Tasks of the Usability Study

During the usability study, the participants had to perform two tasks with the three developed prototypes in different scenarios to collect data about the usability of the different prototypes. For each task, three scenarios were defined to add variation to the tasks. In "Task 1", the participants had to send money to the other chat participant after receiving the instruction for the payment by the facilitator. In "Task 2" the participants actively chatted with the other chat participant and during the conversation, the other chat participant requested a payment from the participant. Depending on the context of this conversation the participant had to perform a payment to the other chat participant. The messages of the other participants for each scenario of "Task 2“ were scripted to guarantee reliable study results (see Appendix A).

In the following subsection, the two different tasks including their scenarios will be described.

4.3.1 Task 1

	Task 1: Scenario 1
Task	The participant has to transfer 8.50 Euro to the other chat participant. Furthermore, the participant should add the payment note “the 8.50 for last night’s diner“ to the transaction. This task begins when the user opens the prototype’s user interface and ends after the successful transaction of 8.50 Euro. The transaction must contain the given transaction note.
Script	You have borrowed 8.50 Euro for last night’s dinner from your friend. Now you want to return that amount within this chat application. Additionally you want to add the message “the 8.50 for last night’s dinner“ to the transaction. Talk about your thoughts while conducting the task.
Inputs	Transaction amount: 8.50/8.5 Payment note: “the 8.50 for last night’s diner”

Table 4.1: Task 1: Scenario 1

	Task 1: Scenario 2
Task	The participant has to transfer 12.99 Euro to the other chat participant. Furthermore, the participant should add the payment note “here 12.99 for the birthday gift” to the transaction. This task begins when the user opens the prototype’s user interface and ends after the successful transaction of 12.99 Euro. The transaction must contain the given transaction note.
Script	Your friend is collecting money for a common friend’s birthday gift. You want to make a transaction of 12.99 Euro to your friend that includes the message “here 12.99 for the birthday gift”. Start the transaction by typing “here 12.99 for the birthday gift” into the chat text input field. Talk about your thoughts while conducting the task.
Inputs	Transaction amount: 12.99 Payment note: “here 12.99 for the birthday gift”

Table 4.2: Task 1: Scenario 2

	Task 1: Scenario 3
Task	The participant has to transfer 10.00 Euro to the other chat participant. Furthermore, the participant should add the payment note “Happy birthday! I send you 10 euros”. This task begins when the user opens the prototype’s user interface and ends after the successful transaction of 10.00 Euro. The transaction must contain the given transaction note.

Table 4.3 continued from previous page

Script	Today is your friend's birthday. You want to send him 10.00 Euro as a birthday gift. Additionally you want to add the message "Happy birthday! I send you 10 euros" to the transaction. Talk about your thoughts while conducting the task.
Inputs	Transaction amount: 10/10./10.0/10.00 Payment note: "Happy birthday! I send you 10 euros"

Table 4.3: Task 1: Scenario 3

4.3.2 Task 2

	Task 2: Scenario 1
Task	The participant has to transfer 2.49 Euro to the other chat participant. The participant will be asked to lend 2.49 euros to buy a train ticket. Furthermore, the participant should add the payment note "the 2.49 for your ticket" to the transaction. This task begins when the user receives the message about the payment request and ends after a successful transaction of 2.49 euros. The transaction must contain the given transaction note.
Script	You are chatting with your friend. Depending on the conversational context you will be asked to perform a payment to your friend. Additionally you want to add the message "the 2.49 for your ticket" to the payment. Talk about your thoughts while conducting the task.
Inputs	Transaction amount: 2.49 Payment note: "the 2.49 for your ticket"

Table 4.4: Task 2: Scenario 1

	Task 2: Scenario 2
Task	The participant has to transfer 8.80 Euro to the other chat participant. The participant will be asked for 8.80 Euro for today's restaurant reservation. Furthermore, the participant should add the payment note "the 8.80 for today's restaurant reservation" to the transaction. This task begins when the user receives the message about the payment request and ends after a successful transaction of 8.80 Euro. The transaction must contain the given transaction note.
Script	You are chatting with your friend. Depending on the conversational context you will be asked to perform a payment to your friend. Additionally you want to add the message "the 8.80 for today's restaurant reservation" to the payment. Talk about your thoughts while conducting the task.

Table 4.5 continued from previous page

Inputs	Transaction amount: 8.80/8.8 Payment note: "the 8.80 for today's restaurant reservation"

Table 4.5: Task 2: Scenario 2

	Task 2: Scenario 3
Task	The participant has to transfer 9.00 Euro to the other chat participant. The participant will be asked to return 9.00 Euro to the chat partner for their last common museum visit. Furthermore, the participant should add the payment note "the 9 euro for the last museum visit" to the transaction. This task begins when the user receives the message about the payment request and ends after a successful transaction of 9.00 Euro. The transaction must contain the given transaction note.
Script	You are chatting with your friend. Depending on the conversational context you will be asked to perform a payment to your friend. Additionally, you want to add the message "the 9 euro for the last museum visit" to the payment. Talk about your thoughts while conducting the task.
Inputs	Transaction amount: 9/9./9.0/9.00, Payment note: "the 9 euro for the last museum visit"

Table 4.6: Task 2: Scenario 3

4.4 Task Order

All participants conducted two tasks on three different prototypes, resulting in six tasks to be completed per participant in total. The order of the prototypes used for the different participants was counterbalanced to prevent any potential learning effects. Any kind of learning effect could cause unreliable study results [RC08]. To avoid unreliable study results the prototypes had been ordered according to the Latin Square technique (see Figure 4.1). The same counterbalancing technique was also used for the sequence of the scenarios within the tasks.

4.5 Study Procedure

First of all, the facilitator provided the participants with an instruction sheet that contained detailed information about the study and the facilitator answered all upcoming questions of the participants. It also provided the participants with all the important instructions for the upcoming tasks. For example, the participants must not change the default payment method, because any payment method changes were not supported

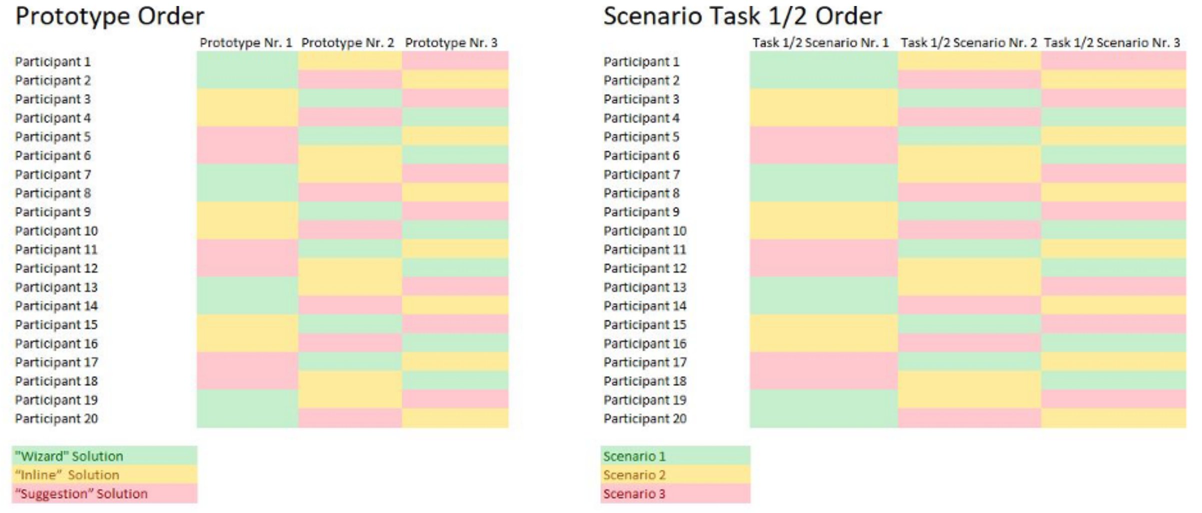


Figure 4.1: Prototype and Task Scenario Order

in the prototypes. Furthermore, the instruction sheet contained a passcode, that was necessary for the payment process in the different prototypes.

The participants had to read and sign the consent form (see Appendix B) of the usability study which was required to participate. Any upcoming questions were answered by the facilitator and the participants received a copy of the signed consent form. Before conducting the tasks, the participants had to fill in an online background survey on the testing device to gain general information about the participants (e.g gender, age, messenger usage, etc.) (see Appendix C). To start the survey the facilitator typed in a unique participant number that helped to associate the collected data with the participant anonymously.

With each of the three prototypes, two tasks had to be conducted. Before each task, the facilitator set up the task and provided the participants with the scenario and instructions of the task written on paper. To start the task the participant had to press a button that opened the user interface. To avoid unreliable data, the facilitator was not allowed to answer questions or help the participant during the tasks. While conducting the tasks, the participants were asked to think out aloud. Think-aloud is a method to gather data of the user's thinking process [Cha03].

After finishing the tasks of one prototype the participants received a printed questionnaire according to the System Usability Scale (SUS) in order to collect data about the user's satisfaction. After the participants filled in the questionnaire, the study continued with the tasks of the next prototype.

After all tasks of the three prototypes were conducted, the facilitator held a semi-structured post-test interview to get qualitative data about the perceived trust and usability of the prototypes. The facilitator recorded the audio of the interview with his smartphone and took notes with pen and paper. The permission for the recordings was approved through the consent form (see Appendix B).

4.6 Collected Data

During the usability study, quantitative data about efficiency, effectiveness and satisfaction was collected. The prototypes used for the tasks logged as an event containing a timestamp and sent them to a *Firestore* database. Furthermore, the screen of the testing device were recorded. The screen recordings contained the voice of the participants to capture think-aloud. The log files in combination with the screen recordings were used to calculate the attributes of efficiency and effectiveness. Additionally, qualitative data were collected through a semi-structured post-test interview.

4.6.1 Efficiency

Efficiency describes the speed and accuracy a user performs a task [Wei20]. To evaluate the efficiency of the prototypes the task completion time, time spent on each activity and number of errors were considered.

Task Completion Time

The task completion time is the time the user requires to complete a task. The starting point of the task completion time varied between the two tasks. In "Task 1", the task completion time began after the participant opened the prototypes user interface and ended after the successful transaction. In "Task 2", the task completion time began when the user received the message about the payment request and ended after a successful transaction.

Time Spent on each Activity

Analysing the user interfaces of the already existing solutions had shown that mobile chat applications require the following four activities to perform a P2P payment.

- Activation of the payment process
- Specification of the amount
- Adding of an optional payment note
- Confirmation of the payment

The time required for these 4 activities were recorded and compared between the prototypes.

Number of Errors

The number of errors the user made during a task were counted manually with the collected log files and the screen recordings. An error was defined as a tap on a wrong user interface element (button, icon, etc.). Typing mistakes that occurred during the input of text or amounts did not count as errors.

4.6.2 Effectiveness

Effectiveness is the user's ability to complete a certain task and it was evaluated with the number of successfully completed tasks [Wei20]. The Task Success Rate is defined as the number of successfully completed tasks in relation to the conducted tasks per prototype. A task was defined as successfully completed if the user transferred the right amount of money and added the right payment note to the transaction.

4.6.3 Satisfaction

The user's satisfaction is the level of comfort, pleasure and fulfilment of expectations [Wei20]. The satisfaction regarding the prototypes was evaluated using the System Usability Scale (SUS) [Bro95]. The SUS is a standardised questionnaire that consists of 10 statements the participant needs to rank from 1 to 5 (1= Strongly disagree, 5= Strongly Agree) that provides the subjective evaluation of satisfaction. The SUS questionnaire of each prototype was evaluated by calculating the SUS score.

