# Titel

## STUDIENARBEIT

des Studiengangs Angewandte Informatik

an der Dualen Hochschule Baden-Württemberg Karlsruhe

von

**Mehmet Ali Incekara & Tom Wolske**

Abgabedatum 29. November 2016

| | |
|---|---|
| **Bearbeitungszeitraum** | 12 Wochen |
| **Matrikelnummer** | 12345678 & 1156973 |
| **Kurs** | TINF14B2 |
| **Gutachter der Studienakademie** | Prof. Dr. Kay Berkling |

# Erklärung

Gemäß §5 (2) der „Studien- und Prüfungsordnung DHBW Technik“ vom 18. Mai 2009 erkläre ich hiermit,

1. dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
2. dass die Übernahme von Zitaten und Gedankengut anderer Autoren gekennzeichnet wurde.
3. dass die eingereichte elektronische Fassung exakt mit der schriftlichen übereinstimmt.
4. dass ich die Projektarbeit keiner externen Prüfung vorgelegt habe.

| | |
|---|---|
| Karlsruhe, den 29. November 2016 | |
| Ort, Datum | Tom Wolske |
| Karlsruhe, den 29. November 2016 | |
| Ort, Datum | Mehmet Ali Incekara |

# Inhaltsverzeichnis

# Abkürzungsverzeichnis

# Abbildungsverzeichnis

# Listings

# 1 Einleitung

Über Spiele zu spieleplattform und darüber auf den lernende Effekt oder über lernen auf multimediales lernen zu einer Plattform zum lernen bestehend aus einem spiel

Spiele sind ein Bestandteil unserer Kultur schon seit tausenden Jahren. Das erste Spiel soll das Königliches Spiel von Ur gewesen sein, welches bereits 2600 vor Christus existierte. Spiele haben sich seit dem jedoch weiterentwickelt und dienen heutzutage zum munteren Zeitvertreib. Ob als Brett, Karten oder Glückspiel, Spiele sind überall zu finden und jeder kann sie spielen. Seit 1972 entwickeln sich darüber hinaus weitere Spiele, Videospiele. Sie nutzen die immer größer werdende Rechenleistung von Computern aus, um uns immer realistisch aussehender Spiele zu liefern. Um den Überblick über die Vielzahl an Videospielen zu behalten, haben sich in den letzten Jahren verschiedene Plattformen etabliert, die versuchen dem Nutzer das zu bieten, was sie suchen. Dabei bieten diese viele verschiedene Arten von Spielen an, die einen beim Spielen die Zeit vergessen lassen. Allerdings können Spiele uns nicht nur die Zeit vergessen lassen und für heitere Stunden sorgen, sie können uns auch wissen vermitteln. Sei es durch eine Geschichte die sich real abgespielt hat, wie der erste Weltkrieg, oder anderes. Dieses Wissen wird vermittelt unterbewusst an den Nutzer vermittelt, ohne das er aktiv versucht dieses zu lernen. Für diesen Zweig hat sich eine eigene Branche entwickelt, welche sich mit Lernspielen befasst und versucht uns, über Videospiele, diese Wissen zu vermitteln. Diese Spiele werden hauptsächlich in den Schulen eingesetzt um den Kindern wissen spielerisch zu vermitteln. Jedoch profitiert nicht jedes Kind von diesem Vorteil. Sei es, weil die Schule keine Computer hat, oder weil das Kind nicht zur Schule gehen kann. Für diesen Zweck wurde die Plattform Hone entwickelt, mit der Kinder, die nicht zur Schule gehen können, die Möglichkeit haben, wissen zu erlangen.

