Software Requirements Specification

für

OBS-Plugin

Version 1.0

Prepared by Dominik Deller, Julia Grötsch, Khang Ho, Philipp Weber

Universität Regensburg, Lehrstuhl für Medieninformatik

11.06.2018

Inhalt

Table of Contents ii

Revision History ii

1. Einführung 1
   1. Zweck 1
   2. Dokumentformat 1
   3. Leserzielgruppe & Aufbau 1
   4. Produktumfang 1
   5. Referenzen 1
2. Überblick 2
   1. Produktumriss 2
   2. Hauptfunktionen 2
   3. Zielgruppen und Charakteristika 2
   4. Betriebssystem 2
   5. Vorgaben in Design und Implementierung 2
   6. Benutzerdokumentation 2
   7. Annahmen und Abhängigkeiten 3
3. Anforderungen an externe Schnittstellen 3
4. Other Nonfunctional Requirements 7
   1. Performance Requirements 7
   2. Safety Requirements 7
   3. Security Requirements 7
   4. Software Quality Attributes 7
   5. Business Rules 8
5. Other Requirements 8

Appendix A: Glossary 8

Appendix B: Analysis Models 8

Appendix C: To Be Determined List 8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Geändert | Durch |
| 1.0 | 11.06.2018 | Initialdokument | Alle |
| 2.0 | 01.10.2018 | Finales Dokument | Alle |

# Einführung

## Zweck

Mit der vorliegenden Software soll es möglich sein, über die Open Broadcaster Software einen (Remote-)Usability-Tests zu durchzuführen. So soll es möglich sein, diese Tests ohne kostenintensive Spezial-Software durchzuführen. Die Funktionen sollen sich an bereits bestehenden Usability-Testing-Software-Produkten (wie z.B. Morae) orientieren.

Ziel des Projekts ist die Fertigstellung von Version 1 der Software, welche die im Folgendem näher beschriebenen Anforderungen erfüllen soll.

## Leserzielgruppe & Aufbau

Das Dokument richtet sich einerseits an Personen, welche die Implementierung der Software nachvollziehen wollen, an Entwickler, welche ein Interesse an der Vorgehensweise bei der Entwicklung und Einbindung einer Usability-Testing-Umgebung in OBS Studio haben, sowie an Benutzer, welche die Software für eigene Usability-Tests verwenden wollen.

Im Dokument werden zunächst Hauptfunktionen und Zielgruppe des Projekts vorgestellt, sowie grundsätzliche Vorgaben für die Entwicklung erläutert. Hierauf werden Schnittstellen vorgestellt, welche die Software integriert. Die Hauptfunktionen der Software werden daraufhin erläutert, sowie ein Überblick über die nicht-funktionalen Anforderungen gegeben.

## Produktumfang

Die entwickelte Software soll es dem Benutzer ermöglichen, durch die Open Broadcaster Software (Remote-)Usability-Tests durchzuführen. Da sich die Funktionen an bereits bestehenden Usability-Testing-Software-Produkten (wie z.B. Morae) orientieren sollen, soll es durch die vorliegende Lösung möglich sein, solche Tests professionell, aber ohne finanziellen Aufwand durchzuführen.

## Referenzen

* + OBS Plugin API Documention. (2018). Retrieved from https://github.com/jp9000/OBS/wiki/OBS-Plugin-API-Documentation
  + Morae. (2018). Retrieved from https://www.techsmith.de/morae.html
  + Open Broadcaster Software | Home. (2018). Retrieved from https://obsproject.com/

# Überblick

## Produktumriss

Das Usability-Plugin für die Open Broadcaster Software (kurz OBS), wurde für Personen entwickelt die Usability Test durchführen möchten. Der Hauptfokus liegt hierbei auf der Möglichkeit, die Tests Remote abzuhalten. Es handelt sich um ein Open Source Project und wurde für das Betriebssystem Windows entwickelt.

