TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Fakultät Informatik

Analyse und Überarbeitung des Graphical User Interfaces von EB GUIDE Studio 6 zur Steigerung der Usability

Bachelorarbeit im Studiengang Medieninformatik

vorgelegt von

Sandra Schumann

Matrikelnummer 302 0357

Erstgutachter:	Prof. Dr. Korbinian Riedhammer
Zweitgutachter:	Prof. Dr. Matthias Teßmann

Hinweis: Diese Erklärung ist in alle Exemplare der Abschlussarbeit fest einzubinden. (Keine Spiralbindung)

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Vorname: Matrikel-Nr.:

Fakultät: Studiengang:

Semester:

Titel der Abschlussarbeit:

Ich versichere, dass ich die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ort, Datum, Unterschrift Studierende/Studierender

Erklärung zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit ☐ genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,

☐ genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format auf einem Datenträger beigefügt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Ort, Datum, Unterschrift Studierende/Studierender

Formular drucken

Kurzdarstellung

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Abstract

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Inhaltsverzeichnis

1. Einleitung **1**
1.1. Elektrobit Automotive GmbH 1
1.2. Abteilung User Experience 1
1.3. Motivation 2
1.4. Zielsetzung 2
1.5. Aufbau der Arbeit 2

2. Theorie **3**
2.1. Grafische Benutzeroberfläche 3
2.2. Ergonomie 3
2.3. Usability 3
2.4. User Experience 6
2.5. Kognitive Last 7
2.6. Human - centered design process 7
2.7. Gestaltprinzipien der Usability 9
2.8. EB Guide Studio 14

3. Analysen **17**
3.1. Voruntersuchungen 17
3.1.1. Ausgangssituation 17
3.1.2. Vorgehensweise 18
3.1.3. Ergebnisse 19
3.2. Verbesserungen 24
3.2.1. Auswahlkriterien 24
3.2.2. Gewinn für den Nutzer 25
3.2.3. Design der Verbesserungen 25

4. Umsetzung **27**
4.1. Prototyp Multiselektion 27
4.1.1. Zielsetzung 27
4.1.2. Axure RP 27
4.1.3. Interaktionsmöglichkeiten 27
4.1.4. Vorgehensweise 27

4.2. Implementierung Filter . . . 27
4.2.1. Zielsetzung . . . 27
4.2.2. Projektaufbau . . . 27
4.2.3. Vorgehensweise . . . 27

5. Usability - Test . . . 29
5.1. Lookback . . . 29
5.2. Remote Usability - Test . . . 29
5.3. Arbeitsaufgaben . . . 29
5.4. Ergebnisse altes Interface . . . 29
5.5. Ergebnisse überarbeitetes Interface . . . 29
5.6. Vergleich . . . 29

6. Fazit . . . 31
6.1. Ergebnisse . . . 31
6.2. Ausblick . . . 31

A. Supplemental Information . . . 33

Abbildungsverzeichnis . . . 35

Tabellenverzeichnis . . . 37

List of Listings . . . 39

Literaturverzeichnis . . . 41

Kapitel 1.

Einleitung

In diesem einführenden Kapitel wird zunächst kurz das Partnerunternehmen der Abschlussarbeit mit der zugehörigen Abteilung User Experience vorgestellt. Weiterhin wird die Motivation dieser Arbeit und das verfolgte Ziel erläutert, bevor es noch einen Überblick über die folgenden Kapitel gibt.

1.1. Elektrobit Automotive GmbH

Partner der Bachelorarbeit ist die Firma Elektrobit Automotive GmbH - im Folgenden nur noch als EB bezeichnet. EB ist ein vielfach ausgezeichnetes, internationales Unternehmen, welches sich auf die Entwicklung von Produkten und Dienstleistungen im Bereich der Automobilindustrie spezialisiert hat. Mit über 30 Jahren Branchenerfahrung bietet EB seinen Kunden unter anderem innovative Lösungen für das vernetzte Fahrzeug, Human Machine Interface Technologien (HMI), Navigations- und Fahrassistenzsysteme und Steuergeräte. Die Automotive Software von EB befindet sich in über 1 Billionen Geräten die in mehr als 90 Millionen Fahrzeugen weltweit Verwendung finden. Mit über 2300 beschäftigten Mitarbeitern, verteilt auf 3 Kontinente und 9 Länder, und einer durchschnittlichen jährlichen Wachstumsrate von über 10 % ist EB ein weltweit etabliertes Unternehmen mit Hauptsitz in Erlangen[Eleka].

1.2. Abteilung User Experience

Jedes Gerät das für den alltäglichen Gebrauch gedacht ist, sollte eine erfolgreiche Interaktion gewährleisten. Dafür ist eine Schnittstelle zwischen Mensch und Maschine (HMI) die einen intuitiven, einfachen und schnellen Umgang mit diesem Gerät ermöglicht unabdingbar. Um die Erfahrungsqualität im Allgemeinen möglichst hoch zu halten wünschen Nutzer sich auf ihre Bedürfnisse angepasste User-Interfaces in allen Lebensbereichen, womit der Bereich UX auch in der Automobilbranche einen hohen Bedeutungsgrad genießt. Die Abteilung User Experience von EB befasst sich vor allem mit der Entwicklung multimodaler HMIs

für Kombiinstrumente, Head Units und Head-Up Displays. Diese werden von EB von der Konzeptphase bis hin zur Serienentwicklung mit Hilfe von EB-GUIDE entwickelt.

1.3. Motivation

TODO

1.4. Zielsetzung

Ziel dieser Bachelorarbeit ist es, Schwachstellen im User Interface von EB GUIDE zu identifizieren und durch deren Verbesserung die Usability von EB GUIDE zu erhöhen. Dafür ist zuerst eine Analyse der Arbeitsabläufe innerhalb der Modellierungsarbeit nötig um entsprechende Probleme im User Interface zu erkennen. Anschließend gilt es Konzepte zu entwickeln welche, durch Anpassung und Überarbeitung der entsprechenden Komponenten in der Benutzerschnittstelle, diese Probleme beheben oder minimieren. Diese Konzepte werden anschließend noch grafisch und interaktiv visualisiert, um abschließende Usability-Tests durchführen zu können. Dabei wird eine identische Aufgabenstellung von Probanden mit dem Alten und dem Überarbeiteten Interface durchgeführt und durch den Vergleich der Ergebnisse festgestellt, ob die Anpassungen die Usability von EB Guide erhöht haben.

1.5. Aufbau der Arbeit

TODO

Kapitel 2.

Theorie

2.1. Grafische Benutzeroberfläche

Der Begriff Benutzerschnittstelle bezeichnet alle Komponenten eines interaktiven Systems, die dem Benutzer Interaktionsmöglichkeiten mit selbigem System bieten, um ein verfolgtes Ziel zu erreichen. Die grafische Benutzeroberfläche (GUI) bezeichnet hierbei den sichtbaren Anteil des Systems und damit nur einen Teil der gesamten Benutzerschnittstelle, zu der auch nicht sichtbare Teile wie z.B. die Funktionslogik gehören[Saro 16]. Heutzutage sind die meisten Benutzeroberflächen auch grafische Benutzeroberflächen, mit denen in den häufigsten Fällen die Interaktion mit dem Nutzer über direkte Manipulation stattfindet[Niel 95].

2.2. Ergonomie

Unter Ergonomie versteht man im Allgemeinen die "Lehre von der menschlichen Arbeit und die Erkenntnis ihrer Gesetzmäßigkeiten"[Ergo 14]. Hierbei ist es wichtig zu verstehen, dass dabei der Fokus nicht ausschließlich auf einer technischen Komponente liegt, sondern das Zusammenspiel von Mensch, der zugeteilten Aufgabe und den verfügbaren Werkzeugen betrachtet wird[Saro 16]. In Bezug auf Software bedeutet Ergonomie also konkret diese gut handhabbar und benutzerorientiert zu gestalten.

2.3. Usability

Mit immer höherer Komplexität von Systemen und Anwendungen kam der Begriff und das Verlangen nach "Benutzerfreundlichkeit" auf. Dieser Begriff suggeriert das lediglich die einfache Benutzung eines Systems ausschlaggebend ist, vernachlässigt hierbei jedoch die Notwendigkeit den Nutzer beim Erreichen seiner Ziele passend zu unterstützen. Dies ist auch der Grund dafür das bald, statt auf "Benutzerfreundlichkeit" auf "Gebrauchstauglichkeit"(engl. Usability) geachtet wurde. Im Gegensatz zur Ergonomie handelt es sich bei Usability nicht

um eine eigenständige wissenschaftliche Disziplin, sondern um eine qualitative Anforderung an ein System[Saro 16]. Konkret spricht man bei einer Software-Anwendung von einer hohen Usability, wenn sie von der für sie bestimmten Zielgruppe effizient verwendet werden kann, also das verfolgte Ziel zufriedenstellend erreicht wird[Rich 16]. Hierfür ist es entscheidend sich bewusst zu machen, dass ein technisches System oder Software immer Teil eines großen Handlungsablaufes ist und dazu dient Schritte dieses Handlungsablaufes zu erledigen. Deshalb muss das System den Anforderungen dieses Ablaufes entsprechen und darf während der Entwicklung nicht getrennt davon betrachtet werden[Saro 16].

Usability wird üblicherweise dadurch gemessen, indem man Nutzern, die der Zielgruppe entsprechen einige vordefinierte Aufgaben mit dem zu messenden System abschließen lässt. In einigen Fällen kann es allerdings auch unnötig sein eine Aufgabe vorzudefinieren, da die Messung auch im alltäglichen Arbeitsablauf mit dem System stattfinden kann. In beiden Fällen ist es jedoch wichtig die Ergebnisse immer im Kontext des Nutzers und der absolvierten Aufgabe zu betrachten[Niel 95]. Wie bereits erwähnt misst die Usability wie zufriedenstellend ein Ziel erreicht werden kann. Deshalb ist es nicht möglich die Usability für ein komplettes System zu messen, da diese immer Kontextabhängig ist, woraus sich auch die Notwendigkeit ergibt bei einer Usability Studie die Testpersonen identische, oder zumindest ähnliche Aufgaben ausführen zu lassen. Ein Textprogramm kann beispielsweise gut geeignet sein, um einen Brief zu tippen, jedoch völlig ungeeignet um größere Datenmengen zu verwalten. Es würde also für die eine Aufgabenstellung eine hohe Usability und für die andere eine niedrige aufweisen. Würde man diese Ergebnisse kontextunabhängig vergleichen würde keine valide Aussage über die Usability des Systems getroffen werden.