The SUS score is calculated as followed:

- the results of odd question numbers have to be reduced by 1
- the results of even questions numbers have to be subtracted from 5
- the results of step 1 and step 2 have to be added up
- the result of step 3 has to be multiplied by 2.5

After the calculation of the SUS score, the result can be ranked using the SUS Grading Scale (see Table 4.7).

SUS Score	Grade	Adjective Ranking
>80.3	A	Excellent
68-80.2	B	Gook
67	C	Okay
51-66	D	Poor
<51	F	Awful

Table 4.7: SUS Grading Scale [SZN19]

4.6.4 Post-Test Interview

A semi-structured post-test interview was conducted to get qualitative data about the perceived trust and usability of the prototypes.

The following questions were asked during the interview:

- Which of the three tested user interfaces would you prefer to perform P2P payments in the future? For what reason/reasons?
- During this usability study, you have tested three different prototyped user interfaces for P2P payments in mobile chat applications. Which of these three user interfaces did you trust the most while performing a payment? For what reason/reasons?
- Would you refuse one or more of the tested prototypes to perform a payment with? If yes, which one/ones? For what reason/reasons?
- One of the used prototypes provides the user with in-line suggestions for P2P payments depending on the user's input and the conversational context.
  - How did you experience this feature?
  - Was this feature useful while conducting the tasks?
  - Could you imagine in-line suggestions for other tasks in mobile chat applications? if yes, for which tasks?

Additional questions were added to the post-test interview by the facilitator about noticeable difficulties or behaviour that occurred during the usability study.

4.6.5 Data Analysis

The collected quantitative data were analyzed using descriptive statistics and ANOVA tests. The ANOVA test is a statistical method for analysing the mean of a variable and the simplest form is the one-way ANOVA test [BCB04]. It is used to compare groups and is an extension of the independent t-test. If the p-value of an ANOVA test results a value less than 0.05 (5%) implies the existence of significant difference between the groups. The p-value is the probability that ANOVA wrongly assumes a difference within the tested groups. The classic one-way ANOVA is the Fisher's F-test [MH17]. The Fisher's F-test assumes equal variances for calculating the p-value, and this approach increases the probability of a Type 1 error. The Welch's F-test is an alternative to Fisher's F-test that decreases the probability of a Type 1 error by not considering the variances of the groups as equal. However, Welch's F-tests did not provide valid results for certain metrics because of too many identical data entries. For that reason, Fisher's F-tests were conducted for the evaluation.

ANOVA tests provide only knowledge about whether significant differences exist within groups or not. To find out which means within the groups differ significantly Tukey post-hoc tests were conducted.

CHAPTER 5

Evaluation of the Usability Study

In this chapter, the conducted usability study will be evaluated. It contains a summary of the participant's background survey and the post-test interview. Furthermore, the collected quantitative data of the usability study will be evaluated using descriptive statistics and statistical analysis.

5.1 Background Survey

For the conducted comparative usability study, 20 participants were recruited from 22/6/21 until 2/7/21 in Vienna. The recruited participants had to fill in an online background survey on the test device at the start of the usability study in order to get general information about the participants. This survey was created and evaluated with

	Variable	Frequency(n)	Percentage
1. Gender			
	female	6	30%
	male	14	70%
2. Age			
	16-20	1	5%
	21-30	16	80%
	51-60	3	15%
3. Education			
	High School	10	50%
	Bachelor's Degree	7	35%
	Master's Degree	2	10%
	Ph.D	1	5%

Table 5.1: Demographic data of the participants

Google Forms. All of the participants were Austrian citizens living in Vienna, Styria, and Lower Austria.

70% of the recruited participants were male, and 80% were between 21 and 30 years old (see Table 5.1). Three participants were between 51 and 60 years old (15%) and one under 20 years old. Half of the participants stated that their highest level of education is High School. 35% of the participants had a Bachelor's degree, 10% a Master's degree, and 5% a Ph.D. All of the participants were primary *iOS* users, and 30% of them stated that they were also *Android* users.

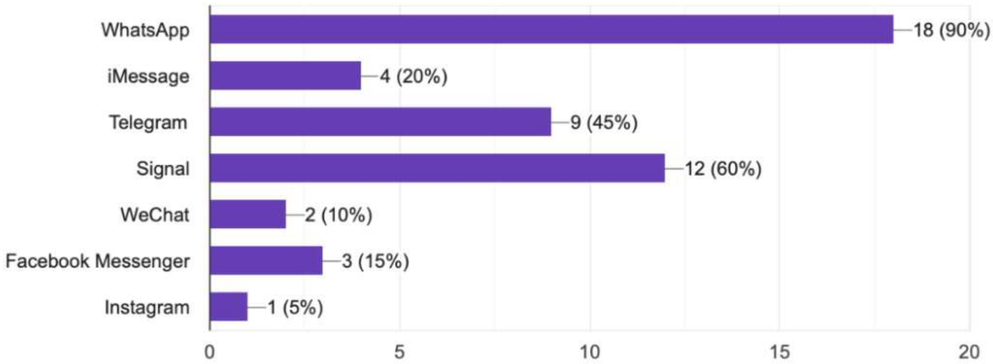


Figure 5.1: Mobile messenger usage of the participants

The most popular mobile messenger among the participants was *WhatsApp* (90% of the participant stated that they use *WhatsApp* frequently) (see Figure 5.1). The second most popular messenger was *Signal*, followed by *Telegram*. 85% of the participants stated that they communicate with mobile messengers multiple times a day, and the rest of the participants (15%) stated that they communicate only daily with mobile messengers.

90% of the participants were mobile banking users. Of these 90%, 40% stated that they use mobile banking weekly, and another 40% answered that they use mobile banking every few weeks. One participant (5%) mentioned that he uses mobile banking multiple times a day.

Only one of the 20 recruited participants stated that he had already transferred money directly in a mobile messenger app. This participant explained that he made payments using *WeChat* during his semester abroad in Hong Kong.

5.2 Efficiency

For the evaluation of the three implemented prototypes on efficiency, the following metrics were considered:

- Task completion time

- Time spent on each of the four identified activities
  - Activation of the payment process
  - Specification of the amount
  - Adding of an optional payment note
  - Confirmation of the payment
- Number of errors

Only data of successfully completed tasks were considered for the evaluation of this metric, and for this reason, the sample size N varies between the prototypes in Task 1 and Task 2. Because of the differences between Task 1 and Task 2, both tasks were evaluated separately.

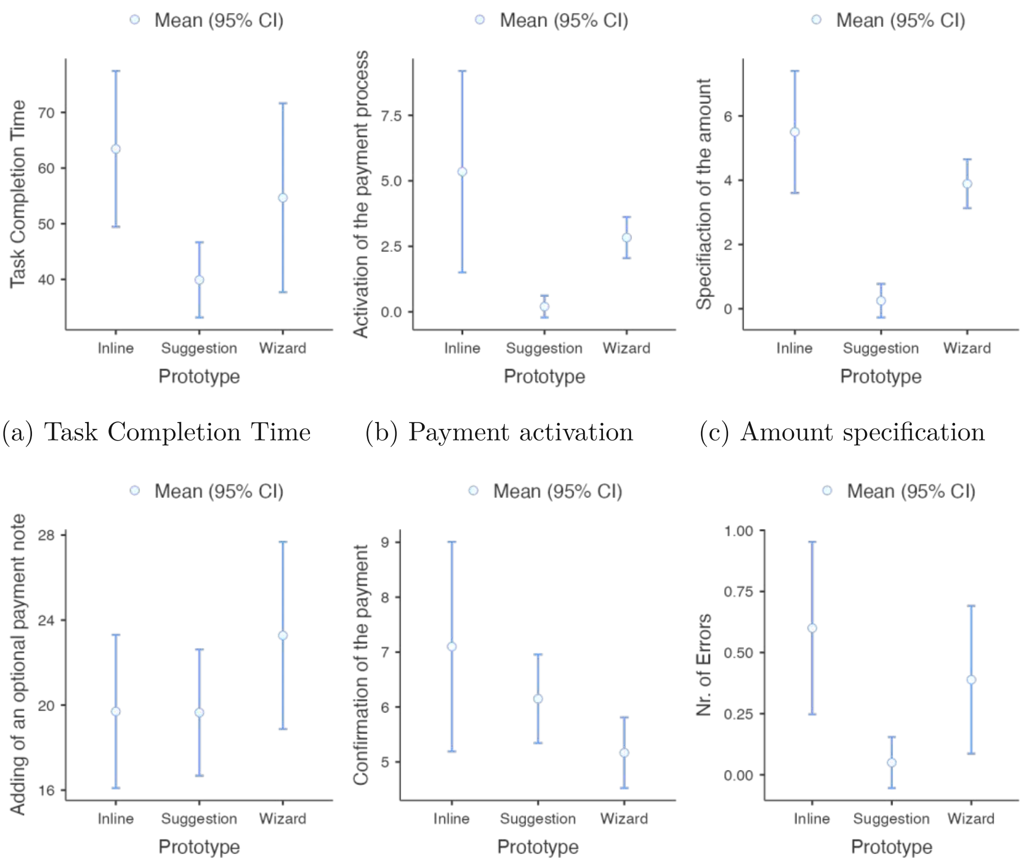
5.2.1 Task 1

Group Descriptives

	Prototype	N	Mean	SD	SE
Task Completion Time	Inline	20	63.4500	29.938	6.6943
	Suggestion	20	39.9000	14.451	3.2313
	Wizard	18	54.6667	34.135	8.0456
Activation of the payment process	Inline	20	5.3500	8.216	1.8372
	Suggestion	20	0.2000	0.894	0.2000
	Wizard	18	2.8333	1.581	0.3727
Specifiaction of the amount	Inline	20	5.5000	4.059	0.9076
	Suggestion	20	0.2500	1.118	0.2500
	Wizard	18	3.8889	1.530	0.3605
Adding of an optional payment note	Inline	20	19.7000	7.706	1.7231
	Suggestion	20	19.6500	6.343	1.4184
	Wizard	18	23.2778	8.857	2.0876
Confirmation of the payment	Inline	20	7.1000	4.077	0.9116
	Suggestion	20	6.1500	1.725	0.3858
	Wizard	18	5.1667	1.295	0.3052
Nr. of Errors	Inline	20	0.6000	0.754	0.1686
	Suggestion	20	0.0500	0.224	0.0500
	Wizard	18	0.3889	0.608	0.1432

Figure 5.2: Descriptive statistics (Task 1)

At first, descriptive statistics were used to summarise the collected data of the six metrics (see Figure 5.2). Additionally, descriptive plots were created for each metric (see Figure 5.3). Each Metric was evaluated separately using ANOVA (see Figure 5.4) including



(a) Task Completion Time (b) Payment activation (c) Amount specification

(d) Payment specification (e) Confirmation (f) Nr. of Errors

Figure 5.3: Descriptive plots (Task 1)

One-Way ANOVA (Fisher's)

	F	df1	df2	p
Task Completion Time	3.81	2	55	0.028 *
Activation of the payment process	5.44	2	55	0.007 *
Specifiaction of the amount	21.06	2	55	< .001 *
Adding of an optional payment note	1.37	2	55	0.262
Confirmation of the payment	2.43	2	55	0.097
Nr. of Errors	4.69	2	55	0.013 *

Figure 5.4: ANOVA (Task 1)

Tukey post-hoc tests, to identify which metrics caused significant differences between the prototypes.