## 1.1 Motivation

Bei Hone handelt es sich um eine Spieleplattform auf der sich Kinder, bevorzugt aus Regionen in denen Bildung mangelhaft ist, anmelden können. Auf dieser Plattform haben Sie dann eine Ansicht Ihrer, durch die Spiele, gelernte Kompetenzen, bzw. können

sich die Kinder dort neue Spiele herunterladen, um weitere Kompetenzen zu erwerben. Diese Plattform ist allerdings nicht reizvoll für Kinder gestaltet und soll deshalb attraktiver für Kinder werden. Das wird durch die Entwicklung eines Spieles umgesetzt, in dem die Kinder ihre Fortschritte einsehen können und dabei spielen können. Durch die Umsetzung als App gelingt es zusätzlich den Kindern eine Offlineplattform zu geben, welche sie unabhängig von der Internetverbindung nutzen könne.

## 1.2 Aufbau der Arbeit

Die Arbeit beginnt mit den Grundlegenden Themen die für das verständniss dieser Arbeit von nöten sind und um die verbindung der Altanwendung aufzuzeigen. In diesen Grundlagen wird auf die verwendeteten Technologien und auf die umzusetzenden Ziele eingegangen, damit verstanden werden kann was getan wurde. Darauf aufbauend wird die Implementierung der zuvor erklärten Ziele erläutert und es wird auf schwirige Stellen in der Umsetzung eingegangen. Abgeschlossen wird diese Arbeit mit einem Fazit und einem kurzen Ausblick, in dem der weitere Werdegang des Projektes geschildert wird.

## 1.3 Ziel der Arbeit

Die Arbeit hat als Ziel die Onlineplattform **Hone** für Kinder interessanter zu gestallten. Dabei wird dies durch eine App realisiert. Diese baut auf der Altanwendung auf und benutzt Schnittstellen zu dieser, um an die benötigten Daten zu gelangen. Die App wird dabei für Smartphones mit dem Betriebssytem Android optimiert.

Das Ziel dieser Arbeit ist es dabei eine Dokumentation über die Vorgehensweise zu liefern, sowie eine Dokumentation, mit deren Hilfe andere arbeiten Können.

# 2 Software Requirements Specification

Das Software Requirements Specification, kurz SRS, ist ein veröffentlichter Standard zur Spezifikation einer Software. Die Struktur eines SRS ist vom Institute of Electrical and Electronics Engineers im Standard IEEE 830-1998 festgehalten.

Der weitere Aufbau dieses Kapitels entspricht dem des Standards.

## 2.1 Einführung

Dieses Unterkapitel des SRS stellt einen Überblick über das gesamte SRS bereit.

### 2.1.1 Zweck

Das SRS beschreibt den Projektumfang und die Anforderungen an die Software. Dabei beschreibt der Verfasser beispielsweise die Funktionalität, die externen Schnittstellen und die Performanz[1].

Die Zielgruppe des SRSs sind zunächst alle, die in irgendeiner Verbindung mit der Software stehen oder jene, die Interesse an der Umsetzung interessiert sind. Zudem dient die Spezifikation zur Kommunikation zwischen Stakeholder und Entwickler.

### 2.1.2 Umfang

Dieses SRS handelt von dem Software Produkt NoRPG. Bei NoRPG handelt es sich um eine Gamifizierung einer Lernspielbibliothek für Android. Allgemein stellt NoRPG nur Lernspiele bereit. Jedoch ist NoRPG wie ein Rollenspiel aufgebaut und erfordert das Spielen von Lernspielen, um weitere Spiele freizuschalten. Jedoch ist es kein Rollenspiel im üblichen Sinne.

[1] vgl. Tripp [1](1998) Seite 3

NoRPG richtet sich an Kinder und soll durch den Aufbau eines Rollenspiels die Benutzer dazu anregen, weitere Spiele herunterzuladen und zu spielen. Im nächsten Kapitel werden unter anderem die Gedanken zur Gestaltung und weitere Ziele von NoRPG genauer betrachtet.

