Bachelorarbeit

Atil Uyanik

Usability Engineering für
Microsoft Surface am Beispiel

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Atil Uyanik

Usability-Engineering für Microsoft Surface am Beispiel

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Stefan Sarstedt

Abgegeben am: 20.07.2011

Atil Uyanik

Thema der Bachelorarbeit
Usability Engineering für Microsoft Surface am Beispiel

Stichworte
Usability, Microsoft Surface

Kurzzusammenfassung
Diese Arbeit untersucht die Einsetzbarkeit von vorhandenen Usability-Untersuchungs-methoden für Microsoft Surface, indem eine Beispielanwendung entwickelt, und Usability-Tests an dieser Anwendung durchgeführt werden.

Anfangs werden die Planungs-, Durchführungs-, und Auswertungsphasen der durchgeführten Tests beschrieben. Die Ergebnisse, die durch die ausgewählten Testmethoden entstanden sind, dienen als Grundlage für die Bewertung von Anwendungen auf dem Microsoft Surface.

Atil Uyanik

Title of the paper
Example for Usability Engineering on Microsoft Surface

Keywords
Usability, Microsoft Surface

Abstract
This paper describes the applicability of existing Usability inspection methods for Microsoft Surface, by developing an example application, and conducting usability tests on this application.

Initially the planning, execution and evaluation phases of the performed test will be described. The results, which were obtained by the selected test methods, provide a basis for the evaluation of application on the Microsoft Surface.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Mit Usability Untersuchungen (Mensch-Computer-Interaktion) wurde schon in den 1980ern angefangen[UsEnHIST]. Die Testpersonen wurden auf der Straße angesprochen und die Programme wurden mit ihnen getestet, wobei die Dauer der einzelnen Tests durch eine Stoppuhr gemessen wurde. Nach jedem Test haben die Programmierer das Programm verbessert. In dreißig Jahren hat sich Usability Engineering weiter entwickelt. Heutzutage werden meistens Desktop-Anwendungen, Webanwendungen und Spiele durch eine Vielzahl von Testmethoden und Testgeräten untersucht, um ihre Anwendbarkeit zu verbessern.

Seit Ende der neunziger Jahre haben sich Projekte im Bereich Surface Computing entwickelt. Im Gegensatz zu „normalen" und herkömmlichen graphischen Benutzeroberflächen (GUI-Anwendungen), die mit Maus und Tastatur bedient werden, werden die Surface-Anwendungen durch Tippen, Wischen oder Berühren bedient. Das nennt man „Natural User Interface" (NUI). Natürliche Benutzeroberflächen, wie bei Microsoft Surface, Touchscreens und Multi-Touch-Geräten, sind berührungsempfindlich und reagieren auf Finger- und Handbewegungen. Damit ermöglichen die natürlichen Benutzeroberflächen eine direkte Interaktion mit der Bedienoberfläche.

Die derzeit vorhandenen Usability-Testmethoden sind hauptsächlich für graphische Benutzeroberflächen geeignet und weniger für natürliche Benutzeroberflächen. Zum Beispiel können die Maus- und Tastatureingaben nicht gemessen werden, da die Geräte für MS Surface nicht Verfügbar sind. Eine andere berühmte Usability-Methode, Eye-Tracking kann hierbei auch nicht eingesetzt werden, da sich die Benutzer während der Bedienung des Surface-Tisches viel bewegen, und nicht wie bei der herkömmlichen Interaktion mit Maus, Tastatur und Monitor in derselben Position bleiben.

1.2 Ziel der Arbeit

Hauptziel dieser Arbeit ist die Benutzbarkeit (Usability) von Microsoft Surface zu testen, auszuwerten und Schlüsse zu ziehen. Dabei ergibt sich das Nebenziel geeignete Testmethoden zu finden, um Usability-Tests durchführen zu können, und diese Testmethoden zu beschreiben. Zu dem Hauptziel wird geklärt:

- wie die Nutzer mit den unbekannten Interaktionstechniken umgehen
- wie die Nutzer mit MS Surface umgehen
- wo die Probleme bei Nutzung des MS Surface liegen
- wie der Lernverlauf aussieht.

Hierfür soll eine Surface-spezifische Beispielanwendung für alltägliche Benutzung geschrieben, und Usability-Tests an dieser Anwendung durchgeführt werden. Wichtig ist bei der Anwendung, dass sie für „casual"-Nutzer geeignet sein soll.

Um die Schwierigkeiten und Probleme bei der Benutzung von Microsoft Surface und der Beispielanwendung (Flug-Informationssystem) zu finden, Verbesserungsmöglichkeiten und neue Erweiterungen zu entdecken, werden Surface-Tisch und Surface-Anwendung zusammen getestet.

1.3 Aufbau der Arbeit

Zentraler Untersuchungsgegenstand dieser Arbeit ist MS Surface, daher sollen in **Kapitel 2** die notwendigen Grundlagen für allgemeines Verständnis beschrieben werden. In diesem Kapitel soll geklärt werden, was MS Surface eigentlich ist, welche Software und Hardware Eigenschaften Surface hat. Außerdem wird ein Blick auf die sogenannten „Gesten" wie Anwählen, Drehen, Scrollen, Skalieren und Verschieben geworfen.

In **Kapitel 2** wird zusätzlich noch der Begriff Usability Engineering allgemein erläutert. Einige der wichtigsten vorhandenen Testmethoden werden beschrieben.

In dem **dritten Kapitel** wird die Beispielanwendung vorgestellt. Dabei werden die Anforderungen an die Software und inhaltliche Informationen, wie Softwarearchitektur, Klassen und Packages, erläutert. Abschließend wird ein möglicher Durchlauf der Anwendung beschrieben.

In **Kapitel 4**, **5** und **6** werden die durchgeführten Tests und die Ergebnisse näher beschrieben. In **Kapitel 4** wird die Planungsphase der durchgeführten Tests und die Erstellung des Testleidfadens erklärt. Außerdem wird die Auswahl der zu realisierenden Testmethoden in diesem Kapitel getroffen.
Bei dem **Kapitel 5** geht es um die Durchführung der ausgewählten Tests. Die Testumgebung wird vorgestellt, Informationen über die Testpersonen werden gegeben, und der generelle Ablauf von Usability-Tests wird beschrieben. Zum Schluss wird die Dauer von den einzelnen Tests gemessen und behandelt.
Die Auswertung der Tests findet in **Kapitel 6** statt. Es werden die Auswertungskriterien, die Auffälligkeiten die es während der Tests gab, und die Auswertung der Fragebögen erklärt.

In **Kapitel 7** werden die Konsequenzen, die nach den Tests entstanden sind, beschrieben. Dazu gehören die positiven Bemerkungen und Verbesserungsvorschlägen für die Beispielanwendung, sowie die Konsequenzen für Software, Usability-Tests, Microsoft Surface und „Casual Anwendungen".

Kapitel 8 fasst die Arbeit nochmal zusammen, und gibt einen Ausblick.

2 Grundlagen

In diesem Kapitel werden die Grundlagen für das Verständnis dieser Arbeit erläutert. Dazu gehören MS Surface, seine Eigenschaften und allgemeine Information, sowie Usability Engineering und einige Untersuchungs-Testmethoden.

2.1 Microsoft Surface

"A computer on every desktop."
Now we say "Every desktop will be a computer."
[BGATES]

Microsoft Surface [MSSurf] ist eine Kombination von Hardware und Software, die einem oder mehreren Benutzern erlaubt, die digitalen Inhalte durch Nutzung von natürlichen Gesten und Handbewegungen zu manipulieren. Surface hat keine Eingabegeräte wie Maus oder Tastatur. Sämtliche Eingaben werden mit der Hand auf der 30" großen Tischplatte vorgenommen.

Abbildung 1 - Microsoft Surface

MS Surface basiert auf der NUI (Natural User Interface) Benutzeroberfläche. Die Objekte, Berührungen oder Bewegungen werden durch fünf, innerhalb von Surface installierte, infrarot Kameras erkannt. Das Resultat des Inputs wird dann per Rückprojektion auf der Oberfläche angezeigt. Das System ermöglicht auch komplexere Eingaben. Die Besonderheit dabei ist, dass der User auch mehrere Aktionen gleichzeitig durchführen kann. Das Gerät ist eigentlich in der Lage eine beliebig große Anzahl von Berührungspunkten zu registrieren und zu verarbeiten. Microsoft gibt vor, dass mindestens 4 Personen mit jeweils 10 Fingern darauf arbeiten können, ohne dass die Performance darunter leidet. Zu den Eigenschaften gehört auch Objekterkennung. So erkennt Surface fast alle Gegenstände die auf den Tisch abgelegt werden, wie z.B. auch elektronische Geräte (Mobiltelefone, Digitalkameras).

Die oben genannten Eigenschaften sind die vier Schlüsseleigenschaften von Surface. Microsoft nennt diese Eigenschaften „Direct Interaction", „Multi-Touch Contact", „Multi-User Experience" und „Object Recognition".

2.1.1 Geschichte

Surface Computing ist ein bedeutender Fortschritt, welcher die traditionelle Benutzerschnittstelle durch eine natürliche Art von Interaktion ersetzen kann. Mit Maus und Tastatur gibt es immer eine kleine Barriere zwischen Mensch und Computer, welche durch die Interaktion mithilfe natürlicher Gesten, Berührungen oder physikalischer Objekte zu beseitigen ist.

Microsoft Surface ist ein Computer der Firma Microsoft, der im Jahr 2007 vorgestellt wurde. Aber mit der Multi-Touch Technologie Forschung wurde schon im Jahr 1982 an der Universität Toronto angefangen. Das eigentliche Surface-Konzept wurde im Jahr 2001 von Stevie Bathiche und Andy Wilson erdacht[MSHist], und erst in 2003 zum ersten Mal als Ikea-Tisch gebaut. Ab 2008 wurde der Einsatzzweck erweitert und Surface ist seither in Hotels, Restaurants, Casinos und Ladengeschäften benutzt worden.

Obwohl Surface Computing noch nicht ganz verbreitet ist, glaubt Microsoft daran, dass es mit der Zeit immer wichtiger und durchdringend für das Alltagsleben sein wird.

2.1.2 Multi-Touch

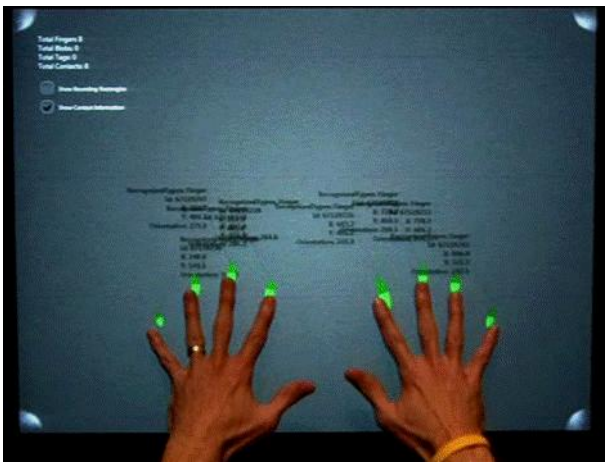

Abbildung 2 - Multi-Touch

Multi-Touch beschreibt ein Natural User Interface Bedienkonzept. Mit der Maus hat man nur eine einzige x/y-Koordinate an der man zur selben Zeit sein und klicken kann. Bei Multi-Touch ist es möglich, zur selben Zeit mehrere gleichzeitige Eingaben zu registrieren und zu verarbeiten. Ein Multi-Touch Bildschirm wird ausschließlich mit den Fingern bedient. Jede Art der Interaktion kann zu einer definierten programmierten Aktion wie sogenannten Gesten (s. Kap 2.1.5) führen. Um beispielsweise Fenster zu skalieren, reicht es, die Gesten mit den Fingern durchzuführen.

Surface ist theoretisch in der Lage 52 verschiedene Berührungen gleichzeitig zu registrieren. Bei Surface werden Berührungen über fünf Infrarot-Kameras erfasst[SurfNUI]. Die Tischform ermöglicht auch Multi-User-Betrieb. Da MS Surface von allen vier Seiten zugänglich ist, können sich idealerweise maximal vier oder fünf Personen um den Tisch versammeln und zusammen gleichzeitig arbeiten.

2.1.3 Objekterkennung

Surface ist in der Lage mehrere Arten von Gegenständen zu erkennen. Zum Beispiel können Objekte über einen proprietären von Microsoft spezifizierten 2-D Barcode (Dotcode s. Abb. 3) erkannt werden und prinzipiell kann mit diesen Objekten interagiert werden. Die Dotcodes gibt es als 8-Bit und zukünftig als 2*64-Bit[SurfNUI].

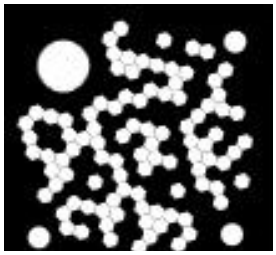

Abbildung 3 - Surface Dotcode

Auch Gegenstände ohne Dotcode können an Hand ihrer Form identifiziert werden, indem Objekte Surface einmal „beigebracht" werden, und dann als direktes physisches Interface agieren. Es ist auch möglich WLAN-fähige Digitalkameras oder Mobiltelefone die mit einem Tag gekennzeichnet sind auf den Tisch zu legen, und die darauf gespeicherten Dateien auf dem Surface-Display anzuzeigen.

2.1.4 NUI (Natural User Interface)

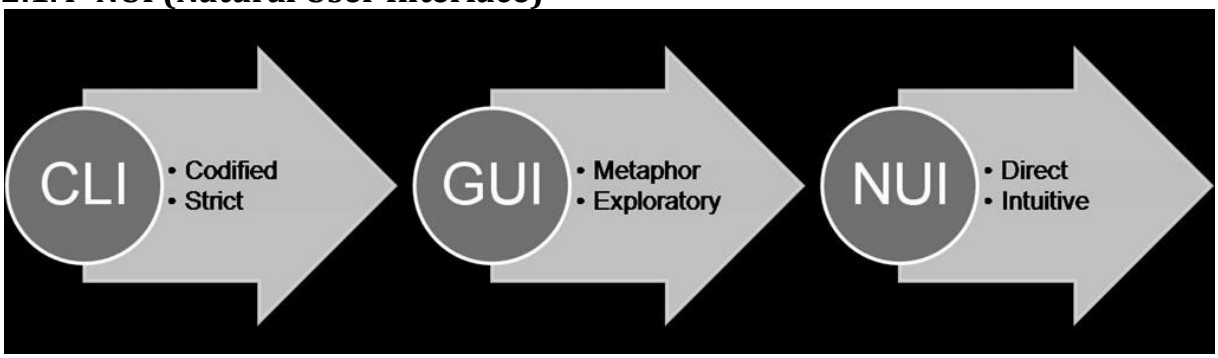

Abbildung 4 - CLI-GUI-NUI

Natural User Interface hat eine neue Art von Mensch-Computer-Interaktion vorgestellt. Bisher wurden meistens „Command Line Interfaces" und „Graphical User Interfaces" verwendet.

Bei Command Line Interfaces werden die Kommandos oder Befehle als Zeichenketten über die Tastatur eingegeben. Die Befehlseingabe wird dann mit dem Betätigen der Eingabetaste abgeschlossen und Carriage-Return wird an das Programm gesendet. Der Command Line Interpreter entnimmt, analysiert und führt die gewünschte Eingabe aus. Zuletzt wird der Output als Textzeilen angezeigt. Obwohl CLI's schon altmodisch sind, werden sie öfters von Programmierern, Administratoren, oder fortgeschrittenen PC-Benutzern verwendet. Ein Beispiel für CLI's wären Unix-basierte Betriebssysteme.

Nach der Erfindung der Maus hat eine neue Ära angefangen, und die Interaktion zwischen Mensch und Computer ist vereinfacht worden. Statt Textzeilen wie bei der CLI, erlaubt GUI, die Interaktionen über graphische Symbole darzustellen. Dies geschieht meistens mittels einer Maus als Steuergerät. Die Maus und das GUI sind die Hauptgründe für die tägliche Benutzung von Computern im Leben, aber die Interaktion ist indirekt. Bei einer direkten Interaktion wie bei NUI, wird die "Handlung" eines Objekts unterstützt und verstärkt. Es wird verstärkt auf traditionelle menschliche Fähigkeiten wie Berührung, Bewegung und Gesten, Sicht, Sprache und Akustik fokussiert. Für den Benutzer würde es einfacher fallen, eine Aufgabe durch direkte, einfache und natürliche Bedienung des Systems zu bearbeiten, statt mit Maus und Tastatur Eingaben wie bei klassischen GUI oder CLI.

2.1.5 Gesten

Microsoft Surface verarbeitet und registriert nur eine Art von Touch, welche durch „Manipulation", „Gesten", oder „Manipulationsgesten" definiert ist.
Wenn ein Objekt durch den Benutzer berührt wird und eine Veränderung durchgeführt wird, spricht man von Manipulationsgesten. Microsoft Surface verarbeitet drei verschiedene Arten von Manipulationsgesten:

- Move (Bewegen)
- Rotate (Rotieren)
- Resize (Skalieren)

Obwohl nur drei Manipulationsgesten verfügbar sind, gibt es mehrere „Bewegungsarten" für den Benutzer um auf den Objekten Veränderungen durchzuführen. Befehle wie[MSUEG];

- **Tap** (Abklopfen); schnell berühren und loslassen, Auswahl eines Elements
- **Slide or push** (Verschieben); Berühren und Bewegen eines Elements
- **Flick** (Werfen); wie „slide", plötzliches Loslassen eines Elements in der Bewegung
- **Touch and Turn**; wie „slide", Element berühren, und Finger im oder gegen den Uhrzeigersinn bewegen, ermöglicht Drehen eines Elements
- **Spin** (Kreiseln); wie beim Flick, plötzliches Loslassen eines Elements in der Rotationsbewegung
- **Stretch**; Vergrößerung eines Elements durch Auseinanderbewegung zweier Finger
- **Shrink**; Verkleinerung eines Elements durch Zusammenführen zweier Finger
- **Twist**; Berührung des Elements durch zwei Finger oder einer Hand und Drehen des Elements im oder gegen den Uhrzeigersinn
- **Pin turn**; ein Finger fixiert die Position der Drehachse, während ein zweiter Finger die Drehung durchführt.

2.1.6 Hardware

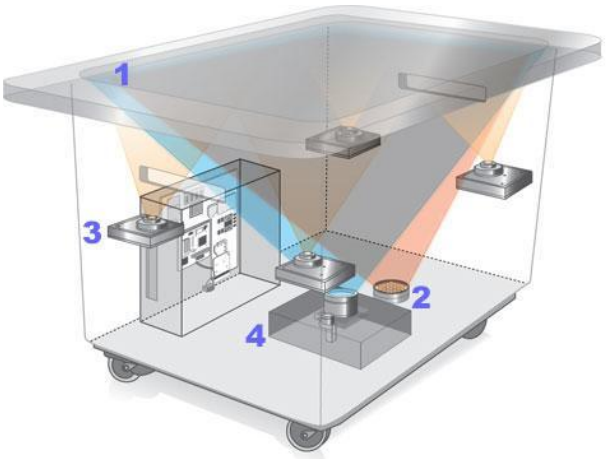

Abbildung 5 - MS Surface Hardware

Auf den ersten Blick erinnert der Surface mit seiner Größe an einen Wohnzimmertisch. Aber innerhalb des Gehäuses befinden sich fünf Infrarot-Kameras, ein Projektor und ein Rechner[MSDS].