## Hauptfunktionen

* Bereitstellung von Tasks durch den Tester
* Chatfunktion für den Probanden und den Tester
* Aufzeichnung des Bildschirms der Testperson
* Task Modellierung durch den Tester
* Streaming des Bildschirms des Probanden
* Aufzeichnung von Tastatur und Maus Events
* Aufzeichnung der Mimik des Probanden
* Möglichkeit zeitsensitive Ereignisse während und nach dem Test zu erfassen

## Zielgruppen und Charakteristika

* Typische Anwender, sind in etwa Studenten oder Dozenten der Informatik, welche sich mit Usability Test auseinandersetzen und diese auch durchführen.
* Fortgeschrittene oder professionelle Anwender, die dieses Plugin für tiefgehende Tests verwenden möchten
* Programmierer die sich mit der Thematik befassen und dieses Projekt weiterentwickeln möchten

## Betriebssystem

* Windows 8
* Windows 10

## Vorgaben in Design und Implementierung

Das Usability Plugin wird mit Python entwickelt und nutzt die API der Open Broadcaster Software. Als Entwicklungsumgebung wurde die Community Version von Visual Studio 2017 verwendet.

## Benutzerdokumentation

Proband:

Mit der Software wird eine kurze Dokumentation zur Inbetriebnahme des Plugins bereitgestellt. Es müssen einige Softwares installiert werden, welche in der Dokumentation näher erläutert werden.

Tester:

In der Dokumentation ist eine kurze Anleitung beinhaltet, die die Inbetriebnahme der Software kurz erläutert.

## Annahmen und Abhängigkeiten

Das Usability Plugin wurde mit Python entwickelt und benötigt eine Python Installation auf dem Rechner des Anwenders. Außerdem muss die Open Broadcaster Software auf dem gewünschten Gerät vorhanden sein. Diese beiden Anforderungen gelten nur für den Rechner des Probanden, der Tester selbst benötigt lediglich einen Webbrowser (mit Zugriff auf Internet, um die Homepage zu öffnen) und OBS selber, um den Screen während des Testings aufzunehmen und evtl. Annotation hinzuzufügen.

# Anforderungen an externe Schnittstellen

## User Interfaces

Das User Interface soll sich an die ISO Norm halten (speziell DIN EN ISO 9241), um eine erfolgreiche sowie effiziente Nutzung des Systems zu ermöglichen.

1. Die Grundoberfläche des Systems wird Open Broadcaster Software (OBS) darstellen. Diese wird durch zusätzliche Szenen ergänzt, die Tasks und zusätzliche Informationen enthalten. Dabei wird der Testleiter das UI per se nicht sehen, da die Szenen in OBS nur von den Zuschauern gesehen werden können. Daher wird ein zweiter Bildschirm benötigt, um den Livestream selber mitzuverfolgen.
2. Die von uns zur Verfügung gestellte Homepage ist die Arbeitsfläche vom Testleiter. Hier kann er verschiedene Funktionalitäten ausführen, beispielsweise Task einblenden lassen, mit dem Probanden kommunizieren (via Chat) oder Annotationen hinzuzufügen. Zusätzlich brauch er noch die Anwendung OBS selber (oder eigenes Recording Programm), um den Test selber aufzunehmen, um eine effektive Nacharbeit zu ermöglichen.
3. Die getestete Person benötigt insgesamt zwei Bildschirme. Auf einem Bildschirm wird der eigentliche Test ablaufen, dieser wird auch über OBS gestreamt, damit der Testleiter dies auch sehen kann. Im rechten Bildschirm wird die Homepage angezeigt. Dort werden Tasks sowie Kommunikation zwischen dem Probanden und Testleiter stattfinden.

## Hardware Interfaces

Für das Livestreamen sollten folgende Hardware Requirements eingehalten werden:

* Mindestens 2,3 GHz Dual-Core CPU
* 8 GB RAM Arbeitsspeicher
* Ordentliche Internet-Verbindung (v.a. Upload)
* Kompatible Grafikkarte

## Software Interfaces

Um die Plugins einzubinden, benötigt man die Software Open Broadcast Software (<https://obsproject.com/de>) sowie die jeweiligen Skripte, die über die Oberfläche von OBS selber eingebunden werden können. Dazu benötigt man jedoch eine kompatible Version von Python (3.5 oder neuere Version). Es gibt verschiedene Ansichten für die jeweiligen User dieser Software. Das User Interface des Testleiters ist etwas kompakter und beinhaltet einen Video Player für den Stream selber. Zusätzlich hat er noch eine „Annotation“ Fläche, wo er Kommentare für sich selber bzw. für die Nacharbeit machen kann. Das UI des Probanden ist relativ simpel gehalten. Wir haben eine Fläche für den Chat sowie einen weiteren Reiter, um die Tasks einzublenden.