## 2.2 Allgemeine Beschreibung

In diesem Unterkapitel werden die allgemeinen Faktoren, die das Produkt und seine Anforderungen betreffen, beschrieben. Dieses Kapitel behandelt nicht die spezifischen Anforderungen sondern stellt den Hintergrund für diese Anforderungen dar.

### 2.2.1 Produktperspektive

In diesem Unterabschnitt des SRS wird das Produkt in unterschiedlichen Perspektiven mit verwandten Produkten betrachtet, denn bei NoRPG handelt es sich nicht um ein selbstständiges abgeschlossenes Projekt.

For each interface: Discussion of the purpose of the interfacing software as related to this software product

Definition of the interface in terms of message content and format. It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.

#### System interfaces

Hone ist das Frontend für Spielentwickler und für Benutzer. Spielentwickler können ihre Spiele bei Hone zur Verfügung stellen und die Benutzer können auf der Webplattform unterschiedliche Lernspiele herunterladen.

NoRPG stellt das verbesserte und kindgerechte Frontend der Benutzer bereit. Der Benutzer kann alle Aktionen aus der App steuern. Jedoch können sich die Benutzer weiterhin auf der Webplattform anmelden.

### User interfaces

The logical characteristics of each interface between the software product and its users. This includes those conÞguration characteristics (e.g., required screen formats, page or window layouts, content of any reports or menus, or availability of programmable function keys) necessary to accomplish the software requirements.

All the aspects of optimizing the interface with the person who must use the system. This may simply comprise a list of do's and don'ts on how the system will appear to the user. One example may be a requirement for the option of long or short error messages.

### Hardware interfaces

This should specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes conÞguration characteristics (number of ports, instruction sets, etc.).

It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.

### Software interfaces

This should specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system). For each required software product, the following should be provided:Name, Specification number, Version number and Source.

### Memory constraints

This should specify any applicable characteristics and limits on primary and secondary memory

### 2.2.2 Produktfunktionen

This subsection of the SRS should provide a summary of the major functions that the software will perform. For example, an SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.

Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level speciÞcation (if one exists) that allocates particular functions to the software product. Note that for the sake of clarity The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time.

Textual or graphical methods can be used to show the different functions and their relationships. Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables.

### 2.2.3 Benutzermerkmale

Grundsätzlich richtet sich die App NoRPG an Kinder, die keine Möglichkeit haben eine Schule zu besuchen oder zusätzlich lernen wollen. Jedoch werden keine anderen Benutzergruppen für diese App ausgeschlossen.

Der Benutzer sollte Erfahrung mit der Verwendung eines Smartphones, insbesondere mit Android-Systemen, haben. Dazu zählt die Bedienung der Android-Oberfläche und insbesondere die Bedienung des Google Play Stores.

Da es sich bei den Benutzern in den meisten Fällen um Kinder handelt, sollten diese englische Texte lesen und verstehen können. Denn zum voranschreiten muss der Benutzer die Unterhaltungen mit NPC zum herunterladen von Spielen verstehen können um die richtige Aktion auszuwählen.

### 2.2.4 Einschränkungen

Einschränkungen für den Entwickler

Regulatory policies, Hardware limitations, Interfaces to other applications, Parallel operation, Audit functions, Control functions, Higher-order language requirements, Signal handshake protocols, Reliability requirements, Criticality of the application and Safety and security considerations

### 2.2.5 Annahmen und Abhängigkeiten

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption may be that a speciÞc operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

## 2.3 Spezifische Anforderungen

This section of the SRS should contain all of the software requirements to a level of detail sufÞcient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisÞes those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output. As this is often the largest and most important part of the SRS, the following principles apply:

SpeciÞc requirements should be stated in conformance with all the characteristics described in 4.3.

