Karlsruher Institut für Technologie

Fraunhofer IOSB

Web-basierte Schnittstelle zur Analyse und Adaption von Serious Games

Masterarbeit

von

Christian Biegemeier

Lehrstuhl für Interaktive Echtzeitsysteme am KIT

Abteilung Interoperabilität und Assistenzsysteme am Fraunhofer IOSB

Betreuer:	Prof. Dr.-Ing. Jürgen Beyerer
Betreuender Mitarbeiter:	Dipl.-Inf. Alexander Streicher
Abgabetermin:	11. Juli 2016

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und mit keinen anderen als den angegebenen Quellen und Hilfsmitteln angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Karlsruhe, 11. Juli 2016

_____

(Christian Biegemeier, B. Sc.)

Danksagung

Mit diesen Zeilen möchte ich mich bei all denjenigen bedanken, die mich während des Studiums und der Anfertigung dieser Masterarbeit unterstützten.

Zunächst gilt mein gebührender Dank Prof. Dr.-Ing. Jürgen Beyerer, der diese Arbeit ausgeschrieben und somit diese spannende Aufgabe erst ermöglicht hat. Auch möchte ich im gleichen Atemzug Dipl.-Inf. Alexander Streicher danken, der durch konstruktives Feedback und hilfreiche Anmerkungen zu dieser Master-Thesis beigetragen hat.

Daneben gilt ein besonderer Dank meiner Freundin, die mir während der kompletten Studienzeit eine große moralische Unterstützung war. Des Weiteren ließ sie mir den nötigen Freiraum, sodass ich mich des Öfteren bis spät in den Abend hinein in unterschiedlichste Vorlesungen einarbeiten sowie die Bachelor- und Masterarbeit vollenden konnte.

Nicht zuletzt danke ich auch herzlichst meiner Familie, da sie mich während des Studiums nicht nur finanziell, sondern vor allem auch emotional unterstützte und immer für mich da war.

Vielen Dank!

Zusammenfassung

In Rahmen dieser Ausarbeitung wurde ein Konzept zur interoperablen Adaption von Serious Games und Simulationen umgesetzt sowie eine systemübergreifende Webschnittstelle für die Analyse von Nutzungsdaten und Adaption von Spielelementen entworfen.

In Lernspielen ergibt sich die Problematik, ein bestmögliches Mittelmaß zwischen einer erhöhten Motivation durch Spielelemente und den Lerninhalten zu finden. Für die effektive und effiziente Aufrechterhaltung der Motivation und das Erreichen der Lernziele, müssen digitale Lernspiele an das Nutzungs- und Lernverhalten angepasst werden. Hierfür wird eine didaktisch ausgeprägte externe Adaptionslogik benötigt, um die Informationen von verschiedenen Anwendungen verstehen, auswerten und darauf reagieren zu können.

Die Problemstellung dieser Arbeit betrifft die grafische Darstellung der internen Adaptionsmechanismen und die mögliche Anpassung eines solchen externen Systems. Weiterhin wird die wissenschaftliche Fragestellung behandelt, welche Daten sich für eine einfache Form der Adaptivität nutzen lassen und wie sich diese auf ein Spiel auswirken können.

Gegenstand dieser Arbeit ist die prototypische Umsetzung eines solchen Systems. Hierfür wurde als externes System die E-Learning Artifical Intelligence (ELAI) Architektur entwickelt [1]. Für eine einheitliche Kommunikation zwischen den zu adaptierenden Spielen und der ELAI, werden xAPI-Statements mit zusätzlichen Metadaten verwendet. Die ELAI bietet zusätzlich eine Web-basierte Nutzungsschnittstelle (ELAI-UI) an. Die ELAI-UI ermöglicht unter anderem die transparente Darstellung der ELAI-Aktionen, die Analyse von Nutzerdaten und die Möglichkeit der Einflussnahme auf ein Spiel. Diese Einflussnahme resultiert durch zyklische Parameterabfragen in einer Adaptivität des Spiels. Als Parameter wurden prototypisch die Spielschwierigkeit und der Hilfelevel eingesetzt. Zur Evaluation der Gesamtarchitektur wurde weiterhin ein Suche-und-Finde Spiel entwickelt und um einen ELAI-Adapter zur Kommunikation mit der ELAI erweitert. Hierbei wurde ein virtueller Assistent umgesetzt, welcher dem Spieler die Aufgabe textuell erläutert und Hilfestellungen gibt. Diese Hilfestellungen werden durch den Parameter Hilfelevel adaptiert. Der Schwierigkeitslevel beeinflusst direkt die Größe des gesuchten Objektes sowie die Wolkenbildung, die wiederum die Sichtverhältnisse beeinflusst. In einer durchgeführten Studie wurde die Adaptionsfähigkeit der ELAI anhand des entwickelten Suche-und-Finde-Spiels mit n = 12 Probanden evaluiert.

Abstract

In this master thesis, an architecture for automatic interoperable adaptation and analyzation of serious games and simulations was developed.

The problem of learning games is to create a high playing motivation. Therefore, the developer has to keep the entertainment elements balanced to the learning elements. To handle this trade-off, the game needs a didactic adaptation component which influences the game's elements respectively the player's usage and learning behavior. Further, an external didactic adaption logic is needed in order to understand the different information from various applications, to evaluate them and to send a reaction back to application.

In order to address these concerns, a prototypical external architecture called E-Learning Artifical Intelligence (ELAI) was implemented [1]. Additionally, the ELAI provides a user interface named ELAI-UI which is authorized to load statements and to present them in different ways. The ELAI-UI can be used to make the ELAI actions transparent, to analyze the user data and to adapt the game. To evaluate the whole concept, a prototypical 'seek and find' serious game with an integrated ELAI adapter was developed. The ELAI adapter handles the communication between serious game and ELAI. As a part of the serious game implementation, a virtual assistant was implemented showing players the task description and helping information. The ELAI uses the variables *difficulty* and *helping* levels to adapt games. The *helping* variable influences the messages from the virtual assistant. The *difficulty* variable influences the size of target and also the quantity of clouds which are used to change the viewing conditions during the game. In a study the adaption skills of ELAI were evaluated with n = 12 test persons who played the developed 'seek and find' game and completed a questionnaire about the applied adaptation strategy.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1. Motivation

Früher noch als reine Freizeitbeschäftigung angesehen, finden heutzutage Videospiele immer mehr Beachtung. In den letzten Jahrzehnten erfuhr diese Branche im technischen sowie auch im sozialen Bereich einen enormen Wandel [2]. Betrachtet man die Umsatzentwicklung der Videospielbranche in den letzten vier Jahren von 23,9 Billionen Euro pro Jahr (2012) auf 40 Billionen Euro pro Jahr (2015) weltweilt [3], wird deutlich, dass Videospiele genauso wie Bücher, Filme oder das Fernsehen Teil unserer Kultur geworden sind [4]. Allerdings werden die genannten Medien heutzutage nicht nur zur Unterhaltung sondern auch zum Erlangen von Wissen verwendet. Bücher werden als Fachliteratur, Filme als Dokumentation und Fernsehen als Nachrichtensendungen zur Beschaffung von Information herangezogen. Warum also nicht auch Videospiele?

Educational Serious Games beschreibt eine solche Kombination von Wissenserwerb und Spiel und wird meist für digitale Spiele mit einem ernsthaften Lernhintergrund verwendet. Entwickler von solchen Spielen wandern dabei auf einem schmalen Grat zwischen Unterhaltung und Wissensvermittlung. Einerseits sollte das Serious Game nicht zu wenig Lerninhalt übermitteln und andererseits verliert das Spiel den *Entertainment-Faktor*, falls das Verhältnis des zu vermittelnden Wissens zu den spielerischen Elementen zu hoch ist. Bei der Entwicklung von Educational Serious Games spielt daher der so genannte Flow-Zustand eine wichtige Rolle. Dieser beschreibt das vollständige Eintauchen in ein Spiel, was im besten Fall zu einer Nicht-Wahrnehmung des Lernprozesses führt. Das wird durch eine bestmögliche Adaption des Videospiels an den Wissenstand und die Fähigkeiten des Spielers erreicht [5]. Dies bedeutet, dass Interaktionsmechanismen, Inhalte oder Vorschläge durch eine intelligente Tutoring-Komponente an die Lernbedürfnisse und den Lernfortschritt des Spielers angepasst werden müssen [6].

1.2. Projektumfeld

An einem Lösungskonzept für interoperables, adaptives E-Learning forscht die Abteilung „Interoperabilität und Assistenzsysteme“ des Fraunhofer Instituts für Optronik, Systemtechnik und Bildauswertung (IOSB) in Karlsruhe. In dem Projekt „Intelligente Bildauswertung in verteilten Systemen“ wurde die so genannte ELAI (E-Learning A.I.) Architektur entworfen, die es ermöglicht, einen adaptiven interoperablen Tutoring-Agenten zu kreieren, dessen Ziel es ist, den Nutzer im Flow-Kanal zu halten [1]. Die Interoperabilität wird bei der ELAI-Architektur durch eine Entkopplung der Spiele-Engine von der externen ELAI-Tutoring Komponente erreicht (Abbildung 1.1). Die Simulation oder das Serious Game wird mithilfe eines ELAI-

Adapters an die ELAI angepasst. Daten von Nutzerinterkationen können auf der Kommunikationsschicht z.B. mithilfe der High-Level-Architektur (HLA) [7] oder mit xAPI-Statements [8] zur ELAI übertragen werden. Für die Adaption ist ein virtueller Agent oder eine Spielmechanik vorgesehen. Während der virtuelle Agent nur angepasste Hinweise auf den aktuellen Zustand des Spiels gibt, werden über die Spielmechanik dynamisch tiefergehende Änderungen eingebracht [1].

Das Konzept der ELAI-Architektur basiert dabei auf einer automatischen Adaption der Simulation oder des Serious Games. Dabei sind die Schritte und Zustände der ELAI nach außen nicht sichtbar. Somit ist eine explizite Einflussnahme und die Analyse der Nutzerdaten nicht möglich.

1.3. Problemstellung

Die Problemstellung dieser Arbeit betrifft die Fragen, wie ein interoperabler Adaptivitätsmechanismus umgesetzt werden kann und entstehende Nutzungsdaten visuell dargestellt werden können. Die Nutzungsschnittstelle muss Zugriff auf Lernfortschrittsprofile, Nutzerprofile und Nutzungsdaten bieten. Weiterhin soll mit dieser eine Analyse der Daten sowie die Einflussnahme auf das Spiel ermöglicht werden.

Die wissenschaftliche Fragestellung ist, welche Nutzungsdaten sich für eine einfache Form der interoperablen Adaptivität nutzen lassen und wie sich diese Adaptivität manifestieren lässt.

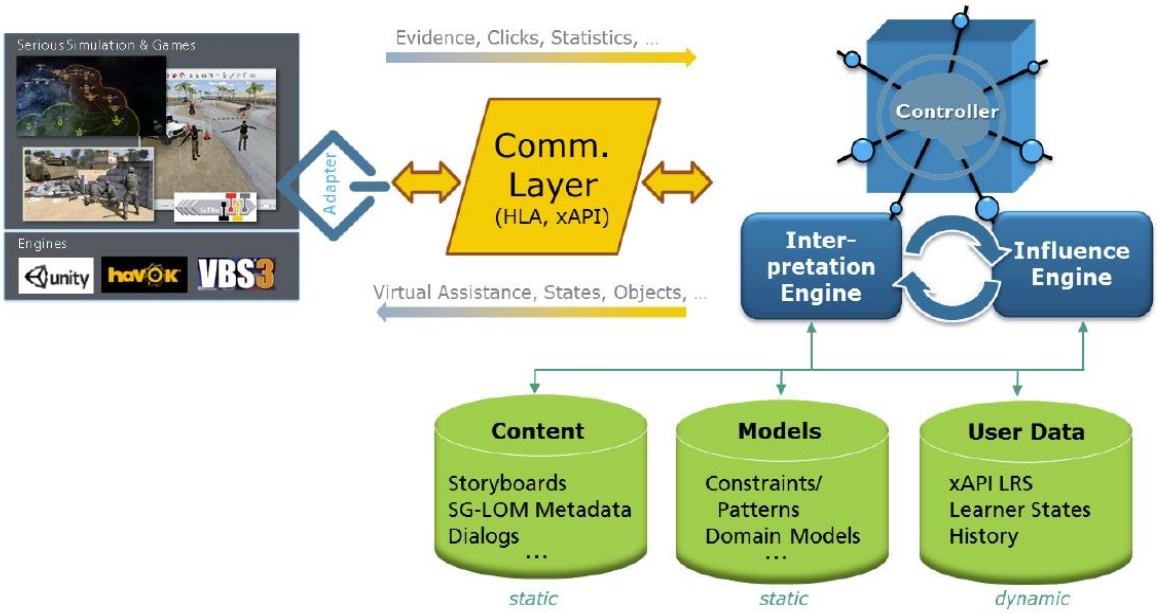


Abbildung 1.1 Grobarchitektur der ELAI (Quelle: [1])

1.4. Zielsetzung

Ziel dieser Arbeit ist es, eine Web-basierte ELAI-Nutzungsschnittstelle (ELAI-UI) zu entwickeln, die über die ELAI an ein existierendes Serious Game des Genres Seek & Find angeschlossen wird. Die ELAI-UI soll dabei die Funktionalität einer Analyse der anfallenden Nutzerdaten sowie die Möglichkeit zur Einflussnahme in das Spiel bieten. Durch die Nutzungsschnittstelle sollen die Entscheidungen der ELAI transparent und die Auswirkungen der Adaption durch Parameterkontrolle und die Möglichkeit zur manuellen Einflussnahme besser sichtbar gemacht werden. Zudem soll eine einfache Umsetzung der ELAI-Architektur für den Zugriff auf die Nutzungsinteraktionsdaten entwickelt sowie die Einflussnahme auf ein Serious Game durch z.B. einen virtuellen Agenten, der angepasste Hinweise in Textform gibt, realisiert werden. Zur Evaluation der ELAI-UI soll ein Anwendungsszenario konstruiert und untersucht sowie eine Studie zur Adaption mit einem prototypischen Serious Game zur Evaluierung der Funktionsweise der ELAI durchgeführt werden.

1.5. Aufbau und Gliederung

Im Folgenden wird der Aufbau der Arbeit erläutert.

Kapitel 1

Kapitel eins führt den Leser zum Thema der Arbeit über eine Motivation ein und beschreibt das Arbeitsumfeld sowie die Aufgaben, die Problemstellung und die Ziele dieser Master-Thesis.

Kapitel 2

Um diese Arbeit abzugrenzen, gibt Kapitel zwei einen Überblick über den aktuellen Stand der Forschung und Technik. Dabei liegt der Fokus vor allem auf den Bereichen Analyse von Daten sowie Adaption und Interoperabilität von Educational Serious Games.

Kapitel 3

Das folgende Kapitel erläutert unter anderem grundlegende Begriffe wie Serious Game und Adaption. Weiter werden auf verschiedene interoperable Kommunikationsmöglichkeiten von E-Learning Systemen eingegangen sowie auf das verwendete Web-Framework und dessen grundlegenden Konzepte. Zuletzt gibt dieses Kapitel einen Einblick in spezielle Entwurfsmuster, welche in dieser Arbeit ihren Einsatz finden.

Kapitel 4

Kapitel vier behandelt das entwickelte System und dessen Entwicklungsschritte. Dies beinhaltet die E-Learning A.I. (ELAI) und dessen Kommunikationsschnittstellen sowie die Entwicklung der Webschnittstelle (ELAI-UI).

Kapitel 5

Gegenstand des fünften Kapitels ist die Durchführung eines experimentellen Anwendungsszenarios eines Tutors im Umgang mit der implementierten ELAI-UI. Der Abschnitt schließt mit einer Auswertung und kritischen Prüfung der ELAI-UI-Nutzbarkeit.

Kapitel 6

Gegenstand des fünften Kapitels ist die Beschreibung der grundlegenden Erweiterungen des SaFIR Serious Games sowie die Ausführung einer wissenschaftlichen Studie mit Probanden. Diese haben mehrere Aufgaben des SaFIR Serious Games durchlaufen und einen Fragebogen ausgefüllt. Am Ende des Kapitels werden die Beobachtung sowie die Auswertung der Studie diskutiert.

Kapitel 7

Der letzte Abschnitt fasst die Arbeit zusammen und liefert einen Ausblick auf weitere mögliche Forschungen, die sich speziell im Hinblick auf die Erweiterung der ELAI und der ELAI-UI ergeben.

Kapitel 2

Stand der Forschung und Technik

Durch die Fokussierung der Arbeit auf die Bereiche Analyse und Adaption von Daten wird in diesem Abschnitt ein Ausschnitt der aktuellen Konzepte und Technologien dargelegt und mit dem Einsatz in dieser Arbeit verglichen.

2.1. Analyse

Die Analyse von Daten ist in den verschiedensten Bereichen eine wichtige Funktionalität, um den Einsatz eines Systems bewerten und gegebenenfalls anpassen zu können. Ein Beispiel hierfür bietet Google mithilfe des Tools *Google Analytics* [9], welches ermöglicht, eine Datenverkehrsanalyse von Webseiten zu erstellen. Mit diesem können Inhaber der Seite untersuchen, wie Nutzer ihre Webseite verwenden und wie sie auf diese hingeleitet wurden. Weiterhin werden Tools für die Erstellung von individuellen Berichten bereitgestellt, welche Webseiten analysieren, um darauf aufbauend diese für Nutzer noch attraktiver zu gestalten.

Bezieht man die Analyse von Daten auf den Bereich des digitalen Lernens, werden oft so genannte *Learning Management Systeme* (LMS) eingesetzt, um eine Lehr- und Lernumgebung bereit zu stellen [10]. Zu den bekanntesten und meist genutzten Systemen gehören dabei Moodle, Edmondo und Blackboard [11]. Solche Systeme stellen unter anderem Lernressourcen bereit und bieten die Möglichkeit administrativer Tätigkeiten innerhalb eines Kurses sowie meist auch die Möglichkeit einer Kommunikation zwischen den einzelnen Teilnehmern. Bei einem Einsatz solcher Systeme wird eine Vielzahl an Daten generiert, die richtig eingesetzt zu einer Lernsteigerung führen können [10]. Zwei Forschungsbereiche, die sich mit der Analyse solcher Daten beschäftigen, sind *Learning Analytics and Knowledge* (LAK) und *Educational Data Mining* (EDM). Beide Bereiche befassen sich mit der Auswertung großer Datenmengen und besitzen starke thematische Überschneidungen [10][12]. LAK beschäftigt sich hauptsächlich damit, den Lernprozess zu verstehen, zu visualisieren und zu verbessern. Im Gegensatz dazu fokussiert der EDM-Ansatz die automatische Erkennung von Mustern, um so Entscheidungen zu treffen. Neuhold beschreibt in seiner Arbeit weiter, dass sich die beiden Bereiche auch im Kontext der Adaption und Personalisierung unterscheiden [10]. LAK nutzt die gewonnen Erkenntnisse der Daten um Lehrende und Lernende zu informieren, wohingegen EDM großen Wert auf die Erstellung von Modellen legt, um darauf basierend Entscheidungen zu treffen.

Ein Tool, das sich im Bereich LAK an Nutzerdaten bedient und sich dabei speziell auf die Lehrenden fokussiert, um deren Lehrprozess und Lernmethoden zu evaluieren, ist *eLAT* (Exploratory Learning Analytics Toolkit) [13]. Mit dem in der RWTH Aachen entwickelten Tool, können Lehrende auf einer grafischen Oberfläche, ähnlich einem Dashboard, zur Analyse Ansichten betrachten, die das Nutzerverhalten im LMS darstellen. Wichtige Faktoren der

Applikation sind nach Dyckhoff et al. das Aktivitätsverhalten, die Anzahl eingeloggter Nutzer, Aktivitätsbereiche, Top 10 Ressourcen, Verwendung des Forums und die Einführungszeit [13]. In Abbildung 2.2 sind solche Ansichten dargestellt. Sie sollen dem Tutor helfen, die Fähigkeiten der Nutzer einzuschätzen und so seinen Lehrprozess bzw. seine Lehrmethode anzupassen.

Eine Arbeit, die sich ähnlich der Vorliegenden auf eine webbasierte Analyse von Daten fokussiert, ist *Grockit* [14]. Die 2011 von Bader-Natal und Lotze entwickelte Lernplattform baut auf einer analytischen Pipeline auf, die aus Datensammlung, Datenselektion, Datenanalyse sowie Darstellung und Verteilung von Daten besteht. Ziel der Datensammlung ist die Sammlung relevanter Nutzerdaten. Diese werden in einem weiteren Schritt mit Hilfe von SQL-Anfragen selektiert sowie durch das Datenanalyse-Paket R analysiert und dargestellt. Durch diesen Framework-Aufbau kann leicht eine neue Ansicht erstellt werden. Bereits durch das Hinzufügen einer SQL-Datei für die Datenselektion sowie einer R-Datei kann eine neue Ansicht generiert werden. Da sich immer mehr Lehrende und Lernende für Ihre produzierten Daten interessierten, wurden viele Statistiken und Analysedaten den Benutzern zugänglich gemacht. In Abbildung 2.1 ist eine Ansicht für Studenten dargestellt, welche den Zugriff auf einen einfachen Report über deren Accuracy bzgl. verschiedener Fähigkeiten visualisiert. Hierbei können die Studenten nicht nur zu jeder Zeit analytische Ergebnisse betrachten, sondern diese aus einer Kombination von 35 verschiedenen Faktoren zusammenstellen.