Für gewöhnliche ist Usability an fünf Attributen gut messbar, welche im Folgenden deshalb kurz erläutert werden.

Erlernbarkeit Das Attribut der Erlernbarkeit ist in vielerlei Hinsicht das fundamentalste von allen hier aufgeführten, da die erste Interaktion, die ein Nutzer mit einem System hat meist das erlernen dessen Funktionen beinhaltet. Gemessen wird dieses Attribut ganz simpel mithilfe von Nutzern, indem man die Zeit misst, die diese benötigen um ein vorher festgelegtes Level an Kompetenz zu erreichen. Um einzustufen ob dieses Level erreicht wurde kann beispielsweise überprüft werden ob der Nutzer eine bestimmte Aufgabe ohne Probleme, oder in einem gesetzten Zeitlimit bewältigen kann. Die Testpersonen sollten der letztendlichen Zielgruppe entsprechen und das System vorher noch nicht genutzt haben[Niel 95].

Effizienz Sobald die Lernkurve des Nutzers nicht mehr rapide ansteigt, wie es bei der anfänglichen Nutzung eines Systems meist der Fall ist, fokussiert er sich darauf effizient und produktiv mit dem System arbeiten zu können. Da effizientes Arbeiten nur ab einem gewissen Wissensgrad möglich ist, sollte bei der Messung dieses Attributes auf Testpersonen

zurückgegriffen werden die bereits Erfahrung mit dem System haben, gegebenenfalls sogar Experten sind. Die Einstufung, ab wann ein Nutzer als Experte gilt, kann entweder der Nutzer selbst treffen, oder man setzt voraus das Personen nach einem festgelegtem Benutzungszeitraum als Experten gelten. Eine typische Art um Effizienz zu messen ist es, mit einer der genannten Möglichkeiten einen Nutzer als Experte einzustufen, Testpersonen zu finden die diesen Kriterien entsprechen und die Zeit zu messen die diese Personen benötigen um Testaufgaben abzuschließen[Niel 95].

Wiedererkennungswert Neben Experten und neuen Nutzern gibt es noch die Gruppe der gelegentlichen Nutzer eines Systems. Für diese Gruppe ist das Attribut der Wiedererkennung besonders wichtig, da sie im Gegensatz zu neuen Nutzern nicht lernen müssen wie das Programm funktioniert, sondern sich lediglich an bereits Erlerntes erinnern müssen. Das Attribut der Erlernbarkeit wirkt also ebenfalls unterstützend um einem System einen hohen Wiedererkennungswert zu geben, aber prinzipiell gilt es zu beachten das das neu Erlernen eines Systems eben nicht mit dem Wiedereinstieg in selbiges gleichzusetzen ist. Diese beiden Attribute können sich also durchaus gegenseitig unterstützen, sind jedoch nicht austauschbar. Messbar ist der Wiedererkennungswert prinzipiell durch zwei Methoden. Eine davon ist die Durchführung eines Standard Tests mit Nutzern, die einige Zeit nicht mit dem System interagiert haben durchzuführen und die Zeit zu messen die sie benötigen um spezifizierte Aufgaben abzuschließen. Alternativ kann man einen Test durchführen, bei dem die Erinnerungen der Testpersonen geprüft werden. Hierbei werden die Nutzer nach dem Test gebeten die Effekte verschiedener Interaktionen, das Aussehen von Icons und die Bezeichnung bestimmter Befehle zu benennen. Die Anzahl der korrekten Antworten, die man hierbei erhält legt den Wiedererkennungswert des Systems fest. Obwohl diese zweite Testvariante leichter durchzuführen ist, hat sie die Schwäche, vor allem bei komplexeren Systemen, nicht sonderlich aussagekräftig zu sein. Es wurde festgestellt das Nutzer sich hier nicht an genaue Bezeichnungen oder Aussehen der Befehle erinnern konnten, jedoch kein Problem hatten diese in den Arbeitsabläufen zu finden oder zu nutzen. Deshalb ist der erste Testansatz empfehlenswerter, auch wenn dieser mehr Aufwand mit sich bringt[Niel 95].

Fehler Nutzer sollten während ihrer Interaktion mit dem System so wenig Fehler wie möglich machen. Unter einem Fehler versteht man in diesem Zusammenhang jede ausgeführte Aktion, durch die nicht das gewünschte Ergebnis erzielt wird. Die Fehlerrate eines Systems wird dementsprechend daran gemessen wie viele Fehler einem Nutzer unterlaufen während er eine Aufgabenstellung bearbeitet, und kann deshalb begleitend zu anderen Attributen untersucht werden[Niel 95].

Zufriedenheit Das letzte Attribut der Usability ist die Zufriedenheit, welche sich in diesem Fall darauf bezieht wie angenehm es für den Nutzer ist mit einem System zu interagieren.

Zum einen kann dieses Attribut medizinisch, zum Beispiel mit Hilfe von EEGs, der Beobachtung der Pupillenerweiterung oder der Herzrate untersucht werden. Da die Testpersonen jedoch meist ohnehin aufgeregt sind, sind solche zusätzlichen medizinischen Maßnahmen einer gewünschten entspannten Atmosphäre nicht zuträglich, abgesehen davon das der Aufwand für solche Untersuchten sehr hoch ist. Eine andere Methode die Zufriedenheit zu messen, ist es, die Nutzer einfach direkt zu fragen. Da man eine objektive Einschätzung der Zufriedenheit der Nutzer erhalten will ist es hier natürlich nötig mehrere Nutzer zu befragen, um aus den einzelnen subjektiven Meinungen eine objektive Meinung zu erhalten[Niel 95]. Diese Befragung findet meist in Form eines Fragebogens statt, da diese recht komplexe Art der Evaluierung in dieser Arbeit jedoch keine Verwendung findet, werden die Vorgehensweisen an dieser Stelle nicht weiter erläutert.

Da es sich bei der in dieser Arbeit untersuchten Software EB Guide Studio, näherer hierzu in Abschnitt 2.8, um eine Anwendung handelt mit der die meisten Nutzer über einen längeren Zeitraum auf hohem Niveau arbeiten, wird in dieser Usabilitystudie hauptsächlich das Attribut der Effizienz untersucht. Ergänzend dazu wird noch dokumentiert wie viele Fehler bei dem Nutzer durch die neuen Ergänzungen im Interface entstehen. Genaueres zu den Untersuchungen und Messungen folgt in Kapitel 5 dieser Arbeit.

2.4. User Experience

Entgegen einer häufigen Annahme bezeichnen Usability und User Experience (UX) nicht das Gleiche. Usability bezeichnet lediglich ein Teil der gesamten User Experience eines Systems[Knig 19]. UX bezieht sich nicht nur auf die reine Nutzungszeit eines Systems, sondern berücksichtigt auch den Zeitraum davor und danach, bezeichnet als Antizipierte Nutzung und Verarbeitung der Nutzungssituation. Usability ist hierbei, wie in Abb. 2.1 zu sehen, als wichtiger Faktor der User Experience in der aktiven Nutzungsphase zu betrachten, jedoch nicht mit dem Begriff gleichzusetzen [Saro 16]. Durch die zusätzliche Betrachtung der Effekte auf den Nutzer vor und nach der Nutzung, wie beispielsweise Erwartungen an das Produkt und Akzeptanz des selbigen, entstehen hier auch Verbindungen zur Gestaltung der Benutzerschnittstelle und dem Produkt-Design[Rich 16]. Zusammenfassend lässt sich festhalten das Usability zwar die funktionsbezogene Betrachtungsweise abdeckt, die User Experience als Ganzes jedoch auch emotionale Faktoren bezüglich des Designs und der Ästhetik berücksichtigt um das Nutzungsvergnügen möglichst hoch zu halten. Zusätzlich ist eine gute User Experience notwendig, wenn ein Produkt auf dem Markt bestehen will. Sobald es mehr als ein Produkt zur Lösung der gleichen Aufgabenstellung gibt, wird das mit der besseren User Experience Verwendung finden[Knig 19].

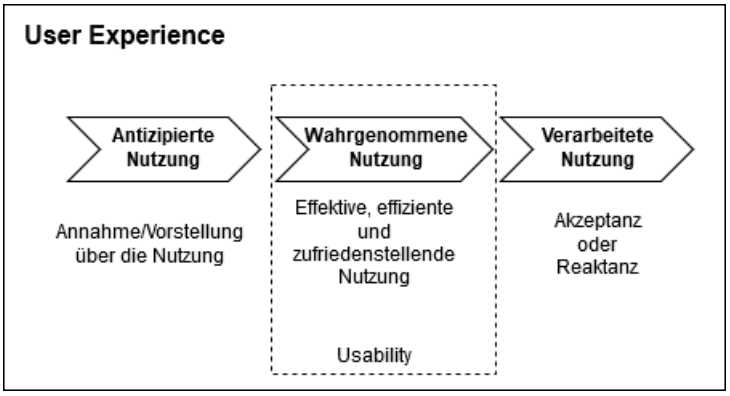


Abbildung 2.1.: Zusammenhang Usability und User Experience (nach [Saro 16])

2.5. Kognitive Last

Die Kognitive Last – auch als mentale Auslastung bekannt – bezeichnet im Allgemeinen die Differenz zwischen den Ressourcen (kognitiv, psychomotorisch oder wahrnehmend) die benötigt werden, um eine Aufgabe zu erfüllen, und den tatsächlich zur Verfügung stehenden Ressourcen. Die Größe der kognitiven Last beeinflusst also direkt die Leistung, die wir bei der Ausführung einer bestimmten Aufgabe erbringen können. Bei einer optimalen Auslastung erbringen Menschen maximale Leistung, ist sie hingegen zu klein oder zu groß geht die Leistung zurück, was in einem Fall auf Unterforderung und damit einhergehende Langweile, im anderen Fall auf Überlastung und Ablenkung zurückzuführen ist[Huma]. Beim Führen eines Fahrzeuges ist die Leistung des Fahrers beispielsweise maximal, wenn seine gesamte Aufmerksamkeit dem Fahren gilt. Allerdings wird er durch Nebentätigkeiten, wie zum Beispiel das Reagieren auf Warnungen, dazu gezwungen sein Multitasking zu betreiben, was seine Performance in Bezug auf beide Tätigkeiten einschränken wird. Es ist also auch wichtig abzuwägen, wie man Warnungen in Benutzeroberflächen darstellt, um keine mentale Überlastung des Benutzers hervorzurufen. Beim Fahren ist beispielsweise die visuelle Modalität durch den ständigen Blick auf die Straße schon sehr ausgelastet, weshalb zur Darstellung von Warnung auch aus Gründen der Auslastung auf Warnungen in Form von Videos oder Animationen verzichtet werden sollte und eher auf die anderen Modalitäten des Menschen, wie Hören und Fühlen ausgelagert werden.