In the following subsections, the six considered metrics will be evaluated separately.

Task Completion Time

The lowest mean value of the task completion time for Task 1 achieved the “Suggestion” prototype with 39.9 seconds. This prototype was followed by the “Wizard” prototype with a mean value of 54.7 seconds. The prototype with the highest mean value was the “Inline” prototype with 63.5 seconds. The conducted ANOVA test resulted a p-value of 0.028. This p-value indicates that the task completion time contains significant differences between the tested prototypes.

Tukey Post-Hoc Test – Task Completion Time

		Inline	Suggestion	Wizard
Inline	p-value	—	0.023 *	0.585
Suggestion	p-value		—	0.226
Wizard	p-value			—

Figure 5.5: Tukey post-hoc test (Task completion time)

To determine which prototype differs significantly, the Tukey post-hoc test was conducted (see Figure 5.5). The result of the undertaken post-test indicates that the “Suggestion” prototype differs significantly from the “Inline” prototype (p-value = 0.023). The p-value between the “Suggestion” prototype and the “Wizard” prototype is 0.226. Between the “Wizard“ and “Inline“ prototype the p-value is 0.585. These two p-values do not indicate significant differences between the “Suggestion“ prototype and the “Wizard“ prototype and between the “Wizard“ prototype and the “Inline“ prototype.

Activation of the payment process

The “Suggestion” prototype reached the lowest mean value for the time required to activate the payment process with 0.2 seconds. For comparisons, the “Wizard” prototype achieved for this activity a mean value of 2.8 seconds. The highest mean value achieved the “Inline“ prototype with 5.4 seconds. The conducted ANOVA test indicated significant differences between the prototypes for the payment activation time (p-value = 0.007). The Tukey post-test revealed significant differences between the “Suggestion” prototype and the “Wizard” prototype (p-value = 0.005) concerning this activity (see Figure 5.6). But the p-value did not indicate significant differences between the “Suggestion” prototype and the “Wizard” prototype (p-value = 0.237) and between the “Inline“ prototype and the “Wizard“ prototype (p-value = 0.267). However, the “Suggestion” prototype has a

14 times lower mean value for the activation of the payment process than the "Wizard" prototypes because of the automatic payment activation, and this indicates that the "Suggestion" prototype caused a reduction of the payment activation process in Task 1.

Tukey Post-Hoc Test – Activation of the payment process

		Inline	Suggestion	Wizard
Inline	p-value	—	0.005 *	0.267
Suggestion	p-value		—	0.237
Wizard	p-value			—

Figure 5.6: Tukey post hoc test (Activation of the payment process)

Specification of the amount

The „Suggestion" prototype reached the lowest mean value for the activity to specify a transaction amount with 0.3 seconds. Followed by the "Wizard" prototype with a mean value of 3.9 seconds. The prototype with the highest mean value was the "Inline" Prototype (5.5 seconds). The conducted ANOVA test indicated significant differences between the three prototypes (p-value < 0.001). The Tukey post-test of this data highlighted significant differences between the "Suggestion" prototype and "Wizard" prototype and between the "Suggestion" prototype and the "Inline" prototype (p-value $<$ 0.001) (see Figure 5.7). The differences between the "Inline" and the "Wizard" prototype are not significant (p-value $= 0.150$). The result highlights that the "Suggestion" prototype requires a significantly shorter time to define the payment amount (13 times faster than the "Wizard" prototypes).

Tukey Post-Hoc Test – Specifiaction of the amount

		Inline	Suggestion	Wizard
Inline	p-value	—	< .001*	0.150
Suggestion	p-value		—	< .001*
Wizard	p-value			—

Figure 5.7: Tukey post-hoc test (Specification of the amount)

Adding of an optional payment note

The shortest mean value for the time required to add an optional payment note achieved the "Suggestion" and the "Inline" prototypes with 19.7 seconds. The "Wizard" prototype followed this mean value with 23.3 seconds. The conducted ANOVA test on this metric did not indicate significant differences between the prototypes (p-value = 0.262) and because of this, no further post hoc test was conducted.

Confirmation of the payment

The mean value for the time required to confirm a payment did not vary much between the prototypes. The "Wizard" prototype achieved the lowest confirmation time with 5.2 seconds. The "Suggestion" prototype reached a mean of 6.2 seconds, and the slowest prototype was the "Inline" prototype with 7.1 seconds. The ANOVA test indicated no significant differences between the prototypes (p-value = 0.097) and because of this, no further post hoc test was conducted.

Number of Errors

The lowest mean value concerning the number of errors per task achieved the "Suggestion" prototype with 0.1 errors. Followed by the "Inline" prototype with the mean value of 0.4. The prototype with the highest mean value achieved the "Inline" prototype with 0.6 errors per task. The performed ANOVA test resulted a p-value of 0.013, indicating significant differences between the tested prototypes. The Tukey post-test highlighted a p-value of 0.01 between the "Suggestion" prototype and the "Inline" prototype (see Figure 5.8). This result indicates that participants using the "Suggestion" prototype conducted tasks with significantly fewer errors than the "Inline" prototype. The p-value between the "Wizard" prototype and the "Suggestion" prototype is 0.172 and between the "Wizard" prototype and the "Inline" prototype 0.497. These results do not provide evidence for significant differences. However, as the calculated mean values of the descriptive statistical analysis show, participants using the "Suggestion" prototype conducted the task with fewer errors than the other two prototypes (3.5 times fewer errors than using the "Inline" prototype).

Tukey Post-Hoc Test – Nr. of Errors

		Inline	Suggestion	Wizard
Inline	p-value	—	0.010 *	0.497
Suggestion	p-value		—	0.172
Wizard	p-value			—

Figure 5.8: Tukey post-hoc test (Number of errors)

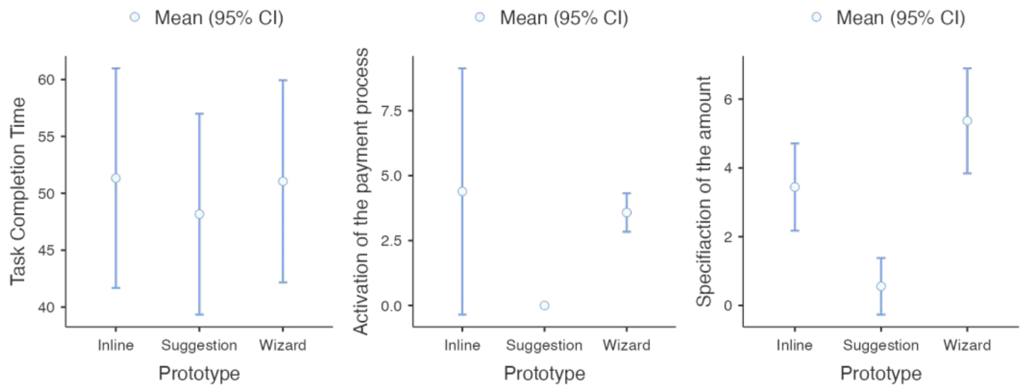
5.2.2 Task 2

For the evaluation of Task 2, the same metrics as for Task 1 were considered. At first, descriptive statistics (see Figure 5.9) were conducted. Additionally, descriptive plots were

Group Descriptives

	Prototype	N	Mean	SD	SE
Task Completion Time	Inline	18	51.333	19.409	4.5747
	Suggestion	18	48.167	17.761	4.1862
	Wizard	19	51.053	18.441	4.2305
Activation of the payment process	Inline	18	4.389	9.531	2.2465
	Suggestion	18	0.000	0.000	0.0000
	Wizard	19	3.579	1.539	0.3531
Specifiaction of the amount	Inline	18	3.444	2.549	0.6008
	Suggestion	18	0.556	1.653	0.3896
	Wizard	19	5.368	3.166	0.7263
Adding of an optional payment note	Inline	18	17.833	8.813	2.0773
	Suggestion	18	22.444	12.339	2.9084
	Wizard	19	16.263	8.962	2.0560
Confirmation of the payment	Inline	18	5.278	1.526	0.3598
	Suggestion	18	7.222	2.981	0.7027
	Wizard	19	5.789	3.011	0.6907
Nr. of Errors	Inline	18	0.722	0.669	0.1577
	Suggestion	18	0.222	0.428	0.1008
	Wizard	19	0.158	0.375	0.0859

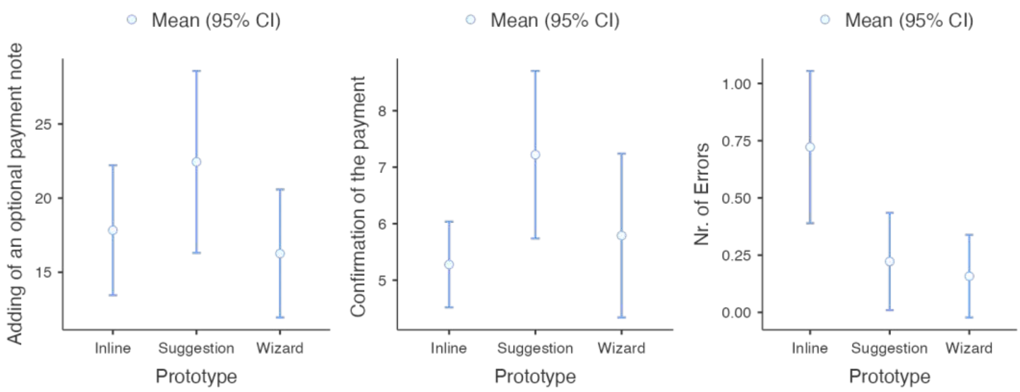
Figure 5.9: Descriptive statistics (Task 2)



(a) Task Completion Time (b) Payment activation (c) Amount specification

Figure 5.10: Descriptive plots (Task 2)a

created for each metric (see Figures 5.10 and 5.11). After this, each Metric was evaluated separately, conducting ANOVA (see Figure 5.12) and Tukey post-hoc tests.



(a) Payment specification (b) Confirmation (c) Nr. of Errors

Figure 5.11: Descriptive plots (Task 2)b

One-Way ANOVA (Fisher's)

	F	df1	df2	p
Task Completion Time	0.162	2	52	0.851
Activation of the payment process	3.231	2	52	0.048 *
Specifiaction of the amount	16.666	2	52	< .001 *
Adding of an optional payment note	1.836	2	52	0.170
Confirmation of the payment	2.694	2	52	0.077
Nr. of Errors	6.827	2	52	0.002 *

Figure 5.12: ANOVA (Task 2)

Task Completion Time

The mean values of the task completion time do not vary much between the three different prototypes in Task 2. The lowest mean for the task completion time achieved the “Suggestion” prototype with 48.2 seconds. Followed by the “Wizard” prototype with 51.1 seconds. The “Inline” prototype reached a slightly higher mean than the “Wizard” prototype with 51.3 seconds. The conducted ANOVA test resulted a p-value of 0.851 for the task completion time and because of this, no further post hoc test was conducted.

Activation of the payment process

The “Suggestion” prototype reached the lowest mean value with 0 seconds to activate the payment process, because detected payment intents automatically initiate the payment process. The prototype with the second-lowest mean value was the "Wizard" prototype with 3.6 seconds. The mean value of the "Inline" prototype was 4.4 seconds. The conducted ANOVA test resulted a p-value of 0.048 (indicating significant differences).