## Kommunikations-Interfaces

Es wird eine Interverbindung benötigt, um das Livestreamen überhaupt zu ermöglichen. Dabei ist vor allem die Upload Rate des Streamers von wichtiger Bedeutung, um einen Verzögerungsfreien Arbeitsablauf zu ermöglichen. Da es auf einen Server gehostet wird, ist die Kommunikation über den Chat möglich, zusätzlich kann über externe Quellen (Skype/Discord) eine mündliche Kommunikation stattfinden.

* 1. **Serveranbindung**

Als temporärer Hostserver wird Digital Ocean benutzt. Dort passieren alle wesentlichen Vorgänge unserer Software und muss bei der weiteren Nutzung selber konfiguriert werden (kurze Anleitung entnehmen Sie bitte dem beigefügten Dokument). Es wurden verschiedene Libraries benutzt, welche ebenfalls initialisiert werden müssen. Eine Datenbank wurde ebenfalls hinzugefügt, die mit dem Server connected werden muss, um Daten zu transferieren.

1. Systemkomponenten

In diesem Abschnitt wurden die verschiedenen Funktionalitäten vorgestellt, die vom Stakeholder genannt und erwünscht worden sind.

## Streaming des Testpersonen-Screens

Um das Testing durchzuführen, muss gewährleistet werden, dass der Stream ordnungsgemäß funktioniert. Dazu muss der Screen des Probanden gestreamed werden. Da es sich um die grundlegende Funktion handelt, ist die Prirorität dieser Funktionalität sehr hoch, da ohne diese Funktion der ganze Ablauf nicht möglich ist.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to see the subject’s screen during the test in realtime

Damit das Streaming ordnungsgemäß funktioniert, müssen die Hardware Requirements erfüllt sein, sowie eine stabile Internetverbindung vorhanden sein. Hierbei ist die Upload Rate des Streamers wichtig, da es sonst zu Verzögerungen sowie Verschlechterung der Stream Qualität kommen kann, was den Ablauf stören würde. Ebenso muss eine stabile Internetverbindung auf der anderen Seite vorhanden sein. Eine verzögerungsfreie Streaming Methode ist momentan nicht möglich, weshalb eine Grundverzögerung von ca. 4-5 Sekunden mit einberechnet werden muss.

## Screen Recording

Der Screen des Streamer muss aufgenommen werden, um nachträgliches Arbeiten zu ermöglichen, da bei einem Live Testing viele sensible Daten nicht bemerkt werden und erst in der Nacharbeit aufkommen. Da es sich hierbei um wichtige Daten handelt, ist auch diese eine Grundfunktion und besitzt daher eine hohe Priorität.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to save a recording of the subject’s screen during the test
* As User I want to record all mouse events during the test
* As User I want to be able to record the subject’s facial expressions during the test

OBS selber hat eine Recording Funktion, die man in den Einstellungen aktivieren kann. Dabei wird jeder Stream in ein gewähltes Videoformat lokal abgespeichert. Da ein solcher Test durchaus länger andauern kann, kann es vorkommen, dass die Datei sehr groß wird und da der Stream beim Probanden läuft - aber beim Tester landen soll - ist dies nicht sinnvoll. Deshalb sollte der Tester entweder über OBS selber aufnehmen oder durch eine externe Software den Screen aufnehmen. Eine Aufzeichnung des Probanden selber ist optional, vorausgesetzt der Proband ist damit einverstanden. OBS bietet die Möglichkeit eine „Face-Cam“ als Quelle anzugeben und in beliebiger Größe an einem Ort auf dem Stream zu platzieren. Dies könnte für manche Tests wichtig sein, muss aber auf Seiten der Probanden manuell eingestellt werden.

Ein Key- sowie Mouselogger ist als Python Skript inbegriffen und muss vor dem Streaming über OBS eingebunden werden (vgl. Doku). Die Logs werden dann als Text Dateien lokal beim Probanden abgelegt. Diese müssen dann dem Tester geschickt werden, dies geschieht am Ende des Tests und wird vor dem Abschluss abgefragt.

