Entwurf und Implementierung eines Android User Interfaces am Beispiel des Hades Simulation Framework

Technische Aspekte Multimodaler Systeme
Fachbereich Informatik
Universität Hamburg

1 November 2014

Author:	Johann Basnakowski
Matrikelnummer:	6316305
Studiengang:	Software-System-Entwicklung

Erstbetreuer:	Dr. Norman Hendrich
Fakultät:	Mathematik, Informatik und Naturwissenschaften
Fachbereich:	Informatik
Arbeitsgruppe:	TAMS

Zweitbetreuer::	Prof. Dr. Christopher Habel
Fakultät:	Mathematik, Informatik und Naturwissenschaften
Fachbereich:	Informatik
Arbeitsgruppe:	WSV

Inhaltsverzeichnis

1 Einleitung

Wir leben in einem mobilen Zeitalter. Immer mehr Menschen greifen zu mobilen Endgeräten um ins Internet zu gehen, E-Mail's zu schreiben oder Spiele zu spielen. Damit lösen mobile Endgeräte wie das Smartphone oder der Tablet Computer den traditionellen Personal Computer immer mehr ab. Eine Studie der Tomorrow Focus Media vom Februar 2014 zeigt auf, dass sich die Anzahl der Tablet Benutzer innerhalb eines Jahres um 100% gesteigert hat [Abbildung: 1]. Aus dieser Entwicklung ergibt sich eine erhöhte Nachfrage nach entsprechenden Tools für den Tablet Computer, kurz "App" genannt, welche bis dahin nur für den Personal Computer existierten. Diese Bachelorarbeit beschäftigt sich mit der Entwicklung eines solchen "App's" an dem Beispiel des "Hades Simulation Frameworks".

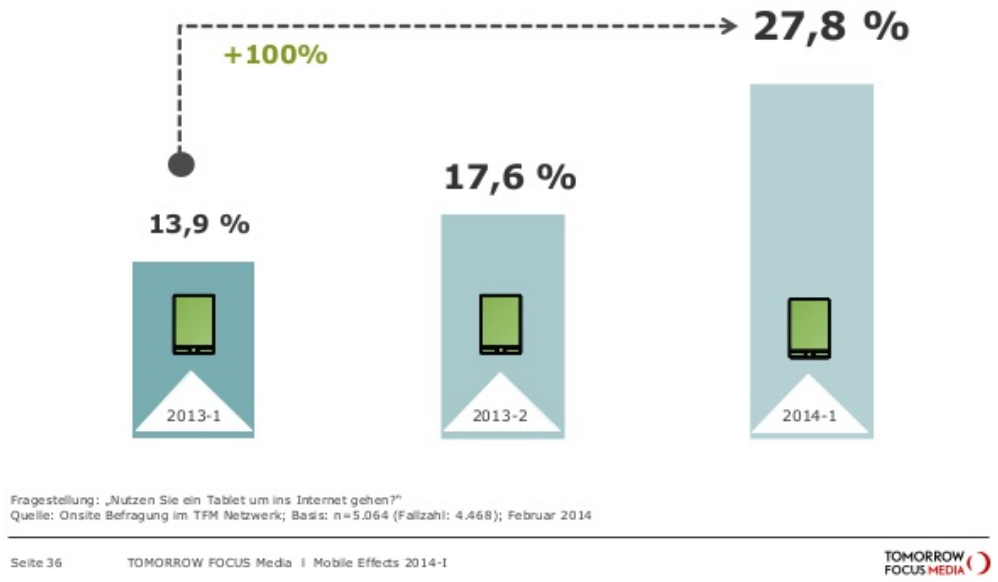

Abbildung 1: Zunahme der Tabletnutzung in Deutschland von 2013 bis 2014, Quelle: [1]

2 Zielsetzung

Das Ziel dieser Bachelorarbeit ist die Entwicklung und Implementierung einer Android Version des Hades Simulation Framework. Hierbei wird das Hauptaugenmerk auf die Entwicklung der Benutzeroberfläche gelegt. Da es sich bei dem Hades Simulation Framework um eine Vielzahl von verschiedenen Werkzeugen handelt, behandelt diese Arbeit nur den Aspekt der Erstellung von digatalen Schaltungen mittels einfacher Schaltelemente und deren Simulation. Funktionalitäten wie der "Waveform Viewer" oder des "Component Browsers" werden hier nicht weiter beleuchtet.

3 Grundlagen

3.1 Hades Simulation Framework

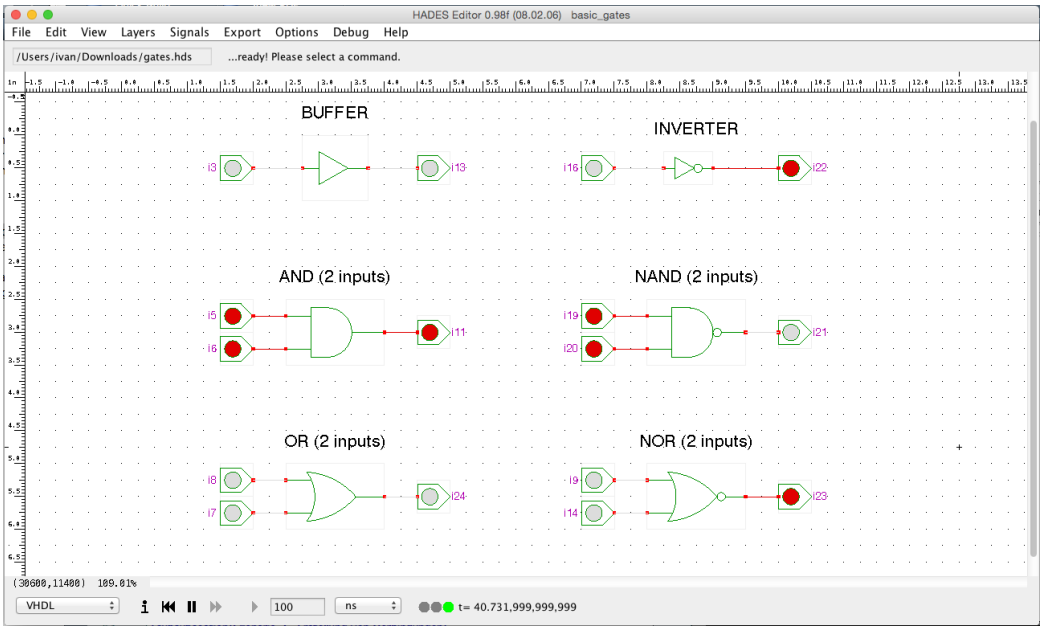

Abbildung 2: Eine geöffnete Schaltung in Hades

Hades is a framework for object-oriented event-based simulation.[1]

Das Hades Simulation Framework [Abbildung: 2] ist eine Ansammlung von Software Werkzeugen zur Entwicklung und Simulation von digitalen Schaltungen. Es setzt hierbei auf ein Event basiertes Simulationssystem, was im Gegensatz zu Konkurrenz Produkten wie Diglog, die interaktive Bedienung erlaubt. Dies bedeutet, dass die digitale Schaltung zur Simulationszeit verändert werden kann und die Wirkung sofort eintritt, es ist kein separater Übersetzungsprozess nötig. Damit eignet sich Hades besonders gut als Lehrmittel für Studenten, um verschiedenste Schaltungen aufzuzeigen und um den Studenten eine praktische Umgebung zu geben, das Gelernte zu verdeutlichen.

[1] Zitat aus Quelle: [14]

Hades Funktionen

Hades bringt eine Vielzahl an Funktionen mit sich, welche es dem Benutzer erlauben einfache Schaltungen zu erstellen und zu simulieren. Hades bringt neben den normalen Funktionen, welche von dem Erzeugen von Elementen, bis hin zur Verdrahtung und der Manipulation von Elementen reichen, noch einige zusätzliche Funktionen mit sich. Im nachfolgenden ist eine Auflistung von sämtlichen Hades Funktionen, mit welchen sich diese Arbeit beschäftigen wird.

- Erzeugung von Schaltelementen
- Verschieben von Schaltelementen
- Drehen von Schaltelementen
- Löschen von Schaltelementen
- Änderungen der Eigenschaften von Schaltelementen (Name, Verzögerungszeit, Farbe von LED, etc.)
- Verbindungen zwischen Schaltelementen erzeugen
- Rein - und Hinaus zoomen innerhalb der Schaltung
- Das Verschieben der Arbeitsfläche
- Das Starten,Pausieren und das Zurücksetzen der Simulationsumgebung

Es sei hierbei angemerkt, das Hades über die Möglichkeit verfügt, hierarchische Schaltungen zu laden. Die Android Version von Hades unterstützt dieses Funktion leider nicht.

3.2 Tablet Computer

Ein Tablet Computer ist ein tragbarer und flacher Computer, welcher keine Maus oder Tastatur als Bedienelement aufweist, stattdessen wird ein Touchscreen als Ersatz verwendet. Aufgrund der enormen Ähnlichkeit mit modernen Smartphones, sowohl was die Bedienung, Design als auch den Leistungsumfang betrifft, verwenden Tablet Computer mobile Betriebssysteme, welche ursprünglich für Smartphones entwickelt wurden.

3.2.1 Eigenschaften

Tablet Computer gibt es in den verschiedensten Größen und Ausstattungen. Die meisten haben eine Bildschirmdiagonale von 7 bis zu 10.5 Zoll, wiegen zwischen 465 und 290 Gramm [2] und verfügen über die selben zusätzlichen Funktionalitäten wie moderne Smartphones, wie z.B über einen Beschleunigungsmesser und GPS. Diese Eigenschaften sind es, welche Tablet Computer immer attraktiver für den Endbenutzer machen.