Tukey Post-Hoc Test – Activation of the payment process

		Inline	Suggestion	Wizard
Inline	p-value	—	0.053	0.896
Suggestion	p-value		—	0.130
Wizard	p-value			—

Figure 5.13: Tukey post hoc test(Activation of the payment process)

Additionally a Tukey post-hoc test was conducted (see Figure 5.13). The p-value between the "Suggestion" prototype and the "Inline" prototype is 0.053, and the p-value between the "Suggestion" prototype and the "Wizard" prototype is 0.13. All three p-values are not < 0.5, and because of this, the Tukey post-hoc test did not indicate significant differences.

Specification of the amount

Participants using the "Suggestion" prototype reached the lowest mean value for the time necessary to specify the payment amount with 0.6 seconds. In comparison, participants using the "Inline" prototype required 3.4 seconds. The highest mean value reached the "Wizard" prototype with 5.4 seconds. The conducted ANOVA test resulted significant differences between the three tested prototypes (p-value < 0.001). The Tukey post-test resulted two significant p-values for this metric (see Figure 5.14). The p-value between the "Suggestion" prototype and the "Inline" prototype is 0.004, and the p-value between the "Suggestion" prototype and the "Wizard" prototype is < 0.001. However, the p-value between the "Wizard" prototype and the "Inline" did not indicate significant differences (p-value $= 0.896$). These results of the Tukey post-hoc test show that the "Suggestion" prototype provided a significantly lower time for the specification of the payment amount than the other prototypes because in this prototype the system automatically defines the payment amount.

Tukey Post-Hoc Test – Specifiaction of the amount

		Inline	Suggestion	Wizard
Inline	p-value	—	0.004*	0.065
Suggestion	p-value		—	< .001*
Wizard	p-value			—

Figure 5.14: Tukey post hoc test (Specification of the amount)

Adding of an optional payment note

The lowest mean value for the time required to add an optional payment note reached the "Wizard" prototype with 16.3 seconds. Participants using the "Inline" prototype required 1.6 seconds more than using the "Wizard" prototype (17.9 seconds). The prototype with the highest mean value for this activity achieved the "Suggestion" prototype with 22.4 seconds. The conducted ANOVA test provided a p-value of 0.170, and this result indicates no significant differences between the tested prototypes for this activity. Because of this, no further post hoc test was conducted.

Confirmation of the payment

The lowest mean value for the time required to confirm a payment reached the "Inline" prototype with 5.3 seconds. This mean is slightly followed by the "Wizard" prototype with 5.8 seconds. The "Suggestion" prototype reached the highest mean value for this activity with 7.2 seconds. ANOVA conducted for this activity resulted a p-value of 0.077. This p-value missed the threshold for the indication of significant differences between the prototypes and because of this, no further post hoc test was conducted.

Number of Errors

The lowest mean value for the number of errors per task achieved the "Wizard" prototype, and the "Suggestion" prototype with 0.2 errors per task. The prototype with the highest mean value completed the "Inline" prototype with 0.7 errors per task.

Tukey Post-Hoc Test – Nr. of Errors

		Inline	Suggestion	Wizard
Inline	p-value	—	0.012*	0.004*
Suggestion	p-value		—	0.921
Wizard	p-value			—

Figure 5.15: Tukey post-hoc test (Number of errors)

The conducted ANOVA test resulted a p-value of 0.002. This p-value indicates significant differences between the three prototypes concerning the number of errors. The Tukey post-hoc test resulted p-values that indicate significant differences (see Figure 5.15). The p-value between the "Inline" prototype and the "Suggestion" prototype is 0.012, and between the "Inline" and the "Wizard" prototype, the p-value is 0.004. The p-value between the "Wizard" and "Suggestion" prototype does not indicate significant differences (p-value = 0.921). The results highlight that users using the "Suggestion" prototype and the "Wizard" prototype perform tasks with fewer errors than using the "Inline" prototype.

5.3 Summary of Efficiency

Metric	Task 1	Task 2
Task completion time	p = 0.028*	p = 0.851
Activation of the payment process	p = 0.007*	p = 0.048*
Specification of the amount	p < 0.001*	p < 0.001*
Adding of an optional payment note	p = 0.262	p = 0.170
Confirmation of the payment	p = 0.097	p = 0.077
Number of errors	p = 0.013*	p = 0.002*

Table 5.2: ANOVA results (Task 1 and Task 2)

In Task 1, the ANOVA test for the task completion time indicates significant differences between the three tested prototypes (p-value = 0.028) (see Table 5.2). The "Suggestion" prototype achieved a significant lower task completion time than the "Inline" prototype. However, the task completion time difference between the "Suggestion" prototype and the "Wizard" prototype is not significant. In Task 2, the task completion time did not vary much between the prototypes. The conducted ANOVA test did not indicate significant differences (p-value = 0.851).

In Task 1, for the activity to activate the payment process, ANOVA detected significant differences between the prototypes (p-value = 0.007). The conducted Tukey post hoc test indicated significant differences between the "Suggestion" prototype and the "Inline" prototype, but not between the "Suggestion" prototype and the "Wizard" prototype. In Task 2, the ANOVA test also provided a p-value that indicates significant differences (p-value = 0.048). The automatic payment suggestion feature of the "Suggestion" reduced the activation of the payment process in both tasks to almost zero seconds.

In Task 1 and Task 2, the time of the activity to specify the payment amount varies significantly between the prototypes (p-values < 0.001). The automatic pre-definition of a detected payment amount in the "Suggestion" prototype reduces the time to specify the payment amount significantly compared to the "Inline" and the "Wizard" prototypes. The "Inline" prototype and the "Wizard" prototype do not vary significantly in both tasks.

For the activity to add an optional payment note, ANOVA did not provide significant differences for Task 1 and Task 2 (p-values > 0.17). To confirm a payment, ANOVA indicated also no significant differences for both tasks (p-values > 0.077).

In Task 1, the ANOVA test for the number of errors indicated significant differences (p-value = 0.013). The Tukey post-hoc test highlighted that participants using the "Wizard" prototype conduct tasks with significantly fewer errors than with the "Inline" prototype. The other p-values of the post-hoc test were not significant. In Task 2, ANOVA also indicated significant differences between the prototypes (p-value = 0.002). Participants using the "Suggestion" prototype and the "Wizard" prototype performed tasks with significantly fewer errors than using the "Inline" prototype.

5.4 Effectiveness

The number of successfully completed tasks was considered to compare the three implemented prototypes on effectiveness. A task was defined as successfully completed if the user transferred the right amount, including the given payment note of the task description. A successfully completed task was given the value"1" and an unsuccessful task the value "0".

All participants transferred the correct payment amount for each task. However, some participants sent the text of the payment note as a separate message before or after the payment. In such cases, the task did count as successfully completed. If the participant forgot to add a payment note or added a wrong note (e.g. a too short payment note), the task counted as unsuccessfully completed.

Because of the differences between Task 1 and Task 2, both conducted tasks were evaluated separately.

5.4.1 Task 1

Group Descriptives

	Prototype	N	Mean	SD	SE
Task Completion	Inline	20	1.000	0.000	0.0000
	Suggestion	20	1.000	0.000	0.0000
	Wizard	20	0.900	0.308	0.0688

Figure 5.16: Descriptive statistics (Task completion)

The mean value of the "Inline" and the "Suggestion" prototype is 1 for Task 1 because all participants passed all tasks with these prototypes (see Figure 5.16). The statistical mean of the "Wizard" prototype is 0.9 because two participants did not pass Task 1 with this prototype.

One-Way ANOVA (Fisher's)

	F	df1	df2	p
Task Completion	2.11	2	57	0.130

Figure 5.17: ANOVA (Task completion)

The conducted ANOVA test with the collected data concerning the successfully completed tasks of Task 1 resulted p-value = 0.13. This p-value indicates no significant differences between the prototypes concerning the successfully completed tasks and because of this, no further post hoc test was conducted. Additionally, a descriptive plot was created (see Figure 5.18).

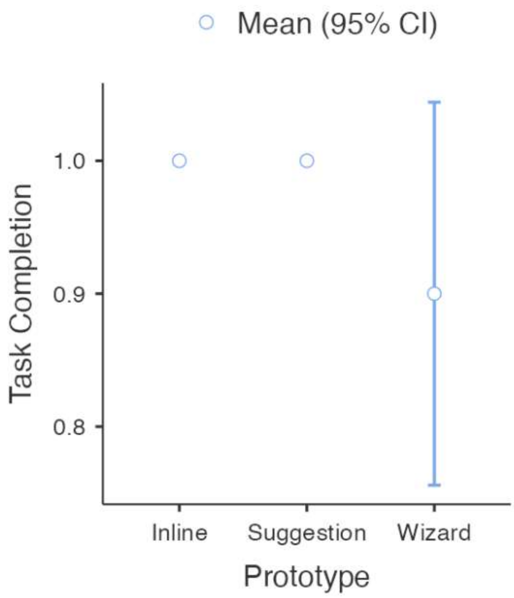


Figure 5.18: Descriptive plot (Task completion)

5.4.2 Task 2

Descriptive statics indicate that the "Wizard" prototype achieved the highest mean value of 0.950 for the number of successfully completed tasks of Task 2 (see Figure 5.19). The "Inline" and "Suggestion" prototypes reached a slightly lower mean value of 0.90.

Group Descriptives

	Prototype	N	Mean	SD	SE
Task Completion	Inline	20	0.900	0.308	0.0688
	Suggestion	20	0.900	0.308	0.0688
	Wizard	20	0.950	0.224	0.0500

Figure 5.19: Descriptive statistics (Task completion)

One-Way ANOVA (Fisher's)

	F	df1	df2	p
Task Completion	0.209	2	57	0.812

Figure 5.20: ANOVA (Task completion)

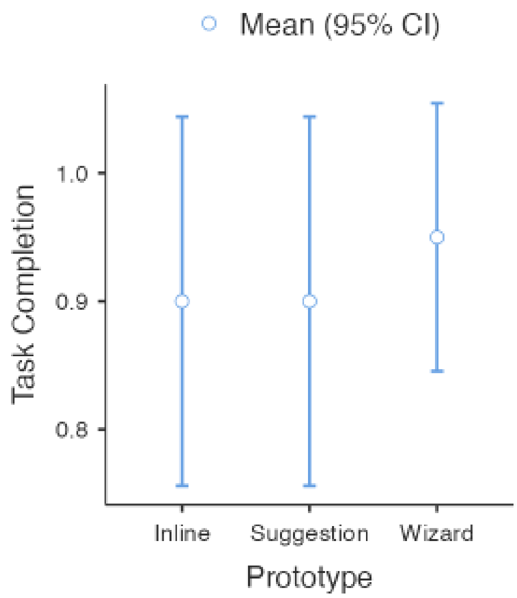


Figure 5.21: Box plot (Task completion)

The ANOVA test concerning the number of successfully completed tasks indicates no significant differences between the prototypes (p-value=0.812) (see Figure 5.20). With the probability of 81.2%, there exist no differences between the tested prototypes. Because of this, no further post hoc test was conducted. Additionally, a descriptive plot was created (see Figure 5.21).

As in task 1, task 2 features no significant differences between the prototypes concerning the number of successfully conducted tasks.

5.5 Summary of Effectiveness

In Task 1, the conducted ANOVA test did not provide significant differences between the prototypes (p-value = 0.13) (see Table 5.3). As in Task 1, in Task 2, ANOVA also resulted a not significant p-value (p-value = 0.812). This result indicates that the different

Metric	Task 1	Task 2
Task Completion	p = 0.130	p = 0.812

Table 5.3: ANOVA results (Task 1 and Task 2)

implemented prototypes do not affect the effectiveness.