Physikalische Dimensionen: 108cm x 69cm x 54cm (L x B x H), Gewicht : 90 kg.

Display: Auf den Stahlrahmen liegt eine kratz- und wasserresistente Acrylglasplatte, in die ein 30" großes Display integriert ist. Microsoft Surface erkennt die Berührungen und verschiedene Arten von Objekten mit Hilfe von fünf integrierten Infrarot-Kameras, verarbeitet die Eingaben und zeigt die daraus resultierende Interaktion durch den Projektor mit Rückprojektion an. Da die maximale Auflösung 1024 x 768 Pixel ist, ist die interaktive Fläche des Surfaces zu klein und niedrig auflösend für kollaborative Anwendungen.

CPU: 2.13-GHz Intel Core 2 Duo Prozessor ermöglicht die Verarbeitung von zwei parallelen Prozessen mit zwei Rechenkernen.

Arbeitsspeicher: 2 GB dual-channel DDR2. Da Microsoft Windows Vista Business Edition schon ca. 1 GB für den Betrieb braucht, bleibt nur noch 1 GB für alle anderen Anwendungen. Die Belegung des Arbeitsspeichers ist abhängig von allen laufenden Prozessen und ihrem jeweiligen Datenaufkommen. Da in der Praxis zunächst keine Anwendung auf dem Surface geschlossen wird, sondern bloß in den Hintergrund tritt, kann der verfügbare Arbeitsspeicher schnell knapp werden.

Grafikkarte: Microsoft Surface besitzt eine ATI X1650 Grafikkarte mit 256 MB Memory, die eine maximale Auflösung von 1024 x 768 Pixel erlaubt.

Audio: Stereo-Ausgabe über zwei eingebaute „flat-panel" Lautsprecher. Audio-Eingänge und Mikrofon sind nicht verfügbar. Weiterhin können noch zwei Kopfhörer angeschlossen werden.

Netzwerk: IEEE802.11b, IEEE802.11g, Bluetooth 2.0 und Gigabit Ethernet. Vernetzung mit einem LAN, WAN oder WLAN sowie Bluetooth-Verbindungen mit anderen Geräten möglich.

Ein- und Ausgabe: 6 USB 2.0 Ports werden zu Entwicklungszwecken zum Anschluss von Maus und Tastatur bereitgestellt. Es existiert ein externer S-VGA Anschluss (ebenfalls für Entwicklungszwecke), welcher eigentlich den primären Windows-Desktop ausgibt.

2.1.7 Software

Microsoft Surface basiert auf einer Windows Vista Plattform. Das aktuelle Modell läuft mit Windows Vista Business Edition als Betriebssystem und speziell installierten Surface-Applikationen. Diese sind in XNA / .NET / WPF entwickelt und müssen demnach auf dem Rechner installiert werden, was derzeit noch den Einsatz von Maus und Tastatur erfordert[SurfNUI].

Das Microsoft Surface SDK 1.0 SP1 bietet die Möglichkeit für die Entwickler ihre eigenen Anwendungen zu entwickeln. Surface SDK enthält Anwendungs- und Codebeispiele und einen „Surface Simulator", um die Anwendungen unabhängig von der Surface-Hardware auf einem PC-System zu entwickeln. Die folgenden Entwicklungswerkzeuge sind in Surface SDK vorhanden[MSSurf];

- **Windows Presentation Foundation (WPF)**: ist ein Grafik-Framework und Teil des .NET-Frameworks von Microsoft zum Erstellen von Windows-Clientanwendungen. Der Kern von WPF ist ein auflösungsunabhängiges und vektorbasiertes Rendering-Modul, das die Funktionen moderner Grafikhardware nutzt. WPF erweitert diesen Kern um einen umfassenden Satz von Anwendungsentwicklungsfunktionen wie; Steuerelemente, Datenbindung(Data Binding), Layout, 2D- und 3D-Grafik und Animationen[MSWPF].

- **Microsoft XNA**: ist ein .NET-basiertes Grafik-und Spiele-Entwicklungs-Framework, das zuerst im Jahr 2006 vorgestellt wurde. Es wird für die Entwicklung von Spielen unter Windows Vista, Windows XP und Microsoft „XBox 360" verwendet[MSXNA].

- **Microsoft Visual Studio**: ist eine Entwicklungsumgebung von Microsoft, die Programmiersprachen wie; Visual Basic, C, C++ und C# unterstützt. Da die Entwicklungssprache von MS Surface C# ist, wird MS Visual Studio als Entwicklungsumgebung für Surface-Plattformen präferiert.

Neben den aufgezählten Werkzeugen, sind auch noch die Entwicklungssprache Microsoft Visual C# Express Edition, und das User-Interface-Entwicklungstool Microsoft Expression Studio 2 in Surface SDK vorhanden.

2.2 Usability Engineering

Der Begriff „Usability" stammt aus dem Englischen. Er setzt sich aus zwei Wörtern zusammen, to use (benutzen) und the ability (die Fähigkeit). Übersetzt wird der Begriff mit ***Gebrauchstauglichkeit*** *oder aber auch* ***Brauchbarkeit****.*
[UsEnREG]

Usability Engineering behandelt die Beziehung zwischen einem Produkt und seinen Benutzern. Es ist ein Teilgebiet der Mensch-Computer-Interaktion und definiert, wie einfach ein Produkt über ihre Benutzerschnittstelle zu bedienen ist.

Nach ISO 9241[ISOUsability]; "Usability ist das Ausmaß, in dem ein Produkt von einem bestimmten Benutzer verwendet werden kann, um bestimmte Ziele in einem bestimmten Kontext effektiv, effizient und zufriedenstellend zu erreichen."

Nach Nielsen[Nielsen93] wird die Gebrauchstauglichkeit von Software durch folgende fünf Attribute bestimmt:

- **Learnability (Erlernbarkeit)**: Das System sollte so leicht wie möglich sein, damit der Benutzer sich schnell daran gewöhnen und Ergebnisse erzielen kann.

- **Efficiency (Effizienz)**: Das System sollte so effizient sein, dass wenn ein Benutzer das System kennengelernt hat, sollte es möglich sein auf hohem Niveau zu arbeiten.

- **Memorability (Einprägsamkeit)**: Ein unkomplizierter Umgang mit dem System muss gewährleistet werden, auch wenn ein Benutzer das System lange Zeit nicht benutzt hat. Wiedereinstieg soll so einfach wie möglich sein.

- **Errors (Fehlerresistenz)**: Das System sollte eine niedrige Fehlerrate haben, damit der Benutzer die Fehler die er während der Benutzung des Systems macht, leicht wieder beheben kann. Schwerwiegende Fehler dürfen gar nicht auftreten.

- **Satisfaction (Zufriedenheit)**: Das System sollte angenehm zu benutzen sein, damit es den Benutzer Zufriedenheit stellt.

Also ist Usability keine eindimensionale Eigenschaft eines Systems, sondern sie besteht aus mehreren komplexen Komponenten. Es ist unmöglich diese Anforderungen ganz am Anfang zu erfüllen. Durch die Usability Engineering Methoden können die Schwachstellen eines Systems während des Entwicklungsprozesses erkannt und behoben werden. Ein solches Vorgehen ist vorteilhaft, da die Lösung von Usability-Problemen bei einem fertig entwickelten System schwierig und teuer ist.

Um die Gebrauchstauglichkeit eines Systems zu untersuchen gibt es verschiedene Untersuchungsmethoden die man einsetzen kann.

2.2.1 Heuristische Evaluierung

Heuristische Evaluierung ist die meist benutzte formative (vor Fertigstellung des Systems) Usability Methode, um die Gebrauchstauglichkeit einer Benutzeroberfläche zu beurteilen. Es bedeutet, dass eine geringe Anzahl von Evaluatoren die Benutzerschnittstelle eines Produktes untersuchen, und überprüfen in wieweit diese mit bestimmten Usability Richtlinien (Heuristiken) übereinstimmen[Nielsen93]. Die Richtlinien können dabei aus unterschiedlichen Quellen kommen, aber auch die eigene Intuition und Vernunft kann als Quelle dienen. Ziel ist es dabei, so viele Fehler wie möglich zu finden. Die Ergebnisse können dann in der Designphase hilfreich werden.

Bei heuristischer Evaluierung kann jeder einzelne Evaluator das Produkt alleine untersuchen, aber die bisherige Untersuchungen zeigen, dass die Ergebnisse von allein durchgeführten Untersuchungen nicht genügend Usability-Probleme aufweisen. Da alle Evaluatoren verschiedene Probleme finden können, werden die Ergebnisse von den verschiedenen Evaluatoren gesammelt, damit möglichst viele Fehler aufgespürt werden können. Erst wenn alle anderen Evaluationen fertig sind, kommunizieren die Evaluatoren miteinander um ihre Befunde zusammenzufassen. So wird eine unabhängige und unbeeinflusste Prüfung gewährleistet.

Die Ergebnisse werden entweder schriftlich von den Evaluatoren selbst oder von einem Beobachter, der während des Tests ein Protokoll führt, festgehalten. Ein Test mit einem Beobachter ist vorzuziehen, da er während des Tests den Evaluatoren bei Fragen oder Problemen behilflich sein kann. Es wäre auch sinnvoll, dass der Evaluator die Probleme und Fragen selbst löst, aber das würde die Arbeitslast-und Dauer erhöhen. Die Prüfungen dauern in der Regel ein bis zwei Stunden, dies ist aber von der Komplexität des Produktes abhängig.

Die Vorteile dieser Methode sind;

- Geringe Kosten im Gegensatz zu anderen Evaluationsmethoden
- Erweiterte Planung ist nicht erforderlich
- Ein lauffähiges System ist nicht zwingend erforderlich
- Flexibel einsetzbar

Die Nachteile dieser Methode sind;

- Die Benutzung wird nicht analysiert, sondern es findet nur eine regelbasierte Analyse statt
- Findet Probleme, aber bietet in den meisten Situationen keine Lösungen an
- Findet wesentlich kleinere und weniger Probleme als andere Usability Engineering Methoden
- Die Evaluatoren sind keine wirklichen Nutzer. Es ist wahrscheinlich, dass die echten Nutzer auch andere Probleme finden können.

2.2.2 Cognitive Walkthrough

Cognitive Walkthrough(CW) ist eine aufgabenorientierte Usability Engineering Methode, bei der ein interaktives Produkt hinsichtlich seiner Erlernbarkeit untersucht wird[Burmester2003]. Es werden gezielt einzelne Abläufe untersucht. Im Gegensatz zu anderen Methoden, wird der Schwerpunkt meistens auf die mentalen Prozesse eines Users, und weniger auf das Interface gelegt. Der CW setzt sich aus einer Vorbereitungs- und einer Analysephase zusammen.

Bei der Vorbereitungsphase werden die zu testende Interface, detaillierte Informationen über die potenziellen Nutzer, die Aufgaben und die Aufgabenschritte bestimmt. Dabei sollte

beachtet werden, dass die zu lösende Aufgaben für den täglichen Gebrauch des Systems sinnvoll sind. Jede Aufgabe und ihre Aufgabenschritte werden ausführlich beschrieben. Für jede Aufgabe wird festgelegt, welchen Weg der User im Idealfall gehen wird, um seine Aufgabe durchzuführen.

In der Analysephase werden die konkreten vorgegebenen Handlungsabläufe analysiert. Dabei werden für jeden Handlungsschritt vier Aspekte beurteilt, die für eine erfolgreiche Mensch-Computer-Interaktion erforderlich sind;

- Versucht der Benutzer den richtigen Effekt zu erzielen?
- Erkennt der Benutzer, dass die korrekte Aktion zur Verfügung steht?
- Versteht der Benutzer, dass der richtige Effekt durch die korrekte Aktion erreicht wird?
- Erkennt der Benutzer den Fortschritt, wenn die korrekte Aktion ausgeführt worden ist?

Falls diese Fragen positiv beantwortet werden, wird eine sogenannte "success story" für jeden einzelnen Schritt erstellt. Und falls die Antworten negativ sind, wird eine "failure story" erstellt, welche zeigt warum der Benutzer die Aufgaben nicht erfüllen konnte. Dadurch werden die Usability-Probleme erkannt.

Die Vorteile dieser Methode sind;
- Schnell und einfach durchführbar
- Geringe Kosten
- Durchführung schon im Entwicklungsstadium

Die Nachteile dieser Methode sind;
- Getestet wird von Experten und nicht von echten Benutzern
- Sehr aufwändige und detaillierte Aufgabenanalyse
- Für jede Aufgabe muss ein eigener Cognitive Walkthrough erarbeitet werden

2.2.3 Usability Testing

Usability Testing ist eine effiziente Usability-Test-Methode um Gebrauchstauglichkeit einer Anwendung zu überprüfen. Ein wichtiger Vorteil dieser Methode ist, dass die Tests von realen bzw. potenziellen Benutzern durchgeführt werden. Dies kann unersetzlich gesehen werden, da ein Test mit echten Benutzern, mehr und wichtigere Informationen über die Benutzung von dem jeweiligen Interface gibt[Nielsen93].

Dabei werden die Testpersonen aufgefordert typische Aufgaben mit dem Testobjekt zu lösen. Während der Tests werden die Testpersonen beobachtet und gleichzeitig von einer Videokamera aufgenommen, um die Schwierigkeiten bei der Bedienung aufzudecken. Laut Nielsen[Nielsen93] besteht der Testablauf aus vier Phasen;

- **Vorbereitung:** Der Testleiter soll sicherstellen, dass der Testraum bereit für den Test ist. Dabei ist es wichtig, dass das Computer-System im Startzustand ist, und alle benötigten Testmaterialen wie Aufgabenzettel und Fragebögen für die Testperson vorliegen. Alle Dateien und Ergebnisse aus vorherigen Tests sollen auf einen anderen Rechner oder Verzeichnis verschoben werden.

- **Einführung:** In der Einführungsphase grüßt der Testleiter die Testpersonen, und gibt Informationen über den Testablauf und Testumgebung. Die Testpersonen sollen sich so bequem wie möglich fühlen, damit sie sich im Laufe des Tests normal und

natürlich verhalten. Es hilft z.B. die Testpersonen darauf hinzuweisen, dass der Testleiter keine Beteiligung an dem Produkt hat, damit die Testpersonen ohne Zweifel ihre Meinung sagen können. Nach der Einführung gibt der Testleiter den Testpersonen die Aufgabenzettel, und bittet sie sich diese durchzulesen, damit vor dem Test die Unklarheiten geklärt werden können.

- **Test:** Während der Tests sollen die Testpersonen die Aufgaben allein lösen. Es sollten keine positiven oder negativen Feedbacks vom Testleiter gegeben werden. Nur in Ausnahmefällen könnte der Testpersonen Hilfe angeboten werden. Z.B. wenn der Testperson im Laufe des Tests festhängt, oder wenn ein aus den vorherigen Tests bekanntes Problem auftaucht, könnte eine Ausnahme gemacht werden. Desweiteren wird der Test normal ausgeführt, und die Auffälligkeiten werden von dem Testleiter notiert.

- **Nachbesprechung:** Nach dem Test wird die Testperson gebeten, die Fragebögen auszufüllen. Danach kann ein Interview mit der Testperson für andere Kommentare und mögliche Verbesserungen durchgeführt werden. Dies kann bei der Redesignphase hilfreich werden, da jeder andere Benutzer verschiedene Ideen hat. Zuletzt sollte der Testleiter die Ergebnisse des Tests nachprüfen, um sicherzustellen, dass alles richtig durchgeführt wurde.

2.2.4 Laut Denken

Laut Denken ist eine Untersuchungsmethode, in dem die Testpersonen aufgefordert werden, während der Tests laut mitzudenken. Dabei sollen die Testpersonen alle Aufgabenschritte, ihre Gedanken und Erwartungen bis zur kleinsten Detail aussprechen. Das ermöglicht dem Testleiter die Auffälligkeiten sofort zu erkennen und zu notieren. Dieses Verfahren wurde zuerst als ein psychologisches Experiment vorgestellt, aber wird heutzutage mehr für Mensch-Computer-Interaktion benutzt[Nielsen93].

Allerdings erscheint vielen Personen diese Methode als unnatürlich, da sie nicht daran gewöhnt sind, während der Bewältigung einer Aufgabe laut zu denken. Das macht die Lösung einer Aufgabe nicht nur schwieriger, sondern kann auch negative Effekte auf das Resultat haben. Zunächst kann es dazu führen, dass die Aufgaben langsamer durchgeführt werden. Wichtig ist dass der Testleiter die Testperson immer daran erinnert laut mitzudenken, da die Testpersonen immer wieder vergessen können laut mitzudenken.

Think-Aloud-Methode kann in verschiedene Varianten durchgeführt werden:

- **Nebenläufig**: Dieser Variante erfordert, dass die Testpersonen noch während der Aufgabenbearbeitungsphase laut denken. Diese Methode gibt detaillierte Informationen über die Bearbeitung einer Aufgabe, da die Testpersonen jede bestimmte ausgeführte Aktionen Schritt für Schritt aussprechen, also ist das Feedback „real-time".

- **Retrospektiv**: Bei dieser Variante wird das Verhalten der Testpersonen während der Tests aufgezeichnet. Die Aufnahme wird dann von dem Testleiter und den Testpersonen angeschaut, und nach Erfüllen jeder Aufgabe versucht die Testperson seine Gedanken zu verbalisieren. Ein wichtiger Vorteil dieser Methode ist es, dass der Testleiter die Aufnahme pausieren und ohne Unterbrechung des schon durchgeführten Tests detaillierte Fragen stellen kann. Es besteht aber trotzdem das Risiko, dass die Testperson sich nicht mehr an alle Gedanken erinnern kann.

- **Konstruktive Interaktion**: Bei dieser Variante wird der Test nicht mit einem User sondern idealerweise mit zwei Usern durchgeführt. Auffällig ist es hier, dass die Testsituation natürlicher erscheint als bei einem Test mit nur einem User. Es fällt den Testpersonen leichter ihre Gedanken zu verbalisieren, während sie zusammen eine Aufgabe lösen, da sie dabei kommunizieren. Dadurch werden oftmals mehr Gedanken geäußert, als bei Untersuchungen nur mit einer Testperson. Ein Nachteil dieser Methode ist, dass nicht immer alle Testpersonen zueinander passen, und miteinander arbeiten können. Diese Methode wird meistens für Projekte eingesetzt, für welche es einfach ist eine Vielzahl von Testpersonen zu finden.

2.2.5 Fragebögen und Interviews

Eine der wichtigsten Methoden zur Evaluation von Software-Usability ist der Einsatz von Benutzerbefragung. Viele Aspekte der Usability sind am besten zu erheben, indem man Benutzer befragt[Nielsen93].