3.2.2 Gesten

Anders als bei herkömmlichen Computern, läuft die Bedienung bei einem Tablet über den TouchScreen ab. Dies bedeutet, dass das Tablet/Smartphone mit dem Finger bedient wird. Hierfür wurden einige Gesten eingeführt, um die Bedienung zu erleichtern. Die Hauptgesten zur Steuerung lauten:

- Single Tap
- Double Tap
- Long Press
- Pinch and Zoom
- Drag and Drop

Single Tap

Abbildung 3: Single Tap Geste, Quelle: [6]

Die Single Tap Geste entspricht im Kontext von Smartphones und Tablet etwa dem Linken Mausklick in einer PC Umgebung. Es handelt sich hierbei um eine simple Berührung des Touchscreen und darauffolgendes loslassen an der ansprechende Stelle. Es ist die Hauptnavigationsgeste auf modernen mobilen Endgeräten.

Double Tap

Abbildung 4: Double Tap Geste, Quelle: [6]

Bei der Double Tap Geste handelt es sich, wie der Name bereits vermuten lässt um die doppelte Ausführung der Single Tap Geste. Dies entspricht etwa dem Rechten Mausklick in einer PC Umgebung.

Long Press

Abbildung 5: Long Press Geste, Quelle: [6]

Die Long Press Geste ist eine eigenständige Smartphone/Tablet Geste. Es handelt sich hierbei um ein langes Klicken mit der Finger. Sie besitzt

kein genaues äquivalent in der PC Umgebung und wird auch in anderen Applikationen auf die verschiedenste Art und Weise verwendet.

Pinch and Zoom

Abbildung 6: Pinch and Zoom, Quelle: [6]

Bei Pinch and Zoom handelt es sich um eine Multitouch Geste. Dies bedeutet das mehr als nur 1 Finger benötigt wird, um die Geste auszuführen. Diese Geste wird ausschließlich für das Hinein-/Heraus zoomen innerhalb einer Applikation verwendet.

Drag and Drop

Abbildung 7: Drag and Drag Geste, Quelle: [6]

Drag and Drop ist wie Pinch and Zoom eine Multitouch Geste. Diese ist äquivalent zu der Drag and Drop Operation mit der Maus innerhalb einer PC Umgebung.

3.3 Android

Bei Android handelt es sich um eine Software-Platform und Betriebssystem für mobile Endgeräte wie Smartphones, Netbooks und Tablet Computer. Es wird von der Open Handset Alliance (von Google gegründet) als Open-Source Projekt entwickelt und ist mit einem weltweiten Marktanteil von 84,6% [[5]], das am weitesten verbreitete mobile Betriebssystem in der heutigen Zeit. Zur Zeit der Erstellung dieser Arbeit befindet sich Android in der Version 4.4. Diese Arbeit bezieht sich daher nur auf die Versionen 4.0 bis zu 4.4.

Google hat zur Entwicklungshilfe für Android Applikationen, die Android SDK bereit gestellt. Hierbei handelt es sich um eine Ansammlung von Applikationen und Klassen zur Analyse und Entwicklung von Android Applikationen. Applikationen für das Android System werden in der Programmiersprache Java geschrieben.

4 Benutzeroberfläche von Hades

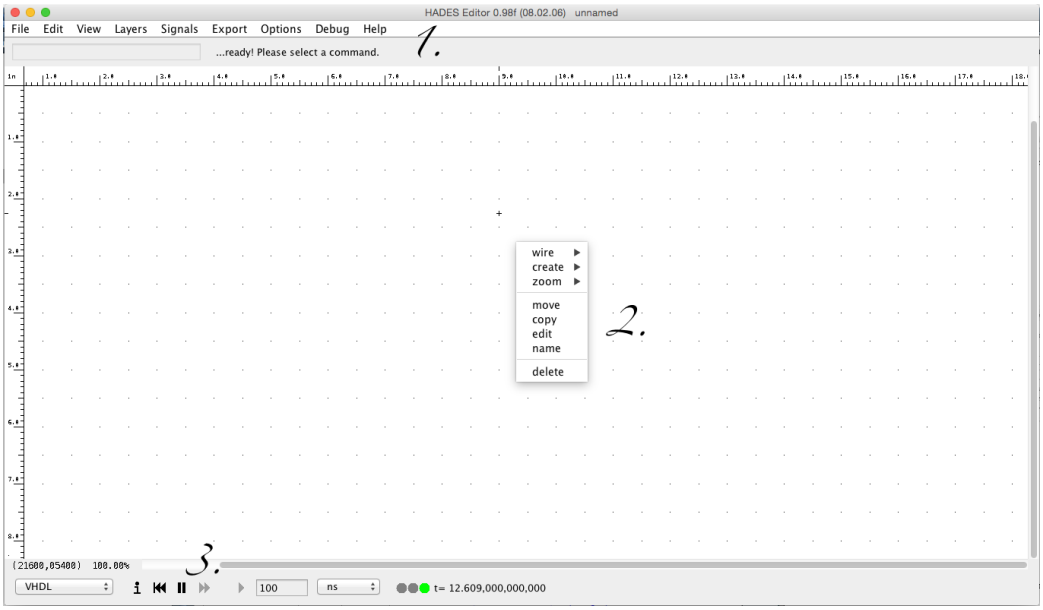

Abbildung 8: Hauptfenster von Hades mit geöffnetem Kontektmenü

Hades obliegt in seiner Bedienung einer Dreiteilung . Punkt 1. zeigt das Navigationsmenü, hier finden sich hauptsächlich Funktionen der Dateiverwaltung, des "Component Browser" und des "Waveform Viewer".

Punkt 2. zeigt das Kontextmenü auf. Das Kontextmenü ist das Hauptnavigation Werkzeug für Hades, wenn es darum geht, Schaltungen zu bauen. Über dieses Menü lassen sich neue Schaltelemente erzeugen, die Eigenschaften bestehender Schaltelemente ändern, Verbindungen zwischen Schaltelementen erzeugen und Schaltelemente löschen. Weiterhin lässt sich über diese Menü innerhalb der Schaltung rein- oder raus-zoomen.

Punkt 3. ist die Schnittstelle zur Steuerung des Simulators. Hierüber lässt sich der Simulator starten, pausieren oder ganz zurücksetzen.

4.1 Benutzerszenarien

Da es sich bei dem Kontextmenü, für die Erstellung von Schaltungen, um die wichtigste Navigationsschnittstelle handelt, soll diese nun noch einmal genauer im Detail beleuchtet werden.

Szenario 1 - Erstellung von Schaltelementen

Eine der häufigsten Benutzerszenarien ist die Erstellung von Schaltelementen. In Hades geschieht dies indem mittels eines Rechtsklick auf die Hauptfläche das Kontextmenü geöffnet wird. Hierüber kann dann über einzelne Kategorien ein Schaltelement ausgewählt werden [Punkt 1.] und über einen linken Mausklick, an der Mausposition, positioniert werden. [Punkt 2.]

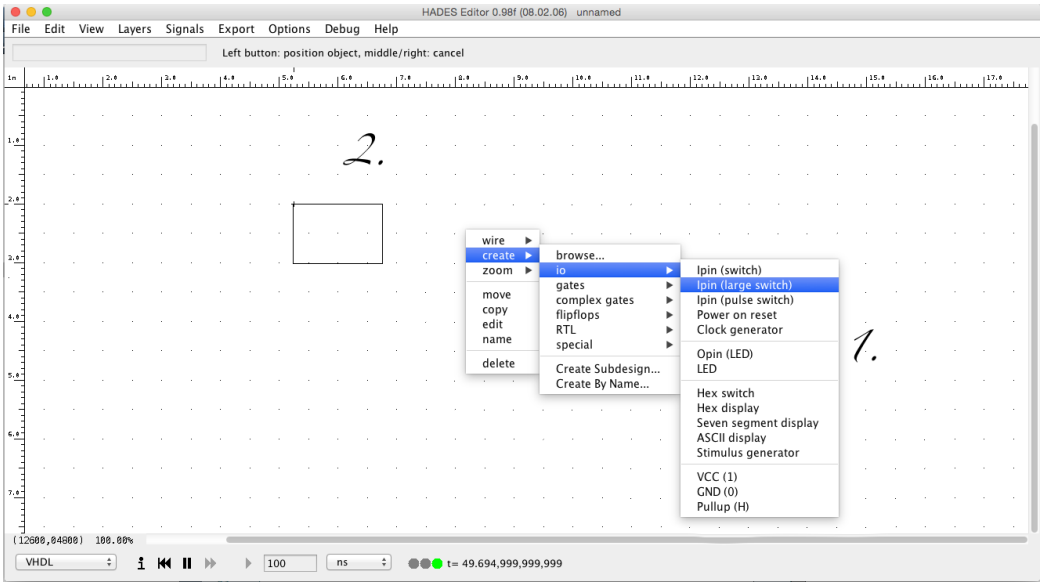

Abbildung 9: Hades Kontektmenü + Schaltelement Positionierung

Szenario 2 - Erstellung von Verbindungen

Verbindungen werden in Hades über das Raster erzeugt. Es muss per linkem Mausklick auf einen der Ports geklickt werden und dann jeweils auf die einzelnen Gitterpunkte, über die die Verbindung laufen soll, bis hin zum Endport.