SpeciÞc requirements should be cross-referenced to earlier documents that relate

All requirements should be uniquely identiÞable

Careful attention should be given to organizing the requirements to maximize readability

### 2.3.1 Externe Schnittstellen

### 2.3.2 Funktionale Anforderungen

Use Cases dokumentieren Funktionalitäten eines Systems auf Basis von einfachen Modellen. In einem Use Case wird das nach außen sichtbare Verhalten eines Systems aus der Sicht der Nutzer beschrieben. Ein Nutzer kann hierbei eine Person, eine Rolle oder ein anderes System sein. Dieser Nutzer tritt als Akteur mit dem System in Interaktion, um ein bestimmtes Ziel zu erreichen.

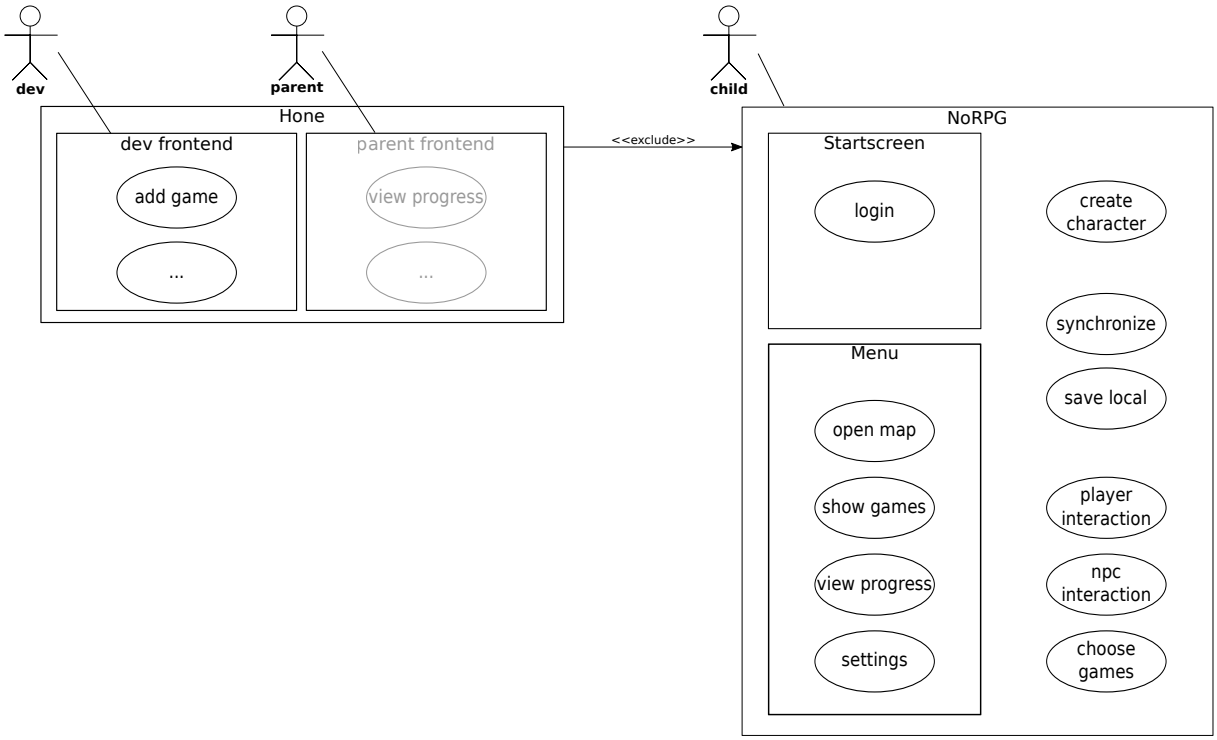


Abbildung 1: Overall Use Case Diagramm

In der Grafik sind 2 Systeme zu sehen. Links das vorhandene System Hone welches ein Frontend für die Entwickler und für die Kinder darstellt. Die Kinder sollen nicht mehr über Hone die Spiele herunterladen sondern nur noch die App NoRPG verwenden. Die Ansicht wird jedoch weiterhin genutzt und soll den Eltern der Kinder die Möglichkeit geben, den Fortschritt des Kindes nachzuschauen. Hone soll von den Rollen Entwickler und Eltern entwickelt werden.