Fragebögen und Interviews sind eigentlich sehr ähnlich, da beide Methoden aus einer Menge von Fragen bestehen und die Antworten von Benutzern aufgenommen werden. Fragebögen werden entweder auf einem Papier oder interaktiv am PC beantwortet. Der Vorteil von Fragebögen ist, dass diese selbstständig, ohne Hilfe von anderen ausgefüllt werden können. Das führt dazu, dass der Benutzer seine Meinung ohne Beeinflussung wiedergeben kann. Laut [Nielsen93] sollten Fragebögen möglichst einfach formuliert sein und dem Nutzer durch präzise Fragestellungen keine Möglichkeit lassen, falsche oder nicht ausreichende Antworten geben zu können.

Im Gegensatz zu Fragebögen, werden bei einem Interview die Fragen persönlich von einem Interviewer dem Benutzer vorgelesen, und die Antworten werden wieder durch den Interviewer notiert. Diese Methode gibt dem Interviewer die Chance, bei unklar formulierten Antworten nachzufragen und präzisere Antworten zu bekommen. Weiterhin kann der Interviewer Folgefragen stellen, die sich aus der Antwort des Nutzers ergeben. Der Nachteil dieser Methode ist, dass die Interviews schwer zu analysieren sind, wenn bei der Befragung viele Folgefragen entstehen[Nielsen93]. Während eines Interviews ist es wichtig, dass der Interviewer neutral bleibt.

3 Beispiel Anwendung

Um den Usability-Test durchzuführen, wurde eine einfache Surface-Anwendung programmiert, welche die Informationen über einen ausgewählten Flug anzeigt. Da diese Anwendung mit Flügen zu tun hat, wird sie möglicherweise in einem Flughafen eingesetzt und von den Passagieren bedient. Deswegen sollte die Anwendung so einfach wie möglich bedienbar sein, damit alle Benutzer jeden Alters, mit oder ohne Erfahrung, sie ohne Komplikationen und Schwierigkeiten bedienen können.

Die Anwendung wurde mit Hilfe von dem Model-View-ViewModel-Entwurfsmuster programmiert. In diesem Kapitel werden; zuerst die MVVM-Muster, dann detailliert die Anwendung mit Klassendiagrammen, Anforderungen und Architektur erklärt.

3.1 Model-View-ViewModel

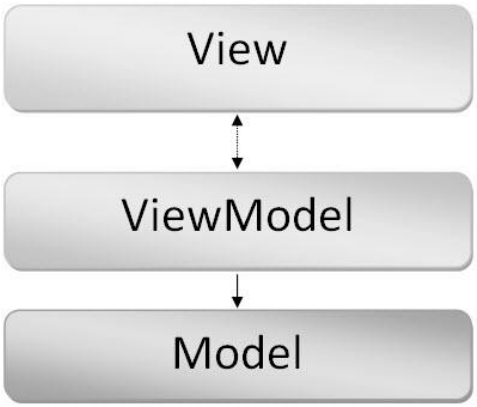

Abbildung 6 - MVVM-Muster

Das Model-View-ViewModel(MVVM) ist eine Variante des Model-View-Controller(MVC)-Konzeptes zur Trennung von Layout und Programmcode. MVVM wird in der Windows Presentation Founding (WPF) und Microsoft Silverlight angewendet[MVVM].

Erklärung der einzelnen Schichten können wie folgt definiert werden;

- **Model**: Wie in MVC, das **Model** enthält die darzustellenden Daten und entspricht der Geschäftslogik. Model ist unabhängig von der Benutzerschnittstelle.

- **View**: Es besteht aus visuellen Elementen wie; Fenster, Buttons, Grafiken und mehrere andere Control-Elemente von GUI. **View** ist die einzige Komponente, mit welcher der Benutzer interagiert. In MVVM ist der View immer aktiv. Das bedeutet; durch „events" und „data-binding", kann diese Komponente mit dem Model und ViewModel interagieren, die eigentlich im Hintergrund stehen.

- **ViewModel**: Das ViewModel steht als Wrapper für die Kommunikation von **Model** und **View**. Die genaue Bedeutung ist „Model of a View". Die Daten werden vom Model genommen, für das Data-Binding konvertiert, und durch Data-Binding an den View gebunden. Das **ViewModel** enthält auch Commands, die der View nutzen kann, um mit dem Model zu interagieren.

Der Zustand des Programms wird vom **Model** gespeichert. **View** erhält den Zustand durch Synchronisation mit dem **ViewModel**.

3.2 Anforderungen

In diesem Kapitel werden die funktionalen und nicht funktionalen Anforderungen der Beispielanwendung erläutert.

Anforderungen beschreiben die Bedingungen die eine Software erfüllen muss. Es gibt zwei Arten von Anforderungen.
Funktionale Anforderungen einer Software sind die fachliche Beschreibung eines Systems und beschreiben die Fähigkeiten eines Systems, die für eine fachliche Problemlösung benötigt werden. Die Beschreibung der funktionalen Anforderungen erfolgt beispielsweise in Form von Anwendungsfällen (sogenannten „Use-Cases").
Und **nicht funktionale Anforderungen** beschreiben die Qualitätseigenschaften, die ein System einhalten soll. Für diese Arbeit werden die allgemeinen Qualitätsmerkmale für Software-Produkte nach ISO 9126 vorgenommen. Diese Eigenschaften werden in folgender Tabelle aufgelistet und später noch in Kap 3.2.2 erläutert.

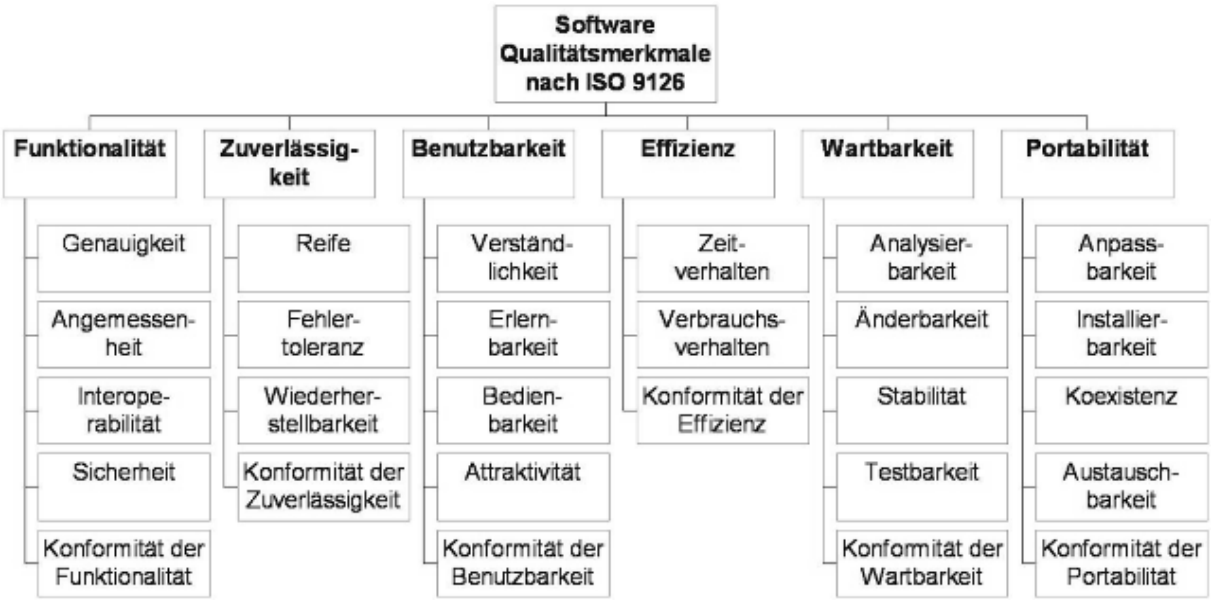

Abbildung 7 - ISO 9126 (Qualitätsmerkmale)

3.2.1 Funktionale Anforderungen

Die wichtigste funktionale Anforderung dieser Software ist, dass sie eine exemplarische Anwendung für „Casual Computing"(kompakte, einfache Anwendungen im Alltag) sein soll.

Die weiteren funktionalen Anforderungen dieser Software sind in Anwendungsfällen beschrieben. Bei jedem Anwendungsfall ist das beschriebene Ziel identisch mit der jeweiligen funktionalen Anforderung. Um die Anforderungen der Anwendung kompakter zu realisieren und alle möglichen Szenarien zusammen zu bündeln, werden in diesem Teilkapitel die folgenden Anwendungsfälle beschrieben:

- SelectOrigin (Abflugsort auswählen)
- SelectDestination (Ankunftsort auswählen)
- SelectFlight (Flug auswählen)
- ShowAircraftSeats (Flugzeugsitzplan anzeigen)

Die folgende Übersicht enthält alle Anwendungsfälle und deren zusammenhänge untereinander. Anwendungsfalldiagramme sind gut geeignet, um den gesamten Lebenszyklus eines Systems übersichtlich darzustellen.

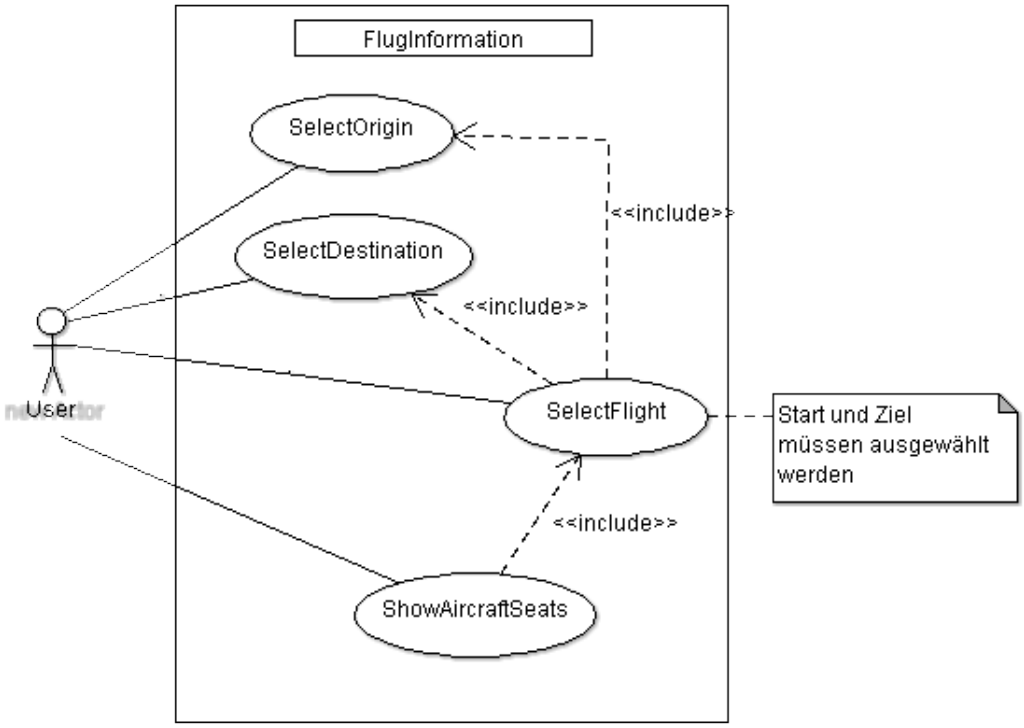

Abbildung 8 - Flug-Inf-Anwendungsfall

SelectOrigin(s. Anhang C.3.1):
Um mit einer Flugsuche anzufangen, muss ein Startflughafen ausgewählt werden. Wenn die Software im Startzustand ist, wird die erste Eingabe als Startflughafen erkannt.

SelectDestination(s. Anhang C.3.2):
Um eine Flugsuche abzuschließen, muss nach der Startflughafenauswahl der Endflughafen ausgewählt werden. Erst wenn der Abflugsort ausgewählt ist, wird die zweite Eingabe als Ankunftsort erkannt.

SelectFlight(s. Anhang C.3.3):
Erst wenn die Start-und-Endflughäfen ausgewählt sind, erscheint ein kleines Pop-Up-Fenster mit den gefundenen Flügen, um einen gewünschten Flug auszuwählen.

ShowAircraftSeats(s. Anhang C.3.4):
Nach der Flugsuche hat der User die Möglichkeit sich den Sitzplan für den ausgewählten Flug anzugucken.

3.2.2 Nicht funktionale Anforderungen

In diesen Abschnitt werden die nicht funktionalen Anforderungen der Beispielanwendung beschrieben. Einige der oben genannten Qualitätsmerkmale werden einzeln betrachtet, und später ausgewertet.

3.2.2.1 Benutzbarkeit (Bedienbarkeit)

Diese Eigenschaft beschreibt wie einfach ein System zu bedienen ist. Leichte Bedienbarkeit wird generell durch anwenderfreundliche Gestaltung der Benutzeroberfläche und einfache Interaktion gewährleistet.

3.2.2.2 Benutzbarkeit (Erlernbarkeit)

Der Aufwand für den Benutzer, um eine Software zu begreifen, beschreibt die Erlernbarkeit eines Systems. Hilfsmittel wie Dokumentationen mit Beispielen und eingebaute Hilfefunktionen sorgen für einen leichten Einstieg. Auch wenn die Software klein und einfach benutzbar ist, können sich die Benutzer die Details der Anwendung einfacher merken. Damit ist die Erlernbarkeit gewährleistet.

3.2.2.3 Effizienz

Effizienz ist die Fähigkeit eines Software-Produktes angemessene Leistung, im Bezug auf die zur Verfügung gestellten Ressourcen und unter festgelegten Bedingungen, zu erbringen. Ein effizientes System soll mit möglichst wenigen Hardwareanforderungen schnell und laststabil sein.

3.2.2.4 Funktionalität (Angemessenheit)

Die Fähigkeit eines Systems geeignete Funktionen zur Verfügung zu stellen, die für die Lösung von Software spezifischen Aufgaben nötig sind. Die Erfüllung funktionaler Anforderungen einer Software gewährleistet diese Eigenschaft.

3.2.2.5 Wartbarkeit

Diese Eigenschaft beschreibt den Schwierigkeitsgrad, künftige Erweiterungen, Änderungen und Wartungen an einem Software-Produkt durchzuführen. Durch aussagekräftige Methodennamen, eine gute Dokumentation und einem stark gegliedertem Aufbau kann dieses Qualitätsmerkmal erfüllt werden.

3.2.2.6 Portabilität

Eine Software erfüllt diese Eigenschaft, wenn sie ohne großen Aufwand bei der Portierung von einer Plattform zu einer anderen Plattform mit unterschiedlicher Architektur, Prozessor, Compiler oder Betriebssystem lauffähig bleibt.

3.3 Die Fluginformations-Anwendung

Durch den technischen Fortschritt erhöht sich die Funktionalität von Multi-Touch Geräten, so dass es immer neue Einsatzgebiete für solche Geräte und für Everyday-Applikationen gibt. Als Beispiel können Flughäfen gezeigt werden, bei denen Zeit eine große Rolle spielt. Um den Usability-Test durchzuführen wurde eine einfache Surface-Anwendung programmiert, die für die tägliche Benutzung in einem Flughafen gedacht ist, und von den Flugpassagieren benutzt wird. Da die Anwendung immer von unterschiedlichen Arten von Personen bedient wird, sollte die Anwendung so einfach wie möglich zu bedienen sein. Die Zeit spielt immer eine große Rolle in Flughäfen für die Passagiere, und deswegen sollte die benötigte Anwendungsdauer so kurz wie möglich gehalten werden.

Was die Anwendung macht, ist wichtige Informationen wie;

- Abflugsort
- Abflugzeit
- Ankunftsort
- Ankunftszeit
- Fluggesellschaft
- Flugnummer
- Gate Nummer
- Flugzeug (Sitzplan)

über einen bestimmten ausgewählten Flug anzuzeigen. Die Daten sind lokal programmiert, und müssen auch lokal in die Anwendung eingegeben werden.

3.3.1 Software-Architektur

Die Architektur der Beispielanwendung basiert auf dem MVVM-Muster. Wie in Kap 3.1 erklärt, sind bei der Software die Business-und-Präsentationschichten getrennt, und die Kommunikation zwischen diesen beiden Komponenten geschieht über das ViewModel. Bei der Darstellung der Architektur wurden die andere ViewModel-Klassen(AircraftViewModel und CityViewModel) außer FlugAuswahlViewModel ausgelassen.

Die benötigten Daten für die Software werden von der „fluginf.Core"-Komponente zur Verfügung gestellt. Die Klassen der Model-Komponente sind als einzige Model-Klasse dargestellt, da sonst die Architektur zu unübersichtlich wäre. Später in diesem Kapitel werden die einzelnen Klassen erläutert.

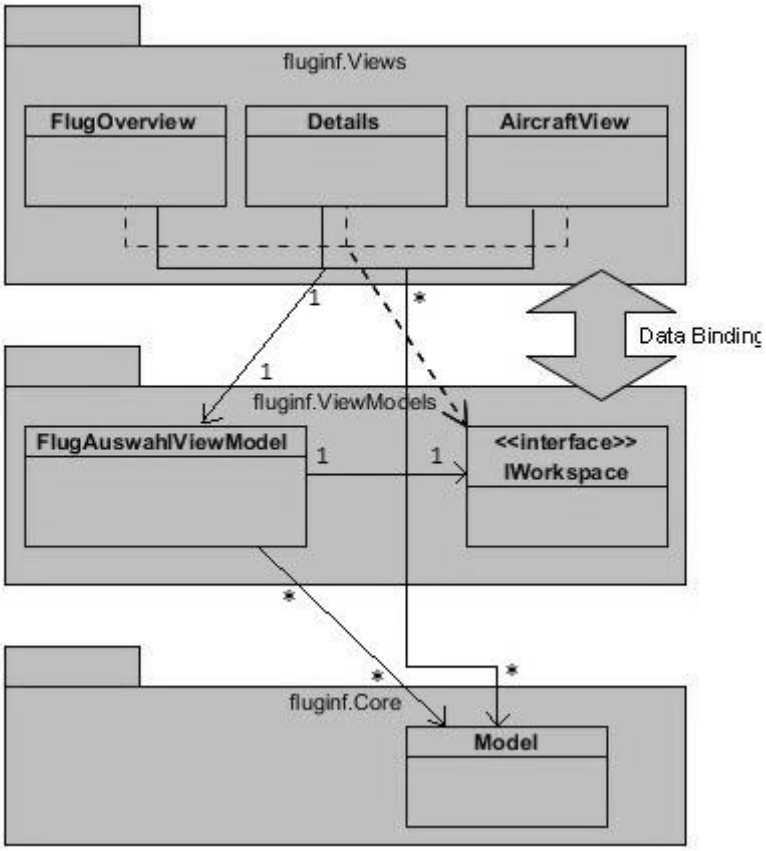

Abbildung 9 - Software-Architektur

Die „fluginf.Views"-Komponente ist für die Anzeige der Benutzeroberfläche zuständig. Bei der Abbildung werden die drei View-Klassen(FlugOverview, Details, AircraftView) und ihre Beziehungen dargestellt. Die View-Klassen können direkt durch Command-und-Data-Binding Befehle der ViewModel-Klasse ausführen, aber das ViewModel kann nur über das IWorkspace-Interface mit View-Komponente kommunizieren. Die View-Klassen können auch mit Model-Klassen zusammenarbeiten, aber dies soll möglichst vermeidet werden, da die komplexe Operationen durch ViewModel ausgeführt werden sollen. Die Kommunikation von Model-Klassen mit View-und-ViewModel-Klassen geschieht durch Events. Wenn eine Änderung auftritt, werden die Daten zwischen View und Model mit „Property Chance Events" über das ViewModel synchronisiert.