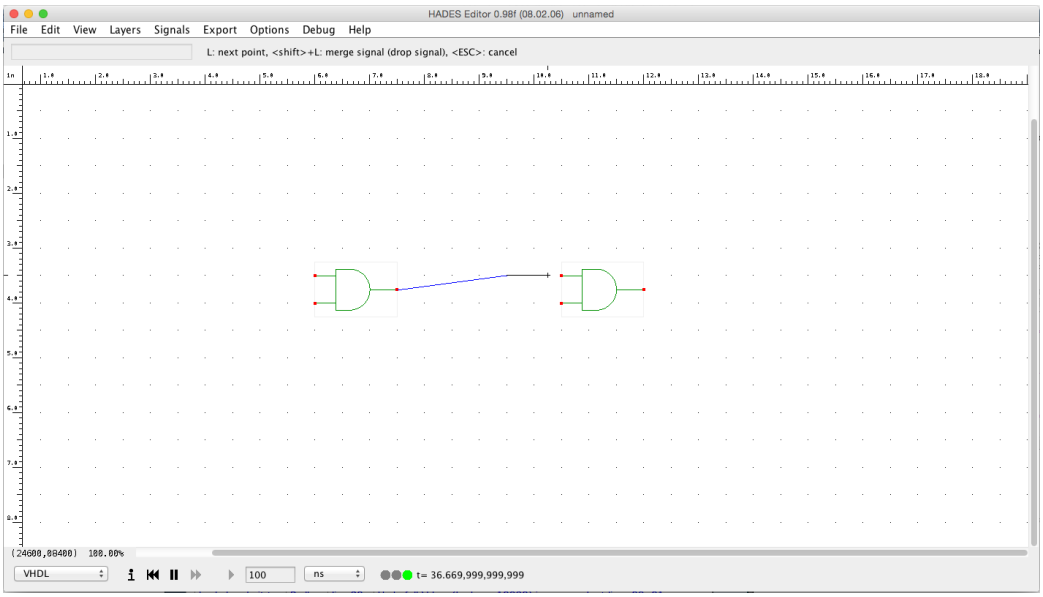

Abbildung 10: Erstellung von Verbindungen in Hades

Szenario 3 - Manipulation von Elementen

Die Manipulation von Schaltelementen erfolgt über das öffnen des Kontextmenü, an einem bestehendem Schaltelement, und das Auswählen der “edit” Option. Über das sich nun öffnenden Eigenschaften Fenster lassen sich die Eigenschaften einzelner Elemente manipulieren.

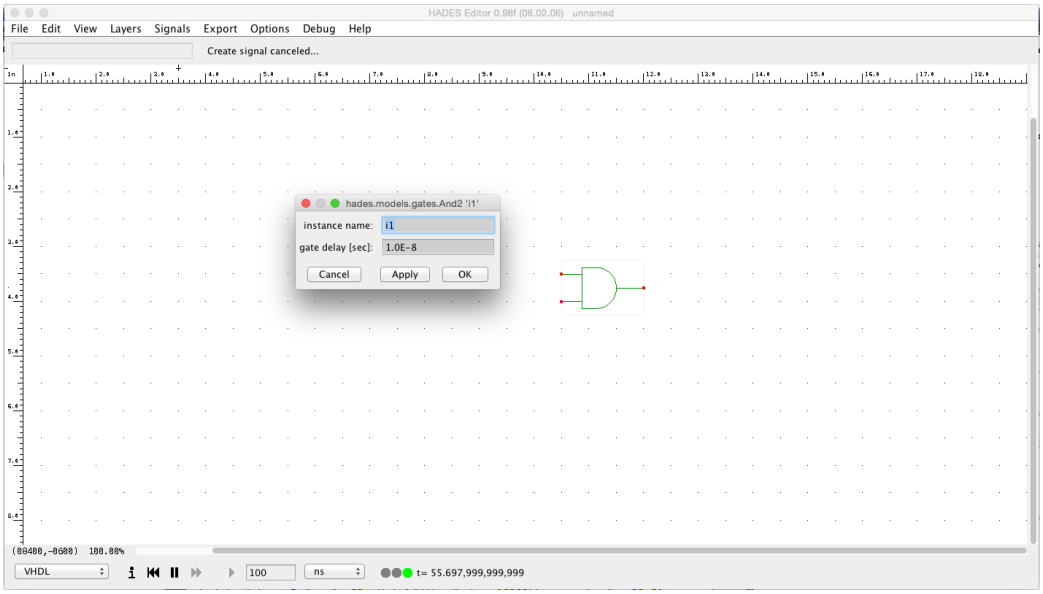

Abbildung 11: Eigenschaftenfenster eines Schaltelements in Hades

5 Problemstellung

Nachdem nun im Detail erläutert wurde, wie die Benutzeroberfläche von Hades funktioniert und welche Eigenschaften Tablets aufweisen, als auch welche Bedienmöglichkeiten ein Tablet aufweist, so ergeben sich einige Probleme, welche die Benutzung von Hades auf Android erschweren.

Platzmangel

Es ist nicht genug Platz vorhanden. Betrachtet man das erste Benutzerszenario, so ergibt sich, dass alle Schaltelemente als eine Liste innerhalb des Kontextmenüs liegen. Bei zu vielen verschiedenen Schaltelementen wird das Kontextmenü zu groß. An einem herkömmlichen PC, mit den heutigen 22-30 Zoll Monitoren mag dies kein großes Problem sein, bei einem Tablet mit, im schlimmsten Fall 7 Zoll Bildschirmdiagonalen, ist dies problematisch. Das Runter-Skalieren des Kontextmenüs ist keine wirkliche Option, wie im Punkt [Feinmotorische Aufgaben] zu sehen sein wird.

Auch große Schaltungen lassen sich auf diese Art und Weise schlecht bedienen. Bei einer großen Schaltung mit zwei Schaltern auf der linken Seite und zwei Ausgabeleuchten auf der rechten Seite, wird bei der Bedienung, immer nur eine der beiden Seiten sichtbar sein. Es ist damit schwer Schalter umzulegen und gleichzeitig deren Wirkung zu beobachten, ohne dem Benutzer unnötige Navigationsgesten aufzudrücken. Das Runter-Skalieren ist auch hier keine gute Lösung wie das nachfolgende Problem aufzeigt.

Feinmotorische Aufgaben

Im Benutzerszenario 2 wurde die Erstellung von Verbindungen erklärt. Das Anklicken von Ports ist mittels eines Cursors einfach durchzuführen, da der Durchmesser eines PC Cursors etwa 5mm beträgt, der menschliche Zeigefinger hingegen hat einen durchschnittlichen Durchmesser von etwa 16 bis 20mm [3]. Dies erschwert die Bedienung der Applikation enorm, da es für den Benutzer nur sehr schwer möglich sein wird, Schaltelemente auszuwählen oder diese zu verschieben. Dies setzt eine gewisse Mindestgröße für jeden Port, jeden Knopf, jedes Element voraus, unabhängig davon, ob es sich um einen Teil der Schaltung oder nur um einfache Knöpfe innerhalb der Navigation handelt.

Es ist somit klar, dass sich das Kontextmenü nicht so einfach übernehmen lässt. Versuche eine andere Art Menü zu finden, wo das jetzige Kontextmenü seine Struktur behält, führen immer wieder zum selben Problem. Die einzige effektive Möglichkeit dieses Problem zu lösen, ist das Kontextmenü aufzuteilen und um dies zu erreichen gibt es mehrere Ansätze, der nachfolgende Designentwurf ist einer dieser Ansätze.

6 Design Entwurf

Das Kapitel [Problemstellung] hat bereits angedeutet, dass die Lösung der Probleme in der Aufteilung des Kontextmenüs liegt. [Abbildung: 12] zeigt eine mögliche Aufteilung auf. Die Funktionen des Kontextmenüs werden hierbei in folgende drei Punkte aufgeteilt:

- Die Erstellung und Manipulation von Schaltelementen
- Das Verbinden von Schaltelementen
- Das Simulieren und Testen der Schaltung

Es handelt sich bei diesen Operationen um separate Prozesse. Dies erlaubt es separate Menüs für jeden dieser Modi zu erstellen, ohne das Abhängigkeiten entstehen.

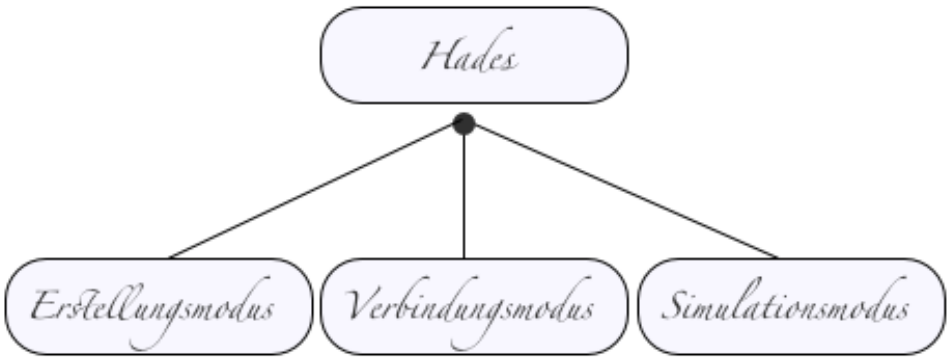

Abbildung 12: Aufteilung von Hades in 3 Modi

Platzmangel

Das Problem des Platzmangel wird gelöst indem die Applikation in drei Modi aufgeteilt wird. Auf diese Weise können mehrere Gesten mehrfach benutzt werden, da eine Double Tap Geste, abhängig davon in welchem Modus sich die Applikation befindet, unterschiedliche Aktionen ausführen kann. Dieser Ansatz ermöglicht es einzelne Menü Punkte auf simple Gesten zu reduzieren und erlaubt es das Kontextmenü in seiner alten Form komplett aus der Applikation zu streichen. Der zweite Fall beim Platzmangel mit großen Schaltungen und deren Simulation lässt sich lösen, indem separate "Übersichtsleistenängelegt werden, welche nur im Simulationsmodus sichtbar sind. In diese können Eingänge/Ausgänge hinzugefügt werden. Über diese lassen sich dann im Laufe der Simulation Schalter betätigen, Zuständen bearbeiten und beobachten ohne direkt Sicht auf den Bereich in der Schaltung zu haben.