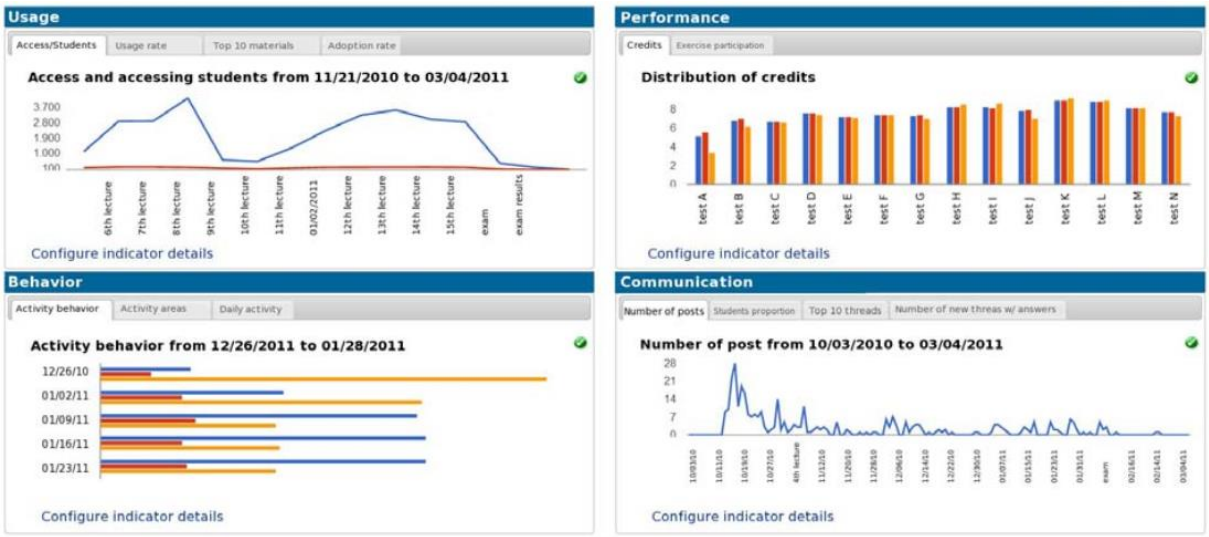


Abbildung 2.2 Monitoring Ansichten der eLAT-UI um Nutzerdaten zu analysieren (Quelle: [13])

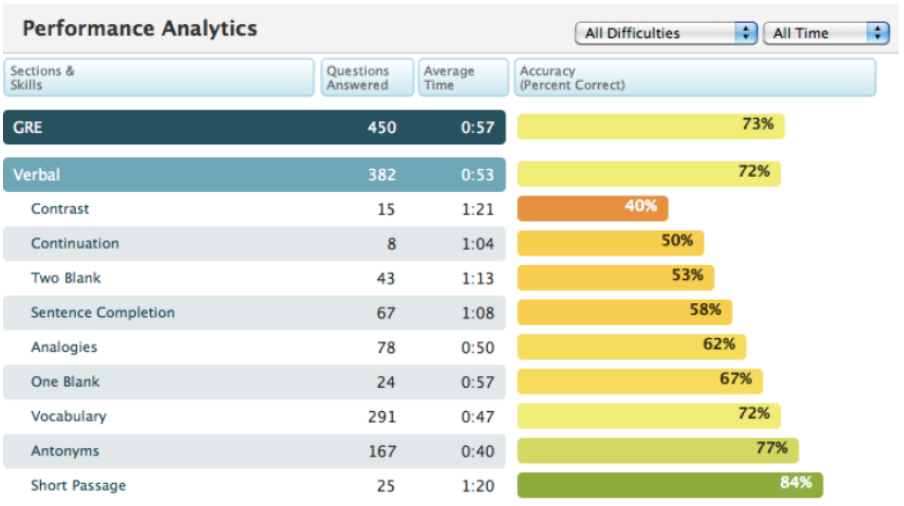


Abbildung 2.1 Grockit Ansicht einer Performance Analyse für Studenten (Quelle: [14])

Die bisher betrachteten Systeme bieten zwar eine Analysesicht für Nutzer und Tutoren an, weisen aber zum ELAI Konzept einen fundamentalen Unterschied auf. Sowohl eLAT als auch Grockit sind alleinstehende Systeme, die nicht mit anderen kooperieren. Im Vergleich dazu bietet die ELAI-UI die Möglichkeit, Daten aus verschiedensten Systemen zu analysieren und bei gleichem Themengebiet diese zusammen zu fassen. Dies wird ermöglicht durch das Konzept einer gemeinsamen Kommunikationssprache.

Auf ein solches Konzept setzt auch die amerikanische Firma Instancy Inc., welches das kostenpflichtige System *iLRS* anbietet [15]. Dieses ermöglicht, Daten aus verschiedenen Quellen darzustellen und zu vergleichen (Abbildung 2.3). Hierfür verwenden sie ein LMS, das verschiedene Optionen anbietet, die automatisch eine Analyse von Lerner-Logins, Registrierungen und Lerner-Aktivitäten ermöglichen. Durch den Einsatz eines gemeinsamen Learning Record Stores (LRS) und der Experience-API (xAPI) als Datenformat ist es dem System möglich, verschiedenste E-Learning Daten auszutauschen und Nutzer miteinander zu vergleichen. Da das System weiterhin Cloud basierte Applikationen anbietet, können die Aktivitäten zu jeder Zeit gespeichert werden und sind unabhängig davon, ob sich der Inhalt innerhalb oder außerhalb der Instancy Plattform befindet. Zur Analyse dieser Daten stellt Instancy Inc. eine Vielzahl an Dashboards bereit, mit welchen spezifische Module und Tools gewählt werden können, um Lern- und Business Perfomance Metriken darzustellen. Beispielweise hilft das Reporting-Feature von iLRS dem Lernenden dabei, seinen Lernfortschritt im Vergleich zu anderen Mitgliedern seiner Gruppe zu vergleichen und Reports über individuelle Personen, Gruppen, Kurse oder Lernressourcen darzustellen. Zur Analyse können die Daten gefiltert, sortiert und auch exportiert werden.

Diese Ausarbeitung fokussiert die Aspekte der Analyse und Adaption, worin auch ein wesentlicher Unterschied zum vorgestellten iLRS-System von Instancy Inc. liegt. Dieses beschränkt sich auf die Analyse der Daten und ermöglicht keine Adaption der Systeme basierend auf den Ergebnissen.

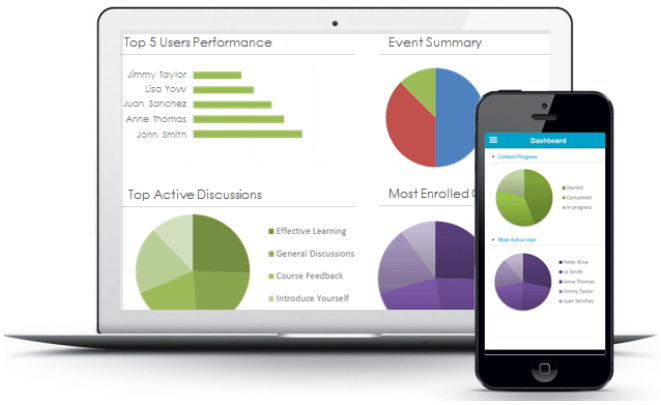


Abbildung 2.3 Instancy Dashboard (iLRS) zur Visualisierung von Nutzeraktivitäten auf dem PC und Smartphone (Quelle: [15])

2.2. Adaption

Die Adaption und Personalisierung von Anwendungen hat eine lange Tradition. Schon seit den 1970er Jahren existiert das Feld der intelligenten Tutoring Systeme (ITS), welche adaptiv in Lernsysteme eingreifen und so gezielteres Lernen durch die Bezugnahme des aktuellen Wissenstands des Nutzers ermöglichen. Ein aktuelles Beispiel bietet das auf Flowcharts basierende ITS namens *FITS* [16]. Dieses hat das Ziel, die Problemlösungsfähigkeit für Programmieranfänger in C++ durch Flowchart basierte Multi-Agenten Systeme mit Bayesian

Technologie zu erhöhen. FITS soll Studenten, im Vergleich zum sequentiellen Nacharbeiten von Inhalten auf statischen Webseiten, besser und dynamischer in neue Programmiersprachen einführen. Dabei soll das Bayes Netzwerk das Wissen des Nutzers darstellen und durch eine eingebaute Text-to-Flowchart Konvertierung diesem ein besser angepasstes Lernerlebnis bieten. In einer anderen Arbeit wird das System *BioWorld* erläutert [17]. In diesem sollen Medizinstudenten virtuelle Patienten diagnostizieren. Mithilfe eines Hidden-Markov-Modells werden dabei die Nutzer-Interaktionsdaten analysiert, um Lernmodelle zu extrahieren und daraufhin adaptiv auf das Verhalten des Studenten einzugehen.

Betrachtet man die Adaption in einem breiteren Spektrum als die vorgestellten adaptiven Lernsysteme, wird von der Spielindustrie oft der ausgedehnte Begriff künstliche Intelligenz (KI) verwendet. Dieser wird dabei von einer einfachen Anpassung der Spielgegner mit vordefinierten einstellbaren Schwierigkeitsgraden bis hin zur komplexen Adaption während des Spiels eingesetzt. Bei komplexen Adaptionen werden typischerweise Machine-Learning Algorithmen implementiert, da diese je nach Tatsachen, Lage oder Umstände schnell und zuverlässig das Spiel dynamisch anpassen können [18]. Dabei ist unter anderem die Adaption der Game AI bzw. der AI der NPCs [19][20], die Szenarien und Aufgaben [21][22], die Story [23][24], die Spielwelt [25][26] oder auch die Spielmechanik [27] möglich [28]. Betrachtet man zum Beispiel die Anpassung der AI bezogen auf das Verhalten der NPCs, kann diese genutzt werden, um adaptiv auf die Spielersituation zu reagieren und somit dem Spieler ein persönlicheres Spielerlebnis zu bieten. *Pro Evolution Soccer 08* adaptiert mit dem entwickelten ‚Teamvision' System die Taktik des gegnerischen Teams anhand des Spielerstils [27]. Wie Valve's Spieleserie *Left 4 Dead* [29] zeigt, kann auch die Spielwelt dem Spieler angepasst werden. In diesem wird durch ein AI-Direktor das Spielverhalten analysiert und verschiedene Events wie das adaptive Erzeugen von Gegnern ausgelöst [28]. Quantic Dream stellt in ihrem Spiel *Heavy Rain* [30] ein System vor, welches die spezifischen Entscheidungen des Spielers basierend auf einem psychologischen Modell analysiert und darauf mit einer Anpassung der Story reagiert [28]. Weiterhin kann die Spielmechanik eines Spiels, wie zum Beispiel die Aktionen Rennen, Schwimmen, Schießen oder Fahren für die Adaption genutzt werden. Im Action Shooter *Max Payne [31]* wird der Schießassistent über die Fähigkeiten des Spielers adaptiert und somit der Schießschwierigkeitslevel verändert [28]. Eine solche Technik der Anpassung des Schwierigkeitslevels, basierend auf der aktuellen Spielerperformanz, wird auch als *Dynamic Difficulty Adjustment* oder *Dynamic Game Balancing* bezeichnet.

Dynamic Difficulty Adjustment (DDA) beschreibt die automatische Anpassung des Schwierigkeitslevels an die aktuellen Fähigkeiten des Spielers. Das Konzept von DDA ist Genre-unabhängig. Die Anpassung kann dabei an verschiedenen Stellschrauben des Spiels geschehen. Diese können sich unter anderem in Form der Anpassung der Geschwindigkeit oder Gesundheit der Gegner, der Anzahl an Gegnern, der Energie des Spielers bzw. des Gegners oder der Länge der Spielerfahrung zeigen. Eine traditionelle Umsetzung ist dabei das Dynamic Scripting oder das Rubber Banding Verfahren [19]. *Dynamic Scripting* ist eine Machine-Learning Technik, die durch stochastische Optimierung charakterisiert ist. Dynamic Scripting beruht auf mehreren Regeln, die jeder Agent enthält. Wird ein neuer Agent erstellt, werden die Regeln genutzt, um ein neues Skript zu generieren, welches das Verhalten eines Agenten beeinflusst. Die Wahrscheinlichkeit, dass eine Regel für ein neues Skript genutzt wird, wird über spezifische Gewichte pro Regel entschieden. Ziel des Dynamic Scripting Verfahrens ist, die Gewichte an die aktuelle Fähigkeit des Spielers zu adaptieren und somit für jeden Agent on-the-fly ein individuelles Verhalten (Game AI) zu generieren (Abbildung 2.4) [19]. Eine beispielhafte Umsetzung dieses Verfahrens bieten Spronck et al. in dem Rollenspiel *Neverwinter Nights,* bei welchem sie die Schwierigkeit des Kampfgegners adaptierten. Das *Rubber Banding* Verfahren ist eine Balancing-Technik, die meist bei Renn- oder Sporttiteln eingesetzt wird. Bei dieser Methode werden die Fähigkeiten des Non-Player-Characters (NPCs) an die persönlichen Fähigkeiten derart angepasst, dass diese im Falle eines gut

platzierten Spielers verbessert und im Falle eines niedrig platzierten Spielers verschlechtert werden. Ein Beispiel für eine derartige Adaption liefert der Spieleklassiker *Mario Kart 64,* bei welchem die Geschwindigkeit der NPCs sowie die Eignung der Popups in Abhängigkeit von der Spielerplatzierung angepasst werden.

Adaptivität findet allerdings nicht nur im reinen Unterhaltungsbereich, sondern auch im Bereich der Serious Games Anwendungen, wie unter anderem Johnson et al. zeigen. Diese stellen 2004 das ernsthafte, taktische Sprach- und Kulturtrainingsspiel *DARWARS* vor [32]. Das für den militärischen Einsatzzweck entwickelte Serious Game soll es ermöglichen, die kommunikativen Fähigkeiten des Nutzers wie Fremdsprachen oder den Kontakt zu anderen Kulturen zu trainieren. Die Lerner praktizieren dabei ihre Fähigkeiten in einem rekonstruierten, simulativen Dorf der Nachkriegszeit, in welchem die Beziehung zu den lokalen Einwohnern vertieft werden sollen. Hierbei hilft dem Lernenden ein intelligentes Tutorien-System durch adaptive Vorschläge und Verbesserungen der Kommunikation. Ein weiteres Beispiel bietet das von der EU geförderte Projekt *ELEKTRA* aus dem Jahr 2008 (www.elektra-projekt.org). In diesem 3D-Adventure-Rollenspiel sollten 13 bis 15 jährige Schüler die optische Physik mithilfe eines Spiels erlernen. Dabei spielten sie die Rolle von Goerge, des Enkels eines gefangen genommenen Wissenschaftlers. Mithilfe des als Geist auftretenden Galileo Galilei muss der Spieler physikalische Aufgaben lösen, um seinen Onkel zu retten [33]. Ziel des EU-Projektes war es, die Vorteile von Computerspielen und deren Design-Prinzipien anzuwenden, um ein adaptives Bildungsspiel zu entwickeln. Dafür arbeiteten mehrere Disziplinen wie Pädagogen, Kognitionswissenschaftler, Neurowissenschaftler und Informatiker zusammen [34].

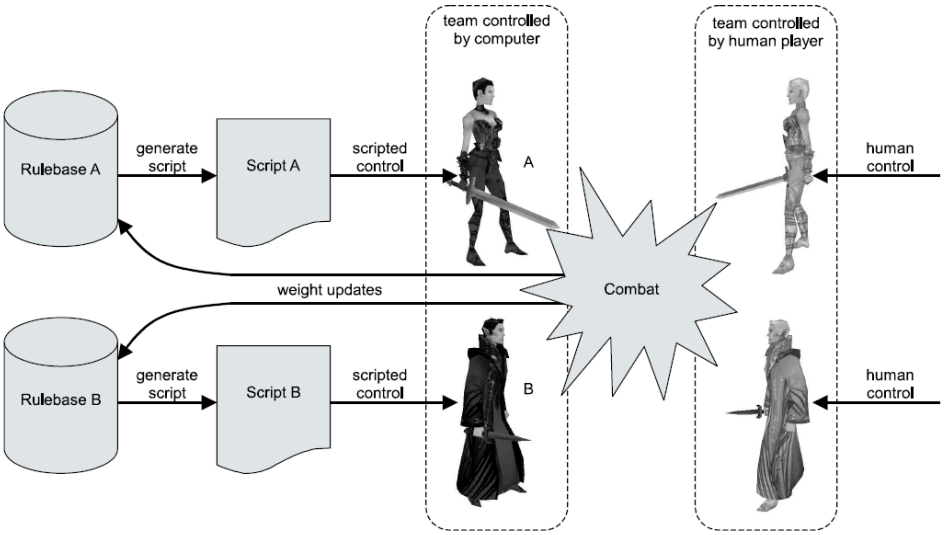


Abbildung 2.4 Dynamic Scripting Prinzip (Quelle: [19])

Eine Adaption des Adventure Serious Games ELEKTRA zeigen Peirce et al. in ihrer Arbeit [33]. In dieser beschreiben sie eine Architektur, mit der es möglich ist, ein Spiel nicht-invasiv zu adaptieren. Die vorgestellte *ALIGN* Architektur entkoppelt die Adaptionslogik von der Spiellogik und ist somit universeller einsetzbar. Sie besteht dabei aus den vier konzeptionellen Aspekten (Abbildung 2.5): (1) Sammeln von Kontextinformationen über den Spielstatus in der ‚Evidence Interpretation Engine', (2) Interpretation und Speicherung der gewonnen Erfahrungen, (3) Verfeinerung der Interventionsbedingungen und zuletzt die (4) Anwendung der Adaptionsregeln des Spiels basierend auf der ‚Recommendation-Engine' [33]. Eine weitere Architektur zur Adaption von Serious Games bietet *ISAT* [35]. Die für den militärischen Zweck entworfene interaktive Geschichten erzählende Architektur soll es ermöglichen, dem Trainierenden eine individualisierte Trainingsumgebung zu bieten. Dabei ist der ‚Director Agent' die zentrale Komponente, da dieser für die Einflussnahme und Individualisierung zuständig ist (Abbildung 2.6). Zur Einflussnahme beobachtet der ‚Director Agent' den Trainierenden während der Lernerfahrung und erzeugt ein Lernermodel. Diese Informationen werden genutzt, um das Szenario auf die Fähigkeiten und das Wissen des Trainierenden anzupassen. Weitere verteilte Architekturen für das Adaptieren von Serious Games oder Simulationen im militärischen Einsatzweck für intelligente virtuelle Agenten sind *TENA* [36] oder *CIGA* [37].

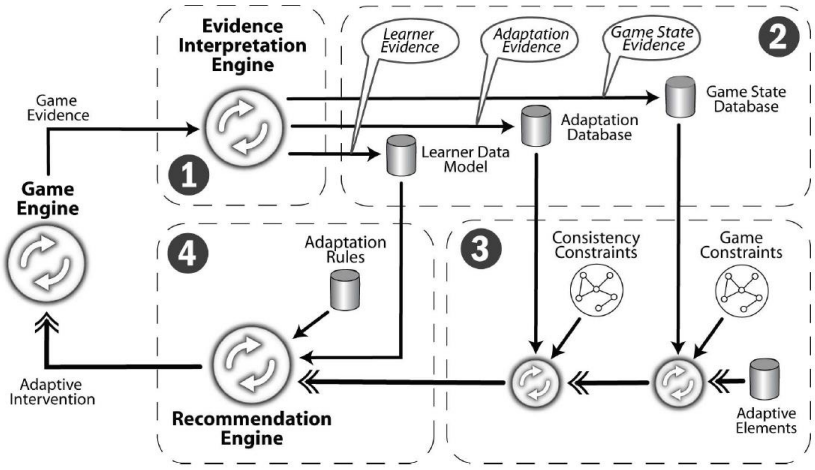


Abbildung 2.5 ALIGN Architektur (Quelle: [33])

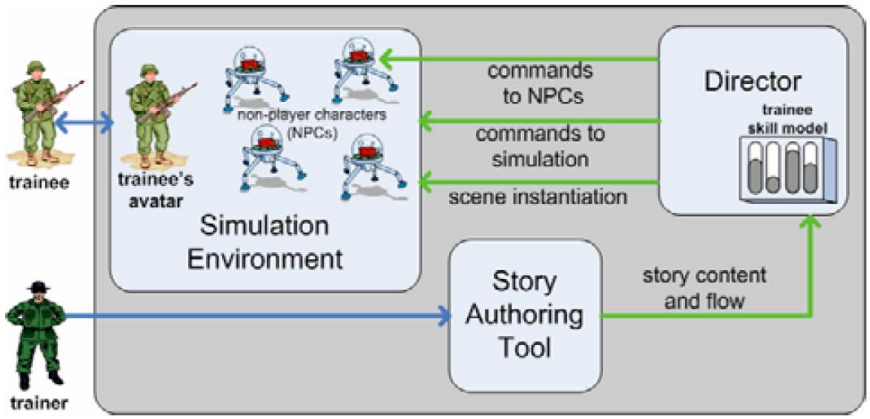


Abbildung 2.6 ISAT Architektur (Quelle: [35])

Kapitel 3
Grundlegende Begriffe und Abgrenzungen

Die folgenden Abschnitte liefern das grundlegende Verständnis für diese Arbeit. Zuerst wird auf den Begriff Serious Game und die Abgrenzungen zu verwandten Genres eingegangen sowie der Begriff der Adaption im Kontext des E-Learning erläutert. Um eine Adaption von Serious Games zu ermöglichen, ist es wichtig, Daten zwischen den Spielen und dem Adaptionssystemen austauschen zu können. Weiter werden die in dieser Arbeit verwendeten Web-Technologien vorgestellt. Zuletzt folgt eine Erklärung verwendeter Entwurfsmuster.

3.1. Serious Game und Abgrenzungen

Serious Games (Ernsthafte Spiele) beschreibt die Kombination eines Spiels mit ernsthaften Themen. Auf den ersten Blick scheinen die beiden Begriffe *ernst* und *Spiel* diametral zueinander. Beobachtet man allerdings, mit welcher Konzentration Computerspieler mehrere Stunden dem Geschehen des Spiels auf dem Bildschirm folgen und dabei mit großer Euphorie zum Teil knifflige Aufgaben lösen, wird deutlich, warum seit einigen Jahren die Wissenschaft versucht, dieses Potenzial für ernsthafte Aufgaben zu nutzen [38].

Das erste Mal wurde der Begriff **Serious Game** von Clark C. Abt in seiner gleichnamigen Publikation 1970 genannt [10]. Allerdings bezog er sich damals auf analoge Brett-, Karten- und Rollenspiele. Erst im Jahr 2002 wurde dieser Begriff durch die Veröffentlichung des Computerspieles *Americas Army*[1] und der Gründung der *Serious Game Initiative* von Ben Sawyer und David Rejeski erneut aufgeworfen und digitale Spiele in die Terminologie mit einbezogen [6]. Da der Begriff ein breites Spektrum von digitalen Lernspielen umfasst, ist es schwierig, eine genaue Definition von Serious Games zu erhalten, weshalb noch keine einheitliche Definition existiert [38]. Dennoch werden im Folgenden drei Definitionen zitiert, die die Idee des Begriffes vermitteln sollen:

> *“A serious game is a game in which education (in its various forms) is the primary goal, rather than entertainment.” [4]*

> *“Serious games are games that are designed to entertain players as they educate, train or change behaviour.“ [40]*

[1] Americas Army ist ein kostenloses First-Person-Shooter Online-Spiel, das als Rekrutierungswerkzeug der US-Armee konzipiert wurde. „Seit der Einführung des Spiels konnte ein Anstieg der Rekrutierungszahlen der US-Armee verzeichnet werden“. [39]

"The purpose of a serious game is to get users to interact with an IT application that combines aspects of tutoring, teaching, training, communications and information, with a recreational element and/or technology derived from video games. This combination aims to make practical, useful content (serious) enjoyable (game). It is achieved by developing scenarios that are at once practical and enjoyable." [41]

Fasst man nun die Gemeinsamkeiten der Definitionen zusammen, so lässt sich festhalten, dass unter Serious Game ein digitales Spiel verstanden wird, das der Bildung und Unterhaltung dienen sollte. Wie eingangs erwähnt, lässt die nicht festgelegte Definition Spielräume in der Wahl der Begrifflichkeit zu. Um den Begriff Serious Game zu ähnlichen Begriffen im Rahmen von digitalisiertem Lernen abzugrenzen und besser zu verstehen, werden im Folgenden die Begriffe E-Learning, Edutainment und Simulation erläutert.