3.3.2 Klassen

Es wurden insgesamt 4 Packages in diesem Projekt erzeugt;

- **fluginf**: Main-Package. Enthält die .exe-Datei. Dient hauptsächlich zum Starten der Anwendung.
- **fluginf.Core**: Enthält die .cs-Klassen (Modells). Dient zur Geschäftslogik. Ist unabhängig von Views.
- **fluginf.Views**: Enthält die User-Control-Klassen. Dient zur Darstellung der Daten aus dem Modell.
- **fluginf.ViewModels**: Enthält die ViewModels. Dient zur Kommunikation zwischen Views-Package und Core-Package.

Als nächstes werden die wichtigen Klassen von den jeweiligen Packages erklärt.

3.3.2.1 Fluginf

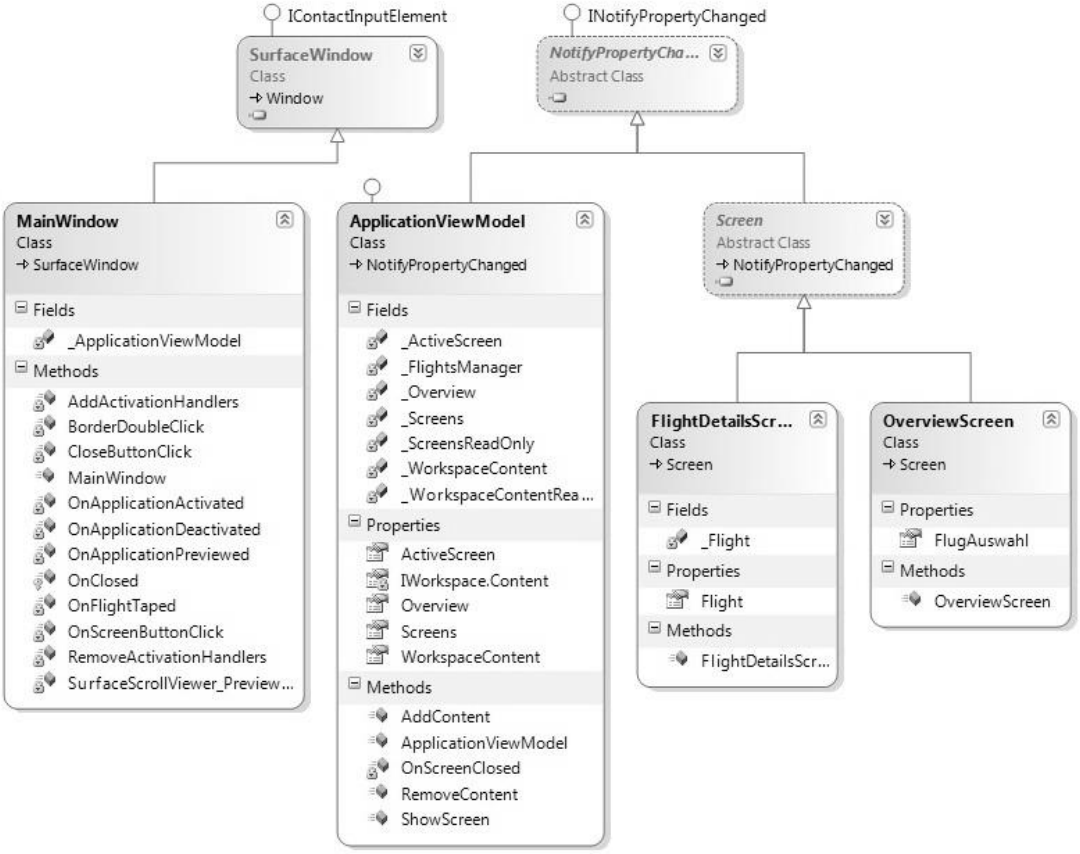

Abbildung 10 - Fluginf-Package

MainWindow-Klasse: Die Start-Up-Klasse der Anwendung. Wenn das Programm gestartet wird, wird zuerst diese Klasse ausgeführt und die Arbeitsfläche für die Anwendung erstellt.

ApplicationViewModel-Klasse: Diese Klasse ist für die Erstellung der Daten und einzelnen Screens zuständig. Durch Ausführen dieser Klasse werden die voreingegebenen Flüge erstellt, und der OverviewScreen wird zur Benutzung bereitgestellt.

FlightDetailsScreen und **OverviewScreen** sind die .cs-Klassen für die jeweiligen Screens. OverviewScreen wird nur ein Mal erstellt, und FlightDetailsScreen wird für jeden ausgewählten Flug neu erstellt. Durch Ausführen der OverviewScreen-Klasse, wird eine ViewModel-Klasse erstellt, die für die Erstellung der Städte und das Suchen von Flügen zuständig ist (FlugAuswahlViewModel-Klasse).

3.3.2.2 Fluginf.Core

Dieses Package enthält die Core-Dateien. Die für die Flug-Information benötigten Daten werden in dieser Klasse erzeugt. Die Klassen in diesem Package haben nichts mit der Anzeige zu tun.

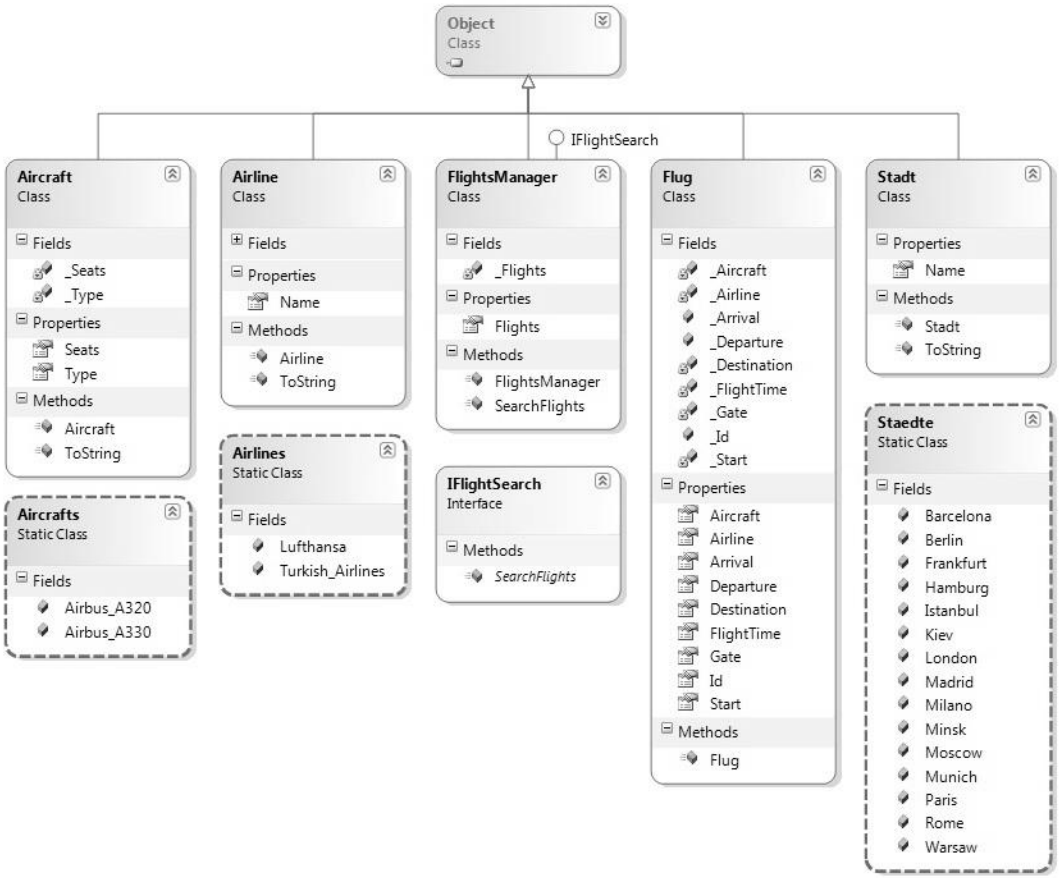

Abbildung 11 - Fluginf.Core-Package

Die Klassen; **Aircraft**, **Airline**, **Flug** und **Stadt** sind die Core-Klassen für die Erstellung der Flüge. In diesen Klassen werden sämtliche Eigenschaften wie Flugzeugtyp, Fluggesellschaft, Fluginformationen und Stadtnamen beschrieben.

Die Klassen; **Aircrafts**, **Airlines** und **Staedte** sind statische Klassen für die Erzeugung der jeweiligen Daten.

FlightsManager-Klasse: Diese Klasse ist für die Erzeugung der Flüge zuständig. Die Daten, die in einen Flug enthalten sind, werden mit Hilfe der statischen Klassen hier zusammengefügt und in einer Liste gespeichert. Die Suche für den ausgewählten Flug geschieht auch in dieser Klasse.

3.3.2.3 Fluginf.ViewModels

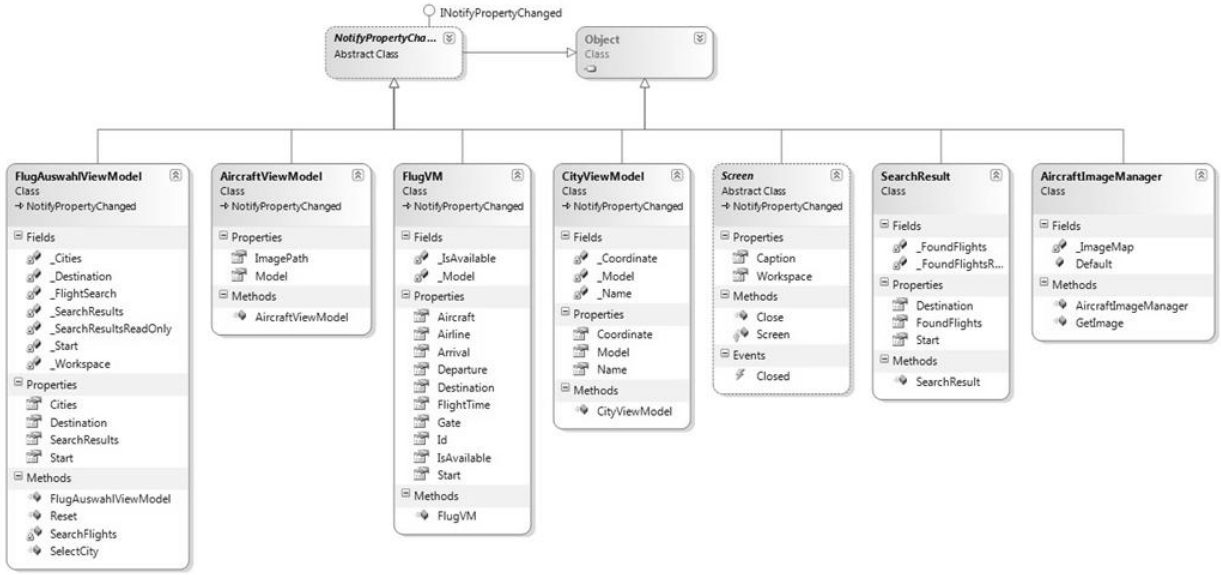

Abbildung 12 - Fluginf.ViewModels-Package

FlugAuswahlViewModel-Klasse: Diese Klasse wird am Anfang der Anwendung durch die OverviewScreen-Klasse erstellt. Sie dient zur Auswahl des Abflugs- und Ankunftsorts, sowie als Wrapper für die Flugsuche. Für die Flugsuche wurde keine extra Methode geschrieben, sondern sie geschieht erst nach der Auswahl des Ankunftsorts automatisch. Bei der Auswahl des Abflug-und-Ankunftsorts müssen die Städte richtig auf der Karte positioniert werden. Hierfür werden die x und y-Koordinate für jede Stadt definiert und in einen List gespeichert.

AircraftViewModel-Klasse: Diese Klasse dient als Wrapper-Klasse um den Flugzeugsitzplan anzuzeigen. Sie wird jedes Mal von der View aufgerufen, wenn sich der Benutzer den Sitzplan anschauen möchte.

FlugVM-Klasse und CityViewModel-Klasse: Diese sind die Wrapper-Klassen für die Flüge und Städte. Alles was einen Flug oder Stadt als Eigenschaft hat, wird in dieser Klasse auch beschrieben und die Daten werden von der Core-Klasse Flug.cs oder Stadt.cs geholt.

AircraftImageManager-Klasse: Die Idee für jedes bestimmte Flugzeug den Sitzplan anzuzeigen wird mit Hilfe dieser Klasse realisiert. Hier werden die Flugzeuge und dazugehörige Sitzplan-Bilder in ein „Dictionary" gespeichert. Wenn der Benutzer einen Flug ausgewählt hat, und sich den Sitzplan anschauen möchte, wird der für diesen Flug gespeicherte Flugzeugsitzplan von dem Dictionary geholt und angezeigt.

3.3.2.4 Fluginf.Views

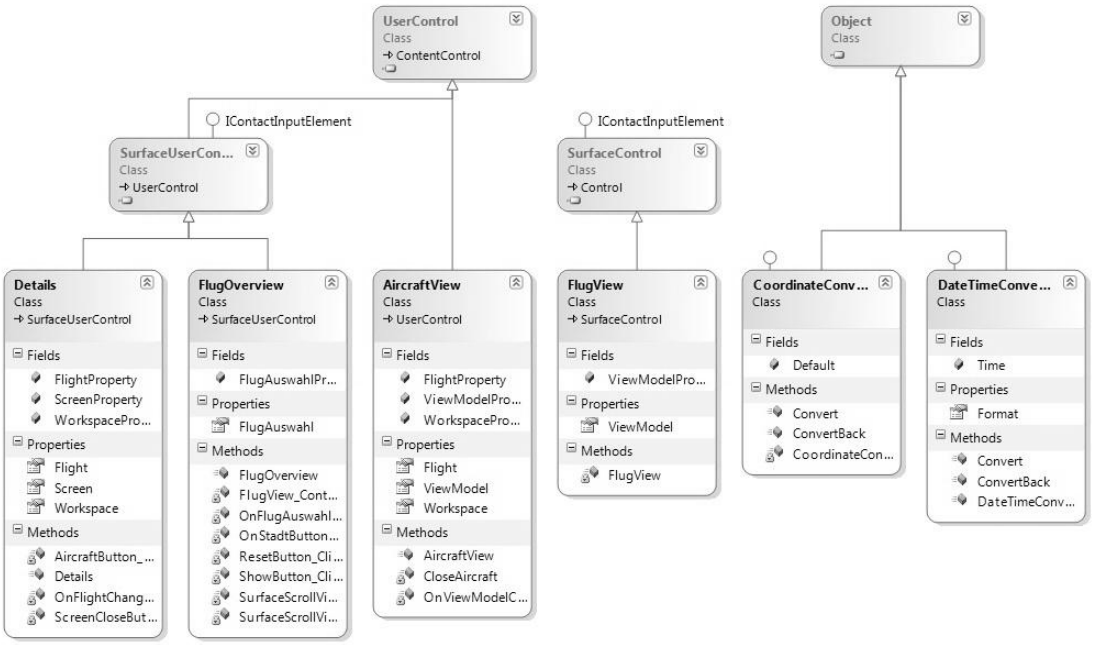

Abbildung 13 - Fluginf.Views-Package

FlugOverview-Klasse: Diese Klasse ist für die Anzeige der Karte zuständig. Da die Karte vergrößerbar/verkleinerbar ist, gab es ein Problem mit der Positionierung der Städte auf der Karte. Wenn die Kartengröße auf Standard ist, sind die Entfernungen zwischen den Städten richtig. Aber wenn die Karte größer oder kleiner wurde, blieben die Städte in dem Punkt wo sie definiert wurden. Das Problem wurde durch einen **CoordinateConverter** gelöst. Wenn die Größe von der Karte geändert wird, wird eine Berechnung (coordinate * actualSize / originalSize) ausgeführt und die Städte werden dynamisch neu positioniert.

Details-Klasse: ist, wie auch die FlugOverview-Klasse, eine SurfaceUserControl Klasse(xaml). Sie dient zur Anzeige der Fluginformation. Nach der Auswahl des Fluges wird diese Klasse ausgeführt und ein neuer Screentab für den ausgewählten Flug erstellt. Für jeden Flug wird eine neue Details-Klasse erstellt.

AircraftView-Klasse: Diese Klasse dient zur Anzeige des Flugzeugsitzplans. Wenn der AircraftButton in der Details-Klasse geklickt/getippt wird, wird diese Klasse ausgeführt und es wird ein Pop-Up-Fenster mit dem Sitzplan angezeigt.

3.4 Durchlauf der Anwendung

In diesem Teil des Kapitels wird ein möglicher Durchlauf der Anwendung mit Screenshots und einem Anwendungsfall erklärt.

Die Anwendung startet mit einer Kartenübersicht, auf welcher die Städte als klickbare Buttons angezeigt werden. Der Benutzer wählt zuerst den Abflugsort durch einen Klick auf die jeweilige Stadt. Zunächst wird der gewünschte Ankunftsort ausgewählt, und es erscheint eine rote Fluglinie zwischen den ausgewählten Städten. Um einen neuen Flug auszuwählen muss der Benutzer den „Reset"-Button klicken, und mit der Auswahl von Abflug-und-Ankunftsort vorne den beginnen.

Abbildung 14 - Karten-Übersicht

Nach der Auswahl des Ankunftsorts erscheint ein Pop-Up-Fenster mit einer Liste von Flügen zwischen diesen Städten. Ganz oben auf dieses Fenster stehen die Abflugs-und-Ankunftsorte, sowie die Anzahl der gefundenen Flüge für diese Strecke. Die einzelnen Elemente in dieser Liste erhalten die wichtigsten Informationen von einem Flug, wie Abflug-und-Ankunftszeit und Fluggesellschaft. Durch einen Klick/Tap wird der gewünschte Flug ausgewählt, und ein neues Fenster mit dem Namen „Details" angezeigt. Die ganzen Fluginformationen werden in diesem Fenster als Textblöcke angezeigt.

Abbildung 15 - Flug-Übersicht

Der Benutzer hat auch noch die Möglichkeit sich den Sitzplan für diesen Flug anzugucken. Durch einen Klick auf den Aircraft-Button erscheint ein Pop-Up-Fenster mit dem Sitzplan von dem Flugzeug für diese Strecke. Die Darstellung des Sitzplans enthält die Sitzreihen und die einzelne Sitzplätze. Die Sitzplätze werden mit 4 verschiedenen Farben angezeigt(Rot, Grün, Gelb und Weiß). Die roten Sitzplätze sind dabei zu vermeiden, da diese Sitzplätze wenig Fußraum haben und nicht nach hinten verstellbar sind. Die gelben Sitzplätze haben entweder nicht genügend Platz oder sind nicht nach hinten verstellbar. Die grünen Sitzplätze sind vorzuziehen, da diese Sitzplätze sowohl genügend Raum haben als auch nach hinten verstellbar sind. Die weißen Sitzplätze sind Standartsitzplätze für dieses Flugzeug. Die Flugzeugsitzplanbilder sind von der Webseite www.seatguru.com genommen worden.