Feinmotorische Aufgaben

Das Problem der Feinmotorischen Aufgaben wird auf drei unterschiedliche Arten gelöst. Zum einen werden alle Elemente innerhalb der Applikation eine Mindestgröße von 48DP[2] erhalten. Dies entspricht der von Google empfohlenden Symbolgröße und garantiert eine angenehme Bedienung.

> On average, 48dp translate to a physical size of about 9mm (with some variability). This is comfortably in the range of recommended target sizes (7-10 mm) for touchscreen objects and users will be able to reliably and accurately target them with their fingers. [3]

Zum anderen muss der Benutzer die einzelnen Verbindungen nicht im Detail erzeugen. Der Benutzer wird nur die zwei zu verbindende Schaltelemente auswählen und das System selbst, wird einen Pfad suchen und die Elemente verbinden. Damit ist kein klicken auf Ports mehr nötig. Und der letzte Punkt ist die Möglichkeit der Zustandsänderung der Schaltelemente über die Übersichtsleisten im Simulationsmodus. Somit existiert innerhalb der Applikation keine Feinmotorische Aufgabe mehr.

[2]Display Independent Pixel : Eine Metrische Einheit bei Android, welche vom System, abhängig von der Displaygröße, auf Pixel umgerechnet wird

[3]Zitat aus Quelle : [7]

6.1 Hauptfenster

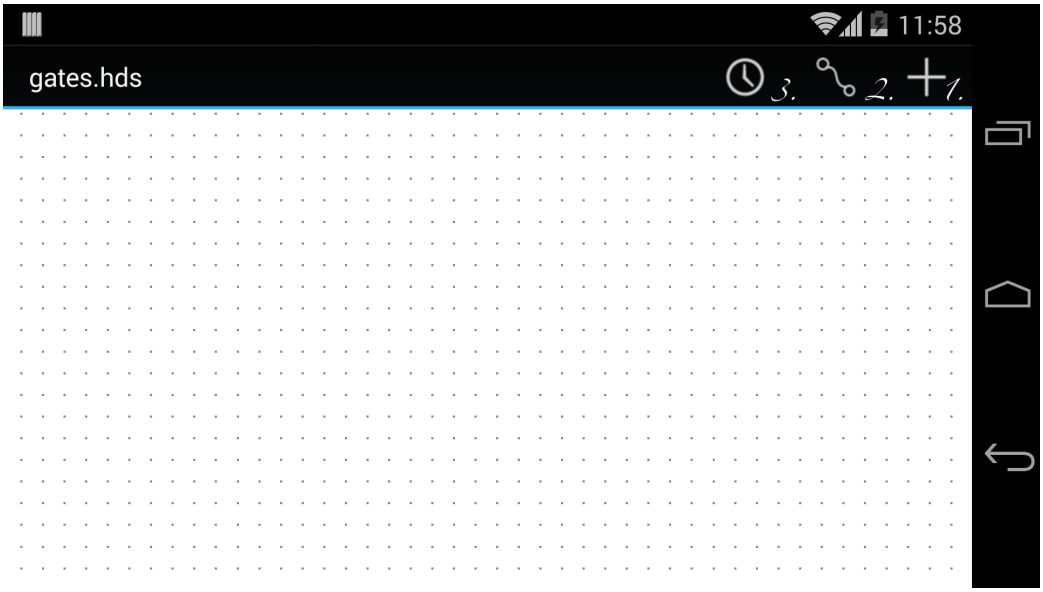

Abbildung 13: Hauptfenster in der Android Version von Hades

Auf [Abbildung : 13] ist das Hauptfenster der Android Version von Hades abgebildet. Oben Rechts in der Action Bar befindet sich die drei Buttons, welche die Applikation in den jeweiligen Modus versetzen. Punkt 1 öffnet hierbei den Erstellungsmodus, in dem können neue Schaltelemente angelegt und bearbeitet werden. Punkt 2 öffnet den Verbindungsmodus, in diesem könne Verbindungen zwischen Schaltelementen erzeugt werden. Punkt 3 öffnet den Simulationsmodus, über diesen lässt sich die Simulation steuern und einzelne Schalter bedienen.

Die Navigation innerhalb des Schaltplans ist intuitiv. Mittels simpler Schiebebewegungen lässt es sich innerhalb des Schaltplan umher navigieren. Über die Pinch and Zoom Geste kann innerhalb der Schaltung hinein- und heraus skaliert werden.

6.2 Erstellungsmodus

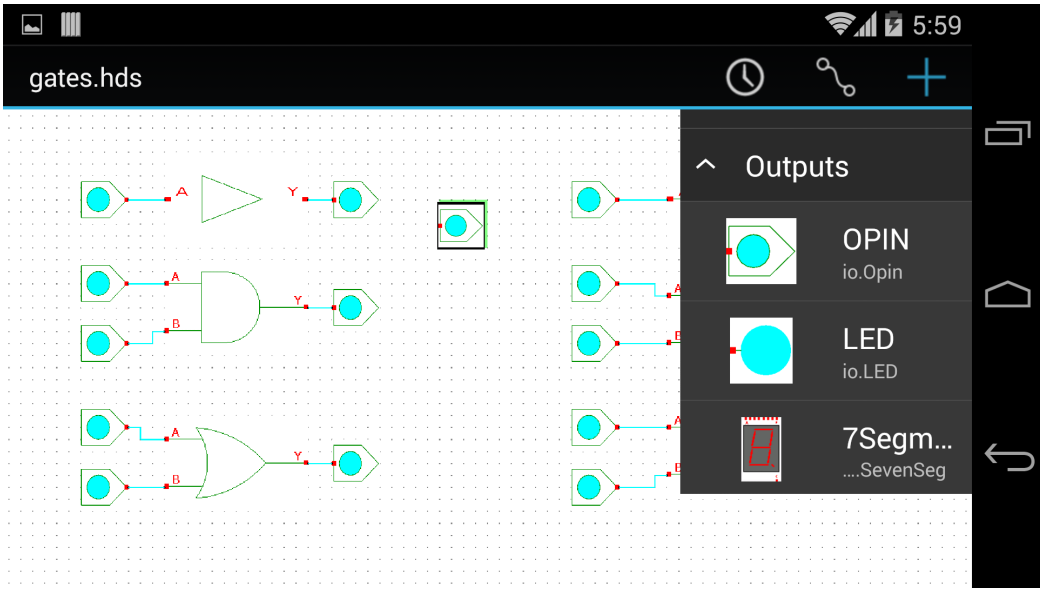

Abbildung 14: Erstellungsmodus in der Android Version von Hades

Dies ist der Modus zum Erstellen von Schaltelementen. Befindet man sich in diesem Modus, so ist am rechten Bildschirmrand eine Leiste aufgeklappt. Innerhalb dieser Leiste können über Kategorien Listen aufgeklappt werden, in welchen sich dann die entsprechenden Schaltelemente finden lassen. Durch die Long Press Geste auf eins der Schaltelemente poppt dieses aus der Leiste hervor und kann dann mittels der Drag and Drop Geste auf dem Schaltplan platziert werden. Der Schaltplan scrollt automatisch mit, falls während der Drag and Drop Geste das Element zu nahe an der Bildschirmrand kommt. Dies macht es sehr angenehm Elemente zu platzieren und ignoriert auf diese Weise die Platzlimitierung des Tablets, indem während der Geste mitnavigiert wird. Ein Double Tap auf ein Element öffnet die Eigenschaften des Schaltelements. Über diese lässt sich das Schaltelement löschen, die Eigenschaften verändern (Verzögerung, Name, etc), aber auch die Drehrichtung festlegen, als auch, ob das Schaltelement als Eingang/Ausgang zu den Simulationsleisten hinzugefügt werden soll.

6.3 Verbindungsmodus

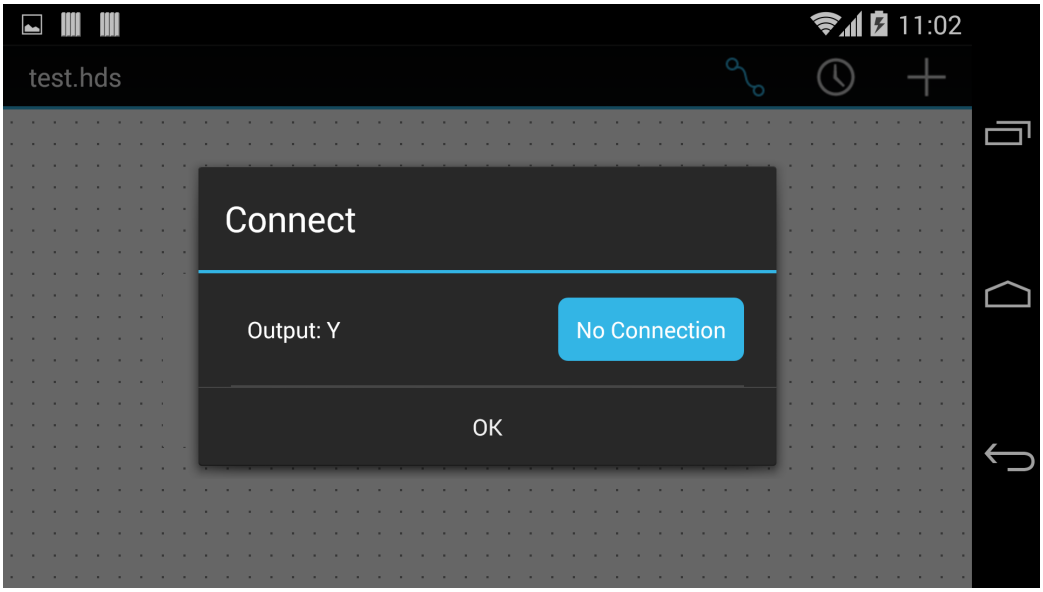

Abbildung 15: Port-Eigenschaften in der Android Version von Hades

Der Verbindungsmodus erlaubt es Schaltelemente zu verbinden. Dies geschieht indem mittels der Double Tap die Port-Eigenschaften geöffnet werden [Abbildung: 15] und mittels Single Tap auf die Blaue Fläche, an dem Port der Verbunden werden soll, geklickt wird. Das Schaltelement ist damit markiert und wird mit einer blauen Umrandung angezeigt. Wird der Schritt nun an einem 2. Schaltelement wiederholt, so erkennt die Applikation das eine Verbindung zwischen den beiden Schaltelementen erstellt werden soll und berechnet mittels des A*-Star Algorithmus eine Verbindung zwischen den beiden Schaltelementen und erzeugt diese.