Ein Begriff, den man häufig in der Literatur findet, ist der Begriff **E-Learning** (Electric-Learning). Unter E-Learning wird laut Johannesson et al. und Doujak das generelle Konzept des Lehren und Lernens mithilfe digitaler Medien verstanden [5][42]. Darunter fallen dementsprechend auch Serious Games (nach obiger Definition). Allerdings beinhaltet dieser nicht nur didaktische Konzepte, sondern beispielsweise auch den Bereich *Motion-based Games for Health*, bei welchem unter anderem der Einsatz von Videospielen genutzt wird, um Bewegung zu fördern. Somit sind nicht alle Serious Games unter dem Begriff E-Learning einzuordnen, sondern nur diejenigen, welche das direkte Ziel verfolgen, Lerninhalte zu vermitteln, welche spezifischer als **Educational Serious Games** bezeichnet werden. In dieser Ausarbeitung wird der Begriff Serious Game nur im Rahmen von Educational Serious Game verwendet, weshalb diese im Folgenden synonym verwendet werden.

Ein weiterer Begriff, den man häufig in der Literatur findet, ist **Edutainment**. Dieser setzt sich aus den Wörtern Education (Bildung, Unterricht) und Entertainment (Unterhaltung) zusammen und bezieht sich – ähnlich zu Serious Games – darauf, über multimediale Lernumgebungen Wissen zu vermitteln. Der Fokus liegt dabei vor allem auf der Vermittlung und Weiterbildung von schulischen Themen wie Mathematik, Physik, Chemie oder Sprachen und somit auch auf ein eingeschränktes Alter. Im Kontrast dazu sind Serious Games für alle Nutzergruppen geeignet und fokussieren das Lernen von neuen Fähigkeiten und Kompetenzen. Der größte Unterschied der beiden Branchen ist allerdings in dem Zusammenspiel von Lernen und Unterhaltung zu sehen. Während bei Edutainment-Titeln die Unterhaltung als Belohnung für das erlernte Wissen eingesetzt wird, sind bei Serious Games die Lerninhalte und Aufgaben integrativer Bestandteil des Spiels. Dies hat zur Folge, dass laut Lampert et al. bei Edutainment-Titeln das Lernen oftmals mit einer fehlenden Motivation und Dynamik einhergeht, wohingegen Serious Games über die Eigenschaften eines kommerziellen Computerspiels verfügen und so z.B. einen hohen Motivationsgrad und ausdifferenzierte Spielumgebungen bieten [39].

Ein weiterer Begriff, den es gegenüber Serious Games abzugrenzen gilt, sind Simulatoren. Ein **Simulator** wird entwickelt, um in möglichst realistischer Spielumgebung Fähigkeiten zu trainieren. Er legt dabei keinen Wert auf die Unterhaltung, was auch den Hauptunterschied zu Serious Games darstellt. Denn bei diesen sollte idealerweise ein Gleichgewicht aus dem Spielerlebnis und dem zu lernenden Inhalt eingehalten werden. Dabei muss das Spiel selbst nicht zwingendermaßen realitätstreu wie bei Simulationen dargestellt werden. Ein gutes Beispiel zur Abgrenzung von Simulationen liefert das surreale Serious Game *Re-Mission* von HopeLap [43]. In dem auf krebskranke Kinder und Jugendliche abgestimmten Spiel steuert der Spieler den Kampfroboter Roxxi durch das Innere des menschlichen Körpers, um bösartige Zellen mit dem Chemo-Blaster abzuschießen. Dabei treten in den verschiedenen Levels unterschiedliche Arten von Krebs in den Vordergrund, die jeweils mit verschiedener Munition bzw. Medikamenten bekämpft werden müssen. Ziel des Spiels ist es, dem Spieler

die Krankheit und deren Symptome verständlicher zu machen und die Wichtigkeit der Einnahme von Medikamenten zu vermitteln. Es wird deutlich, dass ein solches Spiel im Vergleich zu einer möglichst realitätstreuen Simulation große Unterschiede aufweist. Zur Veranschaulichung des Unterschiedes zwischen den beiden Branchen ist in Abbildung 3.1.a das surreale Serious Game *Re-Mission* und in Abbildung 3.1.b die möglichst realitätsgetreue Flugsimulation *FlightAwaySimulation* dargestellt. Ein weiterer Unterschied, der beide Begriffe weiter abgrenzt, sind die Darstellungsmedien. Eine Simulation wird meist am Computer oder in speziell dafür angefertigten Umgebungen eingesetzt, wohingegen ein Serious Game für weit verbreitete Medien wie Computer, Spielekonsolen oder Smartphones entwickelt werden [44].

Abbildung 3.1.a (Surreales) Serious Game: Re-Mission (Quelle: [43])

Abbildung 3.1.b (Realitätsgetreue) Simulation: FlightAwaySimulation (Quelle: www.Flyaway simulation.com)

Abbildung 3.1 Vergleich surreales Serious Game (links) und realitätsgetreue Simulation (rechts)

3.2. Adaption

Ein wichtiger Faktor in der Entwicklung von Computerspielen und Serious Games ist die Gestaltung der spielerischen Inhalte. Diese sollen aus motivationspsychologischer Sicht möglichst attraktiv und langfristig motivierend sein. Ist dies der Fall, bietet das Spiel meist auch den Vorteil des emotionalen Involvements, welches ein wesentliches pädagogisches Element von Serious Games darstellt [45]. Es ist der Zustand des Eintauchens des Spielers in die virtuelle Welt, was auch als *Immersion* bezeichnet wird [5]. Der Spieler dringt dabei komplett in das Spiel ein. Die reale Welt wird vorübergehend vergessen und die künstliche Welt des Spiels wird als realistisch empfunden. Dies führt dazu, dass der Spieler nicht von Außeneinflüssen abgelenkt wird und seine Aufmerksamkeit und Aktionen alleine auf das Spiel konzentriert. Man spricht dabei von dem sogenannten *Flow-Zustand*. Dieser beschreibt laut Doujak „das ultimative Stadium des Eintauchens“, was im Vergleich zu traditionellen Lernmitteln zu einer höheren Motivation und einem stärkeren emotionalen Involvement führt [5][45]. Dabei ist es für das Serious Game besonders wichtig, den Schwierigkeitsgrad des Spiels den Fähigkeiten des Spielers anzupassen. Stehen die Anforderungen des Spiels und die Fähigkeiten des Spielers im Gleichgewicht, sind die Voraussetzungen, in einen Flow-Zustand zu geraten, optimal [5]. Wie Abbildung 3.2 zeigt, resultiert ein Abweichen des Flow-Zustandes einerseits durch eine Überforderung in Frustration und andererseits durch eine Unterforderung in Langeweile. Um einen solchen Zustand bei den Spielern eines Serious

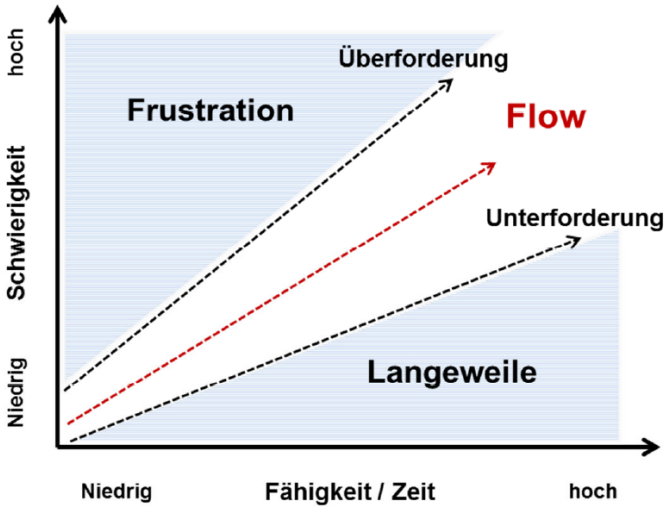


Abbildung 3.2 Flow Zustand (Quelle: [5])

Games zu erreichen, ist es erforderlich, das Spiel den jeweiligen Vorkenntnissen und Fähigkeiten des Spielers anzupassen bzw. zu adaptieren.

Die Adaption im Rahmen eines Serious Games kann als die Fähigkeit bezeichnet werden, den Spielverlauf abhängig von den Eigenschaften und Fähigkeiten des Lernenden zu variieren.

Während der Entwicklung eines Spiels werden die Spielinhalte, die Regeln, die Handlung und die Umgebung meist statisch implementiert, wohingegen der Spieler dynamisch mit dem Spiel interagiert. Ein solche statische Entwicklung ist weit verbreitet und erlaubt es, Spiele und Simulationen robust, testbar und kontrollierbar zu gestalten [28]. Um in diesen trotz allem eine Anpassung an den Spieler vornehmen zu können, werden oftmals Variationsmöglichkeiten abhängig von dessen Profil integriert. Beispielsweise findet man in vielen Entertainment Games die Möglichkeit einen fest definierten Schwierigkeitsgrad, wie die Auswahl zwischen den Stereotypen *Anfänger*, *Fortgeschrittener* oder *Experte* zu wählen. Diese Art der Individualisierung impliziert allerdings direkt das Fehlschlagen der Adaption von Spielern, die ihre Einstufung in die statischen Profile nicht kennen oder sich mit diesen nicht identifizieren können. Eine solche vordefinierte Variation des Spielerprofils führt nach Lopes zu einer unpersönlichen, vorhersehbaren und unflexiblen Form der Adaptivität [28]. Für Spiele, dessen primäres Ziel nicht das reine Entertainment darstellt, ist ein solch ungenaues Spielerprofil ein großes Problem. Zum einen wird ein Maximum der Lerneffizienz nur durch eine maximale Anpassung an die persönlichen Fähigkeiten und Kenntnisse erlangt und zum anderen wirkt sich diese geringe Individualität auf die Motivation und die damit einhergehende Spielwiederaufnahme aus. Die Qualität der Adaption des Spiels kann also entscheidend für den Lernerfolg sein.

Um eine Adaption bestmöglich zu gestalten, entwickelte Shute et al. den in Abbildung 3.3 dargestellten *vierstufigen Adaptionszyklusprozess* mit den Phasen Erfassung, Analyse, Auswahl und Anzeige [46].

In der ersten Phase des Adaptionszyklus, der **Erfassungsphase** (Capture), werden die Daten aus der Interaktion zwischen Mensch und Maschine erfasst. Dabei können die Informationen verschiedenster Natur sein. Es ist die Verwendung von Sensordaten aus z.B. Mikrofonen, Kameras oder Beschleunigungssensoren sowie auch Spieldaten zur Adaption möglich.

In der nachfolgenden **Analysephase** (Analyze) werden die erfassten Daten verarbeitet und aufbereitet. In dieser ‚intelligenten'-Phase wird versucht, die Semantik der Daten mithilfe von quantitativen sowie qualitativen Analysetechniken, Machine-Learning Mechanismen und Data-

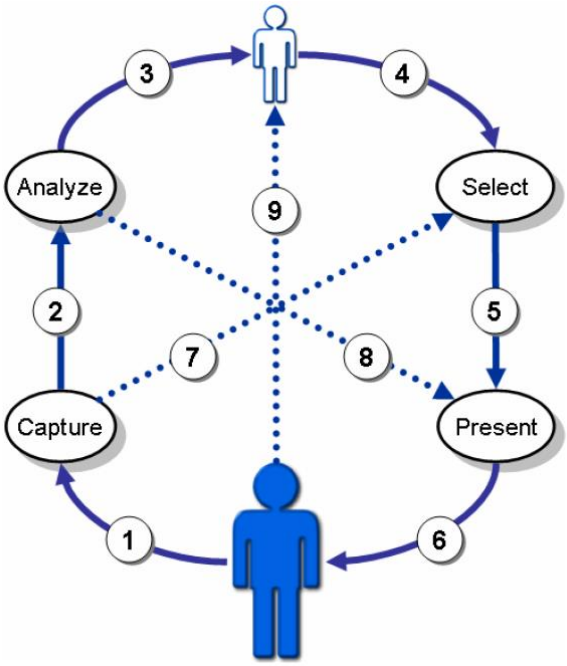


Abbildung 3.3 Vierstufiger Adaptionszyklusprozess (Quelle: [46])

Mining zu verstehen. Wenn das System zusätzlich Zugriff auf weitere Nutzerdaten besitzt, kann es durch Data-Mining Techniken wie Mustererkennung oder mithilfe von kollaborativen Filtern, Muster in den Daten erkennen und so Verhaltensanomalien identifizieren. In dieser Phase trifft man allerdings auf das aus dem Bereich Machine-Learning bekannte *Cold-Start* Problem [47]. Dieses beschreibt die Problematik, von fehlenden Daten im ersten Durchlauf des vierstufigen Adaptionszyklusprozess. Durch die fehlenden Daten ist es dem System nicht möglich, adaptiv einzugreifen, da es keine Vorkenntnisse und Vergleichsmöglichkeiten besitzt. Eine mögliche Herangehensweise ist die Umgehung durch anfängliche Nutzerabfragen. In diesem Ansatz können die ersten Adaptionen mit den Eingabedaten der Abfragen erfolgen und weitere Anpassungen anhand der entstehenden Daten des Spielverlaufs vorgenommen werden. Die verarbeiteten relevanten Informationen werden im Nutzermodell gespeichert und dienen als Basis für die nächste Phase.

Mithilfe des Nutzermodells aus der Analysephase wird in der **Auswahlphase** (Select) entschieden, wie und wann die Adaption in das Spiel intervenieren soll. Abhängig von der Stufe der Adaptionsautomatisierung (automatisch, völlig manuell oder alle Stufen dazwischen), wählt das System diejenigen Informationen des Lernermodells aus, die zum einen relevant sind und zum anderen am besten zur aktuellen Nutzersituation passen. Weiterhin wird in dieser Phase darüber entschieden, wann der beste Zeitpunkt des Eingreifens in das Spiel ist.

In der letzten Phase – der **Anzeigephase** (Present) – wird im Spiel die ausgewählte Intervention adaptiert. Dabei existieren Adaptionen, welche dem Nutzer direkt ersichtlich sind, wie z.B. ein virtueller Assistent, der weitere Informationen bereitstellt oder aber für den Spieler unsichtbare Aktionen wie z.B. das Adaptieren der Parametermodelle der Gegenspieler in einem Rennspiel. Möglichkeiten der Adaption können einfache Parameter wie die Geschwindigkeit der Gegenspieler oder auch komplexere Elemente wie der Inhalt oder die Spielmechanik sein. Lopes und Motyka zählen weiter folgende mögliche Elemente zur Adaption von Spielen auf [28][48]:

- Spielwelt und deren Objekte
- Mechanismus zur Spielsteuerung
- Non-Player-Character (NPC) und dessen künstliche Intelligenz (AI)

- Schwierigkeit der Aufgaben
- Handlungsverlauf
- Spielszenarien und Aufgaben
- Häufigkeit der Inanspruchnahme von Hinweisen und Tipps
- Darstellung von Inhalten abhängig vom Interesse des Lernenden
- Darstellung von Inhalten abhängig vom Vorwissen des Lernenden
- Aufgabengeschwindigkeit
- Präsentationdauer und Antwortdauer
- Zur Verfügung stehende Lernzeit
- Reihenfolge der Aufgaben

Der beschriebene vierstufige Adaptionsprozess beschreibt einen Zyklus, der während des Spiels durchlaufen wird. Dabei interagieren der Nutzer sowie die Adaptionslogik mit dem Spiel. Im folgenden Kapitel werden die Kommunikationsmöglichkeiten zwischen einer Adaptionslogik und Spiel vorgestellt und erläutert.

3.3. E-Learning Interoperabilität

Zur Anpassung eines Spiels an den Nutzer müssen Nutzerdaten gesammelt, analysiert und zu einem Nutzerprofil zusammengefasst werden. Um eine möglichst effiziente und großflächige Abdeckung der Fähigkeiten und Vorkenntnisse des Spielers zu erhalten, müssen Daten über mehrere Spiele und Spieler hinweg gesammelt und ausgewertet werden. Für den dafür benötigten Austausch von Daten im Bereich E-Learning existieren verschiedene Architekturen und Methoden.

Experience API

Zur Realisierung der Kommunikation zwischen verschiedenen E-Learning Systemen ist der Einsatz der Experience Application Programming Interface (xAPI) Spezifikation denkbar. xAPI ist eine Spezifikation für Lerntechnologien (z.B. LMS), die es ermöglicht, verschiedenste Daten zu sammeln und wieder abzurufen. Dabei benutzt sie ein konsistentes Format, das über verschiedenste Technologien (z.B. Webbrowser, Spiele-Engine) hinweg einsetzbar (interoperabel) ist. Entwickelt wurde xAPI, auch bekannt als Tin Can API, von Rustici Software als Teil eines Entwicklungsprojektes, beauftragt von der Advanced Distribtuted Learning (ADL) Initiative [8].

Die Experience API ist als Web Service nach den Prinzipien einer Representational State Transfer (REST) Architektur entworfen worden, die das Datenformat JavaScript Object Notation (JSON) nutzt [49] (Kapitel 3.4). Der Entwurf in REST bedeutet dabei für den Web Service, die grundlegenden HTTP-Funktionen zu nutzen und somit die Datenübertragung mit verschiedensten weltweit verteilten Diensten zu gewährleisten. Dabei müssen dessen Haupteigenschaften ressourcenbasiert und zustandslos eingehalten werden. Ressourcenbasiert bedeutet, dass die Darstellung der Daten als Ressource erfolgen und eindeutig über einen Uniform Resource Identifier (URI) definiert sein müssen. Somit kann jede Ressource direkt über verschiedene HTTP-Request Typen angesprochen werden. Die Zustandslosigkeit einer Architektur bedeutet, dass jede Nachricht die an den REST-Server geschickt wird, eindeutig sein muss und alle Informationen beinhaltet, die der Server zur Ausführung der Anfrage benötigt. Dabei wird kein Zustand gespeichert und so die Bedingung der RESTful-Web Services, dass eine Anfrage an eine Ressource über den eindeutigen URI immer das gleiche Ergebnis liefern soll, erfüllt [49].

xAPI ist eine E-Learning Software Spezifikation, mit der es möglich ist, Informationen zwischen Lernsystemen auszutauschen. Die Informationen werden dabei in Form von Statements in Learning Record Stores (LRS) gesichert. Dieser dient dabei als zentraler Speicher, der Daten empfängt, speichert und zurück an Lernsysteme sendet. Dabei ist es völlig unabhängig davon, welche Art von System mit dem LRS interagiert, da die Daten in HTTP oder HTTPS versendet werden. Die gesammelten Daten können genutzt werden für Lernanalysen, zum Erstellen von Reports oder Dashboards oder für den Austausch von Daten mit anderen Systemen. Die Daten haben jeweils die Form `Actor Verb Object` und werden als Statements bezeichnet.

Ein Statement beschreibt dabei eine Informationseinheit, die von einem Nutzer an ein LRS gesendet oder von diesem geladen wird. Das Statement muss durch die Vorgabe des RESTful Web Services zustandslos sein. Das bedeutet, dass alle Informationen über den Autor, Informationen der Aktivität und das Objekt im Statement vorhanden sein müssen [49]. Es basiert dabei auf dem Konzept von Aktivitäten (Activities), welche bekannt sind von Plattformen wie Facebook, Google+ und Twitter. Auf diesen kann der Nutzer seine Aktivitäten oder diejenigen von anderen Nutzern im zeitlichen Ablauf für Menschen gut lesbar nachvollziehen. Zusammen bilden die Aneinanderreihung von Statements so genannte Activity Streams [49]. Statements enthalten dabei mindestens die Grundform `Actor Verb Objekt` (Abbildung 3.4) [50]. Da xAPI das JSON-Format nutzt, wird ein Statement als JSON-Objekt betrachtet. Ein JSON-Objekt beinhaltet eine unsortierte Menge von [Key, Value]-Tupeln. Der Key gibt dabei den Namen der Eigenschaft an und der Value den dazugehörigen Wert. Dieser kann ein String, Integer, Array oder wieder ein JSON-Objekt sein, womit es möglich ist, Statements ineinander zu verschachteln. Ein einfaches Beispiel für ein Statement könnte sein *Peter like ‚bungee jumping'*. Auf der technischen Ebene wird das Statement im JSON Format übertragen und wird wie folgt codiert:

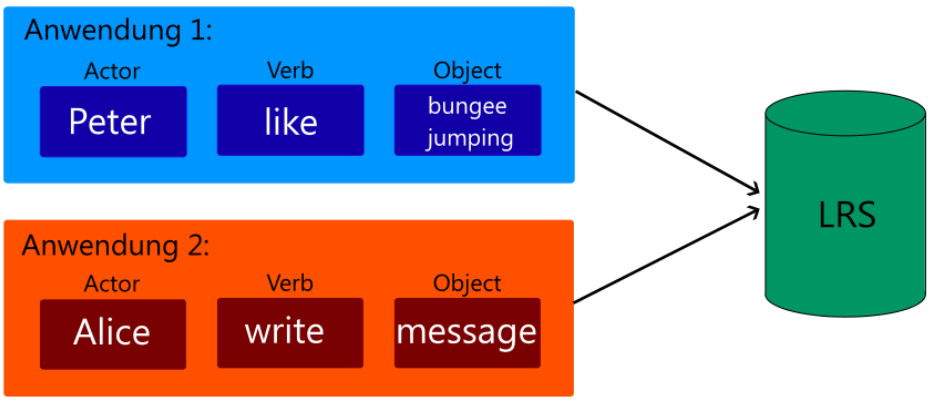


Abbildung 3.4 Einfache Statements der Form Actor Verb Object

```
{
        "actor": {
                "objectType": "Agent",
                "name": "Peter Glider",
                "mbox": "mailto:peter@example.com"
        },
        "verb": {
                "id": "http://adlnet.gov/expapi/verbs/like",
                "display": {
                        "en-US": "like",
                        "de-DE": "mögen"
                }
        },
        "object": {
                "objectType": "Activity
                "id": "http://example.com/activities/bungee_jumping",
                "definition": {
                        "name": {"en-US": "bungee jumping"},
                        "description": {
                                "en-US":"Bungee jumping s an activity that involves jumping from
                                a tall structure while connected to a large elastic cord
                                [Wikipedia]"
                        }
                }
        }
}
```

Im Folgenden wird auf die Bedeutung und den Aufbau der Grundelemente in einem Statement eingegangen und anhand des obigen Beispiels erläutert (Anhang Tabelle 6).