2.6. Human - centered design process

Human-centered design ist ein Ansatz zur Entwicklung interaktiver Systeme, welcher das Ziel verfolgt möglichst nützliche Systeme zu entwickeln und diese gut benutzbar zu machen. Das wird vor allem durch das fokussieren auf den Benutzer und dessen Bedürfnisse und

durch die Anwendung von Ergonomie, Wissen und Techniken der Usability während des Entwicklungsprozesses erreicht[Elekd]. Der in Abb. 2.2 zu sehende User Experience Engineering Process beschreibt den Human Centered Design Process für Elektrobit und basiert auf der ISO Norm 9241 - 210. Das Model sieht Iterationen benutzerorientierter Aktivitäten vor, durch die fortlaufend der Erkenntnisstand der Entwickler erhöht wird, was bei dem ursprünglichen Ansatz eines Wasserfallmodells nicht der Fall war.

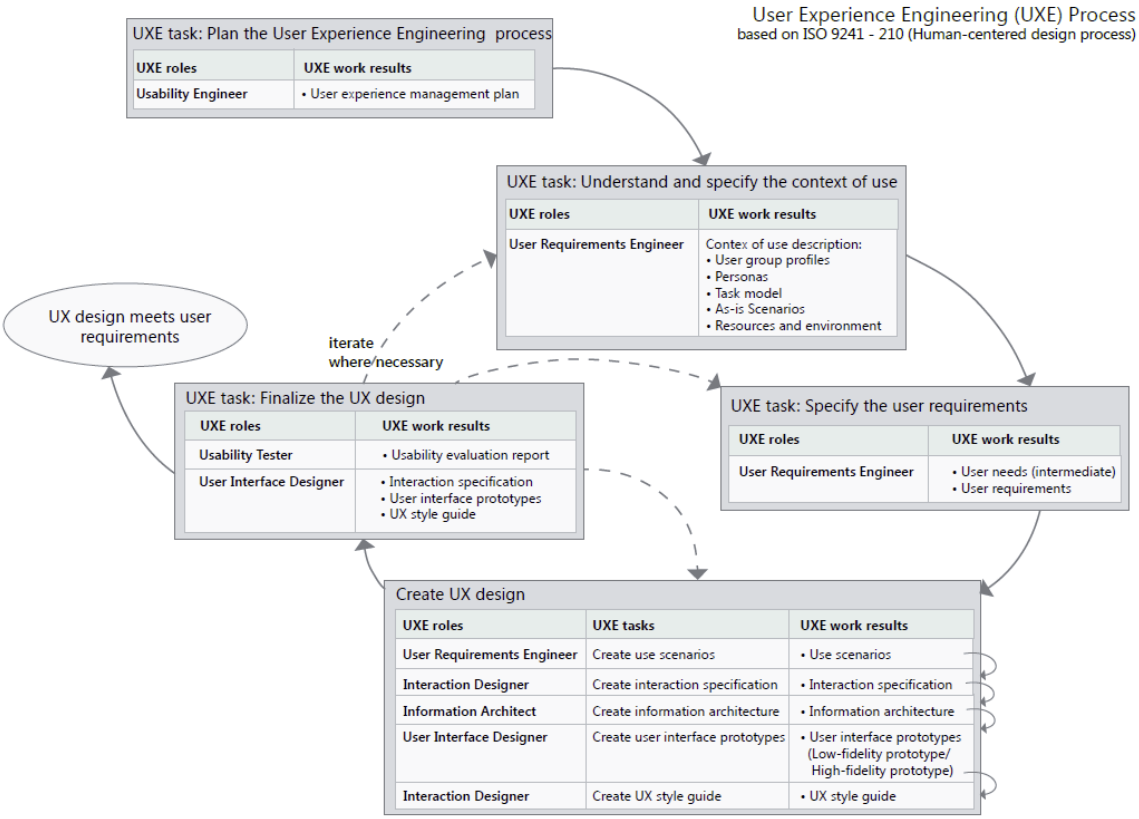


Abbildung 2.2.: Human - Centered - Design - Process bei Elektrobit

Die ISO 9241-210 Norm beschreibt das benutzerorientierte Vorgehen in Entwicklungsprojekten, und gibt Anregungen wie Usability-Engineering-Aktivitäten im Entwicklungsprojekt angewendet werden können. Allgemein definiert die Norm das Vorgehen so, dass die Gestaltung des Systems auf vorangehenden Analysen der Nutzer, deren Arbeitsaufgaben und Arbeitsabläufen aufbaut. Um aussagekräftige Analysen zu erhalten werden die Nutzer während des Entwicklungsprozessen immer wieder aktiv mit einbezogen, was vor allem durch die fest eingeplanten Iterationen geschieht. Optimalerweise vereint das Gestaltungsteam fachübergreifende Kenntnisse und kann die in Abb. 2.2 benannten Rollen optimale besetzen[DIN].

Der Human-centered design process kann und sollte bei der Entwicklung jedes interaktiven Systems verwendet werden, welches eine Benutzeroberfläche aufweist, und wird deshalb auch in der Entwicklung dieser Bachelorarbeit angewendet werden. Aufgrund der Tatsachen, dass bei dieser Arbeit einerseits eine zeitliche Einschränkung vorliegt, und andererseits die Bearbeitung von einer Person und nicht von einem Team durchgeführt wird werden die Iterationen und die einzelnen Schritte jedoch in einer abgeschwächten Form durchgeführt werden. Genauerer Erläuterungen dazu finden sich in den entsprechenden Abschnitten dieser Arbeit.

2.7. Gestaltprinzipien der Usability

Durch die Interaktion mit einem Produkt oder einer Anwendung entstehen bei Nutzern immer Erfahrungen. Ob diese Erfahrungen positiv oder negativ behaftet sind wird maßgeblich dadurch beeinflusst ob das Design der Anwendung den Nutzer unterstützt oder eher behindert, also ob die User Experience gut oder schlecht ist. Bei der Interaktion mit einem Produkt muss der Nutzer immer herausfinden wie er mit ihm umgehen soll, welche Funktionen es bietet oder was überhaupt der Sinn des Produktes ist. Diese Eigenschaft wird als Discoverability oder Erkennbarkeit bezeichnet und ist für eine gute User Experience möglichst hoch zu halten. Discoverability resultiert aus denen im Folgenden erläuterten Gestaltprinzipien der Usability, weshalb diese bei jeden Entwurf eines Systems berücksichtigt und einbezogen werden sollten[Norm 16].

Konsistenz

Konsistenz im Design wirkt sich insofern positiv auf die Usability eines Systems aus, das die Benutzbarkeit steigt, wenn gleiche Funktionen in einem System auch auf identische Art und Weise dargestellt werden. Dadurch wird es dem Nutzer ermöglicht bereits Gelerntes in einem neuen Kontext anzuwenden, ohne effektiv darüber nachdenken zu müssen, neue Dinge schneller zu lernen und sich auf die relevanten Dinge einer Aufgabe zu konzentrieren. Konsistenz in Systemen lässt sich in vier Kategorien unterteilen, die im Folgenden jeweils kurz erläutert werden.

Ästhetische Konsistenz Ästhetische Konsistenz bezieht sich auf Konsistenz in Stil und Aussehen. Dadurch entsteht ein hoher Wiedererkennungswert und es wird Zugehörigkeit signalisiert. Das einfachste Beispiel hierfür ist ein Firmenlogo, welches immer mit konsistenter Schriftart und Farbgebung auftaucht.

Funktionale Konsistenz Funktionale Konsistenz bezieht sich auf die Konsistenz von Bedeutung und Aktion, beispielsweise zeigt eine Ampel immer orange bevor sie rot wird. Der Nutzer kann also sicher mit einer immer gleichen darauffolgenden Aktion auf den aktuellen Zustand rechnen. Usability und Lernfähigkeit werden durch funktionale Konsistenz erhöht, indem diese bereits verinnerlichten Abläufe einfach auf neue Situationen übertragen werden können. So wird beispielsweise ein einheitlicher Playbutton für alle Geräte, von Kassettenrekorder bis Streamingdienst, verwendet. Die Nutzung bereits bekannter Symbole ermöglicht dem Nutzer also eine intuitive Interaktion und macht es möglich die Aufmerksamkeit auf unbekannte Aspekte zu richten, die tatsächlich noch neu gelernt werden müssen.

Interne Konsistenz Interne Konsistenz bezieht sich auf Konsistenz mit anderen Elementen im System, beispielsweise sind Wegweiser in einem Park konsistent zueinander. Dies erzeugt ein Vertrauensgefühl bei den Nutzern, und vermittelt ein durchdachtes Designkonzept und erweckt nicht den Eindruck das alle Komponenten einfach zusammengewürfelt wurden.

Externe Konsistenz Externe Konsistenz bezieht sich auf Konsistenz mit anderen Elementen in der Umgebung. Die Vorteile der internen Konsistenz werden hierbei systemübergreifend erweitert. Das Erreichen externer Konsistenz gestaltet sich jedoch auch schwieriger, da unabhängige Systeme selten exakt gleiche Designstandards haben.

Nicht alle dieser Konsistenzstandards können oder sollten in allen Fällen angewendet werden. Es lässt sich jedoch festhalten das Elemente in einer logischen Gruppe immer ästhetisch und funktional konsistent sein sollten. Insgesamt sollte ästhetische und funktionale Konsistenz, wann immer möglich berücksichtigt werden, da durch ästhetische Konsistenz bei der Einführung einmaliger Identitäten Wiedererkennung gewährleistet werden kann und funktionale Konsistenz aus den bereits genannten Gründen die Usability eines Systems erhöht. Abschließend sollten Systeme immer eine interne Konsistenz aufweisen, sollten bereits Designstandards existieren, gilt es diese zu analysieren.[Lidw 10]

Sichtbarkeit

Das Prinzip der Sichtbarkeit besagt, dass Systeme benutzbarer sind, wenn der aktuelle Status des Systems, mögliche Aktionen und deren Auswirkungen für den Nutzer deutlich erkennbar sind. Es beruht auf der Erkenntnis, dass Menschen schneller und besser Lösungswege finden wenn sie aus einer Reihe von Optionen auswählen können, anstatt sich selbstständig an alle Möglichkeiten aus dem Stegreif erinnern zu müssen[Lidw 10]. Das macht es zu dem wichtigsten Prinzip für hohe Usability in komplexen Systemen, jedoch auch zu dem, welches am meisten verletzt wird[Norm 16]. Häufig wird versucht alle möglichen Optionen,

die ein System bietet sichtbar zu machen, was vor allem in komplexeren Systemen dazu führt das sie relevanten Optionen durch Informationsüberladung auf Seiten des Nutzers schwerer zu erreichen sind. Anstatt die Usabilty zu erhöhen wird also genau das Gegenteil erzielt. Lösungen für diese Problemstellung sind beispielsweise eine Hierarchische Anordnung der Elemente oder Kontextsensitivität des Systems. Bei der Hierarchischen Anordnung werden Funktionen und Informationen in logische Kategorien unterteilt und in übergeordneten Menüs versteckt, welche bei Bedarf ausgeklappt werden können. Ein Beispiel hierfür ist ein Dropdownmenü, wie man es von vielen Websites kennt. In einem Kontextsensitiven System werden Aktionsmöglichkeiten und Informationen je nach aktuellem Status des Systems versteckt oder angezeigt. Beispielsweise bekommt man in einer Modellierungssoftware mit unterschiedlichen Elementen immer nur die Properties angezeigt die für das aktuell ausgewählte Element relevant sind, und nicht alle Properties die im gesamten System existieren[Lidw 10].