6.4 Simulationmodus

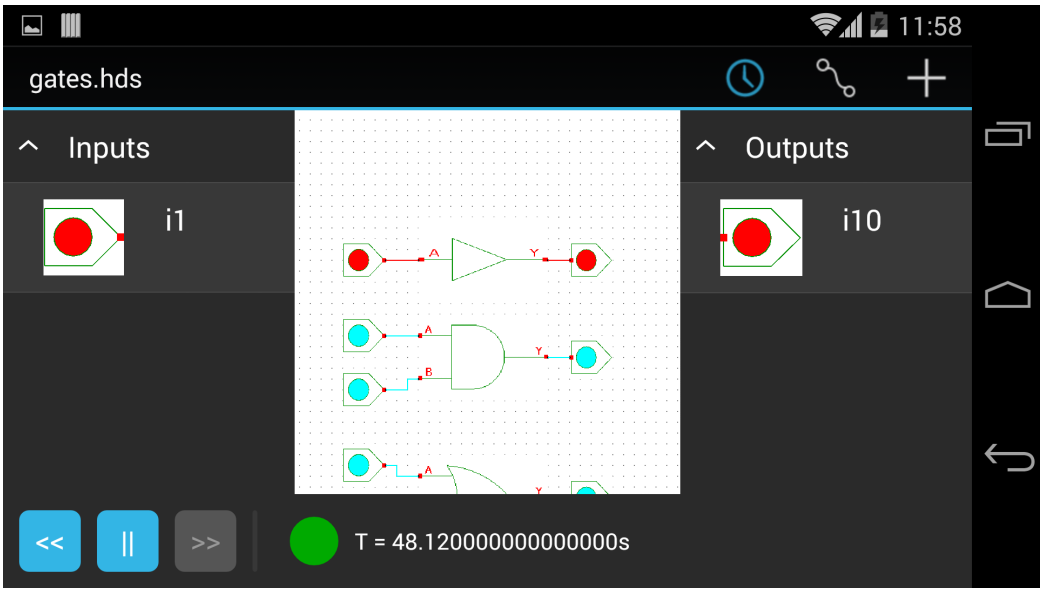

Abbildung 16: Simulationsmodus in der Android Version von Hades

Der Simulationsmodus bietet eine einfache und ideale Benutzererfahrung zur Simulation von bestehenden digitalen Schaltungen. Der Modus enthält zwei Leisten, eine für sämtliche Eingänge(Schalter, Clock Generator, etc) und eine für sämtliche Ausgaben(LED, Ausgangsport, etc) innerhalb der Schaltung. Schaltelemente lassen sich über das Eigenschaften Fenster zu den Leisten hinzufügen und über diese auch bedienen. Dies erlaubt es, bei Schaltungen welche aufgrund ihrer Größe nicht vollständig dargestellt werden können, die Zustände vollständig zu überblicken. Dieser Ansatz löst das zweite Problem des Platzmangels indem es mit den Leisten eine Übersicht/Funktionsschnittstelle bietet, da über die Leisten auch die Zustände der Eingänge ändern lassen. Unten links befinden sich die Steuerungsknöpfe des Simulators. Über diese lässt sich der Simulator starten, stoppen und zurücksetzen. Die Anzeige rechts daneben zeigt die aktuelle Simulationslaufzeit an.

6.5 Projektverwaltung

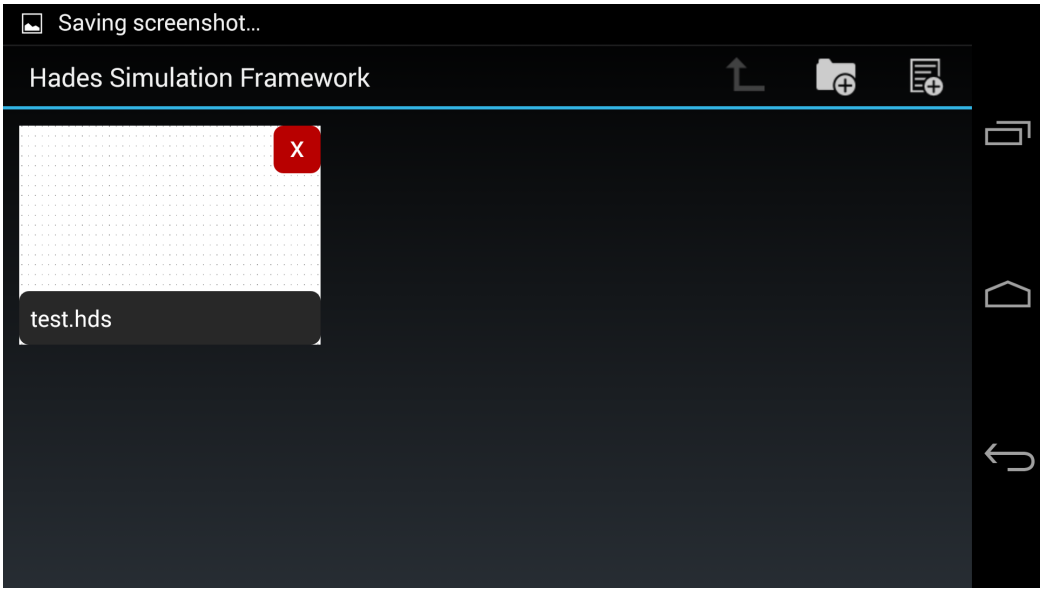

Abbildung 17: Dateiverwaltung in der Android Version von Hades

Dieser Modus erlaubt die Erstellung von neuen Projekten/Ordner auf dem mobilen Endgerät über die Buttons in der Actionbar Rechts oben. Projekte lassen sich über das Rote X oben Rechts an den Projekten wieder löschen. Um ein Projekt zu öffnen oder in einen Ordner hineinzuwechseln, genügt ein Single Tap. Das Pfeil Symbol erlaubt es, innerhalb eines Ordners, wieder aus diesem herauszuwechseln.

7 Aufbau von Android

Die Struktur einer jeder Android Applikation orientiert sich an dem MVC Mustter. Eine Android Applikation besteht auch einer oder mehreren Activities. Eine Activity ist ein Kompletter Bildschirm innerhalb einer Android Applikation. Sie entspricht der Steuerungsschicht innerhalb des MVC Muster.

Layouts werden in Android in XML Dateien definiert. Diese finden sich in dem Verzeichnis */res/layout*. Jedes Benutzerelement, welches innerhalb von Android existiert, erbt entweder von der Klasse View oder Viewgroup. Dabei handelt es sich um rechteckige Boxen, an welchen sich verschiedene Eigenschaften wie die Höhe, Breite, Innerer Abstand, Äußerer Abstand, etc. definieren lassen. Android Layouts werden aus diesen Elementen und deren Unterklassen zusammengesetzt. Ein fertiges Layout in XML Form entspricht der Präsentationsschicht im MVC Muster.

Die Modellschicht aus dem MVC Muster muss vom Entwickler selbst implementiert werden.