5.6 Satisfaction

Group Descriptives

	Prototype	N	Mean	SD	SE
SUS Score	Inline	20	82.3	17.5	3.92
	Suggestion	20	91.8	10.4	2.32
	Wizard	20	91.4	13.3	2.97

Figure 5.22: Descriptive statistics (SUS)

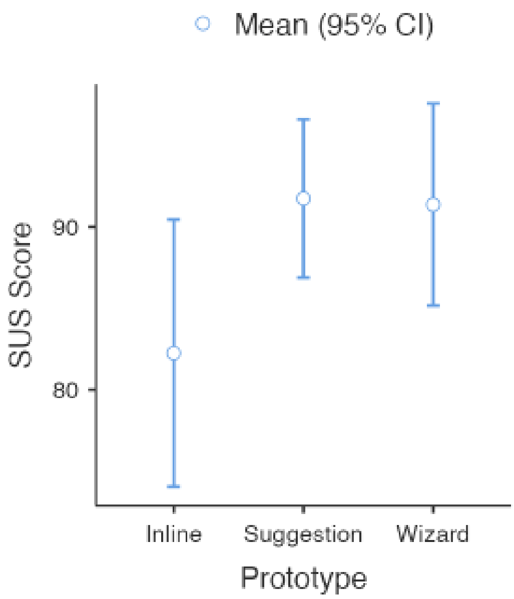


Figure 5.23: Descriptive plot (SUS)

The users' perceived satisfaction with the different prototypes was evaluated using the System Usability Scale (SUS). After the participants had conducted both tasks of a prototype, the participants had to fill in the standardised SUS questionnaire to rate the prototype.

For the evaluation, the SUS scores were calculated and compared. Overall, the statistical mean value of the three tested prototypes have a SUS Score of over 80.3 (see Figure 5.22), which is classified as an excellent ranking [SZN19]. The highest mean value achieved the “Suggestion” prototype with a score of 91.8. This prototype is followed by the “Wizard” prototype with a mean value of 91.4. The “Inline” Solution reached the lowest mean value with a SUS score of 82.3. Additionally, a descriptive plot was created (see Figure 5.23).

One-Way ANOVA (Fisher's)

	F	df1	df2	p
SUS Score	2.94	2	57	0.061

Figure 5.24: ANOVA (SUS)

An ANOVA test was conducted with the collected SUS scores of the three tested prototypes (see Figure 5.24). The conducted ANOVA test achieved a p-value of 0.061. This result indicates that the SUS scores of the three prototypes differ with the probability of 93.9%. The resulting p-value is not < 0.05, and because of this, there are no significant differences between the prototypes and no further post hoc test was conducted.

The results of this statistical analysis indicate that the “Wizard” and “Suggestion” prototype reached a similar SUS ranking. The “Wizard” prototype achieved a 10% lower mean value than the “Wizard” and the “Suggestion” prototype.

5.7 Post-test Interview

After the participant finished conducting all tasks of the usability study, a semi-structured interview was conducted. This interview aimed to get quantitative data about the perceived trust and usability of the tested prototypes.

Question 1: Which of the three tested user interfaces would you prefer to perform P2P payments in the future? For what reason/reasons?

11 of the 20 participants stated that they would prefer to use the “Suggestion” prototype in the future (see Figure 5.25). The most common reasons for this choice were ease of use, faster transaction times, and fewer required steps to perform a payment. Three of the participants described this prototype as convenient to use. Another participant mentioned that he prefers the “Suggestion” prototype because it does not contain pop-ups. 6 of the 20 recruited participants preferred to use the “Wizard” prototype. The main

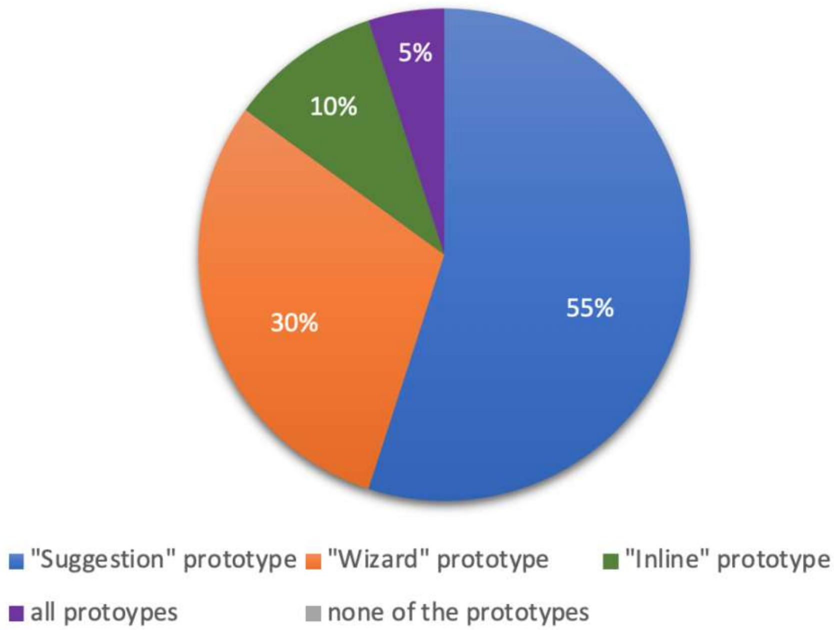


Figure 5.25: Post-test Interview Question 1 result

reason for this choice was the ease of use. Two participants preferred the payment process of the “Wizard” prototype because it takes place in a different view. According to them, such an interaction flow provides more control over payments. “Participant 6“ mentioned that she prefers the “Wizard” prototype because it recalls her on the *George* app (a mobile banking app of an Austrian bank). Two of the participants would like to use the “Inline” prototype in the future. One participant mentioned that, in his opinion, the “Inline“ prototype is the most convenient one to use. Another explained that he likes the “Inline“ prototype because it allows previewing the payment to check the transaction amount before paying. “I think all three prototypes were very easy and not so much different” stated “Participant 1“ and for that reason, she could imagine using each of the tested prototypes.

Question 2: During this usability study, you have tested three different prototyped user interfaces for P2P payments in mobile chat applications. Which of these three user interfaces did you trust the most while performing a payment? For what reason/reasons?

The majority of the participants (11 of the 20 participants) stated that they trust the “Wizard” prototype the most (see Figure 5.26). A popular mentioned reason for this choice was that this prototype seems to be more resistant to errors. Many participants stated that the step-by-step interaction flow and the possibility of manually typing in the amount provide trust. Two of the participants defined ease of use as a reason to trust

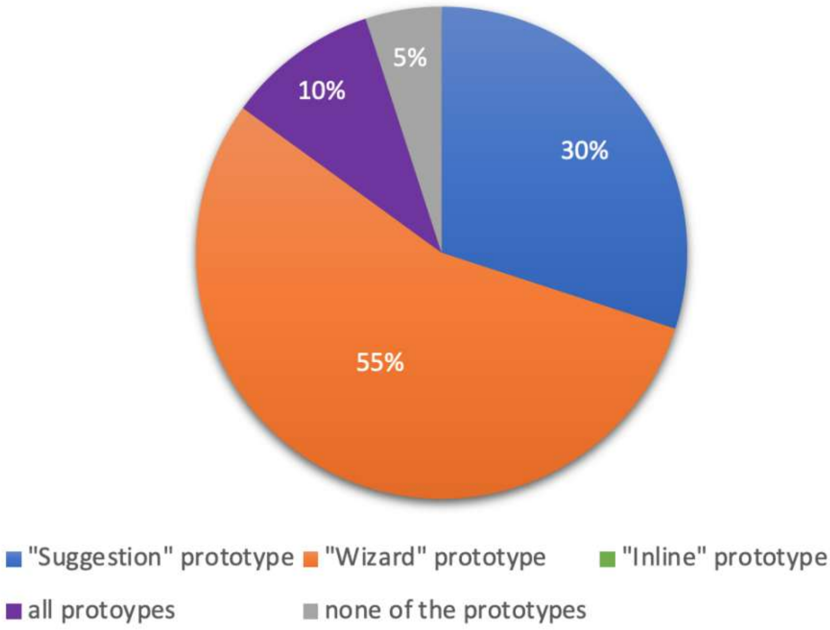


Figure 5.26: Post-test Interview Question 2 result

this prototype the most. 6 of the participants trusted the “Suggestion” approach the most. The majority of the participants did not came up with a reason why they trusted this prototype. One participant mentioned that she trusts the “Suggestion” prototype the most because it felt convenient to use. Another participant stated that it was the easiest prototype to use for him, and because of this, he trusted it the most. Two of the 10 participants trusted all three prototypes. “Participant 15“ mentioned that he does not trust any of the tested prototypes - “Paying with a mobile chat. I don’t know. It feels just unsafe for me”. None of the participants trusted the “Inline“ prototype the most.

Question 3: Would you refuse one or more of the tested prototypes to perform a payment with? If yes, which one/ones? For what reason/reasons?

A quarter of the participants mentioned that they would refuse the “Suggestion” prototype (see Figure 5.27). Most of these participants worried that they could send money unintendedly while chatting. One participant responded that he is concerned about wrong interpretations of the application that could cause annoying false positive suggestions. Another user stated that she perceived this prototype as too complicated and not easy to use. “Participant 20“ explained that the “Suggestion” prototype could harm his privacy while receiving suggestions without consent - “I would be afraid of my privacy. Getting suggestions without my consent to information or things I do not want to see”.

Another quarter of the participants responded that they would refuse the “Inline”

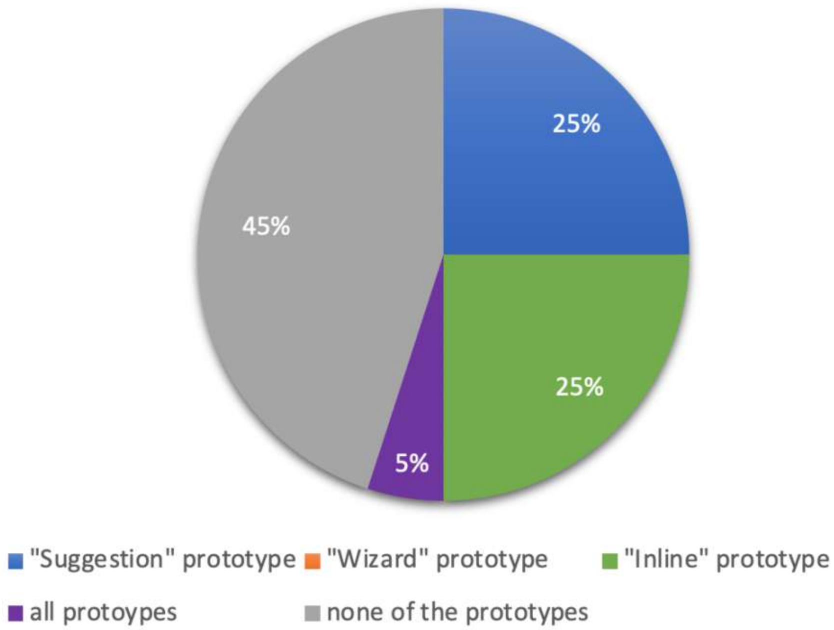


Figure 5.27: Post-test Interview Question 3 result

prototype. Four participants stated that they perceived the “Inline” solution’s user interface as unclear and unnecessarily complicated. Two participants explained that the keypad to define comma amounts was cumbersome to use. None of the participants mentioned refusing the “Wizard“ prototype. 9 of the 20 recruited participants claimed that they would not refuse any of the tested prototypes to send money in mobile chat applications. As a reason, most of the participants stated that they imagine getting used to all of the tested prototypes. Two of the participants answered that they perceived all tested prototypes as good, so they do not have a reason to deny one of them. One recruited participant mentioned that he would not use any of the prototypes if he had a choice because he does not want to make payments in mobile chat applications.