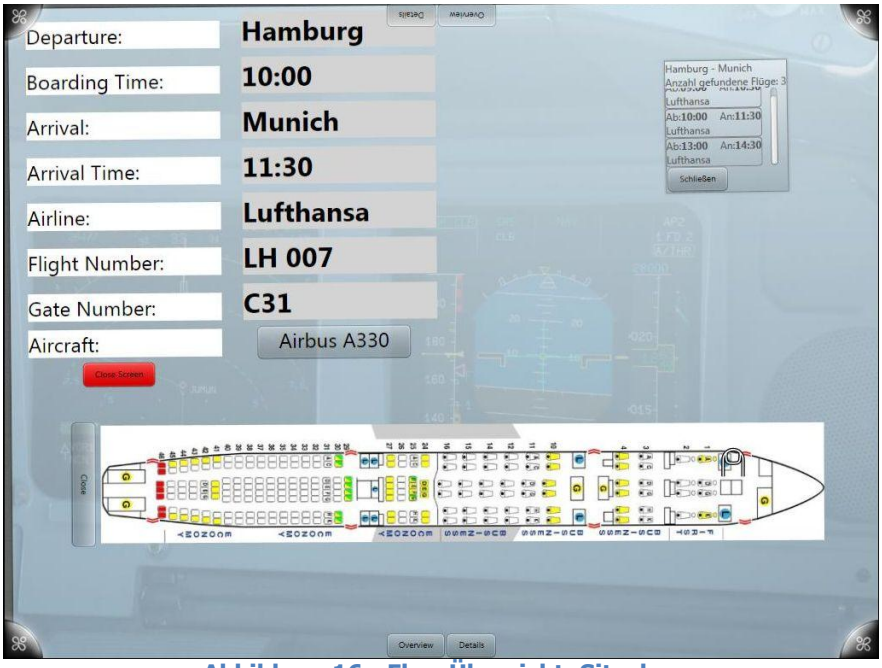

Abbildung 16 - Flug-Übersicht_Sitzplan

4 Vorbereitung der Usability-Tests

Bei diesem Kapitel, geht es um die Vorbereitung der Usability-Tests, die im Rahmen dieser Studie durchgeführt werden. Dazu gehören die Testziele, Testaufgaben, Testpersonen und die Auswahl der verwendeten Testmethoden.

4.1 Testziele

Das angestrebte Ziel der Usability-Untersuchungen, die bei dieser Arbeit durchgeführt werden, ist es zum einen Erkenntnisse über die möglichen Usability-Testmethoden zu gewinnen, die für die Surface-basierten alltäglichen Anwendungen für gelegentliche Nutzer geeignet sind, und zum anderen Usability-Probleme der Beispielanwendung zu identifizieren.

Wie später in Kap. 4.4 erwähnt wird, wurden die Usability-Tests in dem Raum durchgeführt wo sich MS Surface befindet. Die Tests wurden innerhalb von zwei Wochen abgeschlossen. Die einzelnen Tests haben durchschnittlich 20 bis 30 Minuten gedauert. Die für die Untersuchungen benötigte Hardware ist, MS Surface, eine Videokamera für die Aufnahme, und einen Laptop um die Surface-Software und Flughafen-Webseite vergleichen zu können. Für die Tests wurde geplant, mindestens 8 Testpersonen zu finden, die möglichst unterschiedlichen Eigenschaften (Alter, Erfahrung, Geschlecht) haben. Erst wenn die Testpersonen alle Aufgaben erfolgreich gelöst, und alle Bemerkungen und Probleme die ihnen aufgefallen sind, erwähnt haben, gelten die Usability-Tests als „erfolgreich abgeschlossen". Nach jedem Test werden die Testaufnahmen gesichert, um später bei der Auswertung gesichtet werden zu können.

Um einen Konzept für die Usability-Untersuchung von MS Surface zu erstellen wird die GQM-Methode benutzt. GQM (Goal-Question-Metric) ist ein Vorgehen, um in einen Softwareprojekt Ziele mit geeigneten Metriken verfolgen zu können. In Abbildung 15 ist die hierarchische Struktur der GQM-Methode für diese Arbeit zu sehen.

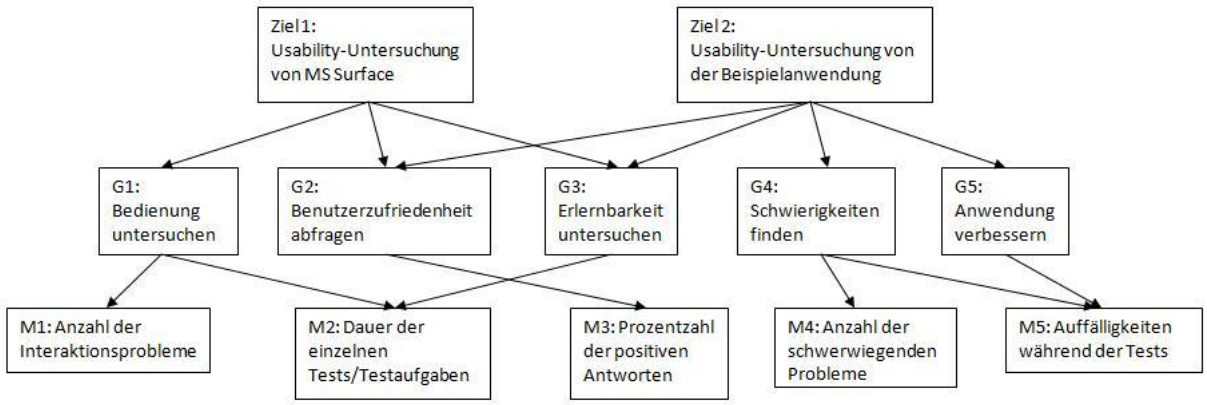

Abbildung 17 - GQM für die Usability Test

Für die Anwendung der GQM werden die Hauptziele in Unterziele zerlegt, und Metriken gesucht, um möglichst viele der Ziele erreichen zu können. Für diese Usability-Untersuchung wurden zwei Hauptziele und fünf Unterziele mit den jeweiligen Metriken geplant.

Ziel 1: Usability-Untersuchung von MS Surface
- G1: Bedienung untersuchen
  - M1: Anzahl der Interaktionsprobleme
  - M2: Dauer der einzelnen Tests
- G2: Benutzerzufriedenheit abfragen
  - M3: Prozentzahl der positiven Antworten
- G3: Erlernbarkeit untersuchen
  - M2: Dauer der einzelnen Testaufgaben

Ziel 2: Usability-Untersuchung der Beispielanwendung
- G2: Benutzerzufriedenheit abfragen
  - M3: Prozentzahl der positiven Antworten
- G3: Erlernbarkeit untersuchen
  - M2: Dauer der einzelnen Testaufgaben
- G4: Schwierigkeiten finden
  - M4: Anzahl der schwerwiegenden Probleme
  - M5: Auffälligkeiten während der Tests
- G5: Anwendung verbessern
  - M5: Auffälligkeiten während der Tests

Um die Unterziele erreichen zu können, müssen die folgenden Fragen gefragt werden:

G1: Bedienung untersuchen
- F1: Wie viele Testpersonen hatten Probleme mit der Bedienung?
- F2: Wie lange haben die einzelnen Testpersonen gebraucht?

G2: Benutzerzufriedenheit abfragen
- F3: Wie viele Testpersonen waren mit der Software zufrieden?

G3: Erlernbarkeit untersuchen
- F2: Hat sich die Dauer einzelner Aufgaben pro Testperson während der Tests verbessert?

G4: Schwierigkeit finden
- F4: Wie viele Probleme haben die Testpersonen gemeldet?
- F5: Welche Probleme sind mir während der Tests aufgefallen?

4.2 Testaufgaben

Um die Benutzbarkeit einer Software einschätzen zu können, werden die Testpersonen gebeten, gezielt gewählte Aufgaben im Rahmen von Kernfunktionen der Software zu lösen. Bei einem aufgabenorientierten Usability-Test ist es wichtig, dass die Aufgaben so natürlich wie möglich gehalten werden. Menschen verhalten sich natürlicher, wenn sie sich ein Szenario vorstellen sollen, anstatt Anweisungen zu befolgen. Hauptsache ist es, die Testaufgaben so einfach wie möglich zu halten, und das Szenario der Phantasie den Testpersonen zu überlassen.

Die Fragen sind besser schriftlich an die Testpersonen zu richten, da man die gesamte Aufgabenstellung nur schwer im Kopf behalten kann. Die Aufgaben sollen kurz genug sein, um in einem bestimmten Zeitraum gelöst werden zu können, aber auch nicht so kurz, dass sie trivial werden. Da die Testpersonen manchmal vor den Tests aufgeregt sein können, hilft es die erste Aufgabe leicht zu stellen, damit die Testperson leicht in den Test einsteigen kann.

Um die Benutzbarkeit von dem Flug-Informationssystem und Microsoft Surface einzuschätzen, wurden während der Tests insgesamt 7 Testaufgaben (s. Anhang C.1) gestellt. Zwei von den Sieben Aufgaben sind am Notebook zu lösen, damit ein Vergleich zwischen einer Flughafen-Webseite und dem Flug-Informationssystem möglich wird. Die Aufgaben wurden so eingestellt, dass die Testpersonen jede wichtige Funktion der Software mindestens einmal testen müssen. Dabei wurden auch Surface-spezifische Eigenschaften getestet.

4.3 Testpersonen

Die Testpersonen haben den größten Einfluss auf einen Usability-Test. Bei der Auswahl der Testpersonen muss der Testleiter daran denken, dass das Endresultat eines Usability-Tests mehr vom Testperson-Typ abhängt. Eine erfahrene Testperson findet vielleicht mehr Usability-Probleme als eine unerfahrene Testperson, aber es kann auch sein, dass eine unerfahrene Testperson andere Usability-Probleme findet, die eine erfahrene Testperson nicht findet.

Da die Usability-Test-Kosten stark von Anzahl der Testpersonen abhängen, darf die Anzahl der Testpersonen nicht zu hoch, aber auch nicht zu gering sein. Eine Testperson verursacht natürlich die geringsten Kosten, dafür werden aber auch weniger Usability-Probleme gefunden als sonst. Obwohl unterschiedliche Testpersonen meistens dieselben Probleme finden, gibt es immer kleine Unterschiede in den Bemerkungen. Es kann sein, dass ein Usability-Problem nicht der ersten Testperson aufgefallen ist, aber da die Menschen immer unterschiedlich sind, ist es möglich, dass dieses Problem während späterer Tests gefunden wird.

Nach Nielsen [Nielsen93] findet man 80% der Usability-Probleme mit 5 Testpersonen. Die Formel für die Berechnung von Prozentzahl lautet: **N(1-(1-L)^n)**, wobei **N** die Anzahl der gesamten Usability-Probleme, **n** die Anzahl der Testpersonen, und **L** der Anteil der durch eine Testperson gefundene Usability-Probleme ist. Generell hat **L** den Wert 31%; das ist der durchschnittliche Anteil von Usability-Problemen, die bei vorherigen Usability-Tests gefunden wurden.

4.4 Auswahl der Testmethoden

Um die Benutzbarkeit von MS Surface zu messen, lassen sich die vorhandene Usability-Techniken nicht leicht einsetzen. In einem voll ausgestatteten Usability-Labor werden meistens Anwendungen oder Webseiten getestet, wobei die Maus-und Tastatureingaben gespeichert werden, und per Eye-Tracking das Blickverhalten des Anwenders festgehalten wird. Diese beiden Techniken sind bei dem MS-Surface-Test leider schwer zu realisieren. Da MS Surface ohne Maus und Tastatur bedient wird, ist es unmöglich Tastatureingabelogging oder Maustracking-Technik zu verwenden. Theoretisch gesehen wäre Eyetracking realisierbar, aber da die Benutzer von MS Surface nicht die ganze Zeit still bleiben, besser gesagt ihr Kopf viel bewegen müssen, wäre diese Technik auch schwer einzusetzen. Zusätzlich ist es auch unpraktisch ein 90-Kilo schweren Tisch hin und herzutragen.

Aus allen diesen Gründen wurde entschieden, den Test in dem Raum durchzuführen, wo sich MS-Surface befindet. Desweiteren wird die Testperson gebeten, während des Tests laut mitzudenken, und gleichzeitig wird sie von Testleiter beobachtet und die Auffälligkeiten werden notiert. Die Methode „lautes Denken" wurde meistens nebenläufig aber auch retrospektiv (s. Kap 2.2.4) eingesetzt. Somit können die Usability-Probleme, die nicht bei dem Test notiert oder verstanden wurden, wiedergefunden werden. Nach jedem Test werden zwei Fragebögen (Microsoft Surface & Flug-Informationssystem) durch die Testperson ausgefüllt. Zum Schluss findet ein Nachinterview statt, um der Testperson zu erlauben, ihre Meinung über das Testobjekt objektiv auszudrücken. Während des Interviews werden auch noch die notierten Auffälligkeiten weiter besprochen, um Unklarheiten und Verständnisprobleme aufzuheben.

5 Durchführung der Usability-Tests

Im folgenden Kapitel wird der Ablauf der Usability-Tests geschildert. Die Testumgebung und die Testpersonen werden vorgestellt, und der generelle Ablauf von den Usability-Tests wird erläutert.

5.1 Testumgebung

Abbildung 18 - Raum 1088

Die Usability-Tests wurden im Raum 1088 der HAW durchgeführt, wo sich Microsoft Surface befindet. In den Raum wurde eine Videokamera aufgestellt, um die Tests aufzuzeichnen. Da eine Bildschirmaufnahme wie bei Eye-Tracking nicht verfügbar ist, wurde die Videokamera so aufgestellt, dass der Surface-Bildschirm und Teil der Testperson erkennbar ist. (Zur Wahrung der Anonymität)

Abbildung 19 - Videokamera-Position

Da der Raum nicht perfekt für eine Usability-Untersuchung geeignet ist, gab es einige Schwierigkeiten, wie z.B. externe Geräusche die auftraten. Die Usability-Labore sind normalerweise schalldicht, und es stehen mehrere Mikrofone zur Verfügung, um eine klare Soundaufnahme möglich zu machen. Da in meiner Situation nur das Mikrofon der Videokamera im Betrieb war, und die Videokamera 2-3 Metern entfernt von den Testpersonen stand, sind externe Geräusche beim Wiederspielen von Videos zu hören.

5.2 Testpersonen

Bei dieser Arbeit haben insgesamt neun Testpersonen den Testgegenstand untersucht. Dabei handelte es sich um sechs Studenten des Informatik-Departments (Angewandte Informatik & Technische Informatik) der HAW Hamburg sowie drei Personen aus meinem persönlichen Umfeld.

In der folgenden Tabelle ist eine gesamte Übersicht der Testpersonen und ihrer Eigenschaften zu sehen. Zwei der Testpersonen hatten sich schon mit Microsoft Surface beschäftigt, sechs der Testpersonen hatten vorher an keinem Usability-Test teilgenommen, und alle Testpersonen hatten schon Erfahrung mit Multi-Touch Geräten. Von diesen Merkmalen ist Surface-Erfahrung das wichtigste. Hilfreich und entscheidend wäre es auch wenn ich Testpersonen gefunden hätte, die keine Multi-Touch-Gerät Erfahrung haben, um präzisere Schlüsse ziehen zu können.

Testperson	Alter	Geschlecht	Surface-Erfahrung	Multi-Touch-Erfahrung	Usability-Erfahrung
t1	26	m	Ja	Ja	Ja
t2	33	m	Nein	Ja	Nein
t3	28	m	Nein	Ja	Nein
t4	33	m	Nein	Ja	Nein
t5	26	m	Ja	Ja	Ja
t6	36	m	Nein	Ja	Nein
t7	25	w	Nein	Ja	Nein
t8	25	m	Nein	Ja	Nein
t9	29	m	Nein	Ja	Ja

Abbildung 20 - Testpersonen

Bei der Auswahl der Testpersonen wurde darauf geachtet, möglichst unterschiedliche Typen von Testpersonen einzusetzen, um zu beobachten, ob jemand mit Erfahrung andere Ergebnisse liefert, als jemanden ohne Erfahrung. Als Endergebnis wurden insgesamt drei verschiedene Gruppen von Testpersonen rekrutiert, die;

- sowohl Surface-Erfahrung, als auch Multi-Touch und Usability-Erfahrung,
- keine Surface-und-Usability-Erfahrung, aber Multi-Touch-Erfahrung,
- keine Surface-Erfahrung, aber Multi-Touch und Usability-Erfahrung

haben. Bei der Auswertung können später die Ergebnisse nach Gruppen aufgelistet werden.

5.3 Testablauf

Die Testpersonen wurden zuerst gegrüßt, und mit dem Testablauf bekannt gemacht. Da die Testpersonen während der Tests aufgenommen werden, wurden sie gebeten eine Einverständniserklärung zu unterschreiben. Einige Testdaten wie; Testzeitpunkt, Nummer des Tests und Informationen über Testpersonen(Alter, Erfahrung) wurden gesammelt. Die Probanden hatten die Möglichkeit die Testaufgaben vor Beginn des Tests durchzulesen, um Unklarheiten oder Fragen im Vorfeld zu klären.

Nach Aufnahmebeginn haben die Testpersonen angefangen die Aufgaben (s. Anhang C.1) zu lösen. Sie wurden gebeten während des Tests laut zu denken. Während der Tests befand ich mich hinter der Videokamera, wo ich sowohl die Testpersonen als auch den Surface-Tisch beobachten konnte. Somit hatte ich auch die Möglichkeit Hilfe anzubieten, falls die Testperson mit einer Aufgabe nicht weitergekommen ist. Dies war auch einige Male nötig.

Nachdem die Testpersonen alle Testaufgaben erfolgreich gelöst haben, wurden sie gebeten zwei Fragebögen (s. Anhang C.2); eine für die Beispielanwendung und eine für Microsoft Surface, auszufüllen. Daraufhin fand ein Nachinterview statt, worin die Testpersonen ihre Meinungen objektiv äußern konnten. Im Nachinterview werden auch Auffälligkeiten hinterfragt, die dem Testleiter während der Tests aufgefallen sind. Diese Gespräche sind für jede Testperson individuell und hängen von den Antworten der Testpersonen ab. Am Ende des Interviews wird den Testpersonen eine Reihe von Fragen gestellt, die für jede Testperson gleich sind. Zum Schluss wird die Testperson verabschiedet, und die Testvideos werden gesichert.