7.1 Android Manifest

Das Android Manifest ist eine globale Konfigurationsdatei innerhalb eines Android Projektes. In dieser wird der Name der Applikation, das Applikationbild, die Android Version und die Rechte definiert, welche die Applikation braucht. Außerdem wird in der Manifest Datei definiert, welche Activity beim Start der Applikation zuerst gestartet werden soll. Die Android Manifest Datei von der Android Version von Hades sieht folgendermaßen aus:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/
apk/res/android"
package="com.uhh.hades"
android:versionCode="1"
android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="16"
        android:targetSdkVersion="20" />
    <uses-permission android:name="android.permission.
    WRITE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/Hades" >
        <activity
            android:name="com.uhh.hades.activities.
            ManagerActivity"
            android:label="@string/app_name"
            android:screenOrientation="landscape">
            <intent-filter>
                <action android:name="android.intent.
                action.MAIN" />
                <category android:name="android.intent.
                category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="com.uhh.hades.activities.
            MainActivity"
            android:label="@string/app_name"
            android:screenOrientation="landscape">
        </activity>
    </application>
</manifest>
```

8 Entwurfsmuster

8.1 Der MVC Ansatz

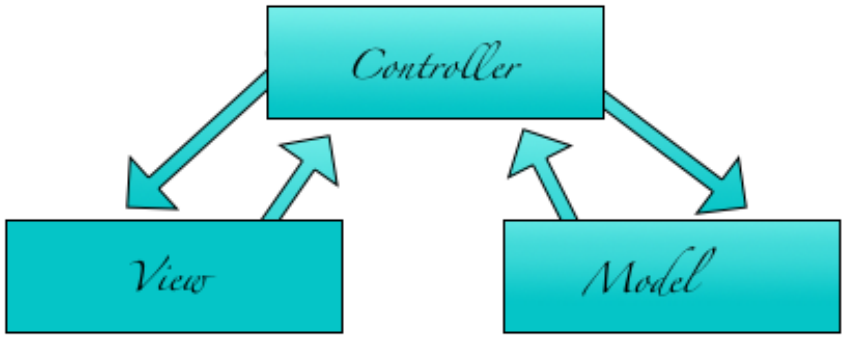

Abbildung 18: Model-View-Controller Entwurfsmuster

Bei MVC (Model - View - Controller) handelt es sich um ein Entwurfsmuster zur Strukturierung einer Applikation. Das Ziel der Verwendung dieses Musters ist es, eine flexible Struktur zu schaffen um das spätere Ändern einer Applikation oder deren Erweiterung zu vereinfachen, als auch eine Basis zu schaffen, welche es ermöglicht Komponenten in anderen Projekten wiederverwenden zu können. Das Entwurfsmuster unterteilt hierbei die Applikation in folgende drei Bereiche:

- Modell(Model)
- Präsentation(View)
- Steuerung(Controller)

Modell

Die Modellschicht beinhaltet die zu verarbeitenden Daten und abhängig von der Implementierung auch die Geschäftslogik einer Applikation. Das Modell hat keine direkte Verbindung zu der Präsentationsschicht einer Applikation. Sämtliche Kommunikation zwischen dem Modell und der Präsentationsschicht findet über die Steuerungsschicht statt.

Präsentation

Die Präsentationsschicht ist zuständig für die Darstellung der Daten. Hierbei kommuniziert diese mit der Steuerungsschicht um die Daten zu erhalten.

Steuerung

Die Steuerungsschicht beschäftigt sich mit der Steuerung der Applikation. Sie kann bis zu mehrere Präsentationsschichten verwalten, Benutzereingaben entgegen nehmen, mit der Modellschicht kommunizieren und die Daten weiter an die Präsentationsschicht leiten. Die Steuerungsschicht kann in vielen Fällen als reine Kommunikationsschicht angesehen werden, es gibt aber auch Fälle wo die Steuerungsschicht Teile der Geschäftslogik enthält.

8.2 Zustandsmuster

Das Zustandsmuster (englisch: State Pattern) ist ein Entwicklungsmuster zur Kapselung von unterschiedlichen Zuständen eines Objektes. Das Muster soll verhindern, dass große Verzweigungen innerhalb einer Applikation entstehen, wenn eine Objekt mehr als einen Zustand hat. [Abbildung: 21] zeigt den Aufbau des Zustandsmusters.

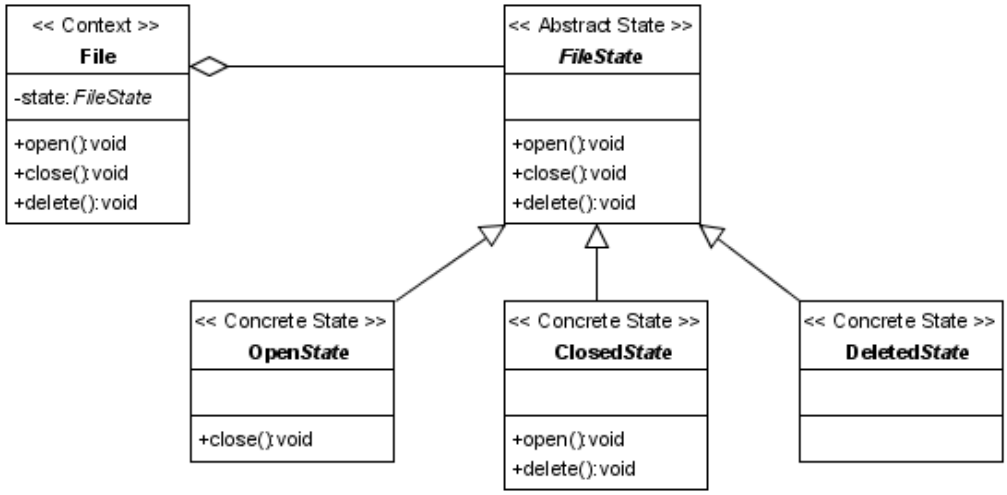

Abbildung 19: Zustandsmuster , Quelle : [8]

Es wird zuerst ein Interface oder eine Abstrakte Klasse definiert, in welche die Methodenaufrufe zur Behandlung eines Zustandes definiert werden. Daraufhin wird jeder Zustand eines Objektes in seine eigene Klasse ausgelagert, welche von der Abstrakten Klasse erbt oder das Interface implementiert. Das Objekt hält jetzt das Zustandsobjekt in welchem es sich befindet, bei einem Zustandswechsel wird das Zustandsobjekt ausgetauscht. Aufgrund der Verwendung einer Abstrakten Klasse oder eines Interfaces, sind alle Methodenaufrufe gleich, somit entfallen Verzweigungen.

9 Lebenszyklus einer Applikation

Das typische Tablet/Smartphone enthält mehr als nur eine Applikation und ähnliche wie ein herkömmlicher PC erlaubt auch Android, das mehrere Programme simultan arbeiten können. Auf einem Tablet/Smartphone ist die Mehrfachverwendung damit realisiert, dass der Benutzer zwischen mehreren Applikation hin und her schalten kann. Die zuvor benutzte Applikation wird dabei eingefroren und sollte der Benutzer Sie wieder benutzen möchten, wieder aufgeweckt. Das System bietet dem Applikationsentwickler die Möglichkeit, auf das Einfrieren oder das Aufwecken der Applikation zu reagieren um gegebenenfalls Daten zu speichern oder diese zu laden. [Abbildung 20] zeigt auf, welche Methoden vom System in welcher Reihenfolge und unter welchen Bedingungen diese aufgerufen werden. Diese Methoden können innerhalb einer Activity überschrieben werden, um gegebenenfalls Aktionen, welche für die Android Version von Hades relevant sind, zu implementieren.

Im Fall der Android Version von Hades bedeutet dies, dass innerhalb der onResume() Methode die Applikation initialisiert werden muss (Schaltplan laden, Simulator starten, etc.) und innerhalb von onPause() der Schaltplan und sämtliche Änderungen gespeichert werden müssen. Es ist wichtig, dass dies innerhalb der onResume() und der onPause() Methode stattfindet, da bei einem herkömmlichen pausieren der Applikation, die Methoden onCreate() und onStop() nicht wieder aufgerufen werden, wie [Abbildung 20] aufzeigt.

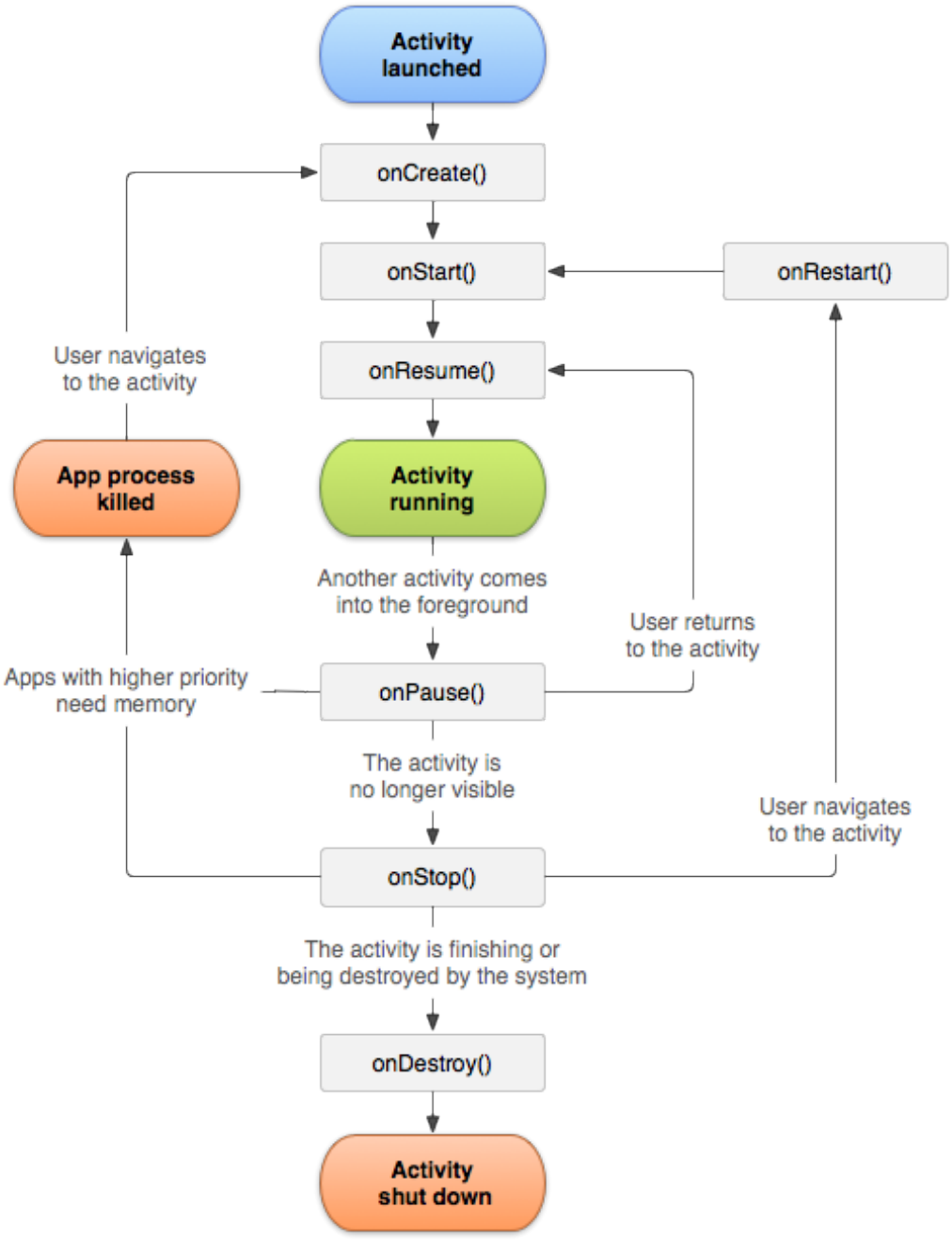

Abbildung 20: Lebenszyklus einer Android Applikation , Quelle : [9]

10 Layout

10.1 Actionbar

Bei der Actionbar handelt es sich um eins der wichtigsten Design Elemente innerhalb von Android Applikationen. Sie dient dem Benutzer als Navigationsleiste innerhalb einer Applikation und wird innerhalb von Google's eigenen Applikation durchgängig verwendet. Es wird daher empfohlen diese zu verwenden [9], um dem Benutzer eine familiäre Navigationsschnittelle zu liefern.

Die Actionbar bietet folgende Funktionen:

- Hinzufügen von graphischen Navigationselementen
- Hinzufügen einer Überschrift innerhalb einer Activity

Die Hades Applikation benutzt die Actionbar um graphische Navigationselemente zum wechseln des Modus darzustellen [Abbildung: 21, rechts], als auch zum anzeigen des Projektnamens[Abbildung: 21, links], in dem der Benutzer sich gerade befindet.

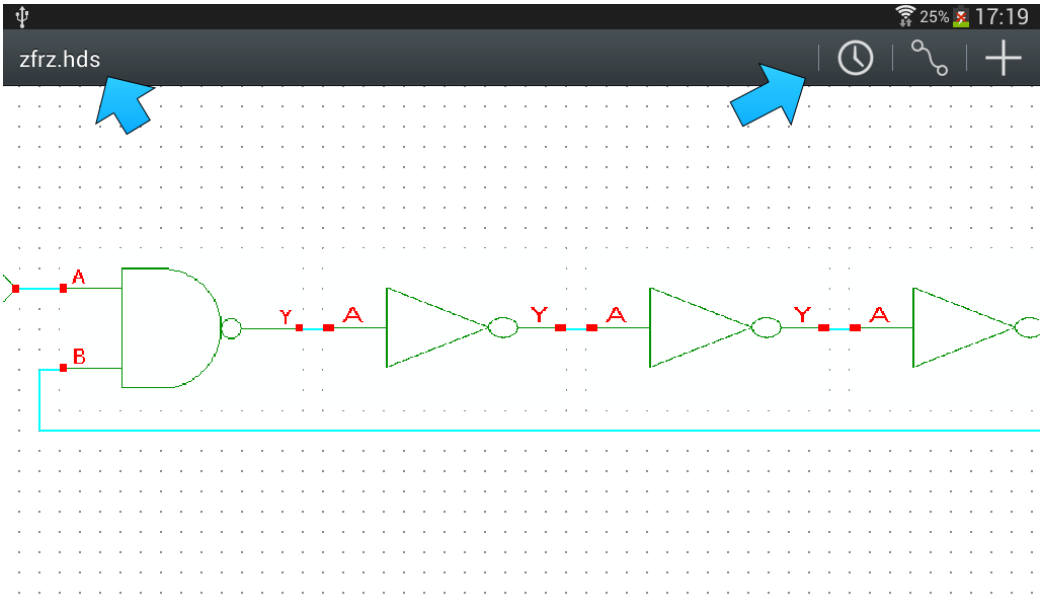

Abbildung 21: Action Bar innerhalb der Android Version von Hades

Das Hinzufügen von Elementen zur Action Bar ist dank der Android SDK sehr einfach. Hierzu muss eine neue Android XML Datei im */res/menu* Verzeichnis angelegt werden, in welcher die Menüpunkte definiert werden. Die XML Datei sieht im Fall von Hades folgendermaßen aus:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:id="@+id/simulationMode"
        android:icon="@drawable/action_bar_simulate"
        android:showAsAction="always"
        android:title="@string/simulation_mode_button_text"/>
    <item
        android:id="@+id/connectionMode"
        android:icon="@drawable/action_bar_connect"
        android:showAsAction="always"
        android:title="@string/connection_mode_button_text"/>
    <item
        android:id="@+id/creationMode"
        android:icon="@drawable/action_bar_create"
        android:showAsAction="always"
        android:title="@string/tool_tip_create"/>
</menu>
```