## Task modelling

Die Tasks sind der wichtigste Bestandteil eines User Testings und besitzt somit auch eine hohe Priorität. Das Hauptziel eines User Test ist es, die Funktionalitäten auf ihre Tauglichkeit zu überprüfen und die Meinungen der Probanden zu berücksichtigen und für das Finale Produkt mit ein zu beziehen. Die Tasks sollen hierbei wesentliche Funktionalitäten der Applikation abbilden und in einem Anwendungsfall soll herausgefunden werden, inwiefern der Proband das Nutzungsszenario erfolgreich absolviert hat.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to model tasks to show them on the tester’s screen
* As User I want to be able to edit tasks

Die Tasks werden auf der Homepage angezeigt und kann nur vom Testleiter hinzugefügt und modifiziert werden. Der Proband kann die Tasks sich anzeigen lassen, während der Testleiter diese hinzufügt/modifiziert. Hierfür existieren auf der Homepage verschiedene Reiter.

## Aufzeichnung von Maus- und Tastaturaktivität

Alle Aktivitäten sollten aufgenommen werden, zwar sind diese nahezu komplett durch das Recording abgedeckt, aber die Maus und Tastatur Events sollten nochmals separat aufgezeichnet werden, um eine nachträgliche Datenauswertung zu vereinfachen. Diese sollen zusätzlich noch mit „Timestamps“ ergänzt werden, um die Datenauswertung zusätzlich zu vereinfachen. Außerdem sollten die Timestamps so gekennzeichnet werden, dass beispielsweise Taskrelevante Daten schneller gefiltert werden können.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to get keylogs of all activity of the subject during the test
* As User I want to record all mouse events during the test
* As User I want to get an overview of hotspots/main areas of interest of the subject
* As User I want to be able to log a timeline for the test

Alle relevanten Daten werden im Regelfall lokal beim Probanden abgespeichert. Diese müssen nach Abschluss des Tests an den Testleiter übermittelt werden. Diese sollten nicht all zu groß werden, da es sich hierbei um mehrere Textfiles handeln wird.

## Aufnahme der Mimik der Testperson

Der Gesichtsausdruck eines Probanden soll bei Erlaubnis aufgezeichnet werden. Gesichtsausdrücke geben Informationen über die Gemütslage des Probanden wieder, was für User Interface spezifische Angelegenheiten wichtig sein kann.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to record the subject's facial expression during the test

Hierfür wird eine Webcam benötigt, sowie muss davor mit dem Probanden abgeklärt werden, ob er damit einverstanden ist, dass man ihn aufnimmt. Man kann in OBS eine Szene erstellen und diese so platzieren, dass Sie nur wenig Workspace in Anspruch nimmt.

## Zeitsensible Anmerkungen

Es soll dem Nutzer möglich sein, Metadaten und Kommentare während des Tests zu machen. Hier soll es möglich sein direkt im Screen einen Kommentar zu verfassen. Diese sollten möglichst einfach hinzugefügt werden, damit der Fokus auf den Test bleiben kann. Ebenfalls werden diese Daten meist vergessen, wenn man diese nicht direkt mitnotiert, weshalb diese Funktionalität sehr wichtig ist.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to add time-sensitive metadata to the recording of the test
* As User I want to be able to log a timeline for the test

Diese Funktion muss in wenigen Schritten erfolgen, da man diese Kommentare während des Testings anfertigt und den Workflow der getesteten Person sowie des Testers selber nicht unterbrechen will. Hierbei wird die Realtime Zeit benutzt.

## Kommunikation mit der Testperson

Die Kommunikation zwischen Testperson und Testleiter spielt eine wesentliche Rolle für ein erfolgreiches Testing. Vor allem hinsichtlich dem Remote Testing, ist die Kommunkation umso wichtiger, da man sich nicht persönlich gegenüber steht, sondern durch eingeschränkte Möglichkeiten kommunizieren muss. Daher muss der Chat reibungslos funktionieren und weder den Leiter noch den Probanden ablenken, während diese den Test absolvieren.

Folgende Use Cases müssen durch diese Funktion abgedeckt sein:

* As User I want to be able to have a live-chat with the subject during the test

Zusätzlich zum Chat wäre es ratsam, über eine Voice Kommunikation nachzudenken. Diese kann über externe Softwares realisiert werden und würde effizienter Funktionieren als ein Chat. Man müsste seinen Workflow bei einem Chat unterbrechen damit man schreiben kann, während bei einer Voice Kommunikation keine Schreibzeit benötigt wird und man kann mehr Information in einer kürzeren Zeit übermitteln.