Der **Akteur** stellt ein Individuum in Form einer Person bzw. eines Agenten oder eines Systems dar. Zwingend erforderlich ist dabei die Angabe von mindestens einem inverse functional identifier (IFI) (Tabelle 1), um die verschiedenen Akteure voneinander unterscheiden zu können. Ein eindeutiger inverser Identifier beschreibt sich entweder durch eine eindeutige E-Mail Adresse (Eigenschaft: `mbox`), dessen Hash (Eigenschaft: `mbox_sha1sum`), eine eindeutige Zeichenfolge (Eigenschaft: `openid`) oder einem Nutzerkonto eines Systems (Eigenschaft: `account`).

Tabelle 1 Inverse Functional Identifier (IFI)

IFI	Typ	Beschreibung
mbox	mailto IRI	Format: ‚mailto:email address' Die E-Mail Adresse sollte dabei eindeutig dem Agenten zugeordnet sein
mbox_sha1sum	String	Hash des Typs mailto IRI
openid	URI	Eine openid ist ein eindeutiger Identifier des Agenten
account	Object	Ein Nutzerkonto eines existierenden Systems z.B. eines LMS oder Intranets

Optional können weiterhin die Eigenschaften `name` und `objectType` dem Akteur zugewiesen werden. Die Eigenschaft `objectType` kann als Agent oder Gruppe definiert werden. Das obige Beispiel nutzt nur einen Agenten – *Peter Glider*. Bei einer Gruppe als Akteur wird zusätzlich die Eigenschaft `member` mit einer Liste von Personen hinzugefügt. Beispielhaft wird folgend die Gruppe *TeamA* mit der Identifikation anhand der E-Mail Adresse *teamA@example.com* und den Akteuren *Andrew Downes*, *Toby Nichols* und *Ena Hills* dargestellt.

```
"actor": {
        "name": "TeamA",
        "mbox": "mailto:teamA@example.com",
        "objectType": "Group",
        "member": [{
                "name": "Andrew Downes",
                "account": {
                        "homePage": " http://www.example.com",
                        "name": "13936749"
                },
                "objectType": "Agent"
        }, {
                "name": "Toby Nichols",
                "openid": " http://toby.openid.example.org/",
                "objectType": "Agent"
        }, {
                "name": "Ena Hills",
                "mbox_sha1sum": "ebd31e95054c018b10727ccffd2ef2ec3a016",
                "objectType": "Agent"
        }]
}
```

Als zweites Grundelement eines xAPI-Statements wird das **Verb** deklariert. Es beschreibt die Aktivität der Lernerfahrung, die der Akteur mit dem Objekt durchführt. Dabei legt die xAPI Spezifikation keine bestimmten Verben fest. Ein Verb ist definiert durch die Eigenschaft `id` und wird mithilfe der optionalen Eigenschaft `display` ergänzt. Die `id` muss als Internationalized Resource Identifier (IRI) angegeben werden, was auch eine aufrufbare URL mit einem erklärenden Inhalt darstellen kann. Im obigen Beispiel enthält `id` die URL *http://adlnet.gov/expapi/verbs/like* und die für den Menschen verständlichere textuelle Darstellung *like* in dem JSON-Objekt `display`. Wie im tatsächlichen Sprachgebrauch auch kann das gleiche Verb abhängig vom Kontext verschiedene Bedeutungen haben. So kann *run* zum einen die Bedeutung haben, dass der Actor gerade rennt oder ein bestimmtes Programm ausgeführt wird. Um dieser Doppeldeutigkeit entgegen zu wirken, werden Verben meist als URL deklariert, unter welcher die Bedeutung beschrieben wird. Zusätzlich ist es möglich, durch verschiedensprachige `display` Elemente das Verb zu beschreiben wie im fortlaufenden Beispiel dargestellt.

Das letzte Grundelement eines Statements ist das **Objekt** (auch als Agent bezeichnet), auf welches sich der Akteur bezieht. Dies könnte wie im Beispiel eine Aktivität sein und wird mit der Eigenschaft `id` eindeutig identifiziert sowie mit den Eigenschaften `objectType` und `definition` erweitert. Allerdings besteht auch die Möglichkeit, das Objekt als Akteur (*Peter love Lora*) oder wiederum als Statement (*Peter love ‚Lora like Books'*) zu initialisieren. Komplexere Aussagen können getätigt werden, wenn man als Objekt die Referenz-ID eines

anderen Statements angibt. So können Aktivitäten bewertet, kommentiert oder allgemein markiert werden [49]. Siehe hierfür das spätere Beispiel „voided“.

Ein Statement kann weiterhin Elemente über die Grundform hinaus enthalten. Es existieren zusätzlich zu Actor, Verb und Object die Elemente Result, Context, Timestamp, Stored, Authority, Version und Attachments. Damit könnte das Tripel ‚Peter like ‚bungee jumping‘‘ mit dem folgenden Code um folgende Informationen erweitert werden:

‚Peter like ‚bungee jumping’ with intensity of 0.6. He made and store this statement at 2016.05.22. The data are verified by the agent ‘anonymous’. The statement are in version 1.0.0. Object ‘bungee jumping’ is a type of ‘sport’’.

```
{
        "actor": { … },
        "verb": { … },
        "object": { … },

        "result":{"score{“scaled”:0.6}},
        "timestamp": "2016-05-22T05:32:34.804Z",
        "stored": "2016-05-22T05:32:34.804Z",
        "authority": {
                "objectType": "Agent",
                "account": {
                        "homePage": " http://www.example.com/",
                        "name": "anonymous"
                }
        },
        "version": "1.0.0",
        "context": {
                "contextActivities": {
                        "parent": {
                                "objectType": "Activity",
                                "id": " http://example.com/activities/sport",
                                "definition": {
                                        "name": {"en-US": "Sport"}
                                }
                        }
                }
        }
}
```

Um zu gewährleisten, dass ein LRS eine stets akkurate und vollständige Sammlung von Daten enthält, ist es nicht möglich, diese logisch zu verändern oder zu löschen [50], was ein entscheidender Faktor für den Einsatz von xAPI in verteilten Systemen ist. Da jedoch Statements eventuell nicht dauerhaft gültig sein sollen, Fehler bei der Erstellung gemacht worden sind oder aus anderen Faktoren Statements deaktiviert werden sollen, ist es möglich, über ein Statement in einem bestimmten Format ein anderes als ungültig (voided) zu deklarieren [50]. Jedes Statement, das ein anderes als ungültig markiert, kann selbst nicht als ungültig markiert werden. Das Statement welches ein anderes in den Zustand ‚ungültig‘ setzt,

muss dabei folgendes Format erfüllen: Das `Verb` muss als *voided* deklariert sein und als `Objekt` die Felder `objectType` mit dem Wert *StatementRef* und `id` mit der Identifikation des zu ändernden Statements gesetzt sein. Die `id` eines Statements wird, falls nicht explizit angegeben, automatisch vom LRS zugeordnet. Folgendes Codebeispiel markiert das fortlaufende Beispiel (mit der automatisch zugeordneten Statement `id` *8f2bf402-f1fb-4039-8128-38d6ab966aae*) als ungültig:

```
{
        "actor": {
                "objectType": "Agent",
                "name": "Peter Glider",
                "mbox": "mailto:peter@example.com"
        },
        "verb": {
                "id": " http://adlnet.gov/expapi/verbs/voided",
                "display": { "en-US": "voided" }
        },
        "object": {
                "objectType": "StatementRef",
                "id": "8f2bf402-f1fb-4039-8128-38d6ab966aae "
        }
}
```

Weitere Architekturen und Methoden

Alternativ zur Nutzung von xAPI-Statements ist der Einsatz der **High-Level Architecture (HLA)** für die Kommunikation zwischen verschiedenen Simulationssystemen denkbar [7]. Das seit dem Jahr 2000 international standardisierte (IEEE 1516) Konzept wurde vom US-amerikanischen Verteidigungsministerium (U.S. Departement of Defense) für die Interoperabilität ihrer verschiedenen Simulationen entwickelt. Der Standard ist dabei nicht für eine spezifische Applikationsdomänen- oder Simulationsmodellierungs-Methodik konzipiert. Anfänglich für das Militär entwickelt, wird der Standard heutzutage auch für den Gesundheitsbereich, die Versorgungskette (Supply Chain) und viele weitere Bereiche genutzt [7]. Das HLA Konzept verbindet mehrere unabhängige Simulationen über eine Verbindungsschnittstelle, die so genannte Runtime-Infrastructure (RTI), miteinander (Abbildung 3.5). Die Kommunikation der verschiedenen Knoten (Föderaten) sowie der Objekte selbst werden dabei in Federation Object Models (FOMs) spezifiziert. Das Ziel der HLA ist es, die Interoperabilität und die Wiederverwendung von Simulationen auszubauen [51].

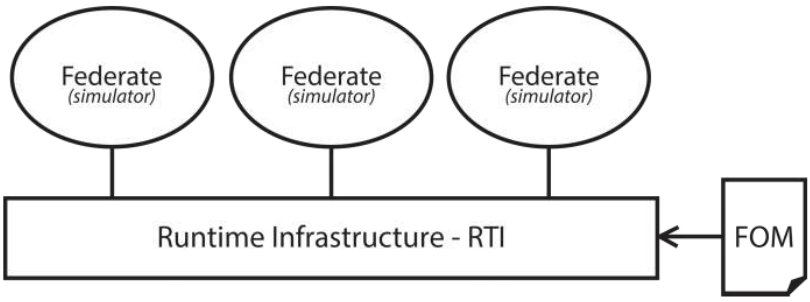


Abbildung 3.5 HLA Diagramm (Quelle: [51])

Ein weiteres Modell, um Lerninhalte auszutauschen ist das **Sharable Content Object Reference Model (SCORM)** [52]. SCORM ist ein von der Advanced Distributed Learning (ADL) Initiative 2004 entwickeltes Referenzmodell für den Austausch elektronischer Lerninhalte in einem Learning Management System (LMS). Es gilt als Vorgänger des xAPI-Standards [50]. SCORM umfasst eine Sammlung an Spezifikationen aus verschiedenen Quellen, um die verschiedenen Umgebungen von E-Learning Inhalten zu ermöglichen. SCORM besteht in der Grundversion aus den drei Elementen *Metadata*, *Content Aggregation Model* (CAM) und *Run-Time Environment* (RTE) [53, 54]. Die Metadaten werden spezifiziert in dem Format Learning Object Metadata (LOM). Das Ziel dieser Spezifikation ist es, digitale Lernressourcen im Rahmen von computergestütztem Lernen zu beschreiben. Ein grundlegender Aspekt ist dabei, dass jedes Lernobjekt eindeutig identifiziert werden kann. Lernobjekte können dabei verschiedener Art sein. Ein LOM-Objekt kann z.B. ein Kurs, Buch, Lehrender oder ein multimediales Objekt sein [55]. Über die Metabeschreibungen von LOM wird das Auffinden sowie die Verknüpfung und Aggregation von Lerninhalten ermöglicht [56]. Das Content Aggregation Model (CAM) ist in XML spezifiziert und beschreibt, welche Lernressourcen verwendet werden können. Die Ressourcen können dabei in Dateien mit unterschiedlichem Format oder in URLs angegeben werden. Diese werden hierarchisch organisiert und in einer Datei hinterlegt, sodass diese in andere Management-Systeme integriert werden können. Mithilfe der Run-Time Environment (RTE) wird die Schnittstelle zum LMS und den Lernobjekten beschrieben. In dieser können unter anderem Lernfortschritte oder Lernziele definiert werden.

3.4. Web Development und Frameworks

In den folgenden Abschnitten werden die Grundlagen für die genutzten Technologien erklärt. Dies beinhaltet unter anderem die Erläuterungen der Kommunikation zwischen den Systemen mithilfe von Servlets sowie das Framework Vaadin, in welchem die ELAI-UI umgesetzt ist. Um ein besseres Verständnis des Aufbaus sowie der Vorzüge von Vaadin zu erhalten, werden weiterhin grundlegende Konzepte wie AJAX und RIA sowie das Toolkit GWT, auf dem Vaadin basiert, vorgestellt.

Servlets und Servlet Container

Das englische Wort Servlet ist zusammengesetzt aus den Begriffen Server und Applet (Mini-Applikation) und ist eine Abkürzung für ein serverseitiges Applet. Als Servlet bezeichnet man Java Klassen, die ein bestimmtes Interface implementieren. Damit diese Klassen auf HTTP-Anfragen reagieren können, müssen diese in einem Servlet Container (= Java Web Server) ausgeführt werden. Bei Anfragen an den Servlet Container entscheidet dieser anhand von Konfigurationsparametern, an welches Servlet die Anfragen weitergeleitet werden sollen und stellt dem Servlet mittels der Objekte `javax.servlet.http.ServletRequest` und `javax.servlet.http.ServletResponse` die Client Anfrage bzw. die Antwort in der `doGet` und `doPost` Methode zur Verfügung. Servlets erweitern somit durch ihre implementierte Funktionalität die Fähigkeiten des Servlet Containers, indem sie Anfragen (Requests) von Clients entgegennehmen, verarbeiten und an die Clients eine Antwort (Response) zurück senden. In der Praxis ist ein Java Servlet ein Teil einer Web Applikation, welche HTML Pages als statischen Kontext und Java Server Pages (JSP) sowie Java Servlet für den dynamischen Inhalt bereitstellt (Abbildung 3.7) [57][58]. Web Applikationen werden in eine Datei des Typs `.war` (Web Application Archive) verpackt und im Servlet Container eingebunden. Ein .war File ist dabei eine Erweiterung des Subtyps JAR (Java Archive). Für die Ausführung eines solchen

.war Files wird ein Servlet Container wie z.B. der Jetty Server [59], der Glassfish Server [60] oder der TomCat Server [61] benötigt.

Um das Servlet auf dem Web Server zu registrieren, werden Metainformationen des Servlets in der Konfigurationsdatei (Deployment Descriptor) mit dem Namen WEB-INF/web.xml im XML-Format bereitgestellt (Abbildung 3.6). In dieser befinden sich die Definitionen der Servlet Klassen und die Zuordnung zu den URL Pfaden des Servlets. Die Klassenpfade für die Servlets und deren Abhängigkeiten sind dagegen in der WEB-INF/classes und WEB-INF/lib Ordnern enthalten. Der WEB-INF Ordner ist ein spezieller Ordner, der die kompletten Metadaten der Web Applikation enthält und nicht über den URL-Pfad erreichbar ist.

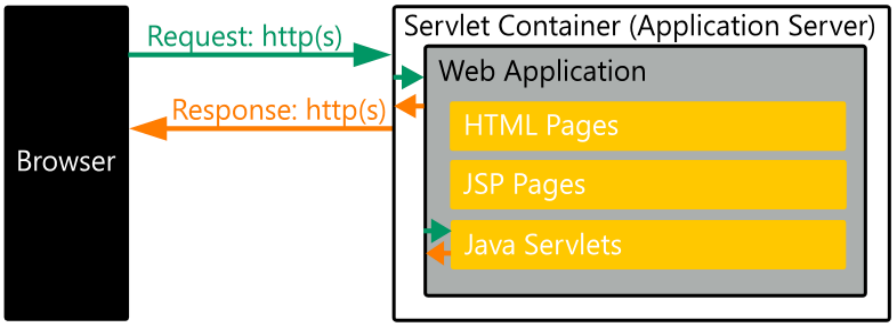


Abbildung 3.7 Servlet und Servlet Container (Eigene Darstellung nach [57])

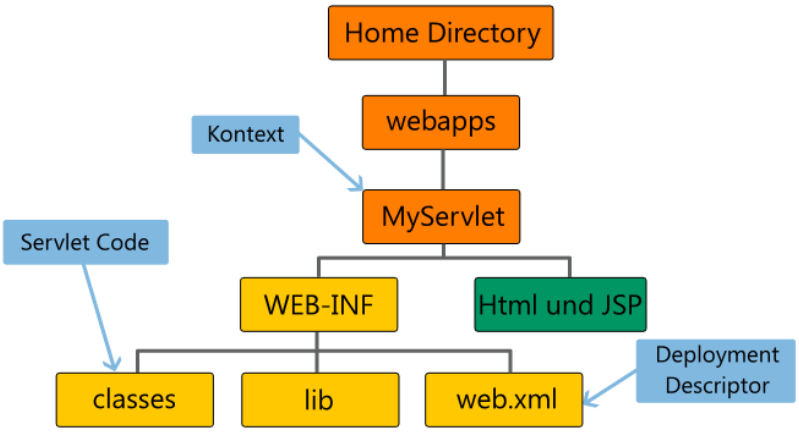


Abbildung 3.6 Servlet Deployment (Eigene Darstellung nach [57])

Asynchronous JavaScript and XML

Im Folgenden wird das AJAX Konzept erläutert, um die Gründe der Umsetzung im Vaadin Framework zu verstehen.

Asynchronous JavaScript and XML (AJAX) beschreibt ein Konzept für die Funktionsweise von Webanwendungen. Das AJAX Konzept basiert dabei zum einen auf der grundlegenden Idee des asynchronen Datenaustauschs zwischen Server und Client, der zwingenden Verwendung der Skriptsprache JavaScript auf der Clientseite sowie auch auf der Serverseite und zuletzt auf dem Datenaustausch zwischen Server und Client im XML-Format.

Der Prozessfluss einer traditionellen Webanwendung wird durch zustandslose HTTP-Anfragen bestimmt. Dies führt durch das damit zusammenhängende Request-Response Verfahren dazu, dass der Nutzer nach Ausführung einer Aktion auf die Antwort des Servers warten muss und somit eine Verzögerung entsteht (Abbildung 3.8). Ziel des AJAX Konzeptes ist es, die Nachteile solcher traditionellen Webanwendungen zu überwinden und die Bedienfreundlichkeit von Desktop-Anwendungen zu erreichen [62]. Zu dem genannten Ziel gehören zum einen die asynchrone und nebenläufige Datenübertragung, sodass der Nutzer während der Übertragung der Daten weiterhin die Oberfläche bedienen kann und zum anderen die Reduzierung der Reaktionszeit dadurch, dass nicht mehr die komplette Seite, sondern nur der veränderte Teil übertragen und neu geladen werden muss (Abbildung 3.9). Somit vermitteln AJAX Anwendungen dem Nutzer den Eindruck, dass diese direkt auf dem Computer ausgeführt werden. Dabei ist die XMLHTTPRequest API von JavaScript eine wesentliche Komponente für das AJAX Konzept. Mit dieser ist es möglich, HTTP-Anfragen in JavaScript an den Webserver zu senden und dessen Antwort wiederum in JavaScript zu verarbeiten und auf der Clientseite durch die Document Object Model API geregelt darzustellen [63]. Die Antwort kann dabei nicht nur wie eingangs beschrieben in XML- sondern auch in HTML- oder JSON-Daten formatiert sein.

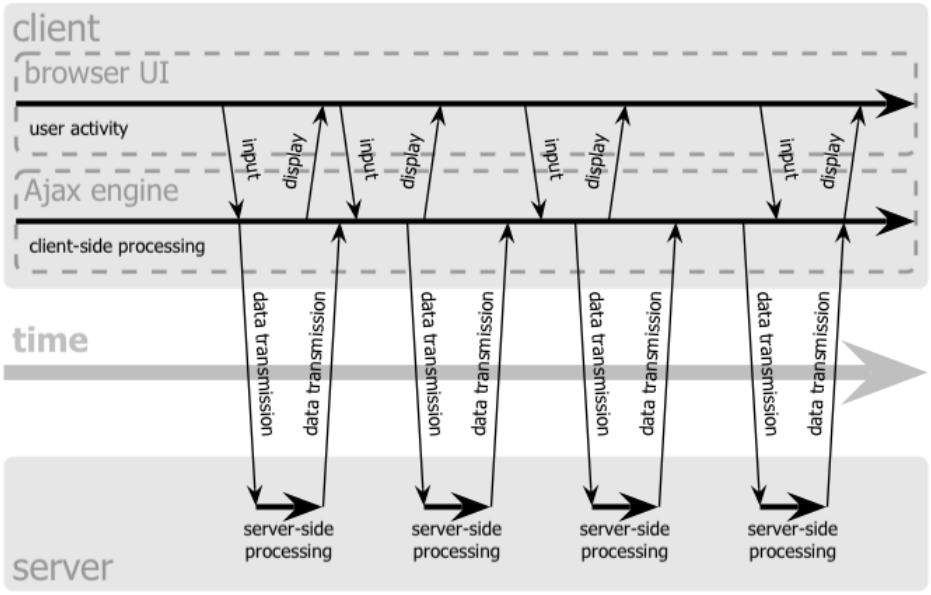


Abbildung 3.9 AJAX Modell einer Web-Anwendung (asynchrone Datenübertragung) (Quelle: [62])

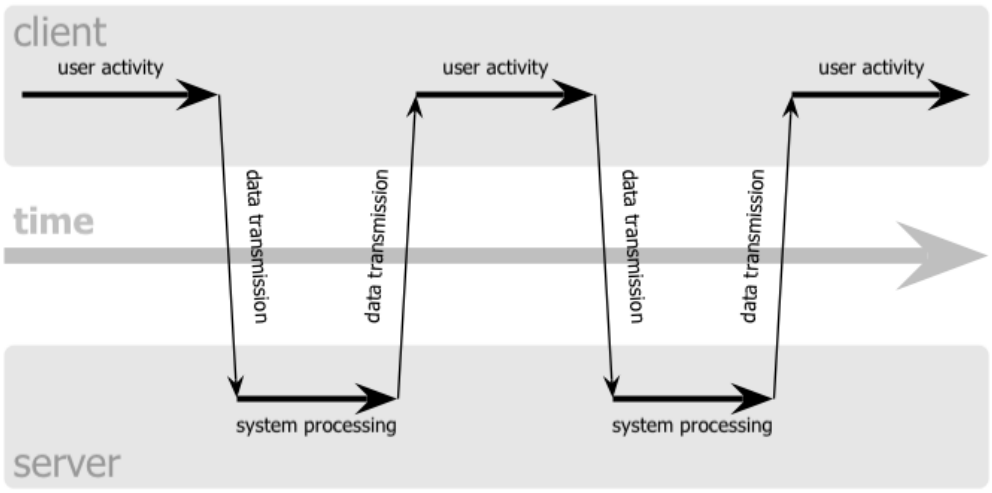


Abbildung 3.8 Klassisches Modell einer Web-Anwendung (synchrone Datenübertragung) (Quelle: [62])

Rich Internet Application

Eine Idee, die Vaadin umsetzt, ist RIA. Rich Internet Application (RIA) ist kein standardisierter Begriff, beschreibt aber im Allgemeinen eine Webanwendung, deren Interaktionsmöglichkeiten über eine klassische Webanwendung hinausreichen. Unter RIA ist der Trend zu dynamischen Webbrowser Desktopanwendungen mit der Möglichkeit der Interaktion mit der Benutzeroberfläche durch bspw. Drag-and-Drop Elemente, 3D-Effekte oder Animationen zu verstehen. Da diese Elemente in der Regel mehr Anwendungslogik als statische Webseiten benötigen, würde dies zu erhöhten Ladezeiten führen. Durch den Einsatz von Techniken wie AJAX kann die Performance und die damit einhergehende Benutzerfreundlichkeit von RIAs verbessert werden [64].