Affordanz

Affordanz bezeichnet die Möglichkeiten mit einem Objekt zu interagieren. Beispielsweise kann eine Checkbox an und wieder abgewählt oder ein Schieberegler nach oben und unten geschoben werden. Die sichtbare Affordanz bezeichnet die Eigenschaft das einem Objekt die Interaktionsmöglichkeiten bereits angesehen werden können, ohne mit ihm interagiert zu haben. Sichtbare Affordanz ist vor allem in der User Interface Gestaltung wichtig, weil praktisch betrachtet alle Pixel auf einem Bildschirm anklickbar sind, jedoch in den meisten Fällen keine Aktion durch das klicken ausgelöst wird. Deshalb ist es wichtig dem Nutzer durch das Aussehen der Elemente zu vermitteln ob diese, wenn sie geklickt werden eine Aktion auslösen, um dem Nutzer wahlloses Klicken durch das Interface zu ersparen, bis ein interaktives Objekt gefunden wird. Solche Probleme können visuell gelöst werden, indem man Objekte im User Interface wie Objekte in der echten Welt aussehen lässt, also beispielsweise einen Button dreidimensional gestaltet. Alternativ kann man beispielsweise alle anklickbaren Objekte etwas anders gestalten als den Rest der Objekte im Interface, um dem Nutzer zu vermitteln das hier eine Interaktion stattfinden kann[Knig 19].

Rückmeldung

Eine der grundlegendsten Richtlinien um die Usability eines Systems zu erhöhen, ist es dem Nutzer immer eine Rückmeldung auf seine Aktionen zu geben. Das bedeutet dem Nutzer immer den aktuellen Systemstatus anzuzeigen und wie seine Aktion vom System interpretiert wurde. Eine Rückmeldung ist vor allem auch dann wichtig, wenn die gewünschte Aktion nicht erfolgreich vom System ausgeführt wurde. Durch fehlende Rückmeldungen kommt Misstrauen beim Nutzer auf, weil ihm nicht vermittelt wird ob auf seine Aktionen

auch eine Reaktion des Systems erfolgt[Knig 19]. Dieses Problem tritt beispielsweise auf, wenn ein System lange braucht um eine Eingabe zu verarbeiten und es versäumt dies dem Nutzer mitzuteilen. So ein Verhalten könnte zur fälschlichen Annahme führen das System wäre kaputt, oder dazu das der Nutzer beginnt neue Interaktionsmöglichkeiten anzuklicken. Reagiert das System innerhalb 0.1 Sekunden, nimmt der Nutzer dies als sofortiges Feedback wahr und keine gesonderte Meldung ist nötig. Bewegt sich die Reaktionszeit zwischen 0.1 und 1.0 Sekunden ist für gewöhnlich auch keine besondere Rückmeldung vom System nötig, auch wenn der Nutzer hier bereits nicht mehr das Gefühl hat direkt mit den Daten zu interagieren, da eine kurze Verzögerung stattfindet. Sobald die Wartezeit auf Seiten des Nutzers jedoch eine Sekunde überschreitet sollte sich der Cursor in eine Sanduhr oder ähnliches verwandeln und damit die Rückmeldung geben, dass das System beschäftigt ist. Sollte die systemseitige Verarbeitung der Eingabe länger als 10 Sekunden dauern ist es ratsam die Ladeanzeige durch eine konkrete Fortschrittsleiste zu ersetzen[Niel 95].

Mapping

Mapping bezeichnet die Beziehung zwischen Bedienelementen und den Effekten den deren Aktivierung auslöst. Wenn der Effekt den eine Nutzerinteraktion nach sich zieht, den Erwartungen des Nutzers entspricht, handelt es sich um gutes Mapping. Die ist beispielsweise bei einem elektronischen Fensterheber der Fall. Wird hier der Hebel nach oben bewegt, hebt sich das Fenster, bewegt man den Hebel nach unten senkt es sich. Gutes Mapping wird zum Großteil durch Ähnlichkeit, das Verhalten oder der Bedeutung innerhalb eines Layouts erreicht. Entspricht beispielsweise das Layout von Herdplattenreglern der Anordnung der Platten wird gutes Mapping durch das Layout erzeugt, der bereits beschriebene Fensterheber arbeitet mit dem Verhalten, und die Tatsache das ein Notfallknopf rot eingefärbt wird ist darauf zurückzuführen das die meisten Menschen rot mit Gefahr oder dem Stopplicht einer Ampel assoziieren. In jedem dieser Fälle macht die Ähnlichkeit es möglich den Effekt der Handlung vorherzusehen, und vereinfacht dadurch die Bedienung für den Nutzer. Aktionsmöglichkeiten müssen so platziert werden das ihre Position und ihr Verhalten dem Layout und dem Verhalten die Anwendung angepasst sind. Außerdem sollte es vermieden werden durch eine identische Aktion verschiedene Reaktionen auszulösen[Lidw 10].

Einschränkungen

Einschränkungen in einem System limitieren die Interaktionsmöglichkeiten für den Nutzer. Wird beispielsweise ein Button ausgegraut, der im aktuellen Kontext ohnehin keine Aktion ausführen würde wird der Nutzer rein optisch daran gehindert eine nicht zielführende

Aktion auszuführen. Durchdachte Einschränkungen dieser Art machen ein Design einfacher nutzbar und reduziert deutlich die Wahrscheinlichkeit von Fehlschlägen während der Systeminteraktion [Lidw 10].

Physische Einschränkungen Physische Einschränkungen limitieren den Bereich in den Aktionen ausgeführt werden können, indem sie Eingaben des Nutzers umwandeln oder umleiten. Eine Art der Einschränkungen ist das konvertieren der Eingabe in lineare oder kurvenförmige Bewegung, wie es Beispielsweise bei der Interaktion mit einer Scrollbar der Fall ist. Mithilfe von Achsen können wirkende Kräfte in Rotationsbewegungen umgewandelt werden, was eine Kontrolloberfläche mit unendlicher Größe auf einem kleinen Feld erzeugt, was am Beispiel der Computermaus gut zu erkennen ist. Die Letzte Form der Einschränkung passiert über Barrieren, die die Eingabe verlangsamen, komplett ausbremsen oder umleiten. Die Einfassung eines Computerbildschirms beschränkt beispielsweise physisch die Interaktionsfläche für den Nutzer. Allgemein sind Physische Einschränkungen nützlich um die Anzahl fehlerhafter Eingaben zu vermeiden, oder manche Eingaben erst gar nicht zu ermöglichen[Norm 16].

Psychologische Einschränkungen Psychologische Einschränkungen limitieren die Anzahl möglicher Aktionen durch Nutzung des Wissens über das Verhalten und die Denkweise der Nutzer. Dies passiert durch den Einsatz von Symbolen, nutzen bekannter Konventionen oder durch das bereits erwähnte Mapping. Symbole sind sinnvoll um Dinge zu benennen, zu erklären oder auch um Warnungen visuell, auditiv oder fühlbar darzustellen. Mit Konventionen macht man sich bekannte Interaktionsmöglichkeiten zunutze, weshalb sie sich gut eignen Systeme sowohl konsistent als auch leicht benutzbar zu machen. Mappings sind, aus bereits erwähnten Gründen, nützlich um dem Benutzer zu vermitteln welche Aktionen aufgrund der Sichtbarkeit, Position oder dem Aussehen der Elemente möglich sind[Norm 16].

Zusammenfassend sollten Einschränkungen im Allgemeinen verwendet werden um die Benutzbarkeit eines Systems zu vereinfachen und die Anzahl an Fehlern zu minimieren. Physische Einschränkungen dienen hierbei eher dem Verhindern ungewollter Eingaben oder gefährlicher Aktionen, Psychologische Einschränkungen sollen das Design eines Systems für den Nutzer klarer und intuitiver gestalten.

Vorteile für den Nutzer

Durch das Befolgen der soeben erläuterten Gestaltprinzipien bei dem Entwurf eines User Interfaces, ist es möglich sich die Art und Weise wie Nutzer visuelle Reize verarbeiten zunutze zu machen um die Benutzerfreundlichkeit des Systems zu erhöhen. Das passiert vor allem dadurch, dass man die kognitive Last des Nutzers verringern während er sich mit

dem Interface auseinandersetzt, indem man bereits bekannten Gestaltprinzipien nutzt. Das bedeutet, dass der Nutzer seine Energie nicht darauf verschwenden muss darüber nachzudenken wie mit den Bestandteilen des Interfaces interagiert werden kann, oder was deren Funktionen sein könnten[Knig 19].