1. Other Nonfunctional Requirements

## Performance Requirements

Die Kernfunktion des Projekts besteht darin, einen Usabilitytest mithilfe des OBS Tool durchzuführen. Dieses wird hierbei für die Echtzeitübertragung eines Videostreams verwendet.

Daher ist eine schnelle Internetverbindung der beteiligten Geräte vorausgesetzt.

Auch sollten die für die Tests verwendeten Rechner daher für das Verwenden der Software geeignet sein. OBS selbst besitzt folgende Anforderungen:

Windows:

DirectX 10.1 compatible GPU

Windows 7 or newer

macOS:

Intel CPU (PPC is not supported)

OpenGL 3.2 compatible GPU

macOS 10.10 or newer

Linux/Unix

OpenGL 3.2 compatible GPU

X window system

[https://obsproject.com/wiki/System-Requirements]

Obwohl die offiziellen Anforderungen dazu keine Angaben machen, sollten 8GB RAM und ein aktueller Prozessor vorausgesetzt werden.

## Safety Requirements

Da es sich hierbei um Software handelt, ist eine Gefährdung durch die Benutzung so gut wie ausgeschlossen. Allerdings sollte hier dennoch die folgende Warnung vor photosensitiven Anfällen beachtet werden, auch wenn die Software keine unüblichen Lichteffekte verwendet:

„Bei manchen Personen kann es zu epileptischen Anfällen oder Bewußtseinsstörungen kommen, wenn sie bestimmten Blitzlichtern oder Lichteffekten ausgesetzt werden.  
Diese Personen können bei der Benutzung von Computern einen Anfall erleiden. Es können auch Personen davon betroffen sein, deren Krankheitsgeschiche bislang keine Epilepsie aufweist und die nie zuvor epileptische Anfälle gehabt haben.  
Falls bei Ihnen oder einem Ihrer Familienmitglieder unter Einwirkung von Blitzlichtern mit Epilepsie zusammenhängende Symptome (Anfälle oder Bewußtseinsstörungen) aufgetreten sind, wenden Sie sich bitte vor der Verwendung an Ihren Arzt.“

## Security Requirements

Die Software zeichnet den Tester sowie seine Eingaben während des Tests auf. Alle eventuell relevanten Datenschutzgesetze sind dabei zu berücksichtigen und die Testperson ist von diesem Umstand in Kenntnis zu setzten und muss ihm zustimmen.

Je nach verwendeter Software oder Serverstruktur zur Verarbeitung des Videostreams sollte auch auf die Datensicherheit geachtete werden, da dort ebenfalls die Datenschutzrechte des Getesteten verletzt werden können.

Daher sind die aufgezeichneten Videos mit Umsicht und der angebrachten Sorgfalt im Hinblick auf die Datenschutzrichtlinien zu behandeln.

## Software Quality Attributes

Die Software soll im Rahmen des Kurses eine funktionierende Version des OBS Tools inklusive funktionierender Plugins oder Scripte beinhalten.

Ein funktionierender Server sowie dessen Implementation ist dahingegen nicht vorgesehen.

Für eine Benutzung des Tools ist daher ein geeigneter Server bereitzustellen, der den benötigten Voraussetzungen gerecht wird.

Die Plugins/Scripte sind verwendbar mit einer standard OBS Version und können zu jeder neuen Installation des Programms hinzugefügt werden.

## Business Rules

Bei der Benutzung des Tools können Testpersonen, vorgegebene Aufgaben bearbeiten und Tester können diesen Anweisungen oder neue Aufgaben zukommen lassen.

Die Funktionalitäten sind auf diese beiden Rollen beschränkt und können nicht vom jeweils anderen benutzt werden.

# Other Requirements

OBS ermöglicht es lediglich, den aufgezeichneten Videostream bei der Darstellung zu verändern, bzw. extra Overlays oder Hinweise anzuzeigen.

Daher ist es nicht möglich auf dem Bildschirm der Testperson, auf dem gerade aufgezeichnet wird, auch neue Overlays mit testrelevanten Informationen oder Aufgaben darzustellen.

Das bedeutet, dass für die vorgesehene Benutzung des Tools zwei oder mehr Bildschirme benötigt werden.

Auf dem ersten wird das Verhalten der Testperson aufgezeichnet und auf dem zweiten kann der Tester mit der Testperson interagieren, etwa in dem er Anweisungen über das Tool an den Probanden schickt.

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>