Zusammengefasst sieht der Testablauf wie folgt aus:

- **Einstieg**
  - Begrüßung und Einstieg der Testpersonen
  - Testdaten sammeln
- **Test**
  - Beobachtung der MS Surface spezifische Auffälligkeiten
  - Beobachtung der Anwendungsspezifische Auffälligkeiten
  - Lautes Denken
- **Fragebögen**
  - Fragebogen für MS Surface ausfüllen
  - Fragebogen für Beispielanwendung ausfüllen
- **Nachinterview**
  - Auffälligkeiten abfragen
  - Gesamt-Eindruck

5.4 Testdauer

Wie in Kapitel 4.1 erwähnt, wurde geplant, die einzelnen Tests innerhalb von 20 bis 30 Minuten abzuschließen. In folgender Tabelle steht die Test-und-Aufgabendauer von jeder Testperson. Die Fragen 3, 4 und 5 und Fragen 6 und 7 sind hier jeweils zusammengefasst wie eine Aufgabe dargestellt worden, da die Aufgaben voneinander abhängen. Die Daten werden später in Kapitel 6 zur Auswertung benutzt.

	Gesamtdauer	Aufgabe 1	Aufgabe 2	Aufgabe 3,4,5	Aufgabe 6, 7
t1	22 Minuten	1 Minute	2 Minuten	3 Minuten	5 Minuten
t2	18 Minuten	1 Minute	1 Minute	2 Minuten	3 Minuten
t3	22 Minuten	1 Minute	1 Minute	3 Minuten	3 Minuten
t4	26 Minuten	1 Minute	1 Minute	2 Minuten	3 Minuten
t5	25 Minuten	1 Minute	1 Minute	3 Minuten	2 Minuten
t6	33 Minuten	2 Minuten	1 Minute	5 Minuten	4 Minuten
t7	34 Minuten	1 Minute	3 Minuten	2 Minuten	5 Minuten
t8	35 Minuten	1 Minute	4 Minuten	2 Minuten	3 Minuten
t9	25 Minuten	1 Minute	2 Minuten	3 Minuten	6 Minuten

Abbildung 21 - Testdauer

Die durchschnittliche Dauer für die einzelnen Tests liegt bei 26-27 Minuten. Die Gesamtdauer enthält auch die Zeit, die die Testpersonen für Fragebögen und Nachinterview gebraucht haben. Die maximale Testdauer beträgt insgesamt 35 Minuten und die minimale 18 Minuten.

Die Zeit, die die Testpersonen für die Bearbeitung der Testaufgaben gebraucht haben, beträgt durchschnittlich 10 Minuten, wobei das meiste der Zeit bei den Aufgaben, die am Laptop zu lösen sind, vergangen ist. Die maximale Zeit, die eine Testperson für die Bearbeitung der Aufgaben gebraucht hat, beträgt 12 Minuten, und die minimale Dauer beträgt 7 Minuten.

Die Aufgaben 2 und zusammengefasst 3,4,5 sind ähnliche Aufgaben, und wurden nacheinander gestellt, damit man den Lernverlauf der Testpersonen testen konnte. Es wurde festgestellt, dass die dritte Aufgabe schneller gelöst wurde, nachdem die Testpersonen bereits die erste zweite Aufgabe gelöst hatten.

Von allen Testaufgaben wurde am meisten Zeit für die Aufgaben 6 und 7 gebraucht, obwohl die Aufgaben eigentlich in kürzere Zeiten zu lösen waren. Nur eine Testperson kannte die Webseite von vorher und hat auch die minimale Zeit mit 2 Minuten geliefert.

6 Auswertung der Usability-Tests

In diesem Kapitel werden die Ergebnisse der Usability-Tests erläutert. Dazu gehören die Auswertungskriterien, die Auffälligkeiten die während der Tests aufgetreten sind, und Ergebnisse der Fragebögen und Interviews. Zusätzlich zu den aufgezählten Auswertungsmaterialien standen auch die Videoaufnahmen zur Verfügung, die später bei der Auswertung gesichtet wurden, um weitere Auffälligkeiten zu notieren.

6.1 Auswertungskriterien

Für die Auswertung der durchgeführten Usability Tests wurden folgende Kriterien erdacht:

- Sind alle Aufgaben erfolgreich abgeschlossen worden?
- Kommt die Testperson mit Microsoft Surface selber klar?
- Musste ich während des Tests eingreifen?
- Nach welcher Zeit konnten die einzelnen Aufgaben gelöst werden?
- Welche Schwierigkeiten hatte die Testperson?

Das wichtigste Kriterium ist natürlich, ob alle Testaufgaben erfolgreich abgeschlossen wurden. Da bei den durchgeführten Tests alle Aufgaben von allen Testpersonen erfolgreich gelöst wurden, kam die gesamte Testdauer von einzelnen Tests ins Spiel, wobei ich mit Hilfe von Informationen die ich von den Testpersonen erhalten habe, einen Vergleich ziehen konnte.

Als nächstes muss ich sicherstellen, dass die Testperson ohne Probleme oder Hilfe mit dem Surface interagieren kann. Dazu werden die Testpersonen nach Erfahrungen mit Multi-Touch Geräten gefragt, damit es später für mich möglich wird, eine Aussage zu treffen.

Wenn eine Testperson mit einer Aufgabe nicht weiterkommt, kann es sein, dass der Testleiter in den Test eingreift. Theoretisch gesehen ist es etwas Negatives im Sinne eines unbeeinflussten Ablaufs, aber damit die Testperson bei einer Aufgabe nicht steckenbleibt und damit der Test weitergeht, ist es notwendig Hilfe anzubieten.

Mit Hilfe der Dauer der einzelnen Tests kann ich Informationen erhalten, worin das größte Problem liegen könnte. Wenn einige Testpersonen bei einer Aufgabe viel mehr Zeit brauchen als andere Testpersonen, kann ich mich auf diese Stelle der Anwendung konzentrieren und mögliche Usability-Probleme identifizieren. Außerdem ist es möglich Vergleiche zwischen erfahrenen und unerfahrenen Testpersonen zu machen, um den Grad der Erlernbarkeit zu bestimmen.

Die mir während der Tests aufgefallenen Schwierigkeiten, die die Testpersonen erlebt haben, wurden notiert, und nach den Tests hinterfragt. Die daraus resultierenden Informationen können dann später gesammelt und ausgewertet werden.

6.2 Auffälligkeiten

In diesem Teilkapitel werden die Auffälligkeiten, die während der Tests notiert und nach den Tests hinterfragt wurden, erläutert. Die Auffälligkeiten werden nach Häufigkeit ihres Vorkommens gewichtet. Zur Hilfe ist eine Excel-Tabelle erstellt worden, in welcher diese Bemerkungen nach Testpersonen aufgelistet sind.

	Reset-Button	Stadt-Button-Feedback	Pop-Up-Fenster	Sitzplan-Legende
t1	X	x	x	x
t2			x	x
t3	X			
t4	X		x	
t5	X	x	x	x
t6	X		x	x
t7	x		x	x
t8	x	x		x
t9		x		x

Abbildung 22 - Auffälligkeiten

1- Reset-Button-Problem

Wie in Kapitel 3.4 beschrieben, muss man immer „Reset"-Button tippen, um mit einer neuen Flugsuche anzufangen. Dieser Button löscht die vorherige Suche und führt das System wieder in den Startzustand. Das Problem liegt daran, dass wenn man schon einen Flug gesucht hat, und dann einen neuen Flug sucht, wird wieder die Strecke der vorherigen Suche auf der Karte angezeigt, und die Eingaben werden nicht berücksichtigt.

Sieben der neun Testpersonen hatten Schwierigkeiten hiermit. Meistens sind sie nicht darauf gekommen, extra einen Button (Reset-Button) zu tippen um einen Flug zu suchen. Zusätzlich fanden die Testpersonen die Positionierung des Buttons problematisch, da wenn die Karte größer skaliert wurde, der Button außerhalb des Bildschirms stand, und daher nicht gesehen wurde.

Die Mehrheit der Testpersonen war der Meinung, dass dieser Button überflüssig ist, und stattdessen alles automatisch geschehen sollte, wenn man wieder zur Startübersicht zurückkehrt.

2- Sitzplan-Legende

Um den bestmöglichen Sitzplatz für eine bestimmte Strecke zu finden, wurde die Funktion „Sitzplan-Anzeige" hinzugefügt. Wie in Abbildung 13 zu sehen ist, stehen drei Farben (Rot, Gelb, Grün) die bei der Auswahl des Sitzplatzes Hilfestellung bieten. Da keine Legende für diese Farben zur Verfügung steht, wurde die Bedeutung der Farben nicht von Testpersonen verstanden.

Sieben der Testpersonen waren der Meinung, dass unbedingt eine Legende zur Verfügung stehen muss, damit es klar und deutlich wird, dass es hier um einen Sitzplatzvorschlag geht. Viele der Testpersonen dachten, dass die Farben für Buchungszwecke benutzt werden.

3- Pop-Up-Fenster

Die Anzeige der Ergebnisse der Flugsuche geschieht in einem Pop-Up-Fenster, welches nach der Auswahl des Ankunftsorts geöffnet wird. Hierzu wurden mehrere verschiedene Auffälligkeiten notiert. Erstens fanden die Testpersonen es schlecht, dass das Fenster kleine Ausmaße hat. Zudem wurde es als unübersichtlich empfunden sich Flüge anzugucken, wenn mehrere Flüge für diese Strecke verfügbar sind. Dass das Fenster so klein ist, hat zu weitere Probleme geführt. Es ist mehrmals vorgekommen, dass die Testpersonen das Fenster nicht verschieben, drehen oder skalieren konnten. Dies liegt daran, dass die Elemente (wie angezeigte Flüge, Textfelder und Buttons) die in diesem Fenster auftauchen, schon viel Platz brauchen. Deswegen bleibt nur noch wenig Fenster-Fläche um das Fenster berühren zu können. Desweiteren hat die Platzierung des Fensters den Testpersonen missfallen, da es nach jeder Flugsuche irgendwo anders aufgetaucht ist, und nicht immer an derselben Position.

Es gab insgesamt sechs Testpersonen, die Schwierigkeiten hiermit hatten. Als Endergebnis war die gesamt Meinung, dass es übersichtlicher wäre, wenn sich das Fenster automatisch von der Größe her anpassen, immer in der gleichen Position befindet, und nach einer Flugauswahl automatisch ausblenden würde.

4- Stadt-Button-Feedback

Um die Ergebnisse einer Flugsuche anzusehen, müssen in der Reihenfolge zuerst der Abflugsort, dann Ankunftsort ausgewählt werden. Hier lag das Problem daran, dass die Testpersonen nicht mitbekommen haben, ob die erste Eingabe geklappt hat, oder nicht. Nach dem Tippen leuchtet der Button kurz auf, und blendet dann wieder aus. Dies führte dazu, dass die Testpersonen die Eingabe nochmal versuchten. Die zweite Eingabe wurde von dem System als „Ankunftsort" erkannt, und es wurde eine Flugsuche mit dem gleichen Abflugs-und-Ankunftsort durchgeführt.

Vier der Testpersonen waren der Meinung, dass es Hilfreich und Benutzerfreundlich wäre, wenn der Button nach Tippen beleuchtet oder ausgegraut bleibt, und nicht sofort ausgeblendet wird.

Desweiteren ergaben sich auch andere MS Surface-spezifische Probleme, die von mehreren Testpersonen erwähnt wurden:

Bei manchen Testpersonen ist aufgefallen, dass das Tippen und Berühren nicht einwandfrei funktioniert hat. Das Problem daran ist, dass die Eingaben durch Infrarotkameras aufgenommen werden, und dass es keine Drucksensoren gibt. Dadurch kann es selbst beim starken Tippen passieren, dass MS Surface die Eingabe nicht erkennt, weil die Berührungsoberfläche nicht groß genug ist. Ein Fingernagel wäre ein mögliches Worst-Case-Szenario, da die Berührungsoberfläche zu klein ist, um von den Infrarotkameras erkannt zu werden. Außerdem haben die Testpersonen, die damit Probleme hatten, meistens versucht, den Bildschirm leicht zu berühren oder anzutippen wie bei einem Multi-Touch Mobiltelefon, welches auch zu demselben Problem geführt hat.

Weiterhin waren einige Testpersonen der Meinung, dass die Bildschirmauflösung nicht ausreichend hoch ist. Für die Beispielanwendung reicht die Auflösung aus, da zum Beispiel bei der Software nur einige Großstädte aus Europa zur Auswahl stehen, und keine anderen

Kontinente oder Flughäfen. Wenn die Software um mehrere Städte und Flughäfen erweitert wäre, hätte die Auflösung nicht ausgereicht, um die Karte übersichtlich darzustellen.

6.3 Auswertung der Fragebögen und Interviews

In diesem Teilkapitel werden die Fragebögen (s. Anhang C.2) ausgewertet, die nach den Usability-Tests durch die Testpersonen ausgefüllt wurden. Die beiden Fragebögen enthalten insgesamt 22 Fragen, die entweder mit „Ja" oder „Nein" beantwortet wurden. Die Fragen sind so formuliert, dass sie immer in einer optimistischen Art gefragt werden. Dafür wurden in den Fragen positive Wörter wie; „einfach", „ausreichend", „leicht" eingesetzt. Somit wird es bei der Auswertung möglich sein einzuschätzen, ob die generelle Meinung positiv oder negativ ist.

Um die Auswertung zu erleichtern, sind die Antworten der Fragebögen in MS Excel-Tabellen mit „1" und „0" dargestellt worden. Die „1" steht hierbei für eine positive Antwort, die „0" für eine negative.

Was nicht bei den Fragebögen den Testpersonen gut gefallen hat, war das man nur mit „Ja" oder „Nein" antworten konnte. Es liegt in der menschlichen Natur eine differenzierte Meinung über einzelne Fragen zu haben. Bei manchen Fragen ist es dazu gekommen, dass sich die Testpersonen nicht sicher waren, ob sie mit „Ja" oder „Nein" antworten, da sie gute Gründe für beide Antworten hatten und nicht zwischen diesen abzuwiegen vermochten.

6.3.1 Fragebogen für Software

Der für die Software erstellte Fragebogen enthält insgesamt 10 Fragen. Dazu wurde auch unten Platz für Anmerkungen gelassen, so dass die Testpersonen aufschreiben konnten, was ihnen gut oder schlecht gefallen hat. Jede einzelne Frage hat ein bestimmtes Usability-Ziel. Bei einer Zusammenfassung bestimmter Fragen aus dem Fragebogen können allgemeinere Aussagen zur Benutzbarkeit getroffen werden.

In folgender Tabelle sind die Antworten aller Teilnehmer zu jeder Frage dargestellt. Speziell bei der Frage-Nummer 8 würde die Antwort „Ja" eine negative und die Antwort „Nein" eine positive Bedeutung haben. Hier wird abgefragt ob eine Hilffunktion nötig ist. Und daher sind die Antworten für die Frage-Nummer 8 umgekehrt in die Tabelle eingetragen worden. Hierdurch wird gewährleistet, dass positive Antworten immer durch eine „1" in der Tabelle dargestellt werden, damit eine Gesamtauswertung leichter zu realisieren ist.

	Frage 1	Frage 2	Frage 3	Frage 4	Frage 5	Frage 6	Frage 7	Frage 8	Frage 9	Frage 10
t1	1	1	1	1	0	1	1	1	0	1
t2	1	1	1	0	0	1	1	1	1	1
t3	1	1	1	0	1	1	1	1	1	1
t4	1	1	1	1	0	1	1	1	0	1
t5	1	1	1	1	1	1	1	0	1	1
t6	1	1	0	1	1	1	1	1	1	1
t7	1	1	1	1	1	1	1	0	1	1
t8	1	1	1	1	1	1	1	0	1	1
t9	1	1	1	1	1	1	1	1	1	1

Abbildung 23 - Antworten (Software)

	Ja	Nein	Prozent "JA"
t1	8	2	80
t2	8	2	80
t3	9	1	90
t4	8	2	80
t5	9	1	90
t6	9	1	90
t7	9	1	90
t8	9	1	90
t9	10	0	100

Abbildung 24 - Prozent "Ja"

Die Tabelle links, zeigt die Anzahl der Fragen, die mit „Ja" oder „Nein" beantwortet sind, sowie die Prozentzahl der „Ja"-Antworten. Daraus kann man die Aussage treffen, dass die generelle Meinung für das Flug-Informationssystem mit 87,8% „Ja"-Antworten und 12,2% „Nein"-Antworten positiv ist. Drei von den neun Testpersonen haben in den Fragebogen zwei, fünf der Testpersonen haben nur eine, und eine Testperson hat überhaupt keinen negativen Antworten gegeben.

Die folgende Tabelle enthält die durchschnittlichen Antworten aller Testpersonen zu den einzelnen Fragen. Zuerst werden die Fragen einzeln betrachtet und ausgewertet, bevor eine gesamte Auswertung realisiert wird.

	Frage 1	Frage 2	Frage 3	Frage 4	Frage 5	Frage 6	Frage 7	Frage 8	Frage 9	Frage 10
Ja	9	9	8	7	6	9	9	6	7	9
Nein	0	0	1	2	3	0	0	3	2	0
Ja_Prozent	100,00	100,00	88,89	77,78	66,67	100,00	100,00	66,67	77,78	100,00
Nein_Prozent	0,00	0,00	11,11	22,22	33,33	0,00	0,00	33,33	22,22	0,00

Abbildung 25 - Fragen-Übersicht

Frage 1:

Wie eine durchweg positive Beantwortung der Frage mit 100% zeigt, empfanden alle Testpersonen die Bedienung der Software als einfach. Der Grund dafür ist, dass die Anwendung von Anfang an übersichtlich geplant und umgesetzt wurde. Es gibt keine ablenkenden Elemente oder komplizierte und verschachtelte Menüs, wie bei der Flughafen-Webseite, die während der Tests auch behandelt wurde.

Frage 2:

Auch hier haben die Testpersonen zu 100% positiv geantwortet. Die leichte Erlernbarkeit der Software könnte daraus resultieren, dass alle Testpersonen bereits Erfahrungen mit Multi-Touch-Geräten haben (siehe Frage 1, Surface-Fragebogen). Es müssen keine komplizierten Zusammenhänge erlernt werden. Der Spaßfaktor könnte ebenfalls dazu beigetragen haben, die Erlernbarkeit zu verbessern (siehe Frage 12, Surface-Fragebogen).

Frage 3:

Nur eine von neun Testpersonen hat die Benutzerfreundlichkeit des Interfaces der Software negativ bewertet. Darüber ob dies in Zusammenhang mit dem Alter der Testperson steht kann keine Aussage getroffen werden, da bei dem Bezug auf nur eine Person auch der Zufall ein Grund sein kann. Die positiven Antworten der anderen Testpersonen können daran liegen, dass bei dem Entwurf des Interfaces auf eine intuitive Gestaltung geachtet wurde.

Frage 4:

Der Wechsel zwischen den einzelnen Menüs oder Masken fiel einigen Testpersonen schwer. Die Screen-Tabs welche eigentlich dafür dienen sollten, wurden weitgehend

übersehen. Die Testpersonen, die diese Tabs bemerkt haben, waren auch der Meinung dass die Tabs nicht leicht zu bemerken sind, und auch dieses Problem mit einer farbigeren Gestaltung zu lösen wäre.