Question 4: One of the used prototypes provides the user with in-line suggestions for P2P payments depending on the user’s input and the conversational context.

Question 4a: How did you experience this feature?

The majority of the participants (16 of the 20 participants) positively experienced this feature (see Figure 5.28). Most of them described it as excellent and easy to use. Two of them mentioned that this feature provides faster payments than the approaches of the other payments. Two participants claimed that it is beneficial to reduce the number of taps to perform a payment. One participant explained that he had trouble recognising the suggestions, and because of this, he recommended an additional pop-up to notify the user about a payment suggestion. According to him, such a pop-up could avoid

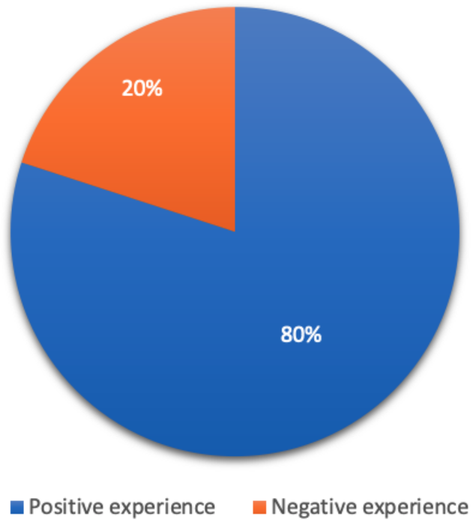


Figure 5.28: Post-test Interview Question 4a result

unintended payments. Four of the participants (20%) had negative experiences with the "Suggestion" solution. One of them stated that he is not interested in suggestions for mobile payments. Another participant mentioned that he is generally scared and uncertain about automatic features - "I am maybe a little bit scared ... but I am always very uncertain if there is automatisation involved in online payment processes".

"Participant 17" explained that it was difficult for him to understand how the suggestions work, and he would have wished for a tutorial before conducting the tasks with this prototype. Another participant mentioned that he would worry about his privacy while receiving suggestions.

Question 4b: Was this feature useful while conducting the tasks?

15 of the 20 asked participants perceived the payment suggestions feature as useful. Two participants (10%) mentioned that this feature could be beneficial for users but not for them. The rest (3 participants) described payment suggestions in mobile messenger as not helpful.

Question 4c: Could you imagine in-line suggestions for other tasks in mobile chat applications? If yes, for which tasks?

10 of the 20 asked participants had no ideas for in-line suggestions besides payments in mobile chat applications. The rest of the participants (10 out of 20 participants) came up with ideas for suggestions. The most popular idea was related to images and GIFs. According to the participants, a mobile messenger application could suggest users to send a photo, GIF or open the photo gallery depending on the conversational context. Two participants mentioned that this feature could be helpful for files stored on the phone or in the cloud. One participant stated that if the application detects the intent to

send a file, the messenger could automatically open *Google Drive*. Another participant had the idea that the messenger could suggest users to create appointments during a conversation. “Participant 19“ said that such a suggestion-based approach could be helpful for forwarding messages to contacts mentioned in a conversation. According to him, this feature could replace groups in some scenarios -”A suggestion to forward a message automatically to more people, yes ... that could replace the step to create a group“. Another participant stated that messengers could provide active suggestions about the already detected keywords for locations, dates and phone numbers.

5.7.1 Summary of the Interview

11 of the 20 participants would prefer to use the novel suggestion-based prototype (“Suggestion“ prototype) in the future. 6 Participants would like to use the “Wizard“ prototype and only two participants the “Inline“ prototype. However, the majority (11 of the 20 participants) mentioned that they trust the “Wizard“ prototype the most. Six participants trusted the “Suggestion“ approach the most, and none trusted the “Inline“ prototype. Furthermore, two participants stated that they trusted all tested prototypes, and one trusted none of the three tested prototypes.

25% (5 out of 20 participants) mentioned that they would refuse to use the “Suggestion“ prototype. Another 25% of the participants explained that they would refuse the “Inline“ prototype. None of the participants stated to refuse the “Wizard“ prototype. The majority of the participants (16 out of 20 participants) positively experience the in-line payment suggestion feature of the “Suggestion“ prototype. Two participants claimed that they negatively experienced the payment suggestions of the “Suggestion“ prototype. 75% (15 out of 20 participants) perceived payment suggestions as useful while conducting the tasks of the usability study. However, 3 participants experienced payment suggestions as not helpful.

Half of the participants came up with their own ideas for in-line suggestions besides payments in mobile chat applications. The most popular idea was related to suggestions for multimedia files (e.g. Stickers, Images, etc.).

5.8 Summary of the Evaluation

In this chapter, the results of the usability study were evaluated using descriptive statistics and statistical analysis. For the evaluation of the three prototypes on efficiency, six metrics were considered. The task completion time for Task 1 was significantly lower for the “Suggestion“ prototype than for the “Inline“ prototype. Participants using the “Suggestion“ prototype were 14.8 seconds faster conducting Task 1 than with the “Wizard“ prototype and 23.6 seconds faster than using the “Inline“ prototype. The inline-suggestion feature caused a significant reduction of the activation of the payment process and specification of the amount. Furthermore, participants using the “Suggestion“ prototype conducted Task 1 with fewer errors than with the other prototypes (3.5 times lower error rate). In Task 2, participants using the “Suggestion“ prototype and the “Wizard“ prototype performed tasks with three times fewer errors.

For the evaluation of effectiveness, the task completion was considered as the metric. The statistical analysis of this metric did not result in significant results, and because of this, the tested prototypes do not affect the effectiveness.

The SUS score was considered for the evaluation of user satisfaction. As for the effectiveness, the statistical analysis did not result in significant differences between the prototypes for this metric.

The qualitative post-test interview identified that 11 of the 20 participants would prefer to use the “Suggestion“ prototype in the future. 30% (6 participants) stated that they prefer the “Wizard“ prototype, and 10% (2 participants) preferred the “Inline“ prototype. Despite the high preference for the “Suggestion“ prototype, the majority of the participants (11 participants) trusted the “Wizard“ prototype the most and only 30% (6 participants) trusted the “Suggestion“ prototype. None of the participants would refuse the “Wizard prototype. However, 25% (5 participants) would refuse the “Suggestion“ prototype. Most of these participants worried that they could send money unintendedly while chatting. 16 of the 20 participants participants positively experienced the automatic payment suggestion feature of the “Suggestion“ prototype and could imagine suggestions for other tasks in mobile messenger.

CHAPTER 6

Discussion

In this section, the defined research questions of this thesis will be answered based on the collected and evaluated quantitative data of the conducted comparative usability study. Furthermore, insights gathered through the semi-structured post-test interview will be discussed.

6.1 RQ1: Is a suggestion-based user interface more efficient than already existing solutions for P2P payments in chat applications?

Efficiency describes the speed and accuracy with which a user performs a task [Wei20]. For the evaluation of this attribute, six metrics were considered. The task completion time, time spent on each of the four identified activities (Activation of the payment process, specification of the amount, adding of an optional payment note, confirmation of the payment) required to perform P2P payments, and the number of errors per task. Because of the differences between Task 1 and Task 2, both tasks were evaluated separately. For the evaluation of effectiveness, descriptive statistics and statistical analysis were used.

Participants using the novel suggestion-based user interface completed Task 1 31.2% faster than with the reimplemented prototype of *WhatsApp/WeChat* and 45.6% faster than with the reimplementation of *iMessage/Facebook Messenger*. In Task 1, the conducted statistical analysis highlighted that participants using the suggestion-based prototype performed tasks significantly faster than using the reimplementation of *iMessage/Facebook Messenger*. However, the statistical analysis did not indicate significant differences between the suggestion-based user interface and the reimplementation of *WhatsApp/WeChat*. In Task2, ANOVA did not identify significant differences for the task completion time between the tested prototypes. The mean value for the task completion time of the novel implemented prototype was 5.8% lower than for the reimplemented user interfaces. The evaluation of this metric indicates that payment suggestions, based on the user's input, reduce the task completion time, which was tested in Task 1. The payment suggestions feature based on the conversational context was tested in Task 2, but the results indicate no improvements for the task completion time.

In Task 1 and Task 2, the mean values for the time required to activate the payment process were between 2.8 to 5.4 seconds for the reimplemented user interfaces of the already existing solutions. The novel suggestion-based prototype reduced the time to activate the payment process to nearly zero seconds (Task 1: 0.2 seconds, Task 2 0.0 seconds). The novel suggestion-based prototype automatically activates the payment process if the system detects a payment intent in the conversational context and the user's input. In the reimplemented prototypes of the existing solutions, the user has to manually activate the payment process, which requires more time.

For both tasks, the mean value of the time required to specify the payment amount was between 3.4 and 5.5 seconds with the reimplemented prototypes of the existing solutions. The conducted ANOVA test indicates that the novel suggestion-based prototype reduced the time required to specify a transaction amount significantly. In Task 1, the mean value for the specification of the amount was 0.3 seconds, and in Task 2, 0.6 seconds. Besides payment intents, the novel suggestion-based prototype detects payment amounts in the conversational context and the user's keyboard input. If the prototype detects a payment intent and a payment amount simultaneously, the system automatically activates the payment process with the detected payment amount already pre-defined. In the reimplementation of *WhatsApp/WeChat* and reimplementation of *iMessage/Facebook Messenger*, the user must manually define the payment amount. For that reason, the novel prototype achieved a significantly lower time to specify the payment amount.

In Task 1, the novel suggestion-based user interface and the reimplemented user interface of *iMessage/Facebook Messenger* achieved a similar mean value for the time required to add an optional payment note. The conducted ANOVA test did not indicate significant differences, but the reimplemented user interface of *WhatsApp/WeChat* achieved a slightly higher mean value. In Task 2, ANOVA also did not indicate significant differences between the three tested prototypes. However, in Task 2, the mean value of the time required to add an optional payment note was with the novel user interface approach 31.5% higher than with the reimplemented user interface of *WhatsApp/WeChat*. In all three tested prototypes, the users need to type the payment note into a text input field during the transaction process. Because of this, the time required for this activity did not vary much between the prototypes.

For the time required to confirm the payment, the statistical analysis did not indicate significant differences between the prototypes in Task 1 and Task 2. In both tasks, the mean value of the time required to confirm the payment was around 5-7 seconds for the tested prototype. The novel prototype achieved in Task 1 a one second higher mean value to perform this activity compared to the reimplementation of *WhatsApp/WeChat*. The reimplementation of *iMessage/Facebook Messenger* achieved the highest mean value for this activity (7.1 seconds) in Task 1. In Task 2, The reimplementation of *iMessage/Facebook Messenger* resulted the lowest time for the confirmation of the payment (5.2 seconds) and the novel suggestion-based the highest mean value (7.2 seconds). The payment confirmation process of the novel prototype was identical to the payment process of the reimplementation of *iMessage/Facebook Messenger*.