2.8. EB Guide Studio

Wie bereits in Abschnitt 1.2 erwähnt dient EB GUIDE der Entwicklung multimodalder HMIs. Um nicht nur das Design sondern auch das Verhalten von User Interfaces bestimmen zu können und eine Auslieferung auf das Zielsystem zu ermöglichen besteht die Produktlinie EB GUIDE aus den verschiedenen, in Abb. 2.3 zu sehenden, Komponenten. Hierbei wird zwischen Komponenten für das Graphical User Interface (GUI) und Komponenten für die Sprachsteuerung unterschieden. Da die Sprachkomponenten jedoch für diese Arbeit nicht weiter relevant sind werden diese in den folgenden Kapiteln auch nicht genauer erläutert. Innerhalb des GUI Bereiches bildet EB GUIDE Studio das tatsächliche Modellierungstool, mit dem das Verhalten und Aussehen der Benutzeroberfläche definiert wird. Für das entwickelte Modell stellt das EB GUIDE Target Framework auf dem Zielsystem die Laufzeitumgebung bereit.[Elekb]

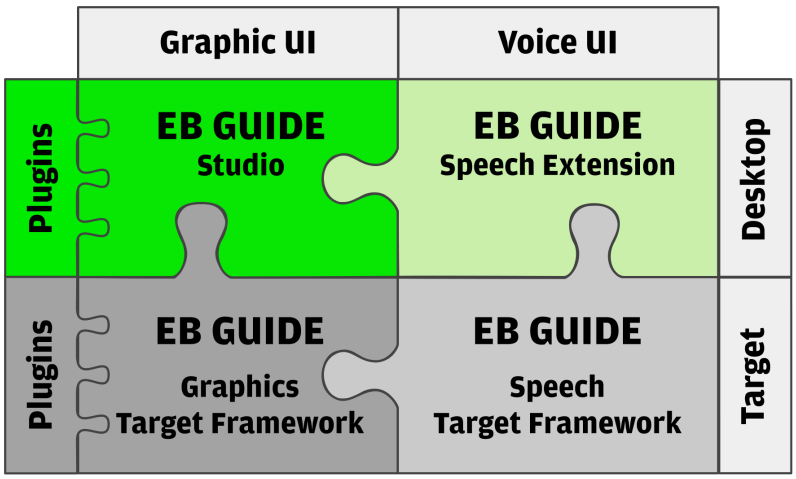


Abbildung 2.3.: Aufbau EB GUIDE

EB Guide Studio ist das Interface von EB Guide mit dem nach dem What-You-See-Is-What-You-Get (WYSIWYG) Prinzip User Interfaces modelliert werden. Durch das WYSIWYG

Prinzip ist es während des Modellierens einer View bereits möglich das Endergebnis des Designs zu sehen. Das Verhalten des Interfaces hingegen wird mithilfe einer Zustandsmaschine, der sogenannten Statemachine, modelliert die auf dem UML- Prinzip aufbaut. Die Trennung der Logik und des Designs wird in EB GUIDE Studio grafisch durch zwei unterschiedliche Arbeitsoberflächen gestaltet in welchen den Modellierern jeweils die entsprechenden Tools und Elemente zur Verfügung stehen.

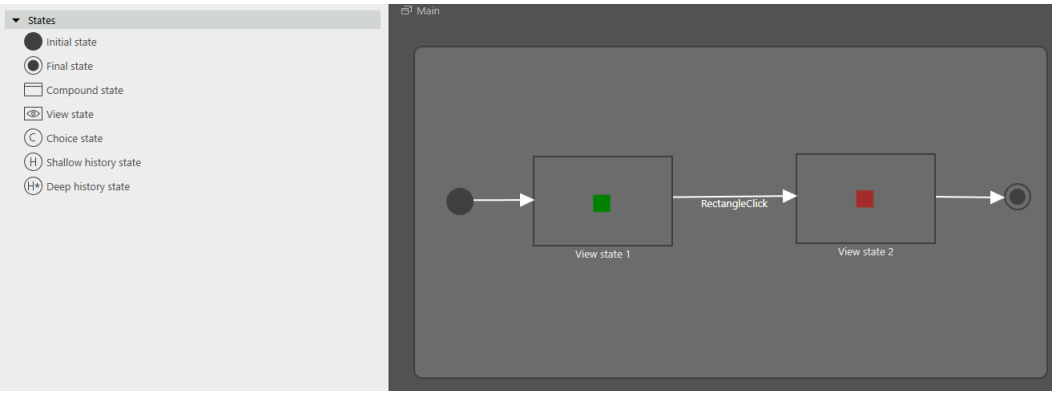


Abbildung 2.4.: EB Guide Studio Statemachine

In Abb. 2.4 ist die Statemachine für ein simples Beispiel zu sehen. Links neben der Arbeitsfläche befindet sich eine Toolbox mit deren Inhalt per Drag and Drop auf der Arbeitsfläche die benötigte Logik definiert wird. Wie bei UML-Diagrammen gibt es einen Initial State der den Startpunkt angibt und einen Final State der die Statemachine beendet. Die ebenfalls zu sehenden View States stehen für einen Screen im Endprodukt, dementsprechend wird in den View States auch das Aussehen der Interfaces definiert. Die Verbindungen zwischen den States werden als Transitionen bezeichnet und mithilfe von Events ausgelöst. Events stellen hierbei beliebige Ereignisse dar die durch Elemente in der View ausgelöst werden können. Der Auslöser für das Event wird mithilfe einer eigens für EB Guide entwickelten Skriptsprache als Trigger für dieses gesetzt. Die häufigsten Auslöser sind Benutzeraktionen mit Widgets, die in Abb. 2.5 zu sehen sind. Widgets sind Elemente, mit denen das Aussehen des Interfaces im sogenannten View Editor bestimmt wird und die sich in Basis- und 3D-Widgets einteilen lassen. Alle Basiswidgets verfügen über Basiseigenschaften wie Höhe, Breite und Farbe sowie über spezifische Eigenschaften wie zum Beispiel "Touch-Released"bei einem Button.[Elekc] Beipspielsweise wird das Event "RectangleClick", was sich in Abb. 2.4 an der Transition zwischen View State 1 und View State 2 befindet in diesem Beispiel durch das Klicken auf das grüne Rechteck in View State 1 ausgelöst. Ist die Benutzeraktion abgeschlossen findet der Übergang von View State1 zu View State 2 statt und der Nutzer sieht nun anstatt dem grünen ein rotes Rechteck. Damit diese Aktion erfolgreich ausgeführt wird muss das grüne Rechteck die Eigenschaft "Touch-Released" zugewiesen bekommen und über die EB Guide Skriptsprache mitgeteilt bekommen das Event RectangleClick zu feuern

sobald das grüne Rechteck berührt wurde. Da sich dieses Event in der Statemachine an der Transition zwischen den beiden States befindet wird nun durch einen Klick auf das grüne Rechteck ein Bildschirmwechsel zwischen den beiden View States ausgelöst.

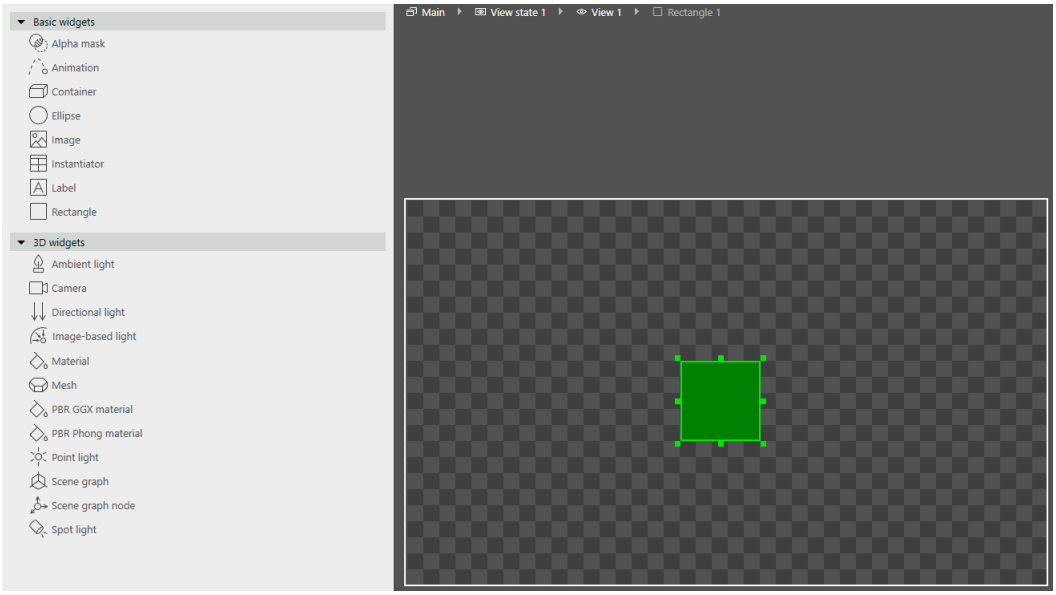


Abbildung 2.5.: EB Guide VIEW

Kapitel 3.

Analysen

3.1. Voruntersuchungen

Wie in Abschnitt 2.3 bereits erwähnt, liegt eine hohe Usability genau dann vor wenn eine System von der für es bestimmten Zielgruppe effizient verwendet werden kann.[Rich 16] Um dies zu gewährleisten ist es vorab nötig diese Zielgruppe zu kennen, zu analysieren und eventuelle Schwierigkeiten in der Benutzung des Systems aufzudecken. Das Erkennen dieser Schwierigkeiten muss, aus Gründen die ebenfalls in Abschnitt 2.3 erläutert wurden, immer in Relation zu der aktuell ausgeführten Aufgabe geschehen, weshalb im Folgenden zuerst die Ausgangssituation beschrieben wird in der die Analysen durchgeführt wurden. Anschließend folgt noch ein Überblick über die Analysemethoden, bevor abschließend die Ergebnisse dargestellt werden.

3.1.1. Ausgangssituation

Im ersten Schritt des Human-centered design process, zu sehen in Abb. 2.2 des vorherigen Kapitels, gilt es den Benutzungskontext zu verstehen. Aus all den möglichen Aufgaben, die der User Requirements Engineer in diesem Schritt des Prozesses bearbeiten soll, wird in dieser Arbeit auf die Nutzergruppe, deren tägliche Aufgaben und die Arbeitsumgebung eingegangen.

Arbeitsumgebung Das Arbeitsumfeld bildet das in Abschnitt 2.8 bereits erwähnte EB Guide 6. Es ist hierbei Situationsabhängig ob die Modellierer mit dem Speech Anteil von EB Guide arbeiten oder nicht, für diese Arbeit werden die Interaktionen mit dem Speech Teil ignoriert und sich nur auf die Usability von EB Guide Studio konzentriert. Auch auf beide Target Frameworks wird im Folgenden nicht mehr weiter eingegangen, da die Nutzerinteraktion mit EB Guide über die Schnittstelle EB Guide Studio stattfindet.

Nutzergruppe Die Zielgruppe für die Analysen im Rahmen dieser Arbeit deckt sich logischerweise mit der Nutzergruppe von EB Guide. Im Rahmen dieser Arbeit wurden nur Personen beobachtet und analysiert die bei Elektrobit beschäftigt sind. Diese arbeiten teilweise bereits sehr routiniert und auch täglich mit der Software, andere benötigen diese nur sporadisch in ihrer täglichen Arbeit. Es werden also Experten und gelegentliche Nutzer in dieser Arbeit untersucht, neue Nutzer werden nicht beachtet. Wie in Abschnitt 2.3 nachvollziehbar ergeben sich hieraus theoretisch die Möglichkeiten das System auf Effizienz, Wiedererkennungswert, Fehler und Zufriedenheit zu testen, welche jedoch im Folgenden noch eingegrenzt werden.