Das rechte System NoRPG stellt die zu entwickelnde App dar. Diese dient als Frontend für das Kind. Es gibt viele Use Cases. Die Use Cases werden in unterschiedliche Gruppen zusammengefasst. Die nächsten Unterkapitel sind die einzelnen Gruppierungen.

### Login

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer sich bei NoRPG anmelden möchte. Eine Anmeldung ist notwendig um NORPG zu starten. Die Besonderheit bei der Anmeldung ist, dass der Account des Benutzers auch für die Anmeldung bei Hone benötigt wird. Der Benutzer kann sich in NoRPG im Startbildschirm anmelden und anschließend das Spiel zu starten.

**Ereignisablauf** Eingeben von Benutzername und Passwort. Klicke auf Login.

Alternativer Ablauf: Abbruch oder Spiel Beenden

**Vorbedingungen** Benutzer ist registriert, Während der Anmeldung ist eine Internetverbindung vorhanden, Kombination von Benutzername und Passwort existiert, es ist kein anderer Benutzer auf dem Gerät angemeldet

**Nachbedingungen** Benutzer angemeldet, kann online oder offline weiterspielen. Beim nächsten Start der App ist der Benutzer automatisch angemeldet.

Oder falls die eingegeben Daten nicht übereinstimmen, Fehlermeldung anzeigen

#### Create character

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer seinen Charakter erstellen möchte. Dies ist eine einmalige Aktion, die beim ersten Anmelden durchlaufen wird.

**Ereignisablauf** Wenn der Benutzer sich zum ersten mal anmeldet hat er die Möglichkeit seinen Charakter zu erstellen. Dafür wählt der Benutzer sich zunächst sein Geschlecht aus und wählt anschließend den passenden Charakter.

Zum Abschluss vergibt der Benutzer seinen Charakter einen Namen.

**Vorbedingungen** Der Account meldet sich das erste mal in der App an.

**Nachbedingungen** Nach der Erstellung beginnt das Spiel und der Charakter ist gespeichert. Bei erneuter Anmeldung muss der Benutzer nicht erneut einen Charakter erstellen.

#### Player interaction

Dieser Use Case beschreibt den Anwendungsfall: Benutzer interaktionen, wie Bewegen oder Bestätigen.

**Ereignisablauf** Klickt auf Pfeiltasten, Charakter bewegt sich in diese Richtung

Klickt auf A, Charakter bestätigt

Klickt auf B, Charakter lehnt ab

(Bild Mockup)

**Vorbedingungen** Spieler befindet sich im Spiel (nicht loading screen und menü ist geschlossen)

**Nachbedingungen** Charakter bewegt sich, bestätigt oder lehnt ab

#### NPC interaction

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer sich in einer Interaktion mit einem NPC befindet. NPC bedeutet Non-Player Charakter und stellt die programmierten Charaktere dar (Unterhaltungen mit NPC, Storytelling)

**Ereignisablauf** Ereignisablauf etc.

**Vorbedingungen** Ingame, nicht loading screen oder menü offen

**Nachbedingungen** Unterhaltung findet statt, etc.

#### Choose games

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer ein spiel zum downloaden auswählt

**Ereignisablauf** Der Benutzer kann sich (wenn vorhanden) zwischen mehrere Spielen auswählen um den Kurs abzuschließen.

**Vorbedingungen** Internetverbindung, darf die SPiele nach dem Standard spielen

**Nachbedingungen** Weiterleitung auf Google Play Store

#### Open map

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer die Karte öffnet. Die Karte dient zur Orientierung der Welt und beinhaltet Symbole etc. um herauszufinden was so ist

**Ereignisablauf** Benutzer öffnet Menü und klickt auf "Map"...

**Vorbedingungen** Menü offen, Benutzer befindet sich nicht in einer NPC Interaktion

**Nachbedingungen** Eine Karte von der aktuellen Welt wird geöffnet

#### Show games

Dieser Use Case beschreibt den Anwendungsfall: Liste der gespielten und heruntergeladneen Spiele wird angezeigt. Zuordnung zu den Standards. Aus NoRPG das Spiel starten können.