The error rate was lower for the novel suggestion-based prototype in Task 1. The novel user interface's error rate was 3.5 times lower than for the reimplemented prototype of *WhatsApp/WeChat* and 6 times lower than for the reimplemented prototype of *iMessage/Facebook Messenger*. In Task 2, the reimplementation of *WhatsApp/WeChat* reached the lowest error rate. The novel prototype achieved a slightly higher error rate than the reimplemented user interface of *WhatsApp/WeChat*, and the conducted ANOVA test did indicate significant differences. The reimplemented prototype of *iMessage/Facebook Messenger* reached a significantly higher error rate than the other two tested prototypes.

The implemented suggestion-based user interface for P2P payments in mobile chat applications improved four of the six evaluated metrics for the user's efficiency compared to the already existing solutions. The statistical analysis of the task completion time for Task 1 indicated significantly lower results for the novel prototype than for the reimplementation of *iMessage/Facebook Messenger*. The task completion time was not significantly lower than for the reimplementation of *WhatsApp/WeChat*. However, the novel prototype achieved a 31.2% lower task completion time compared to the reimplementation of *WhatsApp/WeChat*. In Task 1, suggestions based on the user's keyboard input were tested, and in Task 2, payment suggestions based on the conversation context. In Task 2, ANOVA did not identify significant differences between the three prototypes concerning the task completion time. However, the mean value of the suggestion-based prototype was four seconds lower than for the other reimplemented prototypes. This reduction is because the novel suggestion-based prototype significantly decreases the time required to activate the payment process and to define the payment amount. The suggestion prototype automatically initiates the payment process and defines the payment amount if payment suggestion gets detected. The mean value for the payment activation process in Task 1 was 0.2 seconds, and in Task 2, 0 seconds. Furthermore, the mean value for the specification of the payment amount was in Task 1 0.3 seconds and in Task 2 0.6 seconds. The novel suggestion-based user interface did reduce the time to add an optional payment note and to confirm the payment. Additionally, suggestions based on the user's input reduced the number of errors per task in Task 1 (3.5 times lower than for the reimplementation of *WhatsApp/WeChat* and six times lower than for the reimplantation of *iMessage/Facebook Messenger*). On the other hand, in Task 2, the reimplementation of *WhatsApp/WeChat* and the novel prototype achieved a significantly lower error rate than the reimplementation of *iMessage/Facebook Messenger*.

To sum up, the implemented suggestion-based user interface partly improved the efficiency compared to the reimplemented existing solution. The significant reduction of the payment activation process and definition of the payment amount decreased the task completion time. Suggestions based on the user's input reduced the task completion time and error rate. Suggestions based on the conversational context did not reduce the task completion time and error rate significantly in comparison to the reimplemented solutions.

6.2 RQ2: Is the user's effectiveness higher with a suggestion-based user interface for P2P payments in mobile chat applications than already existing solutions?

Effectiveness is the user's ability to complete a task [Wei20]. This attribute was measured by the number of successfully completed tasks per prototype. Because of the differences between Task 1 and Task 2, both tasks were evaluated separately. For the evaluation of the number of successfully completed tasks, descriptive statistics and statistical analysis were conducted.

Overall, all participants were capable of transferring the right amount to the other participants with all prototypes in all tasks. No or too short payment notes attached to the transactions were reasons for unsuccessfully completed tasks.

All participants successfully conducted Task 1 using the novel suggestion-based user interface for P2P payment and the reimplemented user interface of *iMessage/Facebook Messenger*. Two of the 20 participants (20%) using the reimplementation of *WhatsApp/WeChat* did not successfully complete Task 1. The conducted ANOVA test did not provide a p-value that indicates significant differences between the prototypes. Furthermore, the Tukey post-hoc test did not provide significant p-values.

In Task 2, two of the 20 participants did not successfully conduct the task using the suggestion-based user interface and the reimplemented user interface of *WhatsApp/WeChat*. Only one of the participants did not successfully complete Task 2 using the reimplemented prototype of *iMessage/Facebook Messenger*. As for Task 1, ANOVA and the Tukey post-hoc test did not indicate significant differences between the prototypes.

The implemented suggestion-based user interface for P2P payments did not improve the task completion rate compared to the reimplemented prototypes of the existing solutions. In Task 1, the novel user interface was as efficient as the reimplemented user interface of *WhatsApp/WeChat*. In Task 2, the novel user interface was slightly less efficient than the reimplemented user interface of *WhatsApp/WeChat*. However, the three tested user interfaces are efficient because all participants were capable of transferring the right amount to the other participants. In Task 1, only two of the 20 participants (10%) forgot to attach the payment note using the reimplementation of *WhatsApp/WeChat*. In Task 2, one participant forgot to add the payment note using the reimplementation of *iMessage/Facebook Messenger*. Two participants forgot to add the payment note using the novel suggestion-based prototype and the reimplementation of *WhatsApp* and *WeChat*.

To sum up, the results of the usability study indicate that the user's effectiveness is already high with the reimplemented existing solutions, and because of this, the novel suggestion-based user interface did not further improve the user's effectiveness.

6.3 RQ3: Is the perceived level of satisfaction higher by using a suggestion-based user interface for P2P payments than already existing solutions?

The user's satisfaction is the level of comfort, pleasure, and fulfilment of expectations [Wei20]. It was measured with the standardised SUS questionnaire [Bro95]. During the usability study, the participants had to fill in the SUS questionnaire for each of the three tested prototypes. For the evaluation, the SUS scores were calculated and compared using descriptive statistics and statistical analysis.

The implemented prototype of a suggestion-based user interface for P2P payments in mobile chat applications did not improve the SUS score compared to the already existing reimplemented solutions. The conducted ANOVA test and the Tukey post-hoc test did not provide significant p-values. The novel suggestion-based prototype, and the reimplemented user interface of *WhatsApp/WeChat* achieved a similar high SUS score. The reimplementation of *iMessage/Facebook Messenger* reached a 10% lower SUS score than the novel user interface and the reimplementation of *WhatsApp/WeChat*. However, the novel approach and the two reimplemented user interfaces of the existing solutions achieved the highest SUS rating (SUS score > 82.3) [SZN19]. This result indicates that the current user interfaces for P2P payments provide the user with a high level of satisfaction. Because of this, the novel user interface that provides the user suggestions based on the conversational context and the user's keyboard input does not improve the user's satisfaction.

6.4 Qualitative Findings

During the post-test interview, the majority of the participants (11 out of the 20 recruited participants) mentioned that they would prefer to use the novel suggestions-based user interface instead of the reimplemented user interfaces of the already existing solutions. The main reasons for this choice were the ease of use, faster transactions times, and fewer steps to perform a payment. These results highlight that the novel suggestion-based approach caused higher perceived usability for the majority of the participants than the reimplemented user interfaces.

However, 55% of the participants stated that they trusted the reimplemented user interface of *WhatsApp/WeChat* the most. Mentioned reasons for this choice were the possibility to manually type in the transaction amount and the step-by-step interaction flow. Only 6 of the participants stated that they trust the novel suggestion-based prototype over the other tested prototypes. Most participants could not come up with a reason for this choice. Even 25% of the participants (5 participants) considered refusing to use the novel suggestion based-prototype. The most common reason for this was the fear of sending unintended money using this prototype. These results indicate that a step-by-step interaction flow in an external window increases the trust for payments in mobile chat applications. The initial goal of this thesis was to reduce the necessary steps to perform a payment in order to reduce the transaction time. However, the results of the post-test interview indicate that a faster transaction process may reduce the perceived level of trust. Only two of the 5 participants that refused the suggestion-based prototype mentioned concerns about the payment suggestions. One of them stated that he is afraid of wrong interpretations, and the other participant explained that payment suggestions might harm his privacy. These findings indicate that most participants were unaware or did not mind that a system that provides suggestions based on the conversation context and the user's input needs to analyse their private chat conversations.

25% of the participants (5 participants) would refuse the reimplementation of *iMessage/Facebook Messenger*. In comparison to the reimplementation of *WhatsApp/WeChat*, the step-by-step transaction process in the reimplementation of *iMessage/Facebook Messenger* takes place in the conversation window. The majority of the participants perceived the reimplementation of *iMessage/Facebook Messenger* as unclear and unnecessarily complicated. Only one of the recruited participants mentioned that he would not trust and refuse all three tested prototypes because he does not want to make payments in mobile chat applications at all.

Overall, the majority of the participants (16 our of 20 participants) positively experienced payment suggestions in mobile chat applications. 15 of the participants described this feature as helpful, and they could imagine suggestions for other tasks in mobile messenger. Despite perceived usability improvements, the faster transaction process of the novel suggestion-based prototype may raise trust issues.

CHAPTER 7

Conclusion and Future Work

In this thesis, a suggestion-based user interface prototype for P2P payments in mobile chat applications was designed and implemented. A suggestion-based user interface aims to allow faster transactions by reducing the necessary steps to perform payments and to increase the overall usability of P2P payments in mobile messenger. The suggestions of this novel user interface are based on the conversational context and the user's input.

For the evaluation of the novel prototype on usability in terms of efficiency, effectiveness, and satisfaction, a comparative usability study was conducted to compare the novel prototype with the reimplemented user interfaces of already existing messenger that support P2P payments.

The novel suggestion-based user interface partly improved the efficiency of P2P payment in mobile chat applications. Payment suggestions based on the user's input reduced the task completion time by 31.2% and reduced the number of errors by 3.5 times. Suggestions based on the conversational context did not significantly reduce the time of the transaction process and error rate.

A payment suggestion of the novel prototype automatically activates the payment process if a payment intent gets detected during the chat conversation. If the payment intent contains an amount, the payment amount gets automatically pre-defined. Because of this, suggestions based on the user's input and the conversational context significantly reduced the time required to activate the payment process and to define the transaction amount to nearly zero seconds.

The reimplemented prototypes of the already existing solutions provide the user with high effectiveness. All participants were capable of transferring the right amount with all prototypes in all tasks. However, a few participants forgot to attach a payment note during the payment process, and this caused a reduction of the task completion rate. The task completion rate was considered for the evaluation of effectiveness, but the conducted statistical analysis did not indicate significant differences between the prototypes in terms of the task completion rate.

As for effectiveness, the novel suggestion-based prototype did not improve the user satisfaction. For the evaluation of the user's satisfaction, the SUS score was considered. All tested prototypes achieved a high SUS score, and the statistical analysis did not indicate significant differences between the prototypes.

However, the perceived usability was higher for the majority of the participants while using the novel suggestion-based user interface. The main reasons for this choice were the ease of use, faster transaction times and fewer steps to perform a payment. 15 of the 20 recruited participants described payment suggestions in mobile messenger as useful and could imagine suggestions for tasks besides payments.

However, the novel suggestion-based prototype may raise trust issues. During the post-test interview, the majority of the participants (11 out of 20 participants) stated that they trusted the reimplemented user interface of *WhatsApp*/*WeChat* more than the novel suggestion-based approach. 25% of the participants (5 participants) considered refusing the suggestion-based prototype. The main reason for the refusal of the novel prototype was the fear of unintended payments. The findings indicate that a step-by-step interaction flow like in *WhatsApp* or *WeChat* may increase the perceived trust during a transaction in mobile chat applications. Only one participant mentioned during the interview that suggestions in a mobile chat application harm his privacy.

7.1 Future Work

The implemented suggestion-based prototype for P2P payments in mobile chat applications is currently not a fully functional chat application. The current prototype supports only two-way text communication between two participants. As future work, the prototype could be extended into a fully functional chat application. The functional chat application could then be used to conduct a field study. A field study could provide more knowledge about how participants perceive payment suggestions while chatting in their natural environment. Additionally, the underlying language analysis that detects payment intents should be improved to ensure a high level of usability and to avoid false-positive suggestions during a field study.