Ist dies getan, muss lediglich in der Activity die onCreateOptionsMenu() Methode überschrieben werden, in welcher mit der Menuinflater Klasse das Menü erzeugt wird.

```
MenuInflater inflater = getMenuInflater();
inflater.inflate(R.menu.main_activity_actions, menu);
```

10.2 Struktur

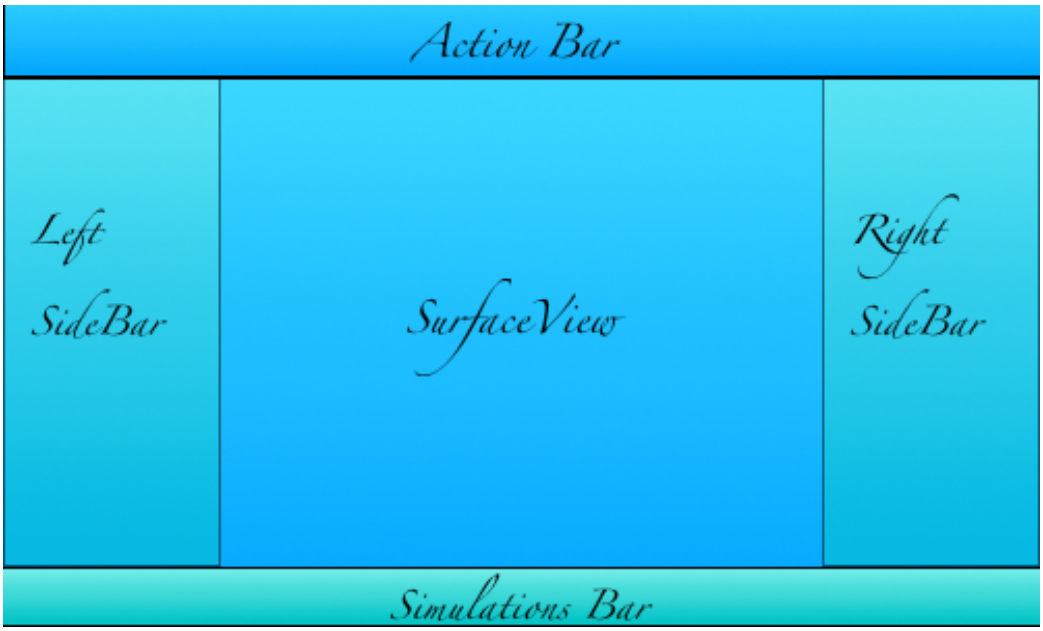

Abbildung 22: Struktur des Hauptlayouts der Android Version von Hades

Die Darstellung von User Interface Elementen in Android funktioniert über die Android View Klassen und deren Unterklassen. Es gibt zwei Möglichkeiten diese in Android zu definieren. Der erste Fall ist das Definieren von Elementen und deren Eigenschaften in XML und die andere Möglichkeit ist die direkte Erzeugung von Instanzen dieser Klassen. Die Android Version von Hades verwendet eine Mischung aus beidem. [Abbildung 22] zeigt die Grundlegende XML Struktur des Hauptlayouts, welches innerhalb des */res/layout/* Ordners definiert ist. Es gibt drei verschiedene XML Layout Elemente zur Positionierung der eigentlichen Benutzerelemente.

Linear Layout

Das Linear Layout organisiert seine Kinderelemente in einer vertikalen oder horizontalen Linie. Dies bedeutet, dass alle Elemente vertikal oder horizontal hintereinander platziert werden.

Relative Layout

Das Relative Layout erlaubt die Positionierung von Elementen relative zueinander. Dies bedeutet, dass Element A Links von Element B sein kann.

List View

Bei einem List View handelt es sich um eine vertikal scrollbare Liste von benutzerdefinierten Elementen. Das Hauptlayout von der Android Version von

Hades ist hauptsächlich mittels Relative Layouts konstruiert. Das äußerste Layout ist ein Relatives Layout, innerhalb diesem befinden sich drei weitere Layouts. Ein Relatives Layout und zwei List Views. Innerhalb des Relativem Layouts befinden sich Android XML Button Elemente und einige Bilder. Dieses Layout stellt die Simulation Bar dar, es ist außer im Simulationsmodus dauerhaft ausgeblendet. Bei den beiden List Views handelt es sich um die linke und rechte Leiste, welche zum einen im Erstellungsmodus als auch im Simulationmodus sichtbar sind. Innerhalb dieser befinden sich entweder die Ein-/Ausgangselemente oder im Fall des Erstellungsmodus sämtliche Schaltungselemente.

11 Erkennung von Gesten

Die Android SDK stellt eine Menge nützlicher Klassen für die Unterschiedlichsten Bereiche zur Verfügung. Auch im Bereich der Gesten-Erkennung nimmt Android dem Entwickler einen Großteil der Arbeit ab, indem Klassen zur Erkennung der Verschiedensten Gesten bereitstellt werden. Es besteht auch die Möglichkeit eigene Gesten zu implementieren. Jedem Android Benutzerinterface Element kann ein onTouchListener gesetzt werden, welche die onTouch Methode implementiert. Innerhalb dieser Methode kann auf Gesten reagiert werden. Die Android Version von Hades verwendet die Klassen GestureDetector und ScaleGestureDetector zur Erkennung von Gesten. Die Android Version von Hades vereinigt alle Gesten an einer Stelle, dem eigenen GestureListener. Der GestureListener implementiert hierzu die Android Interfaces OnGestureListener, OnScaleListener und OnDoubleTapListener und wird an dem SurfaceView gesetzt. Dieser Ansatz erlaubt, dass sämtliche Modi an einer Stelle Zugriff auf die Gesten haben und die Methoden, abhängig von Ihrem Nutzen, überschreiben können.