Frage 5:

Kleine Fehler können nach Meinung jeder Dritten Testpersonen schwerwiegende Folgen haben. Einige der unter Kap. 6.2 aufgeführten Auffälligkeiten können Gründe dafür sein.

Frage 6:

Alle Testpersonen waren der Meinung, dass der Einstieg in die Anwendung leicht ist, und dass sie sich bei einer zweiten Begegnung mit dieser Software sofort wieder zurechtfinden würden. Grund hierfür ist auch wieder die einfache Gestaltung des Programms sowie die intuitive Menüführung.

Frage 7:

Bei der Auswertung dieser Frage fällt wieder auf, dass alle Probanden gleich geantwortet haben. Nimmt man die Frage als Anhaltspunkt dafür, dass die Probanden „Experten" sind, so kann als Schlussfolgerung nur gezogen werden, dass die Anwendung sich für „Experten" gut eignet. Die Eignung für Anfänger könnte bei der vorherigen Annahme nicht eingestuft werden.

Frage 8:

Mit einem Drittel negativer Antworten gehört diese Frage mit Frage-Nr 5 zu den am häufigsten negativ beantworteten. Drei Testpersonen waren der Meinung, dass eine Hilfefunktion nötig wäre. Eine Hilfefunktion wurde ausgelassen, um testen zu können, ob die Bedienung intuitiv und erlernbar ist. Würden die Benutzer eine Hilffunktion nutzen, so würden sie die Software nicht intuitiv erlernen.

Frage 9:

Nur zwei von neun Testpersonen waren der Meinung, dass auf den Bildschirm nicht alle Informationen zu finden waren, die sie brauchten.

Frage 10:

Mit 100% positiven Antworten, fanden die Probanden das Erlernen von Funktionen der Software durch Ausprobieren einfach. Der Grund ist wieder, dass es bei der Anwendung keine komplexen Funktionen gibt. Man kann daraus die Aussage treffen, dass die Software für alltägliche Benutzung geeignet ist.

Die Tabelle unten fasst bestimmte Fragen zusammen, damit zielgerichtet allgemeinere Bewertungen von Usability-Eigenschaften getroffen werden können.

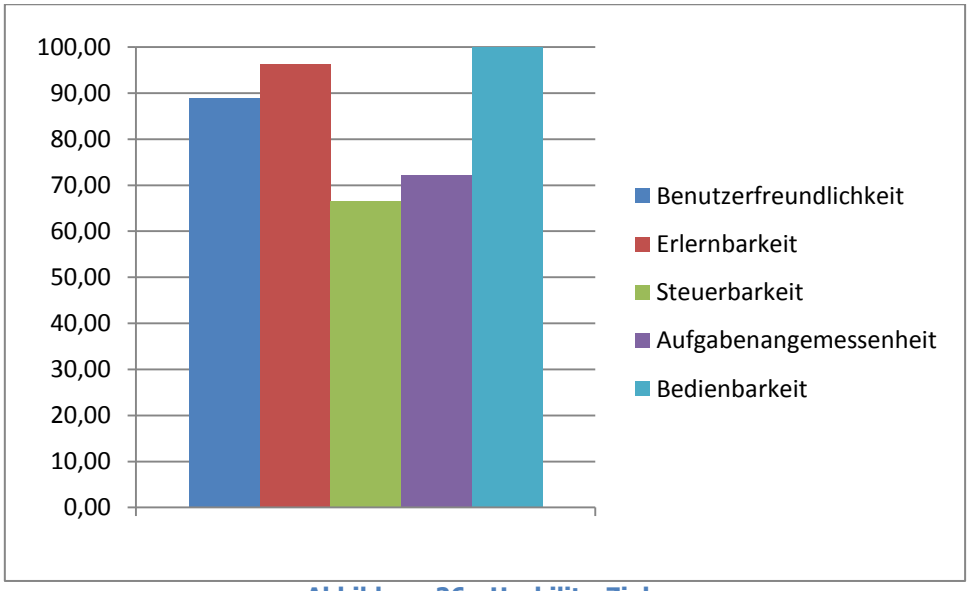

Abbildung 26 - Usability-Ziele

Bei dem Software-Fragebogen wurden die folgenden Usability-Eigenschaften abgefragt:

- Benutzerfreundlichkeit
- Erlernbarkeit
- Steuerbarkeit
- Aufgabenangemessenheit
- Bedienbarkeit

Die Fragen 1 und 4 stehen für Benutzerfreundlichkeit der Software. Die Ergebnisse zeigen, dass die Testpersonen die Nutzungsqualität der Software mit 89% positiven Antworten als „sehr gut" bezeichnen.

Die Erlernbarkeit der Software wurde durch die Fragen 2, 3 und 10 errechnet, und lieferte ein Ergebnis von 96%.

Die wenigsten positiven Antworten hat mit 67% die Steuerbarkeit, welche mit der Frage-Nr. 5 abgefragt wurde. Ich hatte eine höhere Bewertung erwartet, da die Software besonders für leichte Steuerbarkeit entwickelt wurde (siehe Frage 5).

Das Ergebnis für die Aufgabenangemessenheit fiel mit 72% nicht viel besser als das der Steuerbarkeit aus. Ein Grund hierfür könnte sein, dass die beide zusammenhängen. Die Schwierigkeit kann eher als unangemessen empfunden werden, wenn bereits die Steuerbarkeit gering ist.

Am besten wurde mit 100% die Bedienbarkeit bewertet.

6.3.2 Fragebogen für MS Surface

Bei dem Fragebogen für MS Surface sind insgesamt 12 Fragen beantwortet worden. Wie bei dem Software-Fragebogen, sind die Fragen so gestellt worden, dass es durch die positiven Antworten möglich sein wird zu beurteilen, ob die Usability-Eigenschaften erfüllt sind.

Eine Gesamtübersicht über die Antworten ist unten zu finden. Hierbei werden die Fragen-Nr. 3 und 9 mit Unterpunkten dargestellt, da diese Fragen aus mehreren Teilfragen bestehen.

	Frage 1	Frage 2	Frage 3.1	Frage 3.2	Frage 3.3	Frage 4	Frage 5	Frage 6	Frage 7	Frage 8	Frage 9.1	Frage 9.2	Frage 9.3	Frage 9.4	Frage 10	Frage 11	Frage 12
t1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
t2	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1
t3	1	1	1	1	1	0	1	1	1	1	0	1	1	0	1	1	1
t4	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1
t5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
t6	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1
t7	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
t8	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
t9	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1	0	1

Abbildung 27 - Antworten (Surface)

	Ja	Nein	Prozent "JA"
t1	17	0	100,00
t2	15	2	88,24
t3	14	3	82,35
t4	15	2	88,24
t5	17	0	100,00
t6	15	2	88,24
t7	16	1	94,12
t8	16	1	94,12
t9	14	3	82,35

Abbildung 28 - Prozent-"Ja"

Mit ungefähr 90% positiven Antworten kann man davon ausgehen, dass die generelle Meinung der Testpersonen für den MS-Surface-Tisch durchaus positiv ist. Bemerkenswert ist es hier, dass beide der Probanden mit MS-Surface Erfahrung, alle Fragen positiv beantwortet haben. Außerdem haben zwei von den Testpersonen in den Fragebogen drei, drei der Testpersonen zwei und zwei der Testpersonen nur eine Frage negativ beantwortet.

Bei der unteren Tabelle werden die Antworten aus Fragensicht dargestellt. Bei der Auswertung der einzelnen Fragen werden ihre Inhalte auch erwähnt, damit klar ist, was bei der Frage gefragt wird.

	Frage 1	Frage 2	Frage 3.1	Frage 3.2	Frage 3.3	Frage 4	Frage 5	Frage 6	Frage 7	Frage 8	Frage 9.1	Frage 9.2	Frage 9.3	Frage 9.4	Frage 10	Frage 11	Frage 12
Ja	9	9	9	8	8	8	8	9	8	9	7	8	7	7	9	7	9
Nein	0	0	0	1	1	1	1	0	1	0	2	1	2	2	0	2	0
Ja_%	100,00	100,00	100,00	88,89	88,89	88,89	88,89	100,00	88,89	100,00	77,78	88,89	77,78	77,78	100,00	77,78	100,00
Nein_%	0,00	0,00	0,00	11,11	11,11	11,11	11,11	0,00	11,11	0,00	22,22	11,11	22,22	22,22	0,00	22,22	0,00

Abbildung 29 - Fragen-Übersicht (Surface)

Frage 1:

Diese Frage dient zur Informationssammlung über Testpersonen. Hier wird abgefragt, ob die Testpersonen Erfahrungen mit Multi-Touch-Geräten haben. Alle der beteiligten Probanden haben diese Frage positiv beantwortet.

Frage 2:

Mit 100% positiver Antworten kann man bei dieser Frage die Aussage treffen, dass es den Testpersonen einfach fiel MS-Surface zu bedienen.

Frage 3:

Allgemein gesehen, sind die Gesten (3.1 Verschieben, 3.2 Drehen, 3.3 Skalieren) als einfach erlernbar und leicht durchzuführen bewertet worden. Außer Verschieben, hat je eine Testperson für die Gesten Drehen und Skalieren eine negative Antwort gegeben. Als Grund dafür könnten die Schwierigkeiten genannt werden, die die Testpersonen mit Pop-Up-Fenster erlebt haben (s. Kap 6.2 – Pop-Up-Fenster).

Frage 4:

Acht von den neun Testpersonen fanden den Bildschirm (Touch-Screen) ausreichend groß und ergonomisch, um damit arbeiten zu können. Es wurde auch von manchen Testpersonen erwähnt, dass es ein Nachteil ist, dass der Tisch tief und nicht leicht zu transportieren ist, und man auf einem Stuhl setzen muss, um damit arbeiten zu können.

Frage 5:

Bei dieser Frage wurden die Testpersonen gefragt, ob der Abstand zum Bildschirm (Touch-Screen) ausreichend groß ist. Acht von den beteiligten Probanden waren der Meinung, dass die Informationen auf den Bildschirm einfach zu lesen und erkennen sind. Hier spielt die Auflösung (1024x768) der MS Surface eine große Rolle, da Elemente bei einer niedrigen Auflösung größer dargestellt werden, als bei einer höheren.

Frage 6 und 10:

Alle Testpersonen waren der Meinung, dass ein Wiedereinstieg einfach fallen würde, wenn man sich schon mit dem Tisch bekannt gemacht hat.

Frage 7:

Nur eine der neun Testpersonen fand es nötig, das System mit Hilfe eines Handbuchs zu benutzen oder zu erlernen.

Frage 8:

Die Bearbeitungszeiten von dem System sind mit 100% positiven Antworten als „sehr gut" bewertet worden. Der Grund hierfür kann daran liegen, dass die getestete Beispielanwendung nicht sehr aufwendig für die Hardware ist, und dabei keine komplexen Berechnungen durchgeführt werden.

Frage 9:

Bei dieser Frage sind einige mögliche Arbeitsumgebungen (9.1 Flughäfen, 9.2 Restaurants, 9.3 Zuhause, 9.4 Kino) für MS-Surface abgefragt worden. Sieben von den Testpersonen würden den Tisch gerne in Flughäfen (Flüge buchen, Informationen angucken), Zuhause (Als Tisch, Multimedia Gerät) und Kinos (Platz reservieren), und acht Testpersonen in Restaurants (Menü angucken, Essen bestellen) im Einsatz sehen. Als Grund für die negativen Antworten wurde meistens die Größe des Surface-Tisches genannt.

Frage 11:

Bei dem Surface-Fragebogen ist diese Frage mit 22% am negativsten beantwortet worden. Wie vorher in Kap. 6.2 erwähnt wurde, fällt es dem Surface-Tisch schwer, „leichte" Eingaben von Benutzern zu erkennen.

Frage 12:

Mit 100% positiver Antworten kann man bei dieser Frage die Aussage treffen, dass es Spaß macht, MS Surface zu benutzen.

Informationen über die für MS Surface betrachteten Usability-Eigenschaften können wieder durch Zusammenfassung von Fragen gesammelt werden. Die untere Tabelle zeigt diese Eigenschaften.

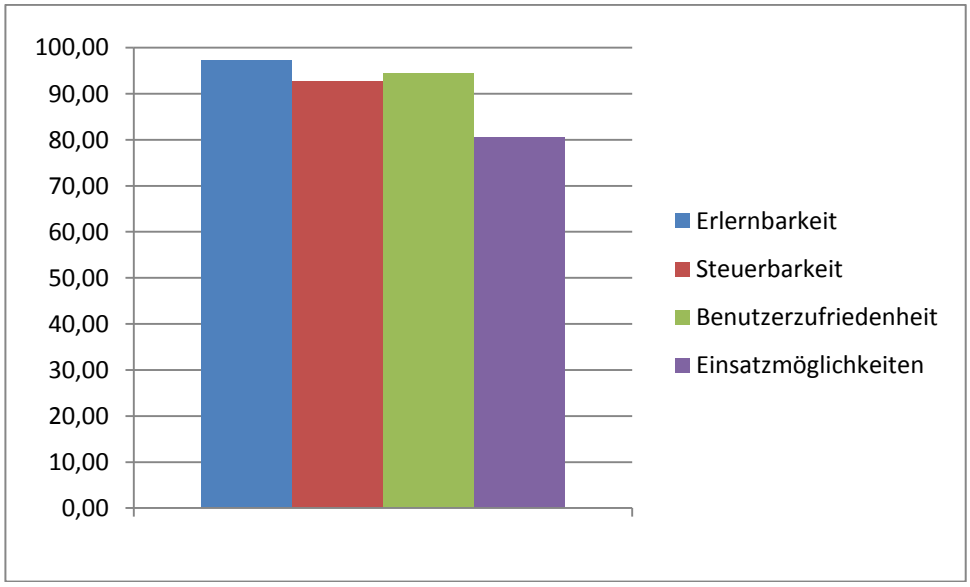

Abbildung 30 - Usability-Ziele (Surface)

Die Fragen-Nr. 1, 6, 7 und 10 beziehen sich auf die Erlernbarkeit des Systems. Die Informationen die bei der Frage-Nr. 1 gesammelt wurden, wäre noch hilfreicher für die Auswertung gewesen, wenn es auch unterschiedliche Antworten gegeben hätte. Da alle Testpersonen Erfahrungen mit Multi-Touch-Geräten haben, kann man nicht beurteilen, ob ein erfahrener Benutzer das System schneller erlernt als ein unerfahrener. Erlernbarkeit ist mit 97% die am meisten positiv bewertete Eigenschaft für MS Surface.

Bei den Fragen 2, 3, und 11 geht es um die Steuerbarkeit des Systems. Die generelle Meinung für diese Eigenschaft ist mit 93% durchaus positiv. Es ist von Bedeutung, dass sich die Testpersonen schon mal mit intuitiven Interaktionsmethoden bei anderen Multi-Touch-Geräten bekannt gemacht haben.

Die Ergebnisse aus Fragen-Nr. 4, 5, 8 und 12 zeigen, dass die Testpersonen das System zu 95% als „Benutzerfreundlich" betrachten.

7 Konsequenzen

In diesem Kapitel werden die Ergebnisse, die aus den durchgeführten Usability-Tests resultieren, erläutert. Es wird beschrieben, was für Konsequenzen die Software und die einzelnen Tests haben, sowie die aus den Tests resultierende Konsequenzen für Microsoft Surface und kausalen Anwendungen. Dabei werden die Verbesserungsmöglichkeiten für die Beispielanwendungen, die die Testpersonen vorgeschlagen haben, und die Erfüllung von den nicht funktionalen Anforderungen aufgezählt.

7.1 Konsequenzen für Software

Die Software wurde hauptsächlich dafür geplant, in kurze Zeit und mit wenigen Aufwand Informationen über Flüge zu sammeln. Die Ergebnisse der durchgeführten Tests zeigen, dass das geplante Ziel erreicht ist, und die nicht-funktionale Anforderungen (s. Kap 3.2.2) generell erfüllt sind (s. Kap 7.1.2).

Während der Tests wurde auch die Flughafen Webseite von Hamburg Airport untersucht, um einen Zeitvergleich für ähnliche Aufgaben bei MS Surface ziehen zu können. Bei einer Flugsuche mit der Beispielanwendung haben die Testpersonen durchschnittlich 2 Minuten gebraucht. Die durchschnittliche Dauer hierbei ist höher als erwartet, und der Grund dafür ist, dass zwei Testpersonen wegen Schwierigkeiten mehr als 2 Minuten gebraucht haben. Wenn man diese beiden Testpersonen nicht betrachtet, kommt man zu einer durchschnittlichen Dauer von ungefähr 1 Minute, welches als „schnell" betrachtet werden könnte. Andererseits beträgt die durchschnittliche Dauer von einer Flugsuche mit der Flughafen Webseite (www.airport.de) ungefähr 4 Minuten. Vom Umfang der Informationen her waren die Testpersonen der Meinung, dass die beide Untersuchungsgegenstände fast gleich gut sind. Die Schwierigkeit bei der Webseite liegt daran, dass die Seite nicht nur für Informationssammlung und Flugsuche geeignet ist, sondern für alles was man in einem Flughafen gebrauchen würde (Shops, Buchung, usw.). Außerdem fanden die Testpersonen die Webseite auch unübersichtlicher, wenn man sie mit der Beispielanwendung vergleicht.

Als Endergebnis waren die Testpersonen der Meinung, dass die Anwendung generell betrachtet einfach, schnell, intuitiv und übersichtlich ist. Wenn man das Anfangsziel der Anwendung, und was am Ende rausgekommen ist vergleicht, waren die Testpersonen vom Umfang der Funktionen her zufrieden, aber es war auch bemerkenswert, dass sie auch noch viele Verbesserungsmöglichkeiten vorgeschlagen haben, die für die Weiterentwicklung der Anwendung hilfreich sein könnten. Im folgenden Abschnitt werden diese Vorschläge einzeln beschrieben.

7.1.1 Verbesserungsvorschläge

Nach den Tests wurden Testpersonen gefragt, welche neuen Funktionen sie gerne noch bei der Software haben würden.

Buchungsfunktion: Viele der Testpersonen waren am Anfang der Meinung, dass die Software mehr für Buchung geeignet wäre, als Informationen über Flüge zu bekommen. Auch deswegen haben viele Probanden vorgeschlagen, eine Buchungsfunktion einzufügen.

Mehrere Informationen: Von der Umfang der Informationen her, wurde die Anwendung von meisten Testpersonen generell als „ausreichend" bewertet. Neben den zur Verfügung gestellten Informationen würden die Probanden gerne auch Informationen wie; Ländernamen in Karten-Übersicht, Flug-On-Time-Info, Wettervorhersage für den Abflugsort, Preise für einzelne Flüge sehen.

Tagesübersicht: Bei der aktuellen Version der Anwendung gibt es nur Informationen über Flüge die in einen bestimmten Tag geplant sind. Es wurde vorgeschlagen auch eine Tagesübersicht zu haben, damit man auch Informationen über Flüge finden kann, die an einem anderen Tag geplant sind.