Arbeitsaufgaben Der Großteil der Modellierer arbeitet in laufenden Projekten von Elektrobit und modelliert dort mithilfe von EB Guide 6 Human Machine Interfaces für die Automobilbranche. Ein anderer Teil der Zielgruppe arbeiten an Kundendemonstrationen mit derern Hilfe dargestellt wird was mit der aktuellen Version von EB Guide 6 modelliert werden kann. Beide Gruppen setzten bei ihrer Arbeit Spezifikationen um die nach den Wünschen des Kunden direkt von diesem oder von Designfirmen erstellt werden. Diese Spezifikationen bestehen meist aus einem schriftlichen Teil der die Logik beschreibt nach der das Interface arbeiten muss, sowie aus einem grafischen Teil der die Anordnung von Icons und Texten zeigt.

3.1.2. Vorgehensweise

Nachdem nun der Benutzungskontext analysiert und verstanden wurde, gilt es im zweiten Schritt des Prozesses die Benutzeranforderungen zu spezifizieren. Diese Benutzeranforderungen beinhalten die Definition identifizierter Bedürfnisse der Nutzer, sind testbar, eindeutig und konsistent. Zu unterscheiden sind qualitative und quantitative Benutzeranforderungen, wobei beide eine Basis für das Design des interaktiven Systems bieten und durch Evaluierung des Systems verifiziert werden können. Qualitative Anforderungen beziehen sich auf die Art und Weise wie das System genutzt wird um das Ziel zu erreichen, quantitative Anforderungen hingegen setzen messbare Ziele für die Usability und User Experience.[Eleke]

Um repräsentative Benutzeranforderungen zu erhalten ist es notwendig, das die Bedürfnisse der Nutzer, aus denen die Anforderungen gebildet werden, tatsächlich auch den Bedürfnissen der Zielgruppe entsprechen. Diese Bedürfnisse erhält man beispielsweise durch Interviews oder Beobachtungen innerhalb der Zielgruppe. Diese Vorgehensweisen wird im Rahmen dieser Arbeit kombiniert angewendet. Die Nutzer werden bei ihrer täglichen Arbeit beobachtet, werden währenddessen aufgefordert ihre aktuellen Arbeitsschritte zu erklären wodurch sie bei Bedarf auch automatisch Kritik am Interface äußern können. Zusätzlich können durch den Beobachter begleitend Fragen gestellt werden.

3.1.3. Ergebnisse

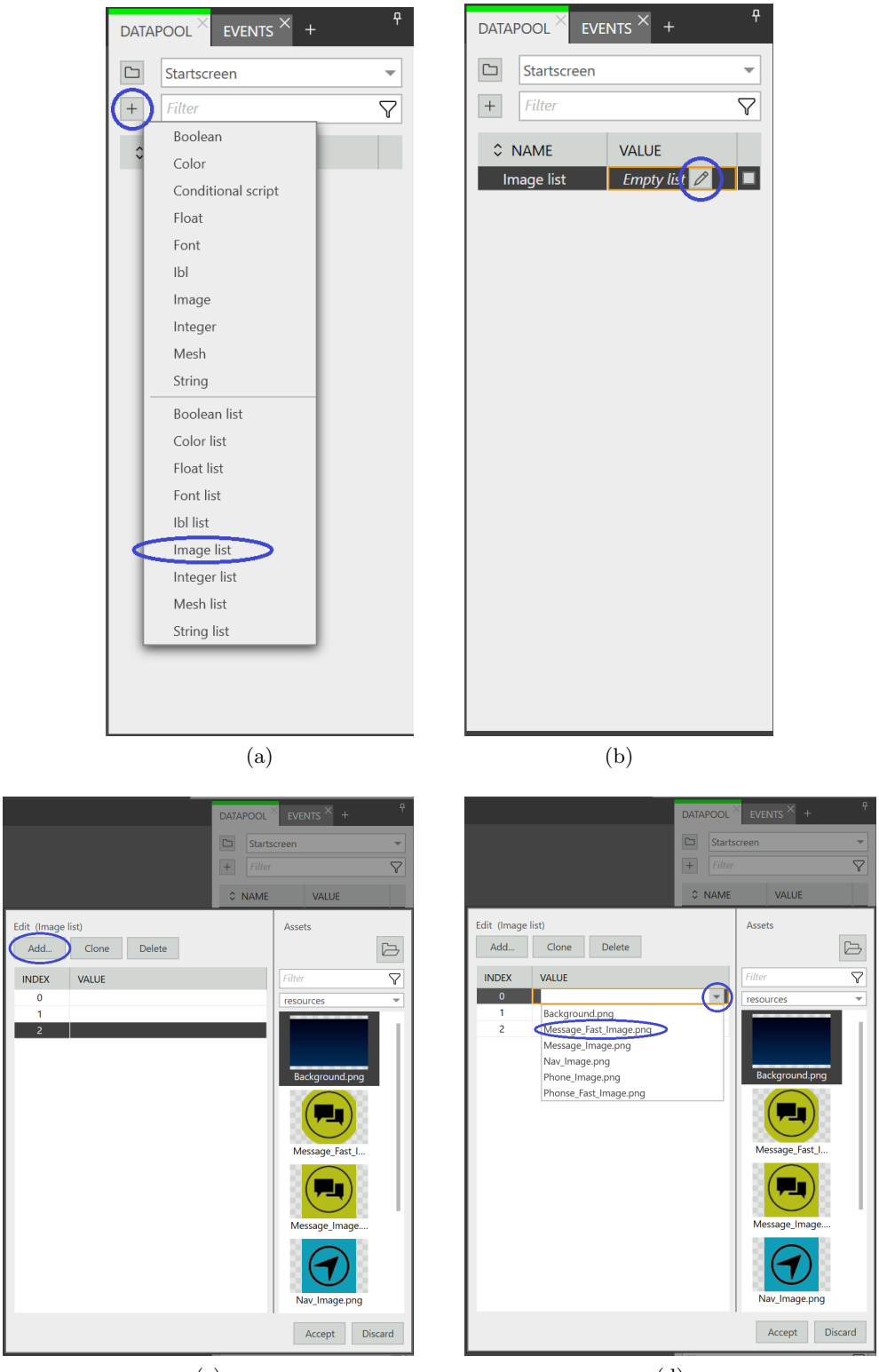
Bei der Durchführung der Beobachtungen fallen einige Dinge auf die die Modellierer bei der Durchführung ihrer Arbeit behindern oder erheblich verlangsamen. Teilweise fällt den Nutzern das selbst auf und sie weisen den Beobachter darauf hin, teilweise haben sie sich bereits so an die Arbeitsschritte gewöhnt das die Behinderung nur einem Außenstehenden auffällt, den Nutzern selbst jedoch kaum noch.

Im Folgenden werden zuerst Beobachtungen allgemein erläutert, bevor die Bedürfnisse der Nutzer formuliert werden, damit es abschließend möglich ist die Benutzeranforderungen zu definieren.

Image List Bei einer Image List handelt sich um ein Datapool Item. Datapool items sind Modelelemente die benutzt werden können um Daten von der Applikation an das Interface zu senden, oder umgekehrt. Ebenfalls können damit Daten gespeichert werden die entweder nur von Seiten des Interfaces genutzt werden, oder nur von der Applikationsseite.[Elekc] in Abb. 3.1 kann das befüllen eines solchen Datapool items am Beispiel einer Image List nachvollzogen werden. In Abbildung b) initialisiert der Nutzer das Befüllen der Liste durch einen Klick auf den SStift"Button, wodurch sich das Pop Up in Abbildung c) öffnet. Die hier zu sehenden Platzhalter für Index und Value müssen einzeln durch einen Klick auf den Ädd.."Button hinzugefügt werden, und danach einzeln, wie in Abbildung d) zu sehen, mit dem gewünschten Image befüllt werden. Für die minimale Auswahl in dem hier zu sehenden Beispiel ist der Aufwand noch erträglich, es ist jedoch zu bedenken das in tatsächlichen Projekten Image Lists mit mindestens 100 Images erstellt werden. Das bedeutet für den Nutzer mehrere Stunden Arbeit, die keinen wirklich Mehrwehrt liefern und den Joy of Use deutlich mindern.

Resultierende Benutzeranforderung Nutzer, die eine Datapool Liste anlegen, müssen die Möglichkeit haben mehrere Images gleichzeitig zu dieser Liste hinzuzufügen.