12 SurfaceView

Der Surface View stellt eine Schnittstelle zum benutzerdefinierten malen, innerhalb der Applikation bereit. In diesem befindet sich die onDraw() Methode. Innerhalb dieser iteriert die Applikation dauerhaft über die Schaltung und malt jedes Element und jede Verbindung welche innerhalb der Schaltung existiert. Das ständige neu malen von Elementen erlaubt es, jegliche Statusänderung eines Objektes sofort zu beobachten. Dieser Ansatz wird häufig in Grafikprogrammen oder auch in PC Spielen verwendet. Dies ist ein Großteil von dem, was die Android Version von Hades interaktiv macht.

13 Übergang zwischen Activities

In der Android Version von Hades existieren zwei Activities. Die erste Activity ist die Projektverwaltung und die zweite ist die eigentliche Hauptapplikation mit den drei Modi. Wenn ein Benutzer eine Projektdatei ausgewählt hat, muss von der Projekt Manager Activity auf die Haupt-Activity gewechselt werden. Die geschieht in Android über Intents. Ein Intent ist eine abstrakte Definition von einer Aktion, welche innerhalb einer Android Applikation ausgeführt werden soll. Zuerst wird ein neuer Intent erzeugt, welche die aktuelle Activity und den Klassennamen der zu startenden Activity als Parameter bekommt. Dann wird an der aktuellen Activity die Methode startActivity() aufgerufen und der Intent wird als Parameter übergeben.

```
Intent intent = new Intent(_activity, MainActivity.class);
_activity.startActivity(intent);
```

14 Autorouter

Der Autorouter ist kein wirklicher Teil der Benutzeroberfläche von der Android Version von Hades, ist aber ein Kernpunkt der Lösung bezüglich des Problems mit Feinmotorischen Operationen. Der Autorouter wird verwendet um eine Verbindung zwischen zwei Schaltelementen zu finden, ohne das diese Schaltelemente überschneidet. Dies dient der Übersichtlichkeit einer Schaltung und vermeidet so, dass der Benutzer den Pfad selbst bestimmen muss. Es sei hierbei angemerkt, dass der Autorouter in seiner jetzigen Form alles andere als perfekt ist, da er immer noch Überschneidungen von Verbindungen zulässt.

Der Autorouter ist mittels des A*-Algorithmuses von Peter Hart, Nils J. Nilsson und Bertram Raphael realisiert. Dabei handelt es sich um einen Algorithmus zur Findungen des kürzesten Pfades zwischen zwei Punkten. Der Algorithmus ist optimal, dies bedeutet, dass immer der kürzeste Pfad zwischen zwei Punkten gefunden wird, falls dieser existiert. Der Algorithmus betrachtet hierzu die umliegenden Gitterpunkte vom Startpunkt aus und ermittelt anhand der Schätzfunktion welcher Gitterpunkt dem Ziel am nächsten liegt und berechnet die Kosten des Pfades anhand der folgenden Funktion:

$$f(x) = g(x) + h(x)$$

Hierbei bezeichnet $g(x)$ die Kosten des bisher gegangenen Pfades, $h(x)$ die Kosten der Schätzfunktion vom Knoten bis zum Zielknoten und $f(x)$ die Gesamtkosten des Pfades. Die Kosten der Schätzfunktion dürfen die Gesamtkosten hierbei niemals überschreiten.

Als Schätzfunktion bieten sich mehrere Funktionen an, die Android Version von Hades verwendet die sogenannte „Manhattan Distanz“, dies ist die Luftlinie zwischen dem Start- und Endpunkt.

15 Potenzial

Die Android Version von Hades erfüllt Ihren Zweck. Benutzer sind in der Lage Schaltungen zu erstellen und erfolgreich zu simulieren. Die PC Version von Hades hat allerdings weitaus mehr Funktionen, als sein kleiner Bruder auf Android. In diesem Sinne hat die Android Version von Hades großes Ausbau Potenzial. Einige Dinge die ausgebaut werden könnten, wären z.b folgende:

- Verbesserung vom Autorouter
- Hinzufügen von komplexeren Schaltelementen
- Support für verschachtelte Schaltungen
- Implementierung des Waveform Viewers
- etc.

Es gibt eine Menge Dinge die noch implementiert werden müssen, bevor die Android Version von Hades die gleiche Funktionalität ausweist wie sein Bruder auf dem PC. Auf der anderen Seite weißt die Android Version von Hades zwei neue Funktionalitäten auf, welche eventuell sogar in die Hades Version auf dem PC übernommen werden könnte, der Autorouter und die Übersichtsseite der Simulationselemente.

Die Android Version von Hades befindet sich erst seit einigen Monaten in Entwicklung, während die PC Version bereits Jahre in der Entwicklung ist. Über die Jahre kann die Android Version von Hades zu einem gleichwertig komplexen oder sogar besseren Produkt entwickelt werden, als sein Bruder.

16 Zusammenfassung

Das Hauptproblem von der Entwicklung einer Android Anwendung von Hades lag in dem mangelnden vorhanden Platz, aufgrund der geringeren Bildschirmgröße auf Tablets/Smartphones. Ein Problem bezüglich Feinmotorischen Operationen innerhalb der Anwendung, ergab sich ebenfalls hieraus. Beide Probleme ließen sich für Hades mit der Aufteilung des Kontextmenüs in drei verschiedene Modi und der Einführung von Übersichtsleisten für Eingänge und Ausgänge, bzw. mit der Implementierung eines Autorouters zur Pfadfindung zwischen Elementen, beheben. Das Layout lässt sich aus einer Vielzahl von Layout XML Elementen zusammensetzen und benutzt zur Navigation die von der Android SDK bereitgestellten Action Bar. Die Erkennung von Gesten läuft über die GestureDetector Klassen und deren Listener ab. Die Android Version von Hades hat einiges an Ausbau Potential bevor es die gleiche Funktionalität von der gleichnamigen PC Version erreicht.

17 Statistische Werte zur Applikation

Gesamtanzahl an Codezeilen:	10564
Anzahl von Klassen:	198
Anzahl von Packeten:	22

Literatur

[1] Michael Kroker.
Mobile Effects 2014 | 2014-l : Das Leben in der digitalen Welt
http://www.tomorrow-focus-media.de/marktforschung/digitalmarkt/info/mobile-effects-2014-i/ ,
21:30 Uhr , 26. Oktober 2014

[2] Will Shanklin.
2014 Tablet Comparison Guide ,
http://www.gizmag.com/tablet-comparison-2014-1-ipad-vs-galaxy-tab-vs-kindle-fire-vs-nexus/32959/,
21:30 Uhr , 26. Oktober 2014

[3] Anthony T.
Finger-Friendly Design: Ideal Mobile Touchscreen Target Sizes
http://www.smashingmagazine.com/2012/02/21/finger-friendly-design-ideal-mobile-touchscreen-target-sizes/,
21:49 Uhr, 26. Oktober 2014

[4] Google Developer Guide
Android, the world's most popular mobile platform,
http://developer.android.com/about/index.html,
22:14 Uhr, 26. Oktober 2014

[5] Tobias Költzsch.
Android läuft auf fast 85 alles Smartphones
http://www.golem.de/news/mobile-betriebssysteme-android-laeuft-auf-fast-85-prozent-aller-smartphones-1408-108290.html,
22:17 Uhr , 26. Oktober 2014

[6] Jason Najarro.
Axure Touch Gestures Widget Library
http://axureland.com/axure-touch-gestures-widgets-library/,
23:48 Uhr, 26. Oktober 2014

[7] Google Developer Guide.
Metrics and Grids
http://developer.android.com/design/style/metrics-grids.html ,
13:44 Uhr, 27. Oktober 2014

[8] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software
5 Auflage. Addison-Wesley, 1996

[9] Google Developer Guide
Adding the Action Bar
http://developer.android.com/training/basics/actionbar/index.html, 19:12 Uhr, 1. November 2014

[10] Google Developer Guide
Android Design Principles
https://developer.android.com/design/get-started/principles.html, 21:45 Uhr, 1. November 2014

[11] IOS Developer Library
iOS Human Interface Guidelines
https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehi
18:42 Uhr, 1. November 2014

[12] Google Developer Guide
Activities
http://developer.android.com/guide/components/activities.html, 21:48 Uhr, 1. November 2014

[13] Google Developer Guide
Layouts
http://developer.android.com/guide/topics/ui/declaring-layout.html, 21:49 Uhr, 1. November 2014

[14] Norman Hendrich.
Interactive Simulation Framework
http://tams-www.informatik.uni-hamburg.de/applets/hades/webdemos/index.html, 23:54 Uhr, 1. November 2014

[15] Google Developer Guide
Intent
http://developer.android.com/reference/android/content/Intent.html, 23.55 Uhr, 1. November 2014

[16] P. E. Hart, N. J. Nilsson, B. Raphael.
A Formal Basis for the Heuristic Determination of Minimum Cost Paths, IEEE Transactions on Systems Science and Cybernetics SSC4 *Seite 100–107, 1968*

Abbildungsverzeichnis

Ich versichere, dass ich die Bachelorarbeit im Studiengang Software-System-Entwicklung selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

Ort, Datum	Unterschrift