Interaktiver Sitzplan, Check-in und Login: Während der Tests haben viele Testpersonen versucht mit dem zur Verfügung gestellten Sitzplan zu interagieren, indem sie die Sitzplätze getippt haben. Daraus resultierend waren sie der Meinung, dass es schön und intuitiv sein würde, wenn man sich vielleicht zuerst einloggen, den gewünschten Sitzplatz selber auswählen und den Check-in-Prozess schneller überspringen würde.

Hilfefunktion mit Video: Da manche Testpersonen beim ersten Schritt Schwierigkeiten hatten, waren sie der Meinung, dass eine Hilfefunktion, in der ein Film mit kurzer Bedienungsanleitung angezeigt wird, den ersten Einstieg erleichtern würde.

Länderauswahl: Falls die Anwendung später mit mehreren Flughäfen und Städten erweitert wird, wäre es dazu gekommen, dass die Kartenübersicht nicht mehr übersichtlich sein könnte. In so einer Situation könnte man die Länder als klickbare Felder betrachten, und die einzelne Länder interaktiver darstellen. Dabei wird nach der Auswahl eines Landes die Kartenübersicht sich ändern, und nur die ausgewählte Land angezeigt.

7.1.2 Nicht funktionale Anforderungen

Diese Teilkapitel beschreibt die Erfüllung von den in Kapitel 3 aufgezählten nicht funktionalen Anforderungen.

3.2.2.1 Benutzbarkeit (Bedienbarkeit)

Die Beispielanwendung ist für den täglichen Gebrauch geplant worden, und muss daher einfach zu bedienen sein. Die natürliche Interaktion mit den Händen und Gesten sorgt für die leichte Bedienung.

3.2.2.2 Benutzbarkeit (Erlernbarkeit)

Da diese Anwendung nicht für eine bestimmte Zielgruppe, sondern für „casual"-Nutzer geplant ist, sollte es von den Benutzern nicht viel verlangen, damit sie sich mit der Software einfach bekannt machen können. Um die Erlernbarkeit zu gewährleisten, wurde die Anwendung von Umfang der Funktionen her möglichst klein gehalten. Daher können die Benutzer die Lösungswege für die Aufgaben sich einfacher merken, und bei einer erneuten Interaktion mit dem System leichter einsteigen.

3.2.2.3 Effizienz

Zeit-und-Verbrauchsverhalten spielt bei der Beispielanwendung nicht große Rolle, da für eine Aufgabe benötigten Daten lokal verfügbar sind, und damit liefert die Anwendung für jede Eingabe immer das gleiche Ergebnis in gleicher Zeit.

3.2.2.4 Funktionalität (Angemessenheit)

Eine Software erfüllt diese Eigenschaft, wenn es ausreichende Funktionen zur Verfügung stehen, um eine Aufgabe mit dieser Software zu erfüllen. Die Beispielanwendung ist nur für

Flugsuche und Informationssammlung geeignet, und daher enthält keinen überflüssigen und komplexen Funktionen.

3.2.2.5 Wartbarkeit

Die Beispielanwendung wurde mit verständliche und aussagekräftige Methoden-und-Variablennamen entwickelt worden. Die Trennung der Model, ViewModel und Model erleichtert die Verständlichkeit und ermöglicht eine zukünftige Weiterentwicklung.

3.2.2.6 Portabilität

Diese Eigenschaft ist leider nicht realisierbar, da die Anwendung nur für MS Surface geeignet ist, und das nur mit einem 32-Bit Windows Vista System lauffähig ist.

7.2 Konsequenzen für Usability Tests

In diesem Abschnitt werden die Konsequenzen beschrieben, die bei der Usability-Untersuchung von Microsoft Surface entstanden sind.

Anders als bei den herkömmlichen Usability-Untersuchungen, ist es bei dieser Untersuchung der Unterschied, dass die Tests nicht in einem Usability-Labor durchgeführt sind. Dies kann man als Nachteil betrachten, da dadurch die zur Verfügung stehenden Testgeräte abgegrenzt werden. Als weiteres, die Untersuchungsgegenstand, MS Surface selber begrenzt die Anzahl von realisierbaren Untersuchungsmethoden, da die vorhandenen Testmethoden mehr für Desktop-Computer geeignet sind. Ansonsten wurden die ausgewählten Testmethoden (s. Kap. 4.4) ohne weitere Schwierigkeiten durchgeführt und realisiert.

Da bei dieser Arbeit nur ein Test pro Testperson durchgeführt wurde, und es weniger Anzahl von Testpersonen gab, war es nicht sehr sinnvoll Usability-Ergebnisse mit Prozentzahlen darzustellen und zu bewerten. Wenn dies der Fall ist, können die Prozentzahlen, die mit Anzahl von „X" Testpersonen entstehen, sehr anders sein, als mit Anzahl von „X+1" Testpersonen. Deswegen wurde es bei der Auswertung versucht, soweit wie möglich präzisere Aussagen für die Ergebnisse zu treffen, und möglichst wenige Prozentzahlen anzugeben.

7.3 Konsequenzen für Microsoft Surface

Die Konsequenzen von den Usability-Untersuchungen können auch auf Microsoft Surface bezogen werden. Die Endergebnisse der Tests haben gezeigt, dass es trotz kleinen Schwierigkeiten Spaß macht, den Tisch zu bedienen. Als weiteres sind das Erlernen und das Ausführen von Manipulationsgesten generell einfach gefallen. Die niedrige Auflösung, die Schwierigkeiten bei der Erkennung von leichten Berührungen, die Größe und schwere Transportierbarkeit des Tisches können als Mängel an „Casual Computing" gezeigt werden.

Diese Mängel werden durch den neu entwickelten Surface-LCD, SUR40 für Microsoft Surface, behoben. Der neue Surface-LCD ist 40-Zoll breit, 10 Zentimeter dünn und hat eine Auflösung von 1920x1080 Pixel. Er wird von Aluminiumfüßen getragen, die auch alle gebundenen Kabel enthalten. Durch den von Microsoft entwickelten „Pixel Sense" Technologie werden auch leichte und kleinste Berührungen erkannt. Die Sensorpunkte stehen auf jedem achtem Pixel der LCD[SUR40]. Mit diesen Eigenschaften wären die obengenannten Mängel behoben, und eine intuitivere Interaktion ermöglicht.

7.4 Konsequenzen für „Casual Anwendungen“

Die Beispielanwendung, die für diese Arbeit entwickelt wurde, ist eine exemplarische Anwendung für allgemeine „Casuale Anwendungen“ an dem Microsoft Surface Tisch. Als weitere Beispiele können Anwendungen für Restaurants und Kinos gezeigt werden.

Durch das Wachsen von Benutzung der Multi-Touch Geräten im Alltagsleben, werden kausale Anwendungen für tägliche Benutzung benötigt. Da sie einfach, kompakt und leicht erlernbar sind, können sie von den kausalen Benutzern ohne Schwierigkeiten benutzt werden.

8 Zusammenfassung und Ausblick

8.1 Zusammenfassung

Ziel dieser Arbeit war zu untersuchen, wieweit die vorhandenen Usability-Untersuchungsmethoden für Microsoft Surface realisierbar sind. Dafür musste eine einfache und intuitive Beispielanwendung entwickelt werden, die für die alltägliche Benutzung geeignet ist. Zuerst wurden Anforderungen und Ziele festgelegt.

Bei der Auswahl der Testmethoden wurden zuerst auf die wichtigste und weitgehend benutzte Testmethoden konzentriert, und dabei festgestellt, dass einige von diesen Methoden nicht realisierbar sind. Um die Benutzbarkeit der Software und Surface zu messen, wurden zuerst Testaufgaben entwickelt, und die ausgewählten Testmethoden durchgeführt.

Bei der Durchführung der Tests wurde es gemerkt, dass die ausgewählten klassischen Testmethoden auf Multi-Touch-Geräte einfach übertragt werden können. Durch das Zusammenspiel von Videoaufzeichnungen, Beobachtungen, Fragebögen & Interviews, und lautes Denken konnte viele Usability-Probleme identifiziert werden.

Mit Hilfe von aus den durchgeführten Tests resultierenden Ergebnissen wurden die Untersuchungsgegenstände ausgewertet, und die Auffälligkeiten und identifizierte Usability-Probleme dokumentiert.

Zum Schluss wurden die aus den Tests entstandenen Konsequenzen für die Usability-Untersuchungen, Beispielanwendung, Microsoft Surface und kausalen Anwendungen erläutert.

8.2 Ausblick

Mit den vorhandenen Methoden war es mit Einschränkungen möglich Benutzbarkeit von einer Anwendung am Microsoft Surface zu messen. Im nächsten Schritt sollten die realisierbare Untersuchungsmethoden für Multi-Touch-Geräte erweitert werden. Als Beispiel könnte die Eye-Tracking-Methode gezeigt werden. Durch Einsatz von dieser Methode könnten noch präzisere Auswertungen möglich sein. Zusätzlich sollte die Testumgebung so angepasst werden, dass es auch möglich wird, mehrere Usability-Metriken wie „Anzahl von Berührungen" zu sammeln, und auch noch die Anzeige auf den MS Surface-Tisch aufzunehmen, um eine effizientere Auswertungsvorgang möglich zu machen und die Ergebnisse zu verbessern.

A Literaturverzeichnis

[BGATES] – Bill Gate's dream: A computer in every home, http://www.telegraph.co.uk/technology/3357701/Bill-Gatess-dream-A-computer-in-every-home.html, Abruf: 01.2011

[Burmester2003] – Michael Burmester: Ist das wirklich gut? Bedeutung der Evaluation für die benutzerzentrierte Gestaltung, 2003

[ISOUsability] – ISO-Norm 9241, http://www.scoreberlin.de/usability-artikel/usability-iso-norm/

[MSSurf] – What is Microsoft Surface, http://www.microsoft.com/surface/en/us/Pages/Product/WhatIs.aspx, Abruf: 12.2010

[MSUEG] – Microsoft Surface User Experience Guidelines, 2009, http://www.microsoft.com/downloads/en/confirmation.aspx?FamilyID=38CC76F1-4A16-4C13-9740-C34DBB5C3012, Abruf: 12.2010

[MSDS] – Microsoft Surface Data Sheet, 2008, http://www.microsoft.com/downloads/en/confirmation.aspx?FamilyID=38CC76F1-4A16-4C13-9740-C34DBB5C3012, Abruf: 12.2010

[MSHist] – Microsoft Surface History – The making of Microsoft's first commercially available surface computer, 2007, http://www.microsoft.com/presspass/presskits/surfacecomputing/docs/SurfaceHistoryBG.doc, Abruf: 12.2010

[MSWPF] – Microsoft Presentation Foundation, http://msdn.microsoft.com/de-de/netframework/aa663326.aspx, Abruf: 12.2010

[MSXNA] – XNA Library, http://msdn.microsoft.com/de-de/library/bb200104.aspx, Abruf: 12.2010

[MVVM] – Introduction to Model/View/ViewModel pattern for building WPF apps, http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx, Abruf: 03.2011

[Nielsen93] – Jakob Nielsen: Usability Engineering, MorganKaufmann, San Francisco, 1993

[SUR40] - http://www.microsoft.com/surface/en/us/WhatsNew.aspx, Abruf: 07.2011

[SurfNUI] – Microsoft Surface und das Natural User Interface (NUI), 2009, http://www.pixelpark.com/de/pixelpark/_ressourcen/attachments/publikationen/090204_White_Paper_MS_Surface.pdf, Abruf: 12.2010

[UsEnHIST] – Frank Spillers: Usability from WWII to the present, 2007, http://www.usabilitytestingcentral.com/2007/02/the_history_of_.html, Abruf: 12.2010

[UsEnREG] – Usability Engineering Regeln, http://www.usability-im-webdesign.de/web_usability_regeln.html, Abruf: 12.2010

B Abbildungsverzeichnis

Abbildung 1 - http://www.microsoft.com/surface/en/us/default.aspx
Abbildung 2 - http://www.codeproject.com/KB/vista/MicrosoftSurfaceIntro.aspx
Abbildung 3 - http://i.msdn.microsoft.com/dynimg/IC353790.jpg
Abbildung 4 - http://en.wikipedia.org/wiki/File:CLI-GUI-NUI.png
Abbildung 5 - http://aco.uwaterloo.ca/newsletters/s08/pics/table1.JPG
Abbildung 6 - http://webenliven-space.de/dotnetblog/image.axd?picture=Abb1-Das-Model-View-ViewModel.jpg
Abbildung 8 - http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/Members/herzwurm/QMvS-abb2.png/image_large

C Anhang

C.1 Testaufgaben

Usability Test Aufgabenzettel

Gehen Sie die Aufgaben nacheinander durch. Bitte versuchen Sie, während der Tests laut mitzudenken.

1- Versuchen Sie die folgenden Gesten an der Kartenübersicht durchzuführen:
   - Verschieben
   - Drehen
   - Skalieren (Zoomen, kleiner oder größer)
2- Finden Sie Informationen über einen Flug von Hamburg nach Istanbul, der um 17:00 Uhr geplant ist.
3- Kehren Sie zurück zur Kartenübersicht, und finden Sie den Flug von Germanwings, der von Hamburg nach München fliegt.
4- Finden Sie den besten möglichen Sitzplatz für Hamburg - München (Flugzeugsitzplan)
5- Schließen Sie alle geöffneten Fenster.

---------- mit Laptop weiter ----------

6- Gehen Sie auf die Webseite www.airport.de
7- Finden Sie einen Flug von Hamburg nach München zwischen 12:00 – 13:00 Uhr und finden Sie Informationen über diesen Flug (falls möglich).

C.2 Fragebögen

Fragebogen für Fluginformationssystem-Software

Der Fragebogen dient zur Einschätzung der Benutzbarkeit von Flug-Informationssystem.
Bitte beantworten Sie die Fragen mit entweder „Ja" oder „Nein".

Frage	JA	NEIN
1- Es war einfach die Software zu bedienen.	☐	☐
2- Die Software ist einfach erlernbar, erfordert wenig Zeit zum lernen.	☐	☐
3- Das Interface der Software ist benutzerfreundlich.	☐	☐
4- Die Software ermöglicht einen leichten Wechsel zwischen einzelnen Menüs oder Masken.	☐	☐
5- Die Software ist so gestaltet, dass kleine Fehler keine schwerwiegenden Folgen haben können.	☐	☐
6- Die Software ist so gestaltet, dass sich einmal Gelerntes gut einprägt.	☐	☐
7- Die Software eignet sich für Anfänger und Experten gleichermaßen.	☐	☐
8- Wäre eine Hilffunktion nötig?	☐	☐
9- Auf dem Bildschirm finde ich alle Informationen, die ich gerade benötige.	☐	☐
10- Die Software ist so gestaltet, dass unbekannte Funktionen durch ausprobieren erlernt werden können.	☐	☐
Schreiben Sie bitte was Ihnen gut oder schlecht gefallen hat. Anmerkungen:		

Fragebogen für
Microsoft Surface

Der Fragebogen dient zur Einschätzung der Benutzbarkeit von Microsoft Surface. Bitte beantworten Sie die Fragen mit entweder „Ja" oder „Nein".

Frage	JA	NEIN
1- Haben Sie schon Erfahrung mit Multi-Touch Geräten?	☐	☐
2- Es war einfach das System zu bedienen.	☐	☐
3- Die Gesten sind einfach erlernbar. - Verschieben - Drehen - Skalieren (Kleiner/Größer zoomen)	☐ ☐ ☐	☐ ☐ ☐
4- Der Bildschirm (Touch Screen) ist ausreichend groß.	☐	☐
5- Der Abstand zum Bildschirm ist ausreichend, um die Informationen gut lesen zu können.	☐	☐
6- Das System ist so gestaltet, dass sich einmal Gelerntes gut einprägt.	☐	☐
7- Das System ist ohne fremde Hilfe oder Handbuch benutzbar.	☐	☐
8- Das System reagiert mit guten Bearbeitungszeiten	☐	☐
9- Würden Sie sich die folgende Arbeitsumgebung für MS Surface vorstellen können? - Flughäfen - Restaurants - Zuhause - Kino	☐ ☐ ☐ ☐	☐ ☐ ☐ ☐
10- Auch bei seltenem Gebrauch ist es kein Problem sich wieder in das System hineinzufinden.	☐	☐
11- Die Eingabe (Berühren, Tippen) funktioniert einwandfrei.	☐	☐
12- Es hat Spaß gemacht, MS Surface zu benutzen.	☐	☐

C.3 Anwendungsfälle

C.3.1 SelectOrigin

Name: Select Origin
Akteur: User
Ziel: Abflugsort auswählen

Auslöser:
Der User möchte eine Flugsuche durchführen.

Vorbedingungen:
Die Software ist in den Startzustand, und Laufbereit.

Nachbedingungen:
Der jeweilige Startflughafen wird ausgewählt und es wird auf die Eingabe der Endflughafen gewartet.

Erfolgsszenario:

1- Der User führt eine Flugsuche durch
2- Der User wählt die Startflughafen aus

Häufigkeit:
Vor jeder Flugsuche muss ein Startflughafen ausgewählt werden.

C.3.2 SelectDestination

Name: Select Destination
Akteur: User
Ziel: Ankunftsort auswählen

Auslöser:
Der User hat schon den Startflughafen ausgewählt, und möchte die verfügbare Flüge anzeigen lassen.

Vorbedingungen:
Abflugsort muss schon ausgewählt sein.

Nachbedingungen:
Der Ankunftsort ist ausgewählt. Die Software sucht die Flüge für die Strecke und gibt die Ergebnisse der Suche in ein Pop-Up-Fenster aus.

Erfolgsszenario:

1- Startflughafen ist ausgewählt
2- Endflughafen ist ausgewählt

Häufigkeit:
Bei jeder Flugsuche muss der Ankunftsort ausgewählt werden.

C.3.3 SelectFlight

Name: Select Flight
Akteur: User
Ziel: Einen Flug auswählen

Verwendete Anwendungsfälle (includes):
„Select Origin" und „Select Destination"

Auslöser:
Der User hat die Flugsuche beendet, und möchte sich jetzt Informationen über einen bestimmten Flug angucken.

Vorbedingungen:
Start-und-Endflughäfen müssen ausgewählt sein, und die Liste mit den gefundenen Flügen muss angezeigt werden.

Nachbedingungen:
Informationen über den ausgewählten Flug werden in einem neuen Fenster angezeigt.

C.3.4 ShowAircraftSeats

Name: Show Aircraft Seats
Akteur: User
Ziel: Flugzeugsitzplan anzeigen

Beschreibung:
Nach der Flugsuche hat der User die Möglichkeit den Sitzplan für den ausgewählten Flug sich anzugucken.

Verwendete Anwendungsfälle (includes):
„SelectFlight"

Auslöser:
Die Fluginformationen stehen zur Verfügung. Der User möchte sich den Sitzplan angucken, und klickt/tippt auf die jeweilige Flugzeugtyp.

Vorbedingungen:
Der Flug muss ausgewählt sein.

Nachbedingungen:
Der Flugzeugsitzplan-Bild wird geöffnet.

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, den _____ _____