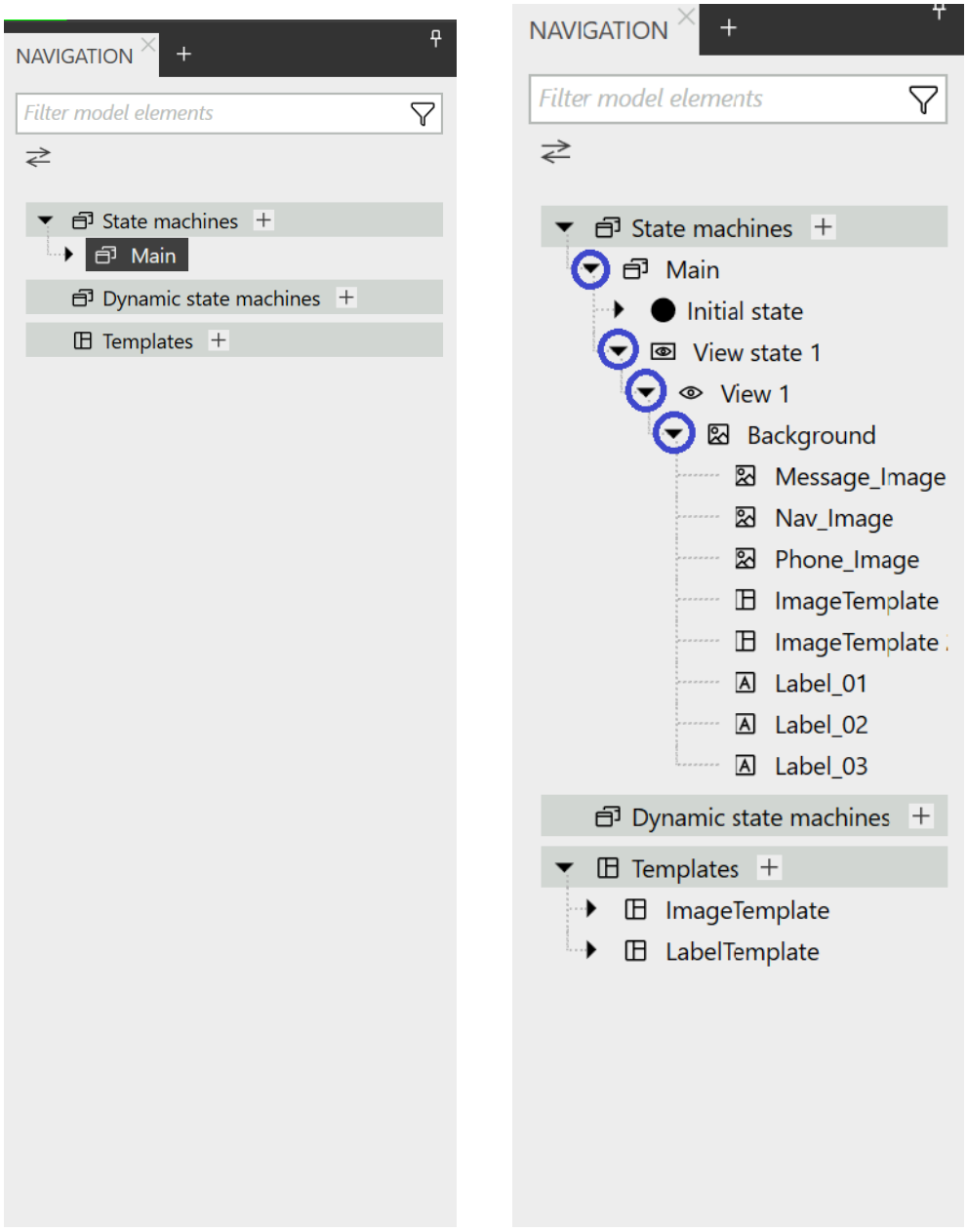
Navigation Wird ein neues Element in der View hinzugefügt wird der WidgetTree in der Navigation, in Abb. 3.2 zu sehen, nicht automatisch ausgeklappt. Durch Beobachtung kann festgestellt werden das die Nutzer, nachdem sie ein neues Element eingefügt haben dieses sofort umbenennen. Für diesen Vorgang müssen sie das Element im WidgetTree erst gesucht werden, was in einigen Fällen sehr viele Klicks, und damit auch Zeit fordert. Sollte der Nutzer in Abb. 3.2 beispielsweise gerade das NavImage eingefügt haben, müssen erst vier Ebenen ausgeklappt werden bevor eine Umbenennung möglich ist.



(a) (b)

(c) (d)

Abbildung 3.1.: Usability - Schwäche Image List



(a) (b)

Abbildung 3.2.: Usability - Schwäche Navigationt

Resultierende Benutzeranforderung Nutzer, die ein neues Element zu ihrer View hinzufügen, müssen die Position dieses Items sofort im WidgetTree nachvollziehen können.

Template Properties Ein Widget Template ermöglicht die Definition eines individuellen Widgets, welches beliebig oft in einem EB Guide Model benutzt werden kann. Es besteht die Möglichkeit Templates zu erstellen die auf bereits existierenden Widgets aufbauen, oder von einem anderen Template abgeleitet sind. Nach der Erstellung kann das Template nach den eigenen Wünschen und Bedürfnissen angepasst werden, beispielsweise durch das hinzufügen von Properties.

Ein Widget Template besitzt außerdem ein Template Interface, welches jene Poperties beinhaltet des Templates beinhaltet welche für jeder Instanz des Templates sichtbar und veränderbar sein sollen. Jede Instanz eines Templates erbst also die Properties des Template Interfaces, welche Template Properties genannt werden.[Elekc]

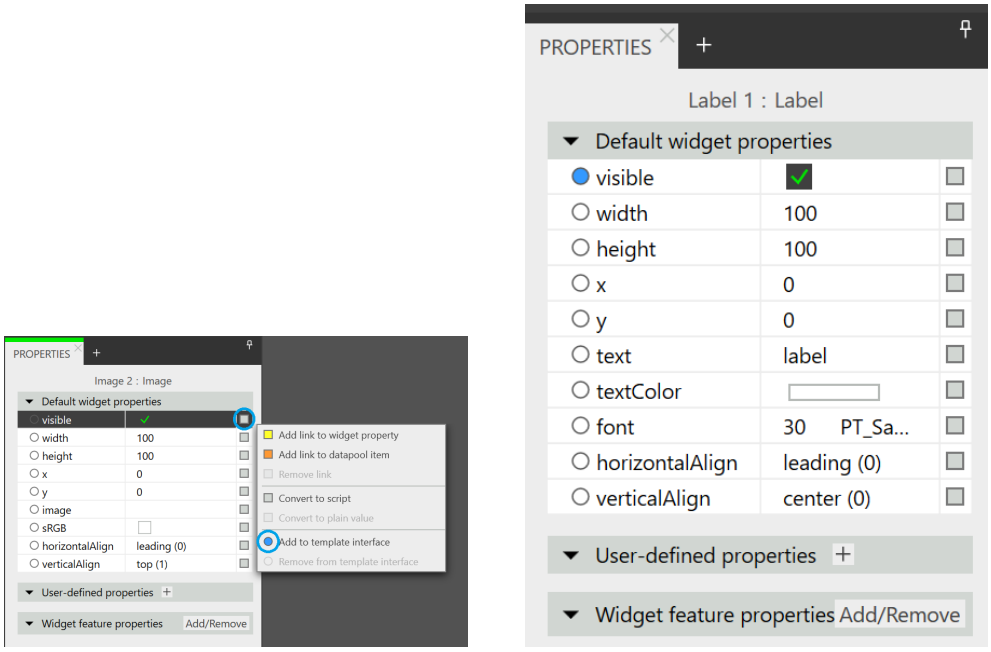
In Abb. 3.3 ist zu sehen wie die Funktion puplish to template interface ausgeführt wird. Hierfür ist zuerst ein Rechtsklick auf das Viereck hinter dem gewünschten Propertie nötig, bevor noch auf Add to template interface geklickt werden muss.

Während der Beobachtung wird deutlich, das vor allem der Rechtsklick den Arbeitsablauf sehr beeinflusst, da dieser in EB Guide nicht häufig verwendet wird. Zum Gropßteil wird mit dem normalen Linksklick gearbeitet, weshalb das auch für diesen Fall wünschenswert wäre In Teil b) von Abb. 3.3 sieht man, das das verlinkte Properties nun einen blauen statt einem weißen Kreis aufweist. Alle anderen Optionen die durch den Rechtsklick geboten werden spiegeln sich Nach Anwendung farblich im Quadrat wieder und lassen den Kreis unberührt. Es wäre also denkbar die Funktion puplish to template interface einfach durch einen Linksklick auf den Kreis zu aktivieren.

Resultierende Benutzeranforderung Nutzer, die für ein Propertie dir Funktion publish to template interface ausführen möchten,müssen dies auf eine intuitive und direkte Art und Weise tun können.

Widget Feature Properties Neben den Default Widget Properties wie Breite und Höhe die jede Widgetinstanz besitzt, existieren noch die so genannten Widget Feature Properties. Diese Features fügen mehr anpassbare Funktionen für das Aussehen und das Verhalten der Widgets bereit. Wie in Abb. 3.4 zu sehen sind die Features in Kategorien unterteilt, die grafisch in Form eines Dropdownmenüs dargestellt sind.

Bei der Beobachtung der Nutzer fällt auf das viele Nutzer genau wissen welches Feature sie hinzufügen wollen, jedoch häufig die zugehörige Kategorie nicht präsent haben. Dies führt



(a) (b)

Abbildung 3.3.: Usability - Schwäche Template Properties

dazu das jedes Dropdownmenü aufklappt wird, bis das gewünschte Feature gefunden wurde. Dieser Problematik könnte leicht mit einer Filtermöglichkeit entgegengewirkt werden.

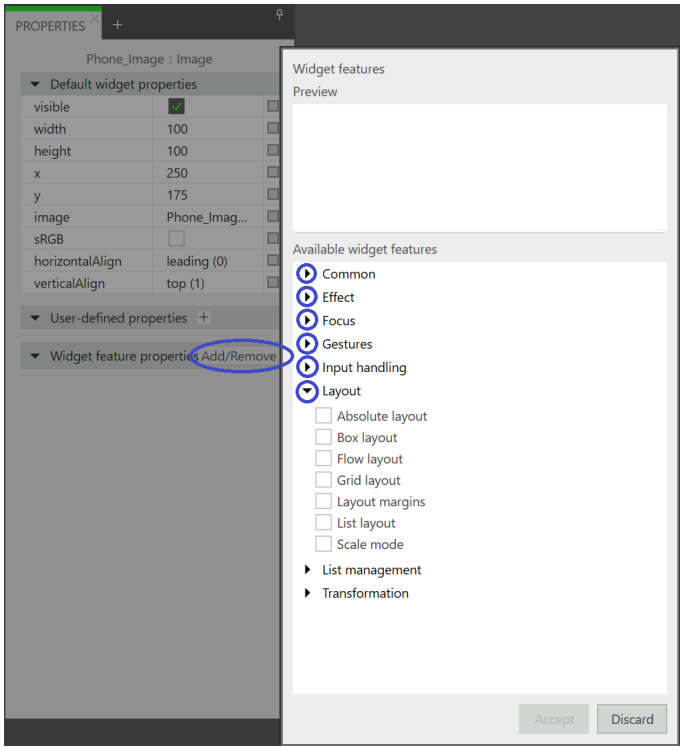


Abbildung 3.4.: Usability - Schwäche Widget Feature Properties

Resultierende Benutzeranforderung Nutzer, die den Namen eine gewünschten Features wissen, müssen die Möglichkeit haben dieses Feature aus allen vorhanden herauszufiltern.

Mehrfachselektion Falls bei mehreren Objekten ein Property auf den gleichen Wert angepasst werden muss, ist es naheliegend für den Nutzer dies durch die zeitgleiche Selektion der betroffenen Elemente zu lösen. Aktuell bietet EB Guide jedoch noch keine Unterstützung für Multiselektion, sind zwei Elemente ausgewählt sind diese lediglich gemeinsam mit der Maus bewegbar der View sieht wie in Abb. 3.5 zu sehen aus.