Suggestion-based approaches for mobile chat applications have already shown significant usability improvements for the suggestion of images, stickers, GIFs, replies, and message-related tasks. Furthermore, 15 of the 20 recruited participants perceived the payment suggestion feature of the novel user interfaces as useful. Two participants mentioned that they could imagine suggestions for files stored on the phone or in the personal cloud. A suggestion-based user interface that suggests users to send files stored on the phone or in the cloud could be implemented and evaluated for future work.

The post-test interview results indicate that the majority (11 of 20 the participants) trusted the reimplemented user interface of *WhatsApp*/*WeChat* over the novel suggestion-based user interface for P2P payments. The findings of this thesis indicate that the short transaction process of the novel suggestion-based prototype may raise the fear of unintended payments. The majority of the participants trusted the reimplementation of *WhatsApp*/*WeChat* because of the step-by step-interaction flow. For that reason, as future work, the factors that cause trust issues in mobile P2P applications could be examined. Furthermore, one participant mentioned during the post-test interview that a suggestion-based user interface harms his privacy. As future work, privacy issues of suggestion-based user interfaces could be analysed, and a suggestion-based user interface that protects the user's privacy could be implemented.

List of Figures

List of Tables

Acronyms

ANOVA Analysis of variance. 4, 67, 68, 71, 73–80, 82, 83, 85, 93–97

API Application Programming Interface. 12

CARS Context-Aware Recommender Systems. 7, 9, 10, 20

CRS Conversational Recommender Systems. 7, 10, 11, 20

GPS Global Positioning System. 9, 17

GSM Global System for Mobile Communications. 9

P2P Person-to-Person. xiii, xv, xvi, 1–5, 7, 8, 20–23, 25, 27, 29, 31, 35, 40, 43, 45, 46, 48, 52, 59, 65, 67, 85, 86, 88, 93, 95–97, 99–101

POI Point of Interest. 9

QOC Question Option Criteria. 32

RS Recommender Systems. 7–9, 20

SUS System Usability Scale. 3, 4, 64, 66, 84, 85, 91, 97, 100

Bibliography

[BCB04] Viv Bewick, Liz Cheek, and Jonathan Ball. Statistics review 9: one-way analysis of variance. *Critical Care*, 8:130–136, 2004.

[BMB17] Imen Ben Sassi, Sehl Mellouli, and Sadok Ben Yahia. Context-aware recommender systems in mobile environment: On the road of future research. *Information Systems*, 72:27–61, 2017.

[Bro95] John Brooke. Sus: A 'quick and dirty' usability scale. In *Usability Evaluation In Industry*, pages 207–212. CRC Press, 1995.

[CAGLGC+18] Sergio Caro-Alvaro, Eva Garcia-Lopez, Antonio Garcia-Cabot, Luis de Marcos, and Jose-Javier Martinez-Herraiz. Identifying usability issues in instant messaging apps on ios and android platforms. *Mobile Information Systems*, 2018:1–19, 2018.

[Cha03] Elizabeth Charters. The use of think-aloud methods in qualitative research an introduction to think-aloud methods. *Brock Education Journal*, 12(2), 2003.

[CXDH19] Fanglin Chen, Kewei Xia, Karan Dhabalia, and Jason I. Hong. Messageontap: A suggestive interface to facilitate messaging-related tasks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–14, New York, USA, 2019. ACM.

[JMCC21] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. A survey on conversational recommender systems. *ACM Computing Surveys*, 54(5):1–36, 2021.

[Juh16] Moosa Juhra Muhammad. How smart is the "smart reply" feature of google allo: from users' viewpoint. In *CHI '16: Proceedings of the 2016*, pages 1–9. 2016.

[KGH+20] Joongyum Kim, Taesik Gong, Kyungsik Han, Juho Kim, JeongGil Ko, and Sung-Ju Lee. Messaging beyond texts with real-time image suggestions. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 1–12, New York, USA, 2020. ACM.

[KRPL19] Kang-Min Kim, Woo-Jong Ryu, Jun-Hyung Park, and SangKeun Lee. mechat: In-device personal assistant for conversational photo sharing. *IEEE Internet Computing*, 23(2):23–30, 2019.

[LHM+20] Abhishek Laddha, Mohamed Hanoosh, Debdoot Mukherjee, Parth Patwa, and Ankur Narang. Understanding chat messages for sticker recommendation in messaging apps. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:13156–13163, 2020.

[LRVRLC21] Juan Lara-Rubio, Angel Villarejo-Ramos, and Francisco Liébana-Cabanillas. Explanatory and predictive model of the adoption of p2p payment systems. *Behaviour & Information Technology*, 40(6):528–541, 2021.

[LW20] Kuan-Yu Lin and Travis K. Wang, Yi-Ting abd Huang. Exploring the antecedents of mobile payment service usage. *Online Information Review*, 44(1):299–318, 2020.

[MBS93] Allan MacLean, Victoria Bellotti, and Simon Shum. Developing the design space with design space analysis. *Computers, Communication and Usability: Design Issues, Research and Methods for Integrated Services*, pages 197–219, 1993.

[MH17] Christopher D. Mellinger and Thomas A. Hanson. *Quantitative research methods in translation and interpreting studies.* Routledge, London, 2017.

[MHE19] Mahdi H. Miraz and Marie Haikel-Elsabeh. Analysis of users' behaviour and adoption trends of social media payment platforms. In *2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, pages 197–202. IEEE, 2019.

[Nap19] Marco L. Napoli. *Beginning Flutter: A Hands On Guide to App Development.* Wiley, n.p, 2019.

[PA19] Laxmi Pandey and Ahmed Sabbir Arif. Context-sensitive app prediction on the suggestion bar of a mobile keyboard. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia*, MUM '19, New York, USA, 2019. ACM.

[RC08] Jeffrey Rubin and Dana Chisnell. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests.* Wiley, n.p, 2 edition, 2008.

[RD19] Shaina Raza and Chen Ding. Progress in context-aware recommender systems — an overview. *Computer Science Review*, 31:84–97, 2019.

[SK12] Basudeo Singh and Jasmine K.S. Comparative study on various methods and types of mobile payment system. In *2012 International Conference* *on Advances in Mobile Network, Communication and Its Applications*, pages 143–148. IEEE, 2012.

[SZN19] Ginanjar Wiro Sasmito, La Ode Mohamad Zulfiqar, and M. Nishom. Usability testing based on system usability scale and net promoter score. In *2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pages 540–545, n.p, 2019. IEEE.

[Wei20] Pawel Weichbroth. Usability of mobile applications: A systematic literature study. *IEEE Access*, 8:55563–55577, 2020.

[YFR+18] Ning Ye, Ariel Fuxman, Vivek Ramavajjala, Sergey Nazarov, J. Patrick McGregor, and Sujith Ravi. Photoreply: Automatically suggesting conversational responses to photos. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, pages 1893–1899, Lyon, France, 2018. ACM Press.

Appendix

Appendix A: Usability Study Chat Script for Task 2

Task 2 Scenario 1

- Hello
- I am on my way home. Could you borrow me please 2.49 euro for my train ticket?

Task 2 Scenario 2

- Hi
- Currently, I am doing the reservation for tonight's diner at your favorite restaurant. Could you send me 8.80 € for the reservation?

Task 2 Scenario 3

- Hey
- Our last museum visit was great. By the way, you still owe me 9 euros for the entry.

Appendix B: Consent Form

TECHNISCHE UNIVERSITÄT WIEN Vienna | Austria

Einverständniserklärung

Ich, ____, erkläre mich bereit, dass die gesammelten Daten und Antworten, die während der Usability Study gesammelt werden, zur Verfügung zu stellen.

Die Usability Study wird zum Testen eines Prototypen verwendet, der im Rahmen einer Diplomarbeit an der Technischen Universität entwickelt wurde. Während der Usability Study werden Daten über die Effektivität, Effizienz und Benutzer Zufriedenheit gesammelt.

Der Diplomant unterliegt der Schweigepflicht und ist dem Datengeheimnis verpflichtet. Damit Ihre Daten gespeichert und ausgewertet werden dürfen, benötigen wir im Rahmen der Datenschutzgrundverordnung Ihre ausdrückliche Einwilligung.

Diese Zustimmung ist freiwillig. Sie können diese Zustimmung jederzeit verweigern oder widerrufen. Sie können auch Antworten zu einzelnen Fragen verweigern. Ein solcher Widerruf oder eine solche Verweigerung hat für sie keine negativen Folgen.

Ihr Widerruf der Zustimmung kann per E-Mail an die angegebenen Kontaktpersonen übermittelt werden. Daraufhin werden alle Ihre personenbezogenen Daten umgehend gelöscht.

Ich stimme ausdrücklich zu, dass die im Zuge der Usability Study gewonnenen Informationen im Rahmen der Diplomarbeit anonymisiert verarbeitet werden dürfen.	□ Ja □ Nein

Kontakt und Ansprechperson:

Name	**E-Mail**	**Matrikelnummer**
Felix Walcher	e1427555@student.tuwien.ac.at	01427555

Ich wurde über das Vorhaben und die Ziele, sowie die Modalitäten meiner Teilnahme aufgeklärt. Mir ist der Ablauf bekannt, alle meine Fragen wurden zu meiner Zufriedenheit beantwortet und ich fühle mich ausreichend informiert.

Ich weiß, dass ich mich bei Fragen oder anderen Anliegen jederzeit an eine zuständige Kontaktperson wenden kann.

Ich bestätige meine Zustimmung zur Teilnahme an der Usability Study und zur oben genannten Verarbeitung meiner personenbezogenen Daten mit dieser Unterschrift:

______________________ __________________________________

Ort, Datum Unterschrift

Appendix C: Background Survey

Background Survey

*Required

1. Participant Number: *

Skip to question 2

1. General information

2. 1.1 What is your gender? *

Mark only one oval.

- ◯ female
- ◯ male

3. 1.2 What is your age? *

Mark only one oval.

- ◯ 16-20
- ◯ 21-30
- ◯ 31-40
- ◯ 41-50
- ◯ 51-60
- ◯ 61+

2. Education

4. 2.1 What is your highest degree or level of education? *

Mark only one oval.

- ◯ High School
- ◯ Bachelors's Degree
- ◯ Master's Degree
- ◯ Ph. D.
- ◯ Other: ______

3. Relevant information concerning the study

5. 3.1 Which mobile operating systems are you using? *

Tick all that apply.

- ☐ iOS
- ☐ Android
- Other: ☐ ______

6. 3.2 Which mobile messengers do you use frequently? *

Tick all that apply.

- ☐ WhatsApp
- ☐ iMessage
- ☐ Telegram
- ☐ Signal
- ☐ WeChat
- ☐ Facebook Messenger
- Other: ☐ ______

7. 3.3 How often do you communicate with mobile messengers? *

Mark only one oval.

- ◯ multiple times a day
- ◯ daily
- ◯ weekly
- ◯ every few weeks
- ◯ never

8. 3.4 Do you use online banking? *

Mark only one oval.

- ◯ yes
- ◯ no

9. 3.5 How often do you perform transactions using mobile banking apps? *

Mark only one oval.

- ◯ multiple times a day
- ◯ daily
- ◯ weekly
- ◯ every few weeks
- ◯ never

10. 3.6 Have you ever transferred money directly with a mobile messenger app? *

Mark only one oval.

- ◯ yes *Skip to question 11*
- ◯ no

11. Specify the mobile messenger app you have used to transfer money: *

This content is neither created nor endorsed by Google.

Google Forms