**Ereignisablauf** Benuter öffnet Menü und klickt auf "Games"...

**Vorbedingungen** Menü offen, Benutzer befindet sich nicht in einer NPC interaktion

**Nachbedingungen** Eine Liste wird angezeigt

#### View progress

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer

#### Ereignisablauf

#### Vorbedingungen

#### Nachbedingungen

#### Settings

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer

#### Ereignisablauf

#### Vorbedingungen

#### Nachbedingungen

#### Synchronize

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer

#### Ereignisablauf

#### Vorbedingungen

#### Nachbedingungen

#### Save local

Dieser Use Case beschreibt den Anwendungsfall, dass der Benutzer

#### Ereignisablauf

#### Vorbedingungen

#### Nachbedingungen

### 2.3.3 Performance requirements

This subsection should specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole. Static numerical requirements may include the following

The number of terminals to be supported, The number of simultaneous users to be supported, Amount and type of information to be handled

### 2.3.4 Logical database requirements

This should specify the logical requirements for any information that is to be placed into a database. This may include the following:

Types of information used by various functions

Frequency of use

Accessing capabilities

Data entities and their relationships

Integrity constraints

Data retention requirements.

### 2.3.5 Design constraints

This should specify design constraints that can be imposed by other standards, hardware limitations, etc.

Standards compliance: This subsection should specify the requirements derived from existing standards or regulations. They may include the following:

Report format, Data naming, Accounting procedures and Audit tracing.

### 2.3.6 Software system attributes

There are a number of attributes of software that can serve as requirements. It is important that required attributes be speciÞed so that their achievement can be objectively veriÞed. Subclauses 5.3.6.1 through 5.3.6.5 provide a partial list of examples.

### 2.3.7 Usability

### 2.3.8 Reliability

This should specify the factors required to establish the required reliability of the software system at time of delivery.

### 2.3.9 Availability

This should specify the factors required to guarantee a deÞned availability level for the entire system such as checkpoint, recovery, and restart.

### 2.3.10 Security

This should specify the factors that protect the software from accidental or malicious access, use, modiÞcation, destruction, or disclosure. SpeciÞc requirements in this area could include the need to

Utilize certain cryptographical techniques;

Keep speciÞc log or history data sets;

Assign certain functions to different modules;

Restrict communications between some areas of the program;

Check data integrity for critical variables.

### 2.3.11 Maintainability

This should specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices.

### 2.3.12 Portability

This should specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include the following:

Percentage of components with host-dependent code;

Percentage of code that is host dependent;

Use of a proven portable language;

Use of a particular compiler or language subset;

Use of a particular operating system.

### 2.3.13 Organizing the speciÞc requirements

For anything but trivial systems the detailed requirements tend to be extensive. For this reason, it is recommended that careful consideration be given to organizing these in a manner optimal for understanding. There is no one optimal organization for all systems. Different classes of systems lend themselves to different organizations of requirements in Section 3 of the SRS. Some of these organizations are described in 5.3.7.1 through 5.3.7.7.

### 2.3.14 System mode

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency. When organizing this section by mode, the outline in A.1 or A.2 should be used. The choice depends on whether interfaces and performance are dependent on mode.

### 2.3.15 User class

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and Þre Þghters. When organizing this section by user class, the outline in A.3 should be used.

### 2.3.16 Objects

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. When organizing this section by object, the outline in A.4 should be used. Note that sets of objects may share attributes and services. These are grouped together as classes.