Google Web Toolkit

Das Google Web Toolkit (GWT) ist ein freies Framework, das die Entwicklung von RIAs in der Programmiersprache Java ermöglicht und bildet einen Grundbaustein für das Vaadin Framework. Diese Desktop-artigen Anwendungen werden typischerweise in der unter den Webbrowsern weit verbreiteten Sprache JavaScript geschrieben [65]. Um die Lücke zwischen Java und JavaScript zu schließen, nutzt das Framework ihre wichtigste Komponente, den Java-to-JavaScript Compiler. Dieser übersetzt den Java-Code in JavaScript-Code, welcher in den Webbrowsern der Nutzer ausgeführt wird. Somit ermöglicht GWT RIAs, basierend auf JavaScript, in Java zu entwickeln [65].

Die Kommunikation zwischen Client und Server erfolgt dabei über so genannte Remote Procedure Calls (RCPs), welche Interprozesskommunikationen realisieren. Dies ermöglicht das Aufrufen einer Funktion oder eines Prozesses in einem anderen Adressraum [64]. Weiterhin bietet GWT die Funktionalität einer maximalen Schnittmenge der Browser Kompatibilitäten, womit Applikationen entworfen werden können, die basierend auf demselben Code in verschiedenen Browsern gleiche Ausführungen liefern. Weiterhin bietet GWT ein Set an fertigen User Interface Widgets, die es erlauben, in kurzer Zeit neue Applikationen zu entwerfen sowie aus bereits existierenden Widgets neue zu kombinieren [66].

Vaadin Framework

In diesem Abschnitt wird das Framework Vaadin erläutert, mit welchem die ELAI-UI entwickelt wurde.

Vaadin ist ein Framework, das es ermöglicht, Web Applikationen basierend auf Java zu entwerfen. Es wurde von dem gleichnamigen Unternehmen Vaadin, Ltd. entwickelt und ist unter der Lizenz *Apache License, version 2.0* frei erhältlich [67]. Es unterstützt dabei sowohl die Programmierung als serverseitiges- wie auch als clientseitiges Modell, wobei die servergetriebene Programmierung laut Vaadin, Ltd. das mächtigere Model darstellt. Entwickelt wird die Vaadin Webanwendung wie eine Desktopapplikation mit den konventionellen Java Toolkits wie AWT, Swing oder WST [57]. Dabei übernimmt das Framework alle Browser zu Server Kommunikationen und Data Transfer Objects für den Entwickler. Für den Nutzer ersichtlich ist nur eine auf HTML5 basierende Webapplikation im Browser [67].

Während bei der traditionellen Webentwicklung viel Zeit zum Erlernen von Webtechnologien wie HTML oder JavaScript erforderlich ist, übernimmt das Vaadin Framework das serverseitige Regeln des User Interfaces im Browser und die AJAX-Kommunikation zwischen Client (Browser) und Server, sodass sich der Entwickler auf die Programmierung der Logik in Java konzentrieren kann. Die Webanwendung selbst befindet sich dabei ausschließlich auf der Serverseite. Durch die Umsetzung des AJAX-Konzeptes in der servergetriebenen Vaadin

Anwendung ist es möglich, Rich Internet Applications (RIAs) zu entwerfen, welche sich an das Endnutzergerät anpassen (Responsive Design) und sich wie eine Desktopapplikation interaktiv anfühlen.

Vaadin beruht auf dem Google Web Toolkit (GWT), sodass dem Entwickler die Anpassung der Webapplikation an verschiedene Browser abgenommen wird. Durch das vollkommene Verschwinden von HTML, JavaScript und anderen Browser-Technologien während der Applikationsentwicklung, kann sich der Programmierer direkt mit dem User Interface (UI) und deren Logik beschäftigen. Das User Interface wird zusammen mit der Business Logik als Java-Servlet auf einem Java-Applikation-Server gestartet. Der in Vaadin integrierte GWT Compiler kompiliert dabei den Java Code in JavaScript-Code. Dieser wird als Servlet im `.war` Format zum Java Applikation Server (genauer: zu einem Servlet Container) ausgeliefert, wo er dann ausgeführt wird.

Die Architektur der Vaadin Applikation teilt sich in den serverseitigen und den clientseitigen Bereich auf. In Abbildung 3.10 ist die Architektur einer Vaadin Applikation dargestellt. Die Clientseite läuft auf dem Browser als JavaScript-Code und stellt das User-Interface (Abbildung 3.10 – Client-Side UI) und die Interaktion der User Events (Abbildung 3.10 – Vaadin Client-Side Engine) mit dem Server dar. Dabei sind keinerlei Browser Plugins für die Nutzung einer Vaadin Applikation nötig. Das Vaadin Servlet wird über HTTP-Anfragen bedient und als normale Servlet Klasse ausgeliefert. Das Servlet empfängt die Client Anfragen und interpretiert diese als Events für die bestimmten User Sessions. Die Events sind dabei mit den UI-Komponenten assoziiert und werden zu den entsprechenden Event-Listener der Anwendung ausgeliefert. Stellt die UI-Logik eine Änderung auf den Serverseitigen UI-Komponenten fest, wird eine passende Antwort für den Client generiert. Die Client-Side Engine empfängt die Antwort und nutzt diese um die nötigen Änderungen auf der Seite im Browser vorzunehmen.

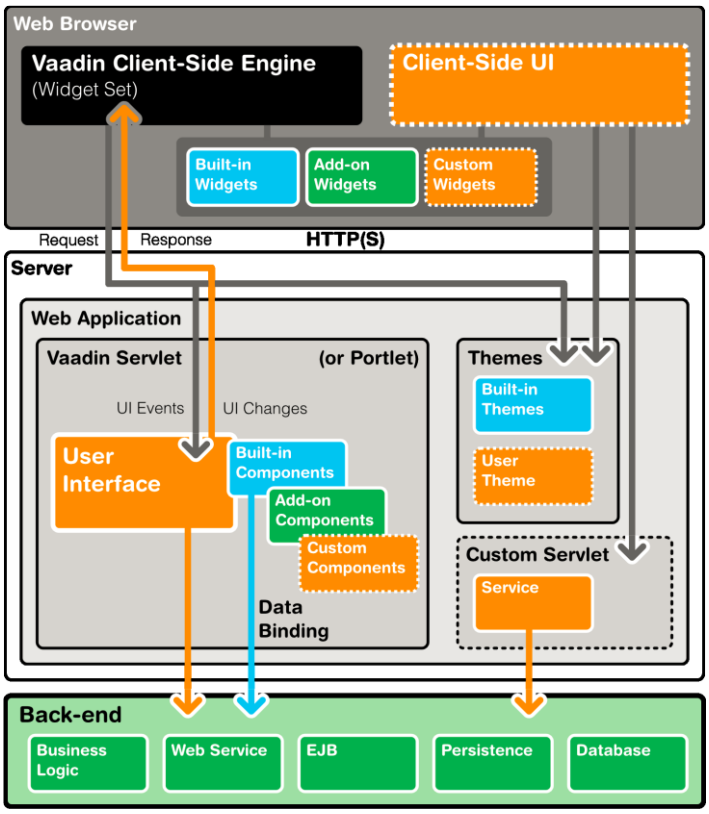


Abbildung 3.10 Vaadin Anwendungsarchitektur (Quelle: [57])

3.5. Entwurfsmuster

Die in dieser Arbeit zu entwickelnde Webanwendung wird mithilfe des Frameworks Vaadin realisiert. Um die Vaadin Anwendung flexibler und modularer zu gestalten wurde das Pattern *Model View Presenter* verwendet. Weiterhin kam das *Event Bus* Pattern zum Einsatz, um zwischen den einzelnen Elementen einer Ansicht und auch zwischen verschiedenen Ansichten zu kommunizieren, ohne die Modularität der Komponenten zu zerstören.

Model View Presenter – Pattern

Das Model View Presenter (MVP) Pattern ist das meist verwendete Pattern bei der Entwicklung von großen Applikationen mit Vaadin und wurde auch in dieser Arbeit eingesetzt [68]. Das MVP-Pattern ist eine Weiterentwicklung des bekannten Model View Controller (MVC) Patterns. Die beiden Architekturmuster unterscheiden sich, wie in Abbildung 3.11 zu sehen, hauptsächlich in der Trennung zwischen dem Model und der View. Dies sorgt beim MVP-Pattern für eine gute Austauschbarkeit der beiden Schichten (seperation of concern). Die Kommunikation der View- und Model-Schicht erfolgt nun über die Presenter-Schicht. Diese greift auf die Model-Schicht zu und leitet die angeforderten Daten an die View-Schicht weiter. Die Presenter-Schicht enthält dabei die User Interface Logik der View. Alle Interaktionen, die der Nutzer auf dem User Interface tätigt, werden direkt von der View zum Presenter weitergeleitet und dort behandelt. Drückt bspw. der Nutzer einen ‚Tabelle aktualisieren' Button, delegiert dessen Event Handler den Aufruf weiter zur festgelegten Methode des Presenter. Dieser lädt die aktuellen Daten vom festgelegten Model und ruft die entsprechenden Methoden zur Aktualisierung der Tabelle mit neuen Daten in der View auf. Durch die Trennung des User Interfaces in der View-Schicht und der User Interface Logik in der Presenter-Schicht ist wiederum die Austauschbarkeit der Schichten gewährleistet. Abbildung 3.11.b zeigt die Struktur des MVP-Patterns für eine Gesamtkomponente.

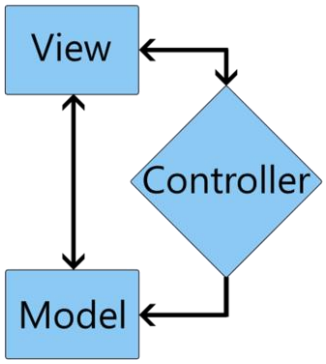


Abbildung 3.11.a Modell View Controller (MVC) Pattern – Komponenten und Abhängigkeiten

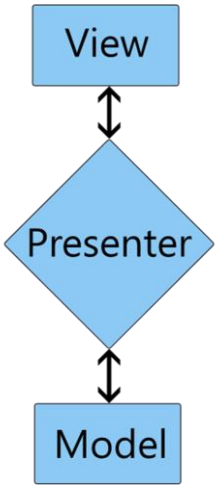


Abbildung 3.11.b Modell View Presenter (MVP) Pattern – Komponenten und Abhängigkeiten

Abbildung 3.11 MVC-Pattern und MVP-Pattern

Event Bus – Pattern

Damit die Ansichten und die einzelnen MVP-Teilelemente in einer Ansicht kommunizieren können, kann das Event Bus Pattern verwendet werden. Durch dieses Architekturmuster ist eine sehr lose Kopplung aller Komponenten auch über mehrere Schichten von MVP-Views hinweg möglich. Für eine komplette Applikation wird nur eine Instanziierung der EventBus-Klasse, meist in der Main-UI des Vaadin Projektes, benötigt [69].

Zur Kommunikation werden Events und Event Handler genutzt. Ein Event ist dabei ein Objekt, das den zuständigen Event Handler (Listener) über eine Änderung informiert. Dabei können auch Parameter eines beliebigen Typs übergeben werden. Wird ein Event gesendet, wird dieses an alle registrierten Event Handler ausgeliefert und die jeweilig implementierte Methode einer registrierten Klasse aufgerufen. Auf diese Weise können verschiedene Klassen auf dasselbe Ereignis unterschiedlich reagieren. In Abbildung 3.12 ist ein beispielhaftes Szenario für den Einsatz des Event Buses illustriert. Objekt `o1` sendet an den Event Bus ein Event des Typs `A`. Da sich nur der `EventHandler 1` und `EventHandler 3` für diesen Typ registriert haben, werden nur diese über das Ereignis informiert und rufen ihre jeweiligen Methoden `onEventA_do_X1(…)` bzw. `onEventA_do_X3(…)` auf.

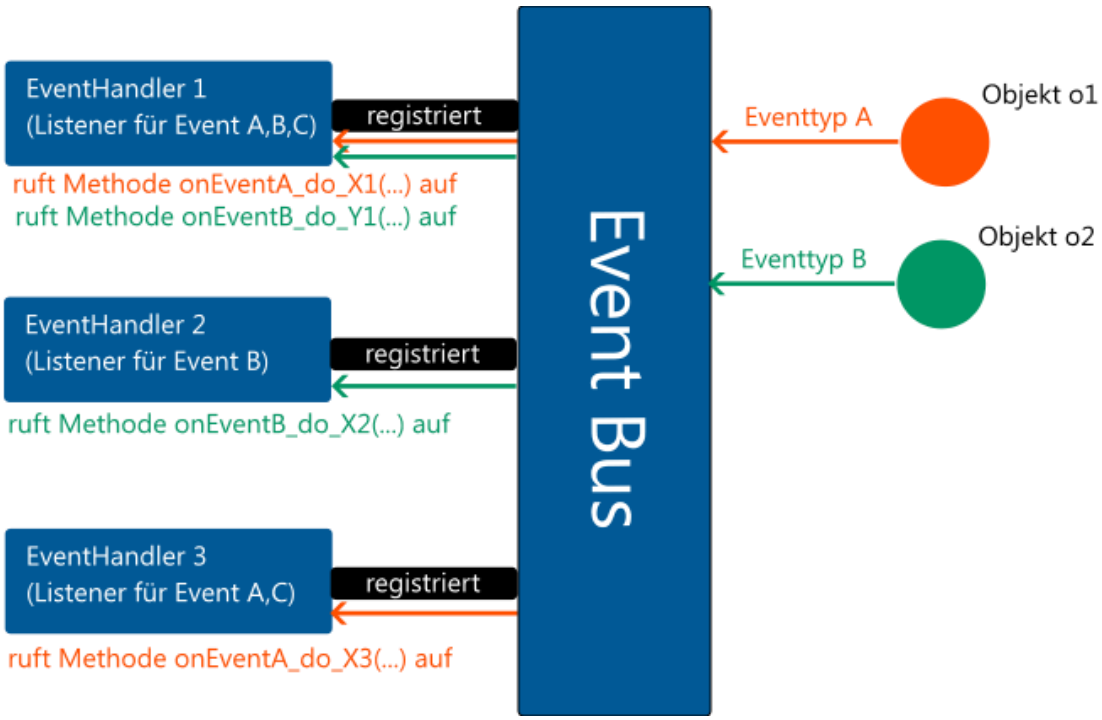


Abbildung 3.12 Event Bus Prinzip illustriert an Beispielen

Kapitel 4
Agentenbasierte Kontrolllogik für adaptive Serious Games

Um eine bessere Erweiterbarkeit und logische Separation zu gewährleisten, wurden bei der Umsetzung der Arbeit die ELAI und die ELAI-UI getrennt [1]. Die ELAI enthält dabei die Kontrolllogik und dient als zentrales Element des Gesamtsystems. Es bietet die Schnittstellen zu den Simulationssystemen und den Serious Games, hält den Zugriff zu den Datenbanken und führt die notwendigen Berechnungen für die Adaptionen durch. Die ELAI-UI hingegen dient der Visualisierung der ELAI und gibt dem Agenten/ Tutor die Möglichkeit, die gewonnen Daten zu visualisieren, zu analysieren und aufgrund der gewonnen Kenntnisse adaptiv auf die ELAI und somit auf die angeschlossenen Systeme zu reagieren. Die Kontrolllogik für adaptive Serious Games kann somit über einen Agenten gesteuert werden. In Abbildung 4.1 sind die grundlegende Architektur der Arbeit sowie die über Servlets realisierten Schnittstellen zu den Serious Games SaFIR und Lost Earth 2307 [70] illustriert. Zu Test- und Studienzwecken wurde das SaFIR (Seek and Find für Image Reconnaissance) Serious Game des Fraunhofer IOSB genutzt, welches im Rahmen dieser Arbeit um einen ELAI-Adapter erweitert wurde (Kapitel 6.1). Dieser realisiert die Kommunikation mit der ELAI auf der Spieleseite. In den folgenden Abschnitten wird auf den Entwicklungsprozess sowie die Umsetzung der einzelnen Komponenten eingegangen.

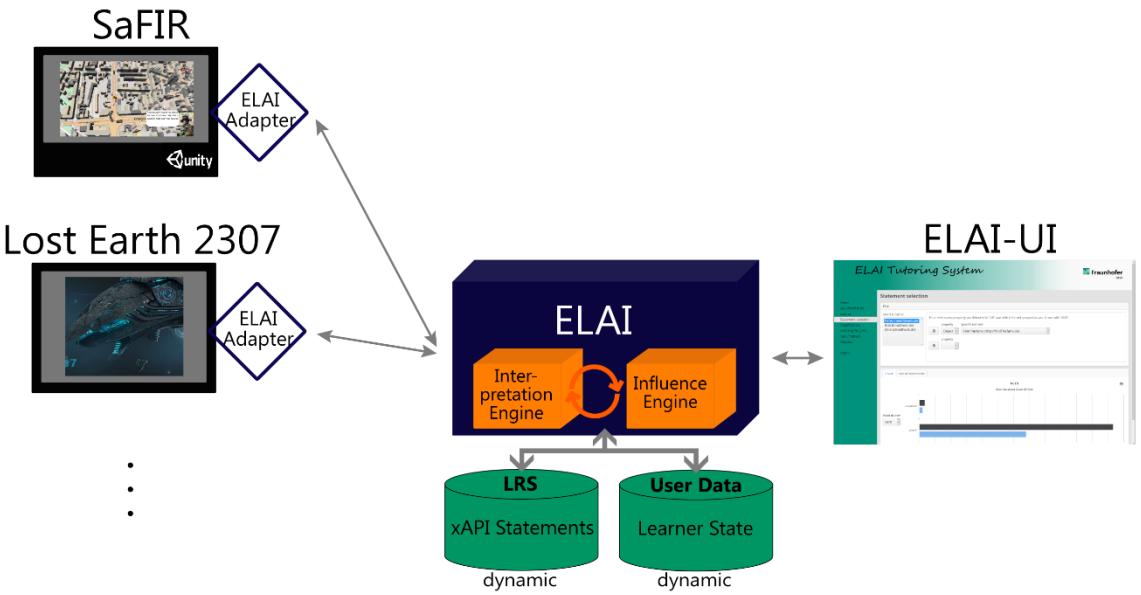


Abbildung 4.1 Grundlegende ELAI-Architektur mit angeschlossenen Serious Games SaFIR und Lost Earth 2307 (Quelle: [70])

4.1. ELAI (Controller)

Die E-Learning Artifical Intelligence (ELAI) stellt das Kernstück des gesamten Systems dar [1]. Um mit den Simulationssystemen und Serious Games kommunizieren zu können, wird an diese ein ELAI-Adapter angeschlossen, welcher eine bidirektionale Kommunikation ermöglicht. Somit können die persönlichen Nutzerdaten systemunabhängig in einer zentralen Datenbank gespeichert werden. Weiterhin werden die erzeugten Spieldaten in einem weiteren Speicher gesammelt und für weitere Anfragen verfügbar gemacht. Mithilfe der implementierten Intepretation- und Influence-Engine, ist es der ELAI möglich, Spiele zu analysieren und auf diese zu reagieren. Die Intepretation-Engine beschreibt dabei einen im Rahmen dieser Arbeit entworfenen Wortschatz, welcher die Aktionen des Lerners Maschinen verständlich beschreibt. Auf dessen Basis können Verhaltensregeln in der Influence-Engine definiert werden. Diese ermöglichen es, das Spiel situationsabhängig zu adaptieren. Zudem bietet die verständliche Beschreibung der elektronischen Daten die Möglichkeit, didaktische Faktoren und systemunabhängige Klassifikationen zu berechnen und so die Leistung des Spielers zu bewerten.

Bevor auf die einzelnen Teilbereiche der ELAI und deren Umsetzungen eingegangen wird, wird die grundlegende Kommunikation zwischen den Systemen betrachtet.

Kommunikationsschicht und ELAI-Adapter

Für die Einbindung verschiedenster Systeme war es nötig, eine Schnittstelle umzusetzen, welche es ermöglicht, netzwerkübergreifend Daten auszutauschen.

Hierfür können unter anderem die in Kapitel 3.3 erläuterten Systeme High-Level-Architektur (HLA) [7] sowie die Experience API (xAPI) [8] eingesetzt werden. Für den Einsatz der HLA spricht die standardisierte Schablone für Datenmodelle zum Austausch der Interaktionen und der Objekte zwischen den Förderaten. Ein weiterer Vorteil liegt in dem für alle Förderaten verständlichen Datenaustauschformat – FOM. Nachteilig allerdings wäre, dass die ELAI die Rolle eines Förderaten übernehmen müsste, um die Daten der RTI lesen zu können. Dies würde allerdings dem Grundprinzip, dass ein Förderat eine Simulation oder ein Spiel ist, wiedersprechen.

Im Gegensatz zum FOM liefert die xAPI ein Format, das zum einen durch seine strukturierte Form und die Nutzung von verständlichen Begriffen für Menschen lesbar ist und es zum anderen ermöglicht, die Daten direkt in einem zentralen externen LRS zu speichern. Da ein wesentlicher Aspekt des ELAI-Konzeptes darin besteht, die Daten der Systeme an die ELAI zu senden und nicht zwischen den Systemen auszutauschen, wurde die xAPI im Rahmen dieser Ausarbeitung eingesetzt. Um Daten wie bspw. xAPI-Statements netzwerkübergreifend zwischen den einzelnen Systemen austauschen zu können, wurden Schnittstellen der ELAI mithilfe von Java-Servlets realisiert. Diese bieten die Möglichkeit, Parameter der Anfrage zu übergeben und dem Anfragenden eine Antwort zu liefern. Um ein systemweites standardisiertes Format der Datenübertragung zu gewährlisten, sind alle Kommunikationen, mit Ausnahme der Parameter, im JSON-Format umgesetzt. In Abbildung 4.2 sind die implementierten Servlets sowie deren Systemzugehörigkeit abgebildet. Insgesamt bietet die ELAI neun Schnittstellen in Form von Java-Servlets an. Von diesen sind zwei dem Kommunikationsaustausch mit den Serious Games zuzuordnen (Spiele Servlets) und die Übrigen dem Austausch von Daten mit der ELAI-UI (ELAI-UI Servlets). Folgend wird deren Einsatzzweck sowie die Form der Datenübertragung erläutert.

Um das Serious Game mit der ELAI mittels der Spiele Servlets zu verbinden, muss jedes Simulationssystem bzw. Serious Game um einen **ELAI-Adapter** erweitert werden. Dieser wird

an ein System angeschlossen und bietet die Möglichkeit, nach dem Polling-Prinzip in regelmäßigen Abständen den Lerner Status des Spielers mithilfe des Pull-Servlets abzufragen und die getätigten Aktionen als xAPI-Statements über das Push-Servlet an die ELAI zu senden (Abbildung 4.3). Wird ein Statement mithilfe des **Push**-Servlets an die ELAI gesendet, wird nach Überprüfung des Formats, dieses im LRS gespeichert. Signalisiert das Statement die Beendigung einer Aufgabe, werden vom LRS die didaktischen Faktoren der erfolgreich gemeisterten Aufgabe berechnet und als gesondertes Statement an das LRS gesendet (siehe Abschnitt *Statements – Intepretation Engine*). Dies hat zur Folge, dass die Berechnungen nachfolgender Klassifikationen nur diese Statements anstelle der jeweiligen kompletten Spielverläufe betrachten müssen, was wiederum der Performancesteigerung dient.