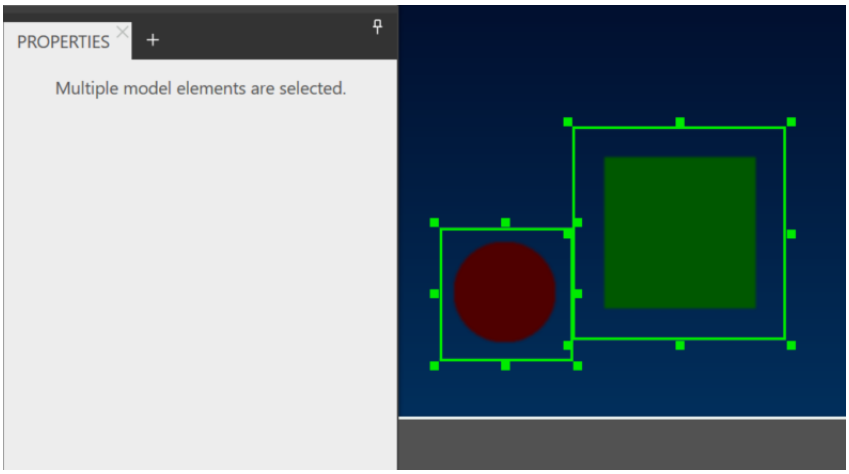


Abbildung 3.5.: Usability - Schwäche Mehrfachselektion

Resultierende Benutzeranforderung Nutzer, die mehrere Objekte gleichzeitig selektiert haben, müssen die Möglichkeit haben all deren Properties nach ihren Wünschen zu verändern.

3.2. Verbesserungen

Da es Aufgrund der zeitlichen Einschränkungen dieser Arbeit nicht möglich ist alle aufgeführten Schwächen weiter zu analysieren wird im Folgenden erläutert auf welchen Grundlagen die Auswahl für die Verbesserungen getroffen wurde, welcher Gewinn für den Nutzer sich durch diese Verbesserungen versprochen wird und welches Design für die Verbesserungen angestrebt wird.

3.2.1. Auswahlkriterien

Ausgangspunkt für weiterführende Untersuchungen

3.2.2. Gewinn für den Nutzer

Effizienzsteigerung intuitiver Bedienung

3.2.3. Design der Verbesserungen

Template Properties

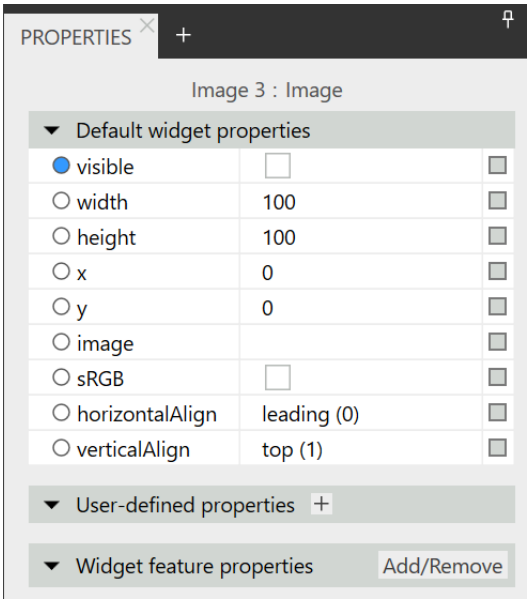


Abbildung 3.6.: Verbesserung Template Properties

Widget Feature Properties

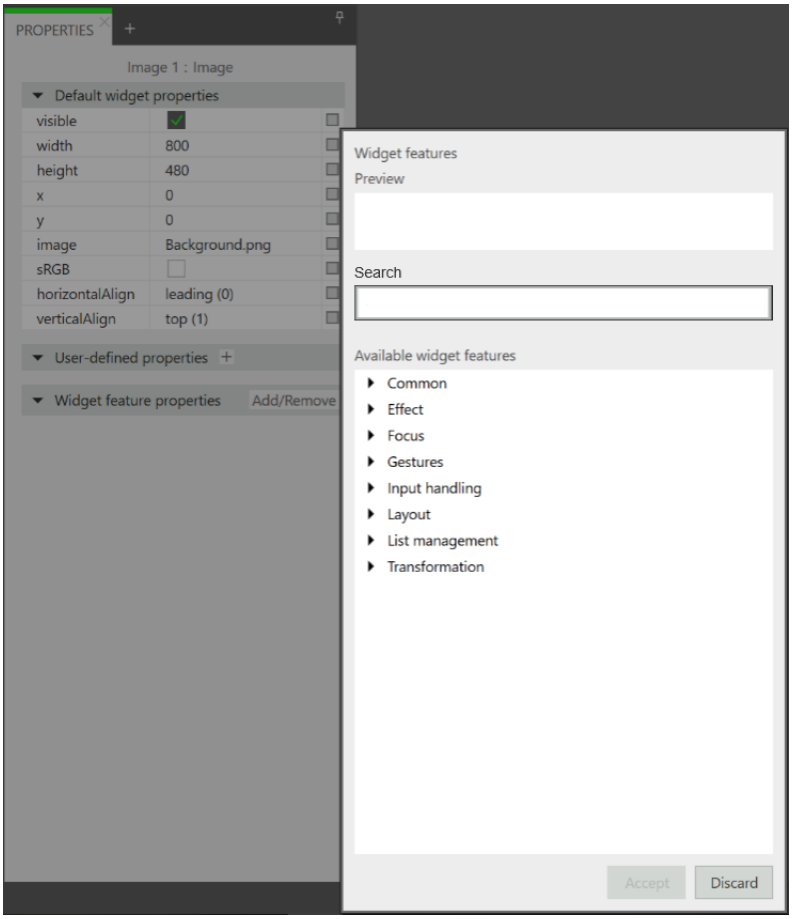


Abbildung 3.7.: Verbesserung Feature Property Properties

Mehrfachselektion

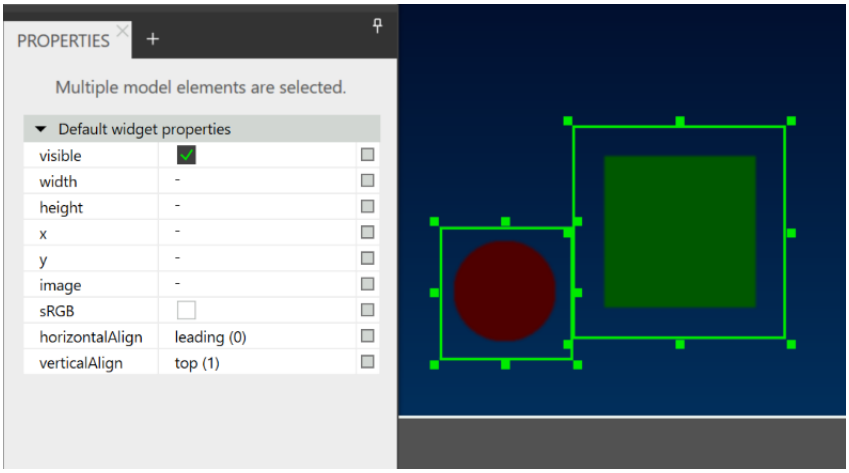


Abbildung 3.8.: Verbesserung Mehrfachselektion

Kapitel 4.

Umsetzung

4.1. Prototyp Multiselektion

4.1.1. Zielsetzung

4.1.2. Axure RP

4.1.3. Interaktionsmöglichkeiten

4.1.4. Vorgehensweise

4.2. Implementierung Filter

4.2.1. Zielsetzung

4.2.2. Projektaufbau

4.2.3. Vorgehensweise

Kapitel 5.

Usability - Test

5.1. Lookback

5.2. Remote Usability - Test

5.3. Arbeitsaufgaben

5.4. Ergebnisse altes Interface

5.5. Ergebnisse überarbeitetes Interface

5.6. Vergleich

Kapitel 6.

Fazit

6.1. Ergebnisse

6.2. Ausblick

Anhang A.

Supplemental Information

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Das hier ist der zweite Absatz. Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Und nun folgt – ob man es glaubt oder nicht – der dritte Absatz. Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Nach diesem vierten Absatz beginnen wir eine neue Zählung. Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Abbildungsverzeichnis

2.1. Zusammenhang Usability und User Experience (nach [Saro 16]) 7
2.2. Human - Centered - Design - Process bei Elektrobit 8
2.3. Aufbau EB GUIDE 14
2.4. EB Guide Studio Statemachine 15
2.5. EB Guide VIEW 16

3.1. Usability - Schwäche Image List 20
3.2. Usability - Schwäche Navigationt 21
3.3. Usability - Schwäche Template Properties 23
3.4. Usability - Schwäche Widget Feature Properties 23
3.5. Usability - Schwäche Mehrfachselektion 24
3.6. Verbesserung Template Properties 25
3.7. Verbesserung Feature Property Properties 26
3.8. Verbesserung Mehrfachselektion 26

Tabellenverzeichnis

List of Listings

Literaturverzeichnis

[DIN] "DIN EN ISO 9241-210 Ergonomie der Mensch-System-Interaktion - Teil 210: Prozess zur Gestaltung gebrauchstauglicher interaktiver Systeme (ISO 9241-210:2010)".

[Eleka] Elektrobit Automotive GmbH. "About Elektrobit (EB) - Elektrobit".

[Elekb] Elektrobit Automotive GmbH. "EB GUIDE Home".

[Elekc] Elektrobit Automotive GmbH. "EB GUIDE Studio - User guide".

[Elekd] Elektrobit Automotive GmbH. "User Experience Engineering".

[Eleke] Elektrobit Automotive GmbH. "User requirements".

[Ergo 14] "Ergonomie - was ist das eigentlich? Definitionen und Forschungsrichtungen". 2014.

[Huma] "Human Machine Interaction".

[Knig 19] W. Knight. *UX for Developers*. Apress, Berkeley, CA, 2019.

[Lidw 10] W. Lidwell, K. Holden, and J. Butler. *Universal principles of design: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design / William Lidwell, Kritina Holden, Jill Butler*. Rockport, Beverly, Mass., [rev. and updated] ed. Ed., 2010.

[Niel 95] J. Nielsen. *Usability engineering*. AP Professional, Boston and London, [new ed.] Ed., 1995?

[Norm 16] D. A. Norman. *The design of everyday things*. Verlag Franz Vahlen, München, überarbeitete und erweiterte auflage Ed., 2016.

[Rich 16] M. Richter. *Usability und UX kompakt [electronic resource]: Produkte für Menschen. IT kompakt*, Springer Vieweg, Berlin, 4. auflage Ed., 2016.

[Saro 16] F. Sarodnick and H. Brau. *Methoden der Usability Evalution: Wissenschaftliche Grundlagen und praktische Anwendung*. Hogrefe, Bern, 3., unveränderte auflage Ed., 2016.