### 2.3.17 Feature

A feature is an externally desired service by the system that may require a sequence of inputs to effect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs. When organizing this section by feature, the outline in A.5 should be used.

### 2.3.18 Stimulus

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc. When organizing this section by stimulus, the outline in A.6 should be used.

### 2.3.19 Response

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc. The outline in A.6 (with all occurrences of stimulus replaced with response) should be used.

### 2.3.20 Functional hierarchy

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by either common inputs, common outputs, or common internal data access. Data ßow diagrams and data dictionaries can be used to show the relationships between and among the functions and data. When organizing this section by functional hierarchy, the outline in A.7 should be used.

# 3 NoRPG

Bei NoRPG handelt es sich um ein Spiel, welches an ein Role Player Game (RPG) erinnern soll. Bei RPGs spielt der Spieler einen Chracter in einer fiktiven Welt. Dabei gibt es eine Story die wärend des spielens erlebt wird. Damit diese Geschichte vorran geht, muss der Spiele verschiedene Quests, dabei handelt es sich um verschiedenste Aufgaben, erledigen, um einen Fortschritt im Spiel zu erlangen. Darüber hinaus sammelt der Spieler Objekte in der Welt die er im Spiel nutzen kann. Ein Beispiel für solche Spiele ist das Spiel The Witcher 3 welches auf diese Prinzipien aufbaut.

Darüber hinaus gibt es noch sogenannte MMORPGs, Massive Multiplayer Online Role Player Game. Dabei Gibt es wie bei RPGs eine Story und Quests allerdings kann man auch auf andere Spieler treffen und mit diesen zusammen spielen. Das wohl berühmteste MMORPG ist dabei World of Warcraft vom entwickler Blizzard. Hier kann der Spieler zwischen verschiedenen Klassen einen Chrktere auswählen, von Menschenähnlichen bis zu komischen Gestalten. Jede dieser Klassen hat verschiedene Fähigkeiten welche sich auf den Spieleverlauf auswirken. Darüber hinaus gibt es in MMORPGs verschiedene Events an denen die Spieler gemeinsam versuchen eine Quest zu erfüllen. Dieser Multiplayer grenzt die MMORPGs von den RPG ab.

NoRPG hat keine Multiplayer möglichkeiten und ist deswegen nur ein RPG. Allerdings muss der Spieler keine Quests erfüllen wie sie aus anderen Spielen bekannt sind. Der Nutzer muss andere Spiele spielen, damit die Geschichte im Spiel vorran geht. Dshalb wurde sich für den Namen NoRPG entschieden, da es sich bei dem Spiel um ein RPG handelt, aber nicht alle klassischen Eigenschaften eines RPGs besitzt.

NoRPG ist ein Spiel welches versucht Bildung für jeden erreichbar zu machen. Dieses Ziel ist dabei in den Global Goals definiert. Dabei handelt es sich um 17 Ziele welche bis 2030 Umgesetzt werden sollen um das Leben für alle Menschen auf der Welt zu verbessern. 2015 haben 193 Weltführer diese Unterzeichnet und begonnen diese Umzusetzen. Dabei sind diese Ziele umfangreich und reichen von einem besseren Umgang mit den uns zur Verfügung stehenden Ressourcen bis hin zu qualitativ hochwertiger Bildung für jeden und kostenlos.

NoRPG unterstützt dabei das Ziel Hochwertige Bildung für jeden Zugänglich zu machen, da die App für jeden frei zugänglich ist. Dieses Ziel hat weitere Unterziele, wobei nun kurz auf die für NoRPG relevanten Unterziele eingegangen wird.

- Bis 2030 sicherstellen, dass alle Mädchen und Jungen gleichberechtigt eine kostenlose und hochwertige Grund- und Sekundarschulbildung abschließen, die zu brauchbaren und effektiven Lernergebnissen führt.