Das **Pull**-Servlet liefert im Gegensatz zum Push-Servlet eine Antwort auf die Anfrage und stellt somit eine bidirektionale Kommunikation dar (Abbildung 4.4). In diesem sendet der Adapter eine Anfrage an die ELAI mit den Parametern Lerner E-Mail und Genre Identifikation. Beantwortet wird diese mit dem Lerner Status im JSON-Format, welcher die Fähigkeiten des Spielers in dem gefragten Genre sowie einen boolean-Wert (Hilfewert) für eine eventuell benötigte Hilfe beinhaltet. Durch die Antwort des Pull-Servlets ist es der ELAI möglich, adaptiv auf ein System zu wirken. Zum einen kann dies über die Schwierigkeit und das Hilfelevel des Spiels im Lerner Status oder über den Hilfewert erfolgen. Im Beispiel des SaFIR Serious Games wird über den Hilfewert der virtuelle Assistent angesteuert, der gegebenenfalls dem Spieler eine weitere Hilfe zur Verfügung stellt. Die Berechnung des Hilfewertes erfolgt zur Laufzeit und wird wahr, falls sich der Spieler länger als zehn Sekunden vom Aufgabenziel entfernt.

Die ELAI-UI Servlets dienen der ELAI-UI als Getter- und Setter-Methoden für Datenbankeinträge und -informationen sowie der Einstellung der Klassifikationsparameter.

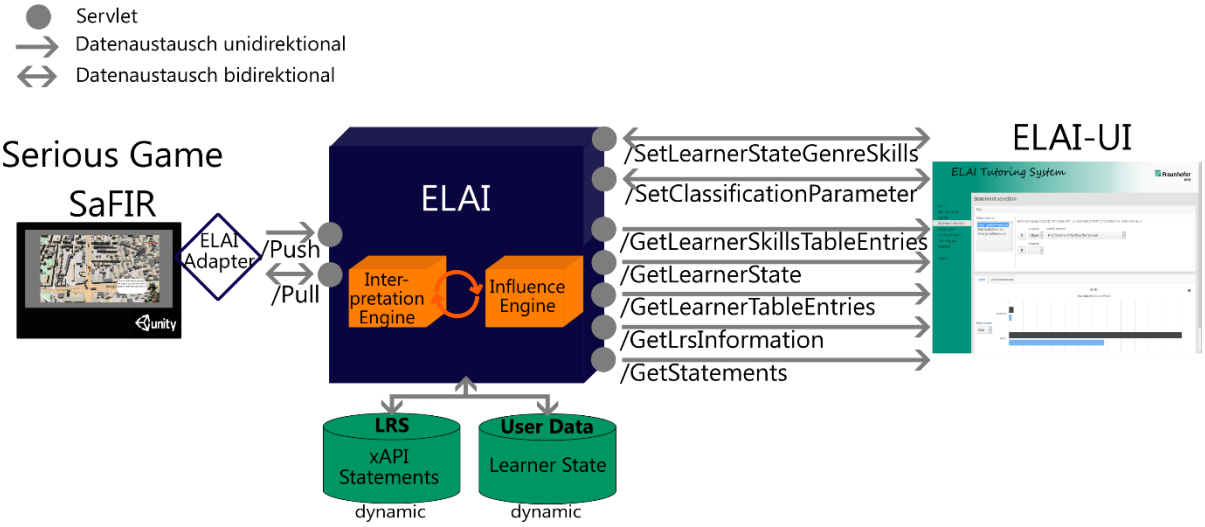


Abbildung 4.2 ELAI-Architektur mit ELAI-Servlets und angeschlossenem Serious Game SaFIR

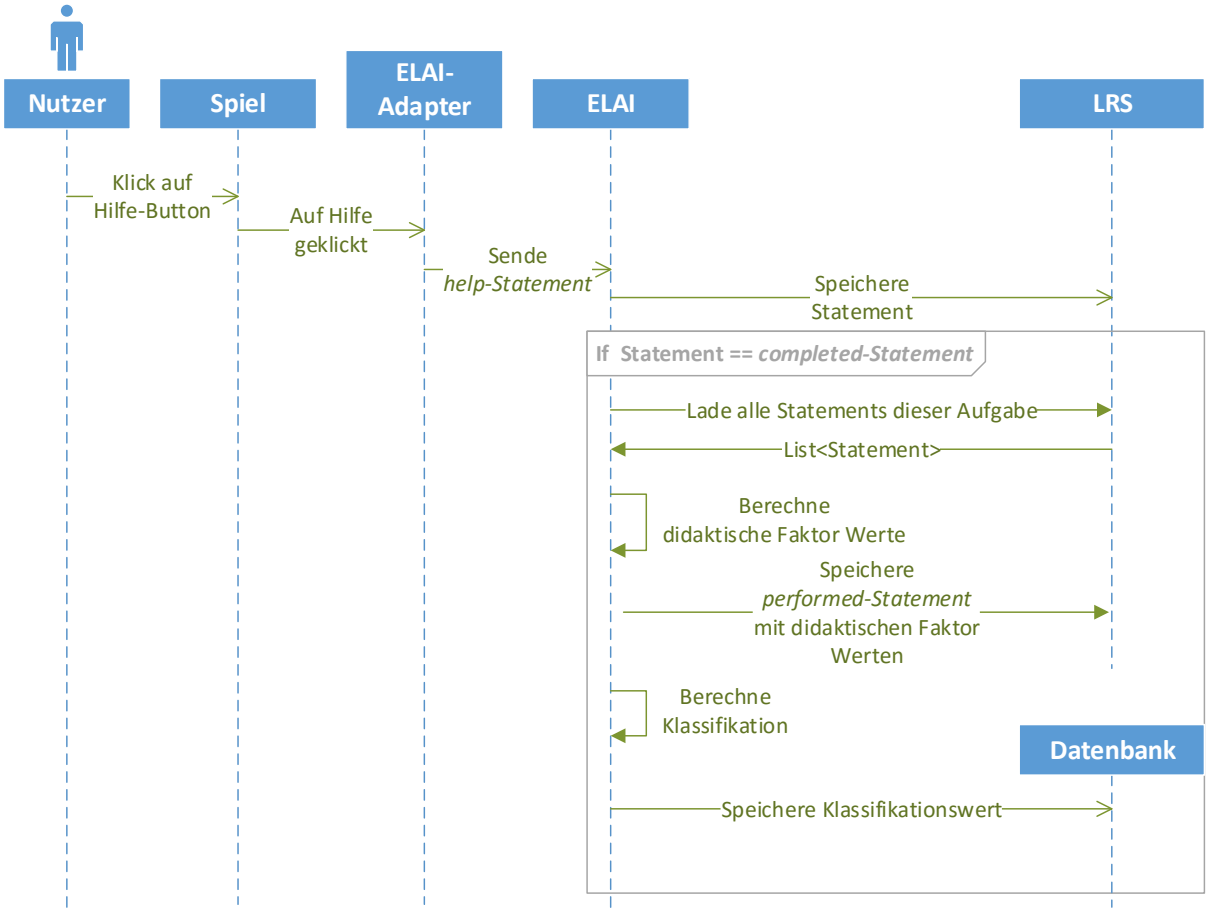


Abbildung 4.3 Ablauf Push-Servlet am Beispiel einer Hilfe Anfrage

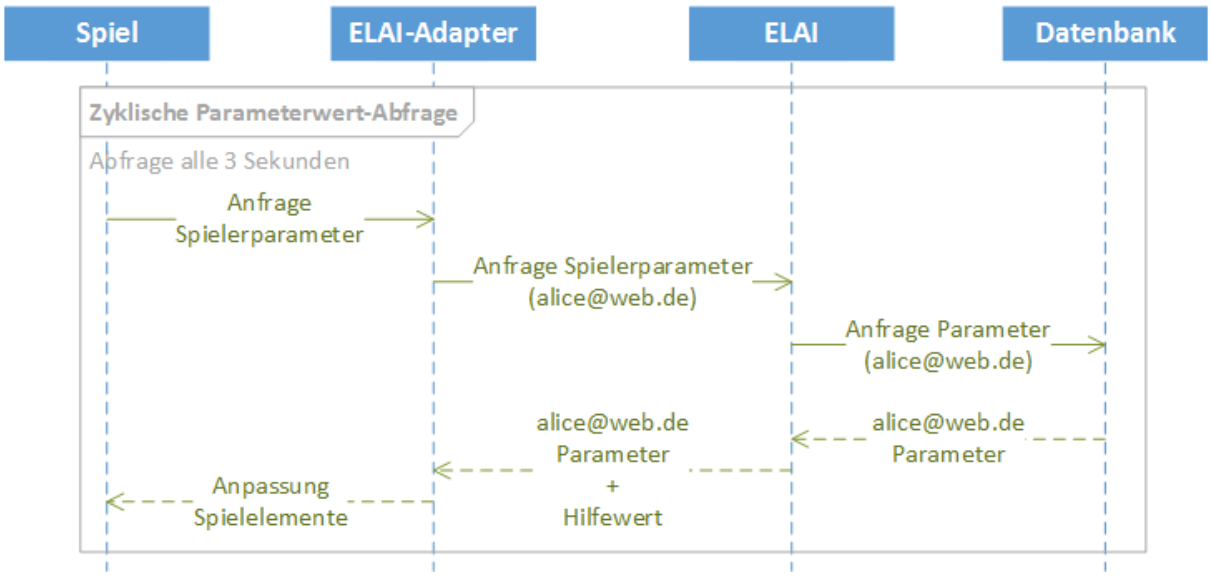


Abbildung 4.4 Ablauf Pull-Servlet: Zyklische Abfrage der Parameter des Nutzers alice@web.de

Datenbanken

Zur Speicherung der Lernerdaten nutzt die ELAI zwei verschiedene Datenbanken. Zum einen ein Learning Record Store (LRS), um die Statements der Spiele zu sichern und zum anderen eine objekterelationale Datenbank, um Nutzerdaten zu speichern.

Zur Speicherung der Nutzungsinformationen (Statements) wurde ein LRS verwendet, da mit diesem die Daten im xAPI-Format abgespeichert werden können und somit eine gefilterte Suche nach den einzelnen Elementen Actor, Verb, Object etc. möglich ist. Dabei empfahl sich, aufgrund der einfachen und kostenlosen Nutzung der Einsatz der SCORM Cloud LRS der Firma Rustici Software [71]. Weiterhin stellt das Unternehmen kostenfrei die TinCanJava Library zur Verfügung [72], welche es ermöglicht, mit dem LRS zu kommunizieren und die gespeicherten Statements gefiltert abzurufen sowie neue Statements zu speichern. Jedoch beschränkt sich der Einsatz der Library nicht nur auf den SCORM Cloud LRS, sondern ist auch für andere Learning Record Stores nutzbar. Um die dafür benötigten Login Daten variabel zu halten, wurden diese auf eine veränderbare Konfigurationsdatei ausgelagert, welche mithilfe der Apache Commons Configuration Library beim Start des Programms eingelesen wird [73].

Zur Speicherung spezifisch ausgewerteter Nutzerdaten wie Name, E-Mail Adresse und Fähigkeiten in bestimmten Genres, wurde eine lokale objektrelationale Datenbank aufgesetzt. Diese wurde verwendet, um den Datenzugriff über Java-Objekte mithilfe von objektrelationalen Mappern zu ermöglichen und trotzdem die Vorteile der höheren Zugriffsgeschwindigkeit von relationalen Datenbanken nutzen zu können. Für einen direkten Zugriff auf die objekt-relationale Datenbank über Objekte wurde Hibernate als OR-Mapper verwendet [74]. Die Datenbank umfasst dabei folgende Tabellen:

- **Learner Table** mit den Attributen Lerner E-Mail Adresse (Identifier) und Lerner Name.
- **Learner Skills Table** mit Lerner E-Mail Adresse und Genre-ID als Identifier und den jeweiligen Genrefähigkeiten und Hilfelevel.

Die Genrefähigkeit dient dabei als Repräsentation der Fähigkeit des Spielers in diesem Bereich. Besitzt dieser bspw. bereits fundierte Kenntnisse im Bereich der Panzersteuerung, können in einem anderen Simulationssystem oder Serious Game diese Fähigkeiten bereits als bekannt vorausgesetzt werden und bedürfen keiner erneuten Erklärung. Weiterhin dient der Hilfelevel der Tabelle als weiterer veränderbarer, genreabhängiger Faktor, der zusätzlich zu den Fähigkeiten das Ausmaß der Hilfestellung eines möglichen virtuellen Assistenten beschreibt. Beide Faktoren haben dabei einen Wertebereich von null bis eins, welcher die prozentuale Fähigkeit des Spielers beschreibt und nach jedem erfolgreichen Beenden einer Aufgabe neu berechnet und klassifiziert wird.

Klassifikation und didaktische Faktoren

Ein wesentlicher Aspekt der Arbeit ist es, mithilfe der im Spiel erzeugten Statements die Fähigkeiten des Spielers zu bestimmen, wofür die implementierte Interpretation-Engine verwendet wird. Ziel ist es, die gewonnenen Informationen zu nutzen, um die Fähigkeiten des Nutzers zu klassifizieren. Hierfür müssen die Daten analysiert und Parameterwerte (didaktische Faktoren) ermittelt werden [1]. Didaktische Faktoren beschreiben relevante Parameter des Spiels, die den Lernzustand des Spielers widerspiegeln. In der prototypischen Implementierung wurden hierfür folgende didaktischen Faktoren genutzt:

- **Spieldauer (TaskDuration)**: Benötigte zeitliche Dauer der Aufgabe
- **Schwierigkeit (TaskDifficulty)**: Eingestellter Schwierigkeitsgrad der Aufgabe
- **Hilfehäufigkeit (TaskHelpingCount)**: Anzahl benötigter Hilfen im Verlauf der Aufgabe

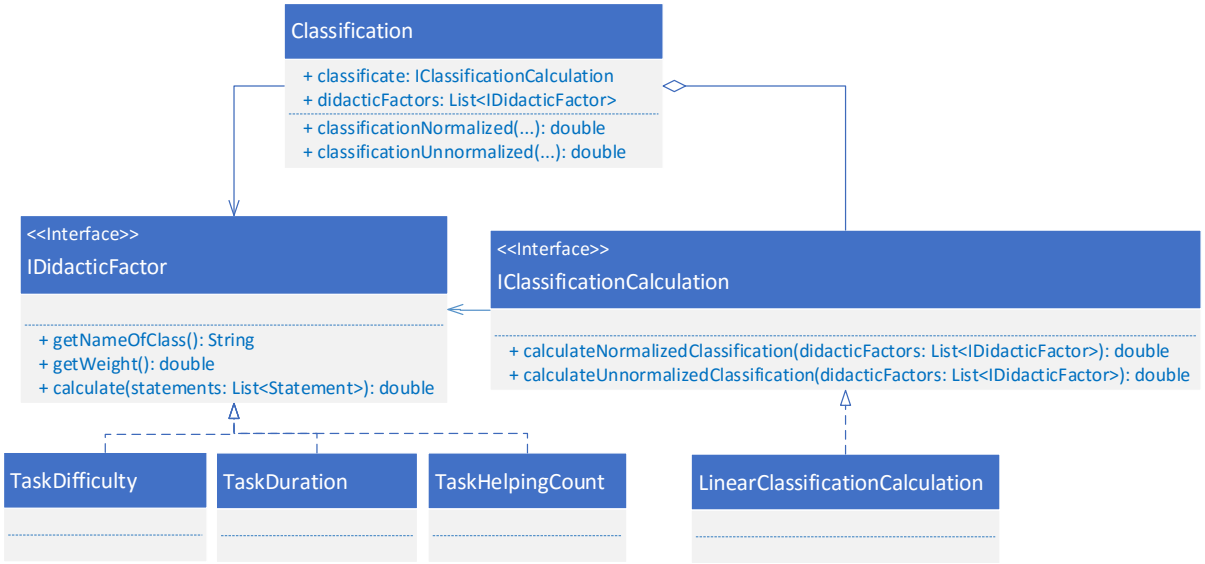


Abbildung 4.5 UML-Diagramm für die austauschbare Klassifikationsmethodik und didaktische Faktoren

Damit die Erweiterbarkeit um zusätzliche didaktische Faktoren gewährleistet werden kann, werden diese über eine Schnittstelle (Interface) beschrieben (Abbildung 4.5). Dieses erzwingt bei der Implementierung didaktischer Faktoren die Festlegung des Parametergewichtes sowie der Berechnungsmethode auf Basis einer Liste von xAPI-Statements. Weiterhin wurde die Klassifikationsberechnung durch ein Strategie-Pattern umgesetzt, was den Austausch der Klassifikationsberechnung erleichtert (Abbildung 4.5). Implementiert wurde ein linearer Klassifikator, basierend auf den drei beschriebenen didaktischen Faktoren.

Nach Beendigung einer Aufgabe des angeschlossenen Systems werden die gesammelten Statements der Aufgabe vom LRS geladen und die individuellen Berechnungen der didaktischen Faktoren $X = (x_1, \dots, x_n)$ basierend auf den Statements durchgeführt. Im Anschluss wird das Ergebnis mithilfe der Formel

$$\bar{x_i} := Normierung(x_i, \vec{x_i}) = \begin{cases} \dfrac{x_i - minWert(\vec{x_i})}{maxWert(\vec{x_i}) - minWert(\vec{x_i})}, & maxWert(\vec{x_i}) - minWert(\vec{x_i}) \neq 0 \\ defaultValue, & sonst \end{cases}$$

normiert. $\vec{x_i}$ repräsentiert dabei die x_i aller zur Verfügung stehenden Aufgaben. Weiter werden die Ergebnisse genutzt, um den Klassifikationswert mit der Formel

$$Klassifikation(X) = \begin{cases} \dfrac{\bar{x_1} * w_1 + \dots + \bar{x_n} * w_n}{w_1 + \dots + w_n}, & w_1 + \dots + w_n \neq 0 \\ 0, & sonst \end{cases}$$

basierend auf den normierten Ergebnissen der didaktischen Faktoren ($\bar{x_i}$) zu berechnen. $w_1, \dots, w_n$ stellen dabei die Gewichte der didaktischen Faktoren da und können über die ELAI-UI justiert werden. Dabei ist es möglich, den Wert des didaktischen Faktors invers in die Klassifikationsberechnung einzubeziehen, d.h. je höher/ niedriger der Wert des didaktischen Faktors, desto schlechter/ besser das Klassifikationsergebnis. Hierfür muss das Gewicht vom Tutor negativ angegeben werden. Ein solcher Fall tritt beim didaktischen Faktor ‚taskHelpingCount' auf. Eine erhöhte Anzahl an Hilfen soll einen negativen Einfluss auf die

Klassifikation haben. Das Ergebnis der Klassifikationsberechnung dient als Genrefähigkeit sowie als inverser prozentualer Wert für den Hilfelevel. Für die Normierung werden Vergleichswerte vorheriger Lösungen der Aufgabe von allen Nutzern herangezogen. Die berechneten didaktischen Faktoren einer Aufgabe werden nach dessen Beendigung in einem spezifischen Statement im LRS gesichert. Diese Speicherung führt zu einer Performanzsteigerung, da nur diese speziellen Statements für die Normierung einer zukünftigen Klassifikationsberechnung herangezogen werden müssen. Unterschieden werden die Statements mithilfe der Interpretation-Engine, welche im folgenden Unterkapitel erläutert wird.

Statements – Intepretation Engine

Wie in Kapitel 3.3 erläutert, besitzen xAPI-Statements die Grundform Actor Verb Object. Sendet ein System ein xAPI-Statement, kann die ELAI durch das maschinenlesbare JSON-Format und den hinterlegten URIs das Statement eindeutig identifizieren. Diese Identifikation basiert auf dem Prinzip des *Semantic Web* und ermöglicht die Interoperabilität zwischen verschiedenen Systemen [75]. In Tabelle 2 und Tabelle 3 sind die Objekte/ Aktivitäten sowie die Verben aufgeführt, die der ELAI als Definitionen vorliegen.

Die Verben der Statements werden für die grundlegende Unterscheidung verwendet. Das Verb started wird genutzt um das Statement als Start eines Systems zu identifizieren. Als Gegenspieler hierzu dient das Verb terminated zur Beendigung des Systems. Weiterhin werden in Systemen verschiedene Aufgaben bzw. Missionen gestartet. Um die Statements für den Start eines Systems und den einer Aufgabe zu trennen, ist das Verb launched für den Start einer Aufgabe reserviert. Eine Aufgabe besitzt dabei die Eigenschaft erfolgreich oder erfolglos mit dem Verb completed beendet zu werden. Die Unterscheidung erfolgt mithilfe einer in den Metadaten gesetzten booleschen Variable. Die bisher vorgestellten Verben dienen dem Start und der Beendigung eines Systems oder einer Aufgabe. In einer Aufgabe können zudem Statements der Form played und asked interpretiert werden. Das Verb played wird dabei verwendet, um zu signalisieren, dass der Spieler gerade die Aufgabe spielt und sich bei einem bestimmten Aufgabenfortschritt befindet. Dieser Fortschritt wird prozentual in den Metadaten Result beschrieben. Eine weitere Interpretationsmöglichkeit innerhalb einer Aufgabe ist mithilfe des Verbs asked umgesetzt, welches beschreibt, dass der Spieler eine Hilfestellung bekam.

Wie in der deutschen Grammatik werden die Objekte/ Aktivitäten in einem Statement zur Zuordnung der Handlung verwendet. In Statements können diese weiter mithilfe von Parent-Activities beschrieben werden. Wird ein System gestartet, sollte mithilfe des Verbs started ein Statement der Form Actor started Genre with Parent-Activity = Game gesendet werden. Hierbei beschreibt das Objekt Gerne ein spezifisches Genre und wird mithilfe dessen Parent-Activity als Spiel (Game) deklariert. Ähnlich wird mithilfe der Aktivität Task eine Aufgabe beschrieben. Beispielhaft könnte die Aktivität mit der Form Actor launched Mission with Parent Activity = Task, Grouping Activity = Genre verwendet werden. Hierbei wird durch das Verb launched deutlich, dass eine Aufgabe mit der Identifikation Mission gestartet wurde. Zur Verdeutlichung, dass Mission eine Aufgabe darstellt, ist dessen Parent-Activity als Task spezifiziert. Um den Kontext weiter zu verdeutlichen, wird mit der Grouping-Activity die Einordnung zum gestarteten Spiel des Genres Genre einbezogen. Die letzten beiden Aktivitäten, die der ELAI als Definitionen vorliegen, sind Help und Alert. Help ist dabei eine Spezifizierung von Alert, was in der URI verdeutlicht wird (Tabelle 3). Verwendung finden diese zusammen mit dem Verb asked, um in einem Statement der Form Actor asked Help with Parent Activity = Alert, Grouping Activity = Genre zu beschreiben, dass der Spieler (Actor) Hilfe bekommen hat.

Um die Verben und Objekte sinngemäß zuordnen zu können, wird nachfolgend ein beispielhafter Verlauf eines Nutzers mit dem System dargestellt (Abbildung 4.6). In der Abbildung sind die einzelnen Statements des Verlaufs mit der Nummerierung (1) bis (8) angegeben. Als Kontext wird angenommen, dass der Akteur Peter Glider eine Aufgabe mit dem Namen Find the Tank T-42 aus dem Genre Seek and Find vollständig absolvieren will.

Startet der Spieler eine Aufgabe, wird zuerst der Kontext des Genres Seek and Find (1) sowie der Kontext des Spiels Find the Tank T-42 geöffnet (2). Um spieleübergreifend den Spielfortschritt verständlich dokumentieren zu können, wurde ein prozentuales Distanzmaß eingeführt, welches den Aufgabefortschritt als Ergebnis (Result) eines Statements beschreibt. Dies wird in Statement 3 und 4 dargestellt. In diesen nähert sich Peter Glider dem Aufgabenziel der Aufgabe Find the Tank T-42 um 0.02 % auf 0.32 % an. Weiter kann der Spieler Hilfe/ Tipps für seine Aufgabe anfordern, was im 5. Statement dargestellt wird. Beendet der Spieler die Aufgabe oder bricht er diese ab, wird zur Signalisierung der Beendigung des Aufgabenkontextes ein Statement mit der eingestellten Schwierigkeit und dem Erfolg der Aufgabe gesendet. Der eingestellte Schwierigkeitslevel wird ebenfalls als Prozentwert im Bereich Null bis Eins angegeben. Somit ist die Menge an Schwierigkeitsgraden für jedes System individualisiert nutzbar. Wurde die Aufgabe erfolgreich beendet, sendet die ELAI selbstständig – auf das completed Statement (6) folgend – ein performed Statement (7) (Abbildung 4.3). Dieses beinhaltet die berechneten didaktischen Faktoren eines Spielverlaufs, welche für spätere Klassifikationen benötigt werden. Zuletzt wird zur Signalisierung der Beendigung des Genre Kontextes ein weiteres Statement gesendet (8).

Tabelle 2 Definierte Verben

Name (en-US)	URI
started	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/verb/started
terminated	http://activitystrea.ms/schema/1.0/terminated/
launched	http://adlnet.gov/expapi/verbs/launched/
completed	http://adlnet.gov/expapi/verbs/completed/
played	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/verb/played
asked	http://adlnet.gov/expapi/verbs/asked/
performed	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/verb/performed

Tabelle 3 Definierte Aktivitäten

Name (en-US)	URI
Game	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/activity/game/
Task	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/activity/task/
Help	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/alert/help/
Alert	http://activitystrea.ms/schema/1.0/alert/

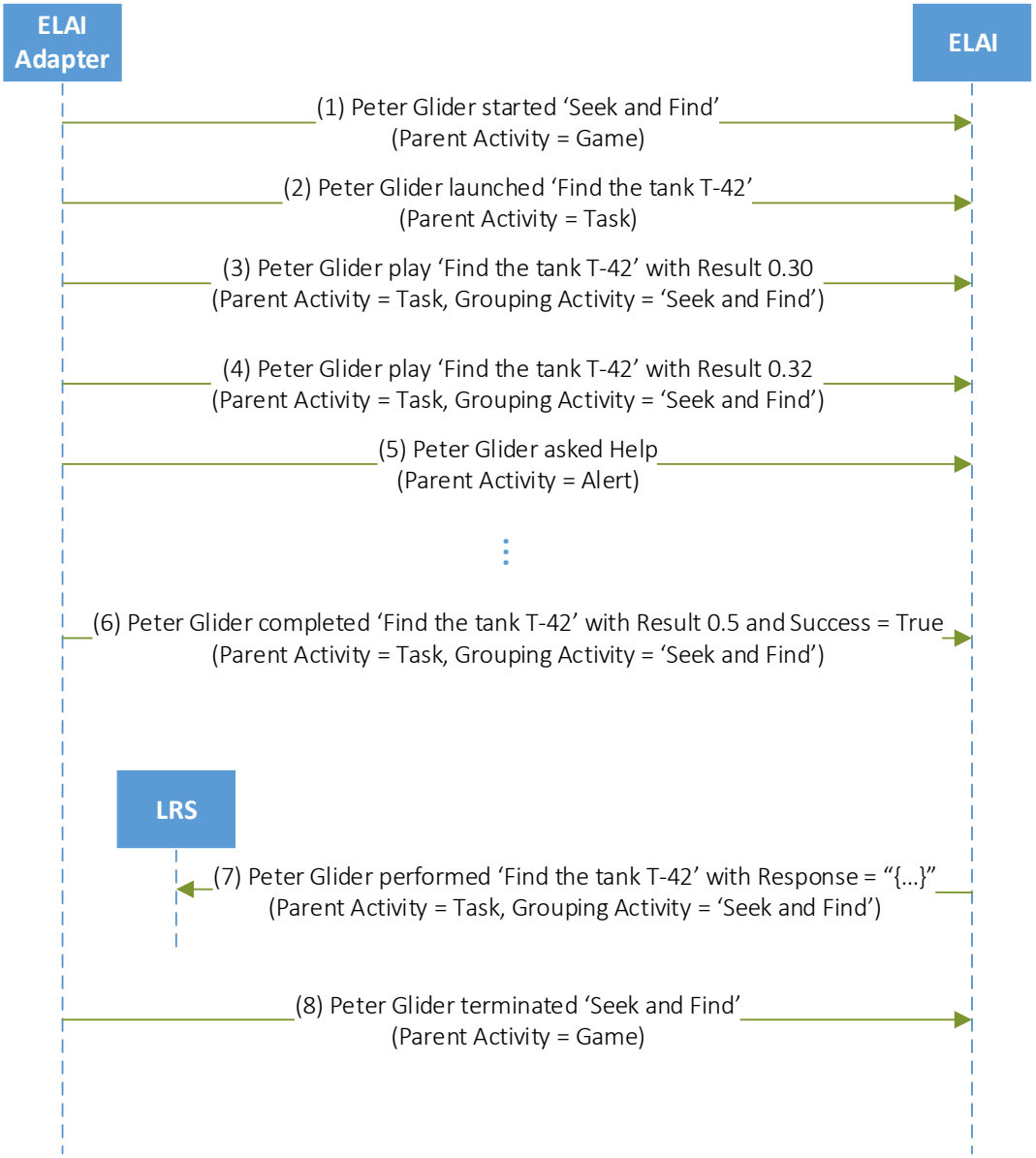


Abbildung 4.6 Beispielhafter Statement Verlauf

4.2. ELAI-UI (Web-UI)

Die ELAI-UI oder das ELAI-Tutoring-System dient dem Nutzer (Tutor) zum einen zur Analyse der ELAI-Daten und zum anderen als Werkzeug zur Steuerung und Adaption verschiedener ELAI-Parameter.

Framework

Zur Realisierung der Oberfläche und der damit enthaltenen Logik der Web-Schnittstelle wurde das Vaadin Framework (Kapitel 3.4) verwendet. Gründe für den Einsatz von Vaadin sind unter anderem auf der serverseitigen UI-Logik zu sehen. Einen ausführlichen Vergleich mit anderen Webentwicklungstools stellt Vaadin, Ltd. auf ihrer Homepage bereit [76]. Clientseitige Module benötigen eine dauerhafte Kommunikation mit dem Server durch so genannte RCP-Calls. Die serverseitig getriebene Entwicklung von Vaadin ermöglicht es, die UI-Logik auf die Serverseite zu verlagern und so die Kommunikation zwischen Client und Server gering zu halten [57]. Dies prädestiniert Vaadin für AJAX (Asynchronous JavaScript and XML) Technologien in besonderer Weise, wodurch das Entwerfen von Rich Internet Applications (RIA) ermöglicht wird [57]. Weiterhin lässt die Entwicklung in Java den leichten Einsatz von z.B. JUnit-Tests zu und die Wahl des Applikations-Stacks offen. Weitere große Vorteile bietet Vaadin durch die automatische Kompatibilität mit verschiedenen Internetbrowsern und mobilen Betriebssystemen mit verschiedenen Bildschirmgrößen (Responsive Webdesign). Dies bedeutet für den Entwickler im Vergleich zur herkömmlichen Webentwicklung eine enorme Zeitersparnis. Zudem wird das Framework von über 570 Add-ons unterstützt [67], was die Entwicklung erleichtert. Ein weiterer wichtiger Punkt ist außerdem die freie kommerzielle Verwendung des Produkts. Dabei ist zu erwähnen, dass Vaadin, Ltd. eine kostenpflichtige Version mit zusätzlich integrierten Tools im Rahmen eines Monatsabonnements anbietet. Dieses enthält unter anderem das Tool Vaadin Charts und den Vaadin Designer, mit welchem die Oberfläche mit Drag-and-Drop Elementen gestaltet werden kann [67].

Grundstruktur

Für den ersten Schritt des Aufbaus der Web-Oberfläche galt es, die Grundstruktur der Arbeit festzulegen. Hierfür wurde das in der Vaadin-Community weit verbreitete *Model-View-Presenter* (MVP) Pattern (Kapitel 3.5) zur logischen Trennung der Datenhaltung, Ansicht und Logik verwendet. Im Unterschied zum bekannten MVC-Pattern besteht keine Kommunikation zwischen dem Model und der View, welche ausschließlich über den Presenter (Controller) realisiert wird. Aus dem eingesetzten MVP-Pattern ergab sich zugleich die Ordnerstruktur des Projektes. Diese beinhaltet für jede Ansicht ein gleichnamiges Package, welches für jede Implementierung auch einen dazugehörigen Presenter für dessen Logik, eine View Komponente zur Darstellung sowie ein Model für den Zugriff auf die Daten beinhaltet. Um die strikte Separation und die Einhaltung der Verbindungen der MVP-Klassen (Abbildung 3.11) zwischen Model und Presenter sowie View und Presenter zu gewährleisten und zudem ein Methodengrundgerüst zu bieten, wurden eine Model, eine View und eine Presenter Klasse erstellt, welche als Superklassen für jede Ansicht verwendet wurden. Eine Ansicht stellt dabei eine Webseite der ELAI-UI dar, die über das Menü erreicht werden kann und den Inhalt der Seite präsentiert. In Abbildung 4.7 ist das Basis-Layout der ELAI-UI verdeutlicht. Bei dem Entwurf des Layouts wurde auf grundsätzliche Usability-Regeln geachtet [77][78]. Zum einen wird versucht, eine visuelle Hierarchie durch die Einteilung in die drei Hauptteile *Header*, *Menu* und *Detailansicht* zu erreichen [77]. Weiterhin werden Ablenkungen durch eine einheitliche Farbe des Menüs und des Headers vermieden und so die Detailansicht fokussiert [77]. Zur besseren Lesbarkeit wurde, wie von Manhartsberger empfohlen, eine serifenfreie Schrift eingesetzt

sowie auf den Kontrast zwischen Text und Hintergrundfarbe geachtet [78]. Für eine benutzerfreundliche Bedienung wurde die Navigationsleiste dauerhaft sichtbar gestaltet, sodass diese auch beim Scrollen der Detailansicht immer ersichtlich bleibt. Weiterhin ist derjenige Menüpunkt, welcher als Link zur Seite, auf welcher sich der Nutzer momentan aufhält, in gleicher Farbe wie die Detailansicht hervorgehoben [78]. Um zusätzlich dem Nutzer das Gefühl zu vermitteln, die Seite nicht zu verlassen, wird bei Aufruf einer Unterseite nur der Inhalt der Detailansicht erneut geladen.

Im Rahmen dieser Ausarbeitung wurden Komponenten implementiert, welche in verschiedenen Ansichten wiederverwendet wurden. Die dafür benötigte komponentenübergreifende Kommunikation zwischen den Ansichten wurde mithilfe des Event Bus Pattern aus der Vaadin-MVP-Lite Bibliothek gelöst [69]. Die Event Handler wurden in diesem Projekt stets im Presenter der MVP Architektur implementiert. Die Kommunikation der Views wird somit über die Presenter realisiert, was einer Einhaltung des MVP Prinzips durch die ausschließliche Logik im Presenter entspricht.

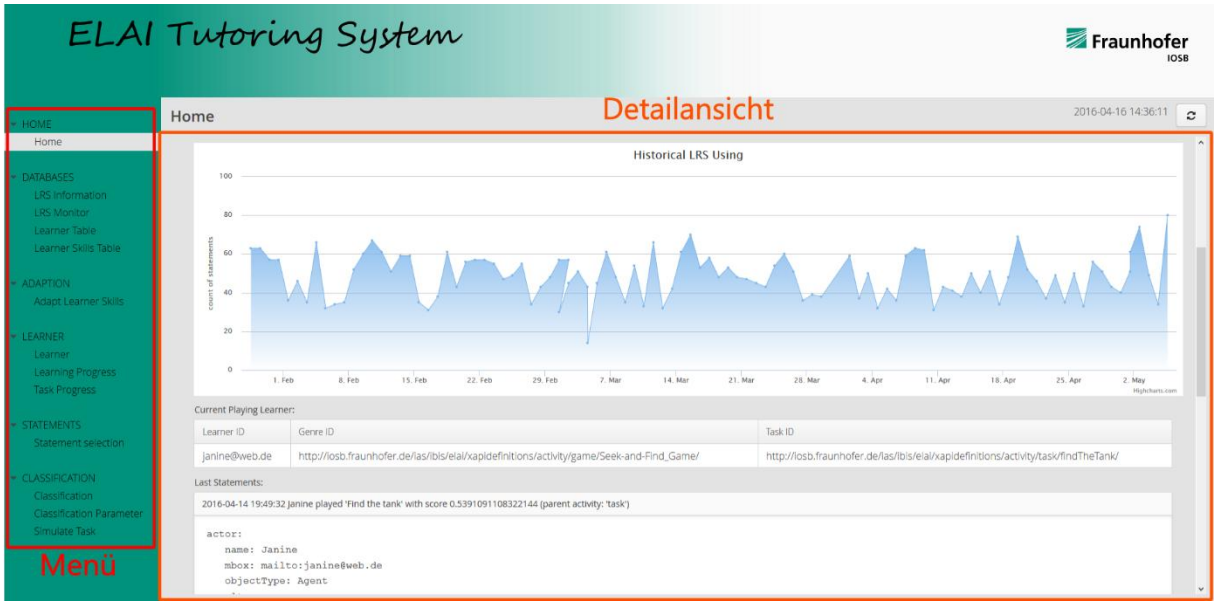


Abbildung 4.7 ELAI-UI Basis-Layout Darstellung in der Home Ansicht

Ansichten der Webseite

Auf den folgenden Seiten werden die verschiedenen Ansichten der Webseite vorgestellt und auf dessen Einsatzzweck eingegangen. Aus Gründen der Übersichtlichkeit wurde nur bei ausgewählten Ansichten eine Abbildung der Detailansicht der Webseite beigefügt. Die restlichen Abbildungen können in Kapitel 5.1 betrachtet werden. Zur Datenvisualisierung in Form von Charts wurde das Highcharts API Add-on von M. Huber eingesetzt [79], welches wiederum das Tool Highcharts verwendet, was für nicht kommerzielle Einsätze frei verfügbar ist [80].

Beim Start der Webseite durch Klicken auf das Home-Element in der Navigationsleiste oder durch Klick auf den ‚ELAI Tutoring System' Schriftzug im Header wird der Tutor auf die **Home** Ansicht geleitet. Diese soll dem Nutzer einen Eindruck über den Einsatzzwecks der ELAI-UI liefern. Hierfür wird diesem unter anderem der zeitliche Verlauf der LRS Nutzung und eine Liste aller aktuell spielenden Nutzer an den angeschlossenen Systemen präsentiert.

Auf den Home Tab folgen die Datenbank-Ansichten. Diese dienen dem erleichterten Zugriff auf deren Einträge und auf zusammenfassende Informationen. Für die LRS-Datenbank können die Ansichten **LRS Information** und **LRS Monitor** genutzt werden. Um dem Nutzer

die Möglichkeit offen zu halten, selbst auf der Webseite des LRS die Daten zu betrachten, werden in der Ansicht LRS Information die Login-Daten des LRS dargestellt. Weiter werden unter LRS Monitor die letzten 100 Statements im JSON-Format dargestellt, um dem Nutzer die Möglichkeit der Betrachtung über den Eingang von Einträgen in das LRS zu bieten und deren Inhalte analysieren zu können. In den weiteren zwei Ansichten werden die Einträge der Lerner-Datenbank präsentiert. Diese stellen unter **Learner Table** die eingetragenen Nutzer sowie unter **Learner-Skills Table** deren Fähigkeitswerte in bestimmten Genres dar.

Für die genreabhängige Adaption der Lerner-Fähigkeiten und des Hilfelevels kann der Tutor in der Ansicht **Adjust Adaptation Level** diese anpassen und in der Datenbank sichern. Somit kann adaptiv auf die aktuellen Spiele Einfluss genommen werden, da die angeschlossenen Spiele über den ELAI-Adapter, die Einträge in regelmäßigen Abständen anfordern und system-eigene Spielanpassungen vornehmen.

In einer weiteren Menükategorie wurden verschiedene Ansichten implementiert, um Daten einzelner Lerner analysieren zu können. Für einen Überblick der verschiedenen Lerner und deren Aktivitäten wird unter der Ansicht **Learner** in einer Tabelle die E-Mail Adresse, der Name, die Anzahl abgesetzter Statements sowie das Datum des ältesten und neuesten Statements angezeigt. Ein wichtiger Aspekt der Analyse ist die Darstellung des Lernfortschritts eines ausgewählten Nutzers in einem bestimmten Gebiet. In der Ansicht **Learning Progress** wird ein Vergleich zum Durchschnitt aller Lerner über die Zeit graphisch dargestellt. Weiter kann der Nutzer den Verlauf einer einzelnen absolvierten oder gerade absolvierenden Aufgabe eines Lerners im Menüpunkt **Task Progress** betrachten. Dabei wird die prozentuale Entfernung zum Aufgabenziel aus den `play` Statements angezeigt. Das Distanzmaß wird dabei vom Spiel definiert und gibt den Spielfortschritt innerhalb einer Aufgabe an. Weiterhin werden die angeforderten Hilfen den jeweiligen Zeitpunkten hinzugefügt. Hierbei ist zu erwarten, dass sich der Spieler dem Ziel nähert, nachdem er eine Hilfestellung erhalten hat. Mithilfe des refresh-Buttons kann der Tutor den Verlauf eines aktuellen Spiels aktualisieren.

Im Anschluss an die Learner Ansichten steht dem Nutzer die selektierbare Betrachtung der Statements unter **Statement Selection** zur Verfügung (Abbildung 4.8). Diese Ansicht soll eine Analyse der Statements im LRS ermöglichen. Dabei können die Statements nach den Eigenschaften Verb, Objekt, Parent Activity und Grouping Activity gefiltert werden. Die Filter beeinflussen dabei direkt einen Balkenchart, der die Menge an gefilterten Statements eines Lerners im Vergleich zu allen Lernern darstellt. Für eine detaillierte Betrachtung steht dem Nutzer zudem die Veranschaulichung der gefilterten Statements im JSON-Format zur Verfügung. Werden mehrere Filter-Eigenschaften des gleichen Typs (z.B. Verb) verwendet, werden diese mit einer OR-Funktion und Filter verschiedener Eigenschaften mit einer AND-Funktion verknüpft.

Die letzte Menükategorie dient der Klassifikationsbetrachtung sowie der Auswahl von Klassifikationsparametern. Im Menüpunkt **Classification** werden dem Tutor 2D-Charts präsentiert, welche das Klassifikationsspektrum der ausgewählten Aufgabe darstellen. Diese Ansicht dient dem Tutor zur Veranschaulichung der Klassifikationsverteilung aller Spieler. Da die Klassifikation (Kapitel 4.1) auf den didaktischen Faktoren der jeweiligen Aufgabe beruht, können diese als Achsen des Graphs gewählt werden. Aufgrund der eingeschränkten Darstellungsmöglichkeiten können dabei nur bis zu drei Parameter gewählt werden. Als weitere Ansicht werden unter **Classification Parameter** die für die Normierung der Klassifikation heranzuziehenden `performed`-Statements eingeschränkt. Diese können zeitlich und/ oder nach Menge gefiltert werden. Die zeitliche Einschränkung bietet dabei die Möglichkeit der Auswahl des Zeitraums oder der dynamischen Filterung nach den Statements der letzten x-Tage. Für eine Kontrolle des Filters werden maximal die letzten 100 `performed`-Statements der gefilterten Liste in Kurzschreibweise angezeigt. Eine weitere Kontrolle der

ausgewählten Parameter bietet die letzte Ansicht der Webseite – **Simulate Task** (Abbildung 4.9). In dieser kann der Tutor durch Eingabe der didaktischen Faktoren für eine bestimmte Aufgabe einen Spielablauf simulieren, um das Ergebnis der Klassifikation abhängig der Klassifikationsparameter zu testen. Durch Betätigen des Run-Buttons wird das Klassifikationsergebnis sowie der passende Stereotyp angezeigt.

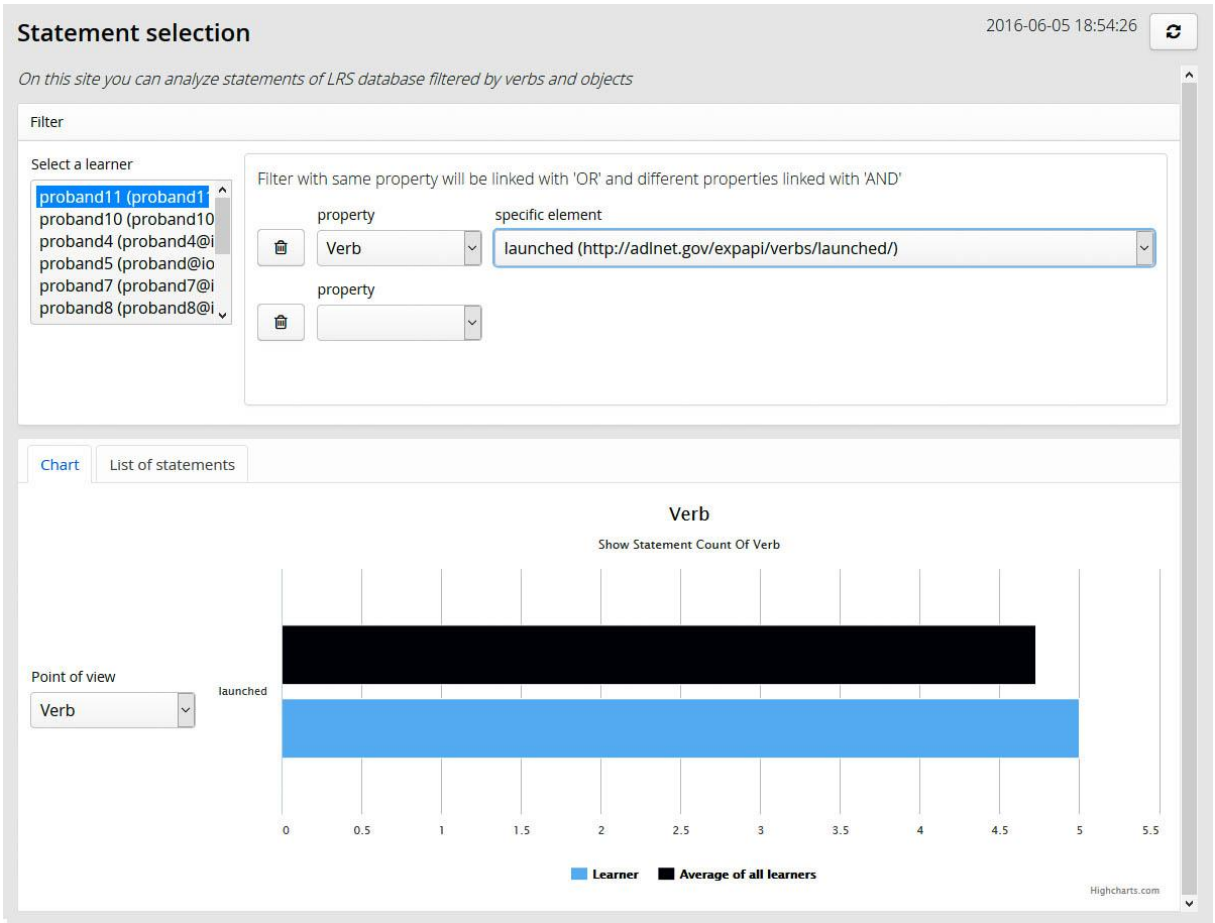


Abbildung 4.8 ELAI-UI Statement Selection Detailansicht

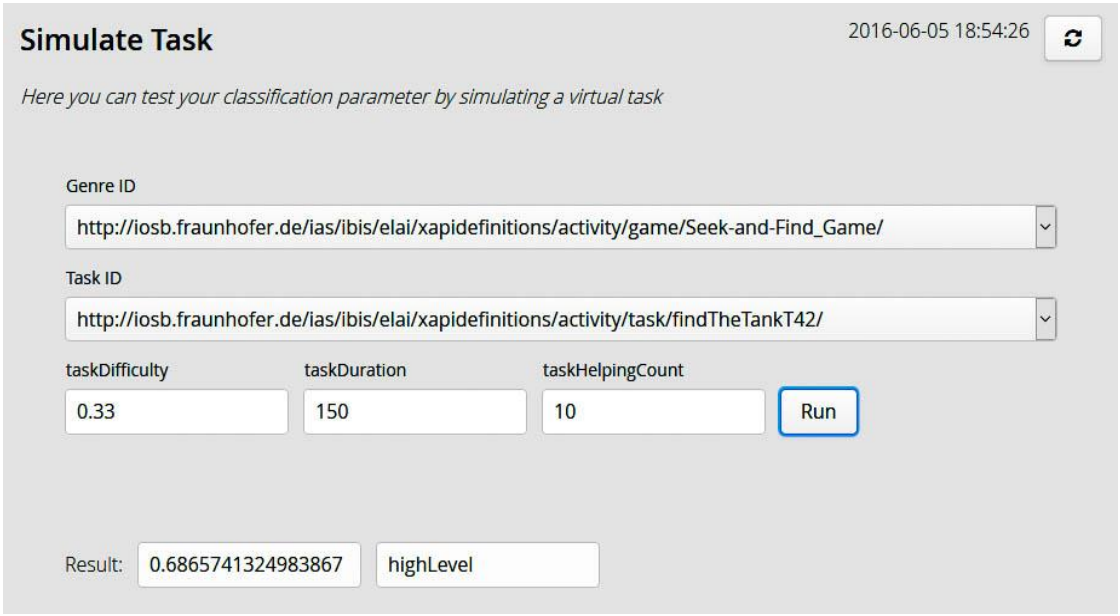


Abbildung 4.9 ELAI-UI Simulate Task Detailansicht

4.3. Entwicklungsprozess

In diesem Abschnitt wird die Umsetzung der einzelnen Komponenten des ELAI-Konzeptes erläutert. Der Fokus liegt dabei auf den verwendeten Tools zur Implementierung, die Vorgehensweise der Arbeit, den durchgeführten Tests und dem Einsatz von Konfigurationselementen.

Entwicklungswerkzeuge und Tools

Die auf Java 1.8 basierte Entwicklung der ELAI und der ELAI-UI wurde mithilfe der Entwicklungsumgebung Eclipse in der Version 4.5 umgesetzt. Dabei wurden für die Projekte das Framework Vaadin sowie das Dependency-Management-System Ivy verwendet. Für die Bereitstellung der Webanwendung sorgte ein Tomcat Server in der Version 7. Für die Weiterentwicklung des SaFIR Spiels sowie für die Neuentwicklung des ELAI-Adapters fand die Game Engine Unity 5.3.4 und die in C# verwendete Entwicklungsumgebung MonoDevelop sowie Visual Studio 2013 ihren Einsatz.

Konzeption

Zu Beginn wurde ein Struktur- und Zeitplan der Masterarbeit angefertigt. Diese erwiesen sich im Verlauf der Arbeit als sinnvoll, um die Planung und den Zeitrahmen einzelner Arbeiten einschätzen zu können. Weiterhin wurde im Rahmen von regelmäßigen Treffen mit dem betreuenden Mitarbeiter das Konzept der einzelnen Komponenten und deren Zusammenspiel diskutiert und mithilfe von Mockups die Gestaltung der ELAI-UI Weboberfläche vorab diskutiert. Mithilfe der Mockups konnten, wie von Balzert et al. empfohlen, bereits vor der Implementierung erste Usability-Tests durchgeführt und die daraus resultierenden Verbesserungen eingearbeitet werden [81]. Aus der Entscheidung, xAPI Statements zur Kommunikation zu verwenden, folgte weiter eine notwendige komponentenübergreifende einheitliche Interpretation der xAPI-Statements, welche in Absprache mit dem betreuenden Mitarbeiter zu Beginn der Arbeit festgelegt wurden (Kapitel 4.1).

Implementierung

Nachdem die Vorbereitungen für die Implementierung und das grundlegende Konzept ausgearbeitet waren, konnte mit der Entwicklung der Benutzeroberfläche begonnen werden. Zu Beginn lag der Fokus auf der Implementierung des Menüs und des Grundlayouts, in welchem die verschiedenen Teilansichten eingefügt werden (Abbildung 4.7). Anschließend wurden die Layout-Elemente grob umgesetzt und im späteren Verlauf vervollständigt. Durch die agile Entwicklung war es zu einem frühen Status möglich, einen Eindruck der Oberfläche zu gewinnen und Bewertungen frühzeitig einzubinden. Zur Entwicklung der Oberflächenkomponenten wurden ausschließlich die von Vaadin bereitgestellten Widgets sowie das Addon Highcharts API von M. Huber für die Erstellung von Charts genutzt [79]. Für eine Evaluation der Oberfläche im Verlauf der Entwicklung wurde ein Skript geschrieben, welches automatisch Dummy-Datensätze generiert und diese in einem LRS speichert. Somit konnten fehlerhafte Darstellungen in Tabellen oder in Charts zum Entwicklungszeitpunkt entdeckt und behoben werden. Da die Entwicklung des Gesamtkonzeptes mit der ELAI-UI begonnen hatte, wurde die Schnittstelle als einfach auszutauschende Verbindung zu den Daten konzipiert. Diese implementierte zu Beginn eine direkte Verbindung zu einem dafür aufgesetzten LRS.

Ein weiterer wichtiger Entwicklungsschritt eines Softwareentwicklungsprozesses bildet das Testen von Funktionalitäten, da hierdurch Fehler vor dem Einsatz der Anwendung aufgedeckt

werden können. Während der Entwicklung wurden aus diesem Grund JUnit-Tests umgesetzt, welche das Verhalten wichtiger Methoden prüfen. Zur Überprüfung größerer zusammenhängender Bereiche wurden manuelle Testszenarien durchgeführt und so deren Funktionalität evaluiert. An dieser Stelle sei erwähnt, dass das Vaadin Framework zwar ein Werkzeug für automatische Oberflächentests bietet, auf dessen Einsatz allerdings aus zeitlichen Gründen sowie dem benötigten kostenpflichtigen Lizenzerwerb verzichtet wurde [67].

Nachdem die grundlegenden Komponenten der ELAI-UI implementiert waren, wurde mit der prototypischen Entwicklung der ELAI begonnen. Da im ELAI-Konzept vorgesehen ist, dass alleine die ELAI eine Schnittstelle zu den Daten besitzt, wurde an dieser Stelle die Anbindung an das LRS umgesetzt und die Nutzerdatenbank erstellt sowie mit Hibernate in das Projekt eingebunden. Nach erfolgreicher Entwicklung von JUnit-Tests zur Abfrage und Speicherung von Daten in den Datenbanken wurde mit der Umsetzung der Schnittstelle zur ELAI-UI über Servlets begonnen. Diese beinhalten nur eine geringe Logik und waren schnell implementiert. Im Vergleich waren die Servlets für die Spieleschnittstellen aufwendiger, da diese die Berechnungen für die Adaption des Spiels enthalten. Aufgrund der Festlegung der Datenstruktur über xAPI-Statements und deren Interpretation, konnte von einem bestimmten Eingangsformat ausgegangen und die Antwort der Servlets schon vor der Implementierung eines Spiels umgesetzt werden.

Für die Evaluation der ELAI und der ELAI-UI wurde anschließend mit der Weiteentwicklung des SaFIR Serious Games und des ELAI-Adapters begonnen (Abschnitt 6.1). Hierbei ergab sich die Problematik, dass es aufgrund der benötigten Bearbeitungszeit der Servlet-Anfragen im Spiel zu Wartezeiten kam. Dieses Problem wurde mithilfe von Coroutinen der Klasse `MonoBehaviour.StartCoroutine` gelöst, die nicht direkt auf den Response des Servlets warten, sondern mithilfe der Rückgabe des `yield` Keywords ein `IEnumerable` Objektes zurückgeben, sobald die Antwort des Servlets eingetroffen ist. Dies ermöglicht einen ununterbrochenen Spielfluss. Die Funktionalität des Adapters und dessen Einfluss auf das Spiel wurde zusammen mit der ELAI Umsetzung durch Testdurchläufe evaluiert. Ebenfalls beinhaltet die Evaluation den direkten Einfluss der ELAI-UI auf das Spiel und die Analyse der Daten mithilfe der ELAI-UI.

Um die Schnittstellenkonfiguration aller Komponenten variabel zu halten, wurden Konfigurationsdateien in der ELAI, der ELAI-UI sowie des ELAI-Adapters eingesetzt. Diese können vom Nutzer vor dem Start des Systems modifiziert werden (siehe Anhang).

Kapitel 5
Anwendungsszenario für die agentenbasierte Kontrolllogik

Dieses Kapitel dient der Darstellung und der Evaluation des Ablaufs einer möglichen Analyse und Adaption mithilfe der ELAI-UI. Zur Einordnung des Anwendungsszenarios in das ELAI-Konzept ist dieses in Abbildung 5.1 illustriert. Im ersten Unterkapitel werden mithilfe eines beispielhaften Szenarios der Verlauf einer Analyse von Nutzerdaten sowie der adaptive Einsatz der Oberfläche dargestellt. Das Kapitel schließt mit einer Erklärung über den Einsatz und die Implementierung der ELAI-UI.

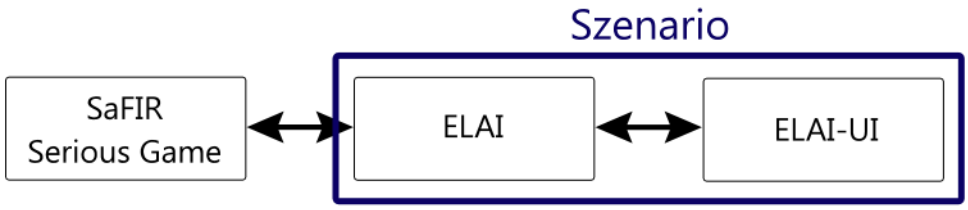


Abbildung 5.1 Einordnung des Szenarios in das ELAI-Konzept

5.1. Durchführung

Um ein Szenario einer Tutor-Interaktion mit der ELAI-UI realitätsnah wiederzugeben, muss zuerst dessen Kontext diskutiert werden.

In diesem existieren zwei Personen. Zum einen Alice (alice@web.de), die Spielerin des Serious Games und zum anderen der Tutor Bob, welcher mit der ELAI-UI interagiert. Alice ist dabei eine Auszubildende im Bereich Bildauswertung der Deutschen Bundewehr. Zum Trainieren ihrer Fähigkeiten soll sie im Rahmen ihrer Ausbildung das vom Fraunhofer IOSB entwickelte SaFIR Serious Game spielen. Bob ist wiederum ein Mitarbeiter des Fraunhofer IOSB, der zum Einstellen der Parameter sowie zur Einsichtnahme der Spieler-daten bemächtigt ist. Das ELAI System befindet sich noch in der Beta-Entwicklungsphase und benötigt daher noch das manuelle Anpassen von Parametern. Es wurde vor einem Monat gestartet.

Um aktuelle Nutzerdaten zu analysieren und die Parameter der Klassifikation anzupassen, startet der Tutor Bob die ELAI-UI Webseite im Browser, woraufhin ihm dessen **Home** Ansicht präsentiert wird (Abbildung 5.2). In dieser werden Bob unter anderem die Genre-Popularität

basierend auf der Anzahl an Statements im Learning Record Store (LRS), die zuletzt eingegangenen Statements sowie die Nutzungsaktivität des LRS angezeigt. Weiterhin wird dargestellt, dass der Nutzer mit der Identifikation *alice@web.de* gerade die Aufgabe *Search the tank* des Genre *Seek and Find* spielt. Um der Identifikation einen konkreten Namen zuzuordnen, wechselt Bob auf die Seite **Learner Table** (Abbildung 5.3). Der Tabelle mit den Spalten *Learner ID* und *Learner Name* entnimmt er, dass der Nutzer *alice@web.de* mit dem Namen *Alice* registriert ist. Im nachfolgenden Menüpunkt **Learner Skills Table** werden weiter die jeweiligen *Skill Level* und *Helping Level* Einstufungen der Nutzer verdeutlicht (Abbildung 5.4). Dieser Ansicht entnimmt Bob, dass Alice im Genre *Seek and Find* bzgl. ihres Fähigkeits-levels und Hilfelevels jeweils beim Wert 0,5 eingestuft ist.

Um die Adaptionsfähigkeiten der ELAI zu untersuchen, beschließt Bob, anhand des aktuell spielenden Nutzers diese zu beurteilen. Hierfür interessiert ihn auch, wie weit Alice im Spiel fortgeschritten ist, weshalb er die Seite **LRS Monitor** aktiviert (Abbildung 5.5). In einer erweiterbaren Ansicht werden Bob die letzten Einträge des LRS angezeigt. Um die Ansicht auf alle Statements von Alice zu reduzieren, filtert Bob mit dem Selektionselement die von Alice heraus. Über einen Klick auf die Kurzfassung des neuesten Statements wird der Tab auf die vollständige Darstellung im JSON-Format erweitert. Diese verdeutlicht, dass vor wenigen Sekunden ein `play`-Statement von Alice gespeichert wurde. Es verrät zudem, dass sie momentan bei einem Aufgabenfortschritt von 70 % ist. Weiter kann Bob anhand der Liste abzählen, dass Alice bereits seit einiger Zeit um diesen Fortschritt schwankt.

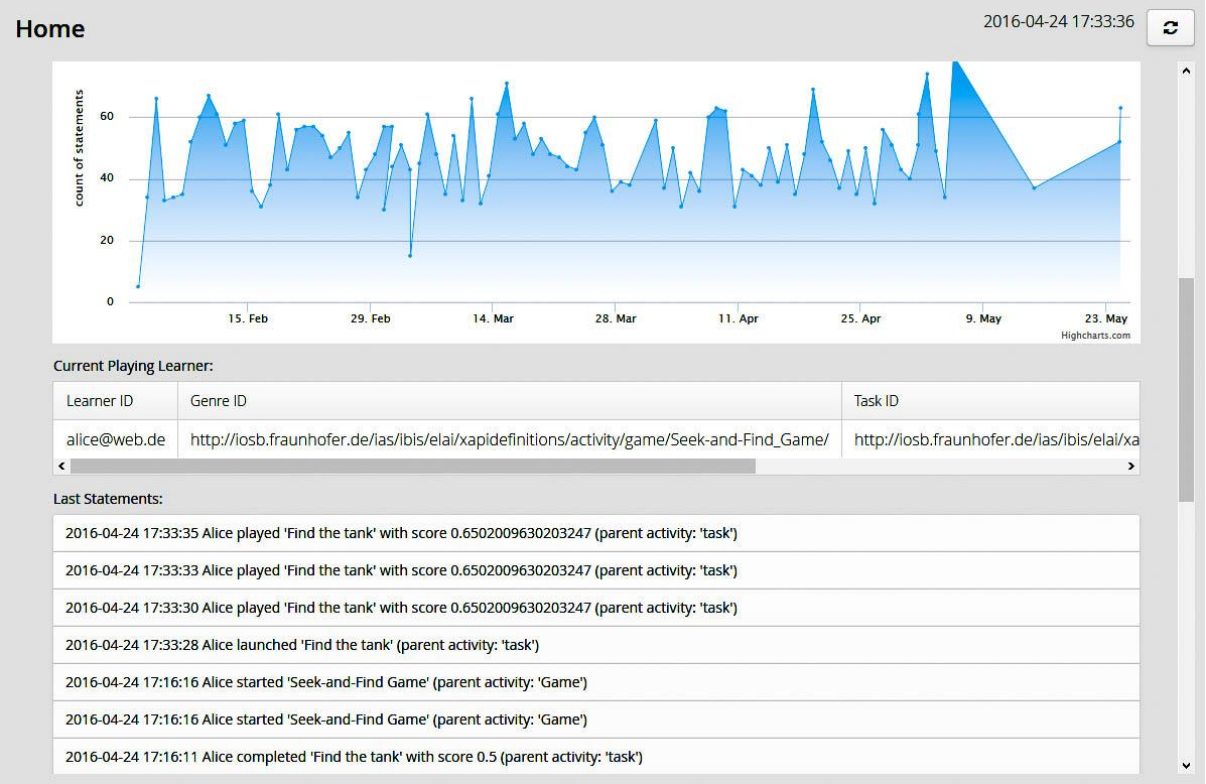
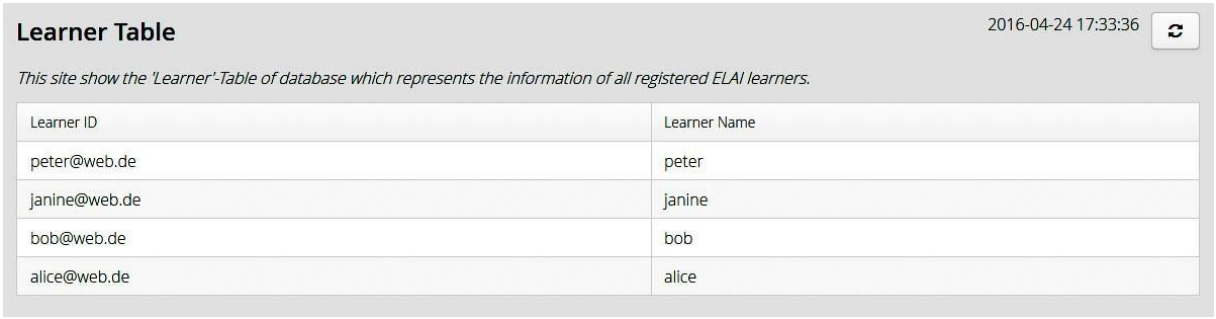


Abbildung 5.2 ELAI-UI Teilansicht der Home-Seite

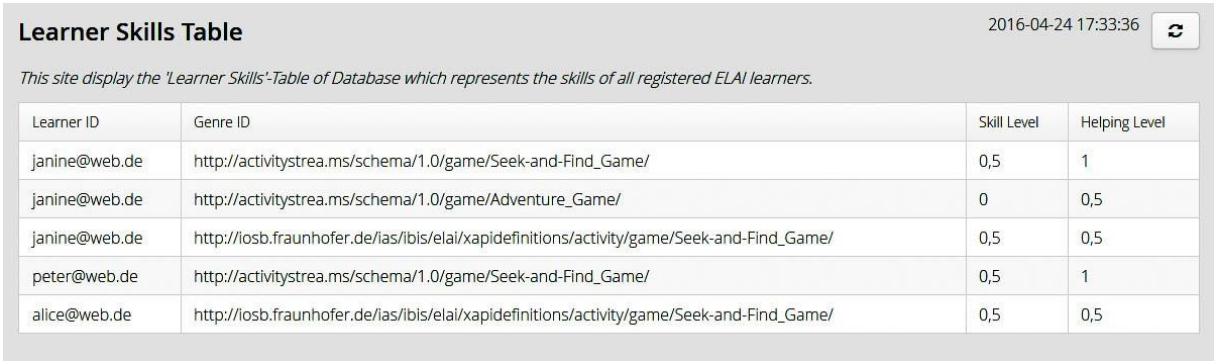


Learner Table 2016-04-24 17:33:36

This site show the 'Learner'-Table of database which represents the information of all registered ELAI learners.

Learner ID	Learner Name
peter@web.de	peter
janine@web.de	janine
bob@web.de	bob
alice@web.de	alice

Abbildung 5.3 ELAI-UI Learner Table Ansicht. Präsentiert alle in der Datenbank enthaltenen Lerner



Learner Skills Table 2016-04-24 17:33:36

This site display the 'Learner Skills'-Table of Database which represents the skills of all registered ELAI learners.

Learner ID	Genre ID	Skill Level	Helping Level
janine@web.de	http://activitystrea.ms/schema/1.0/game/Seek-and-Find_Game/	0,5	1
janine@web.de	http://activitystrea.ms/schema/1.0/game/Adventure_Game/	0	0,5
janine@web.de	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/activity/game/Seek-and-Find_Game/	0,5	0,5
peter@web.de	http://activitystrea.ms/schema/1.0/game/Seek-and-Find_Game/	0,5	1
alice@web.de	http://iosb.fraunhofer.de/ias/ibis/elai/xapidefinitions/activity/game/Seek-and-Find_Game/	0,5	0,5

Abbildung 5.4 ELAI-UI Learner Skills Table Ansicht.
Präsentiert die Fähigkeiten der Nutzer in den