  Dieses Unterziel wird in NoRPG dahingegend unterstützt, dass die Common Core State Standards implementiert. Dies sind standartisierte Standarts für Unterrichtsfächer und beschreiben den zu erlernenden Inhalt für Kinder in den verschiedenen Klassen. Da NoRPG für alle kostenfrei zugänglich ist und Mädchen und Jungen gleichberechtigt sind werden auch diese zwei Aspekte des Unterziels unterstützt.

- Aufbau und Weiterentwicklung von Bildungseinrichtungen, die kinder- und behindertengerecht und geschlechtsspezifisch sind und für alle eine sichere, gewaltfreie, integrative und effektive Lernumgebung bieten

  Da NoRPG keine Bildungseinrichtung ist wird dieses Unterziel nur bedingt erfüllt. NoRPG bietet Kindern jedoch eine sichere, gewaltfreie, integrative und effektive Lernumgebung, wochurch dieses Ziel zum Teil erfüllt wird. Darüber hinaus ist diese Umgebung für Kinder ausgelegt und somit kindergerecht. Geschlechtsspezifisch ist das Spiel nur dahingehend, dass die Kinder zu begin einen Charaktere in ihrem Geschlecht und Alter auswählen können.

`http://www.globalgoals.org/de/global-goals/quality-education/`

## 3.1 Gedanken

GAMIFICATION – was ist gamification wieso wird das gemacht, hier rein passt nicht bei SRS

## 3.2 Story

In NoRPG spielt der Spieler einen Charakere der zu Begin der Geschichte in einem Dorf wohnt. Von diesem können die Einwohner durch Portale in 5 andere Welten gehen, in denen andere Menschen leben und andere Lebensbedingungen herschen. Eines

Tages wurde dieses Dorf von einem Bösewicht heimgesucht und dieser hat die komplette Welt, in der der Spieler lebt, farblos gemacht. Zusammen mit den Farben wurden die Emotionen aus dem Dorf genommen. Nun hat sich der Hauptcharaktere das Ziel gesetzt, die Farben und die Emotionen der Einwohner zurück zu bringen, damit wieder alles wie vor dem Angriff ist. Dazu muss der Spieler in die verschiedenen Welten gehen und verschiedene Aufgaben erfüllen.

In Welt 1, einer Waltwelt, muss der Spieler verschiedene Edelsteine sammeln und Aufgaben erfüllen, um die ersten Farben zurück zu erlangen. Sobald dies geschehen ist kommt die Farbe Grün in das Dorf des Spielers zurück. Nun muss der Spieler in die anderen Welten gehen und die Farbe zurück holen. Über dem Dorf wird sein aktueller Fortschritt als Regenbogen dargestellt, welcher umso bunter und voller wird, unso mehr Edelsteine gefunden und Aufgaben abgeschlossen wurden. Sobald der Spieler alle Edelsteine gefunden und alle Aufgaben abgeschlossen hat, hat er die Welt gerettet und es kommt eine Endsequenz.

## 3.3 Abgrenzung zu MOOC

NoRPG ist allerdings kein Massive Open Online Course (MOOC) sondern baut nur auf einem auf. Die Daten dieses MOOCs werden in NoRPG zur verfügung gestellt, damit die Kinder einen Anreiz haben, diesen zu nutzen. Der eigentliche MOOC ist dabei Hone. Dabei handelt es sich um eine Lernplattform, welche es für Kinder ermöglicht, grundlegendes Verständniss, welches in der Grundschule vermittelt wird, zu erlangen. Dabei stellt Hone den Kindern eine Liste von Spielen bereit. Durch spielen dieser Spiele lernen die Kinder die Dinge die sie auch in der Grundschule lernen würden. Dabei sind diese Lerninhalte in den CCSS definiert und aufgegliedert.

# Literaturverzeichnis

[1] Leonard L. Tripp et al. „IEEE Recommended Practice for Software Requirements SpeciÞcations“. In: (1998).

# Anhang

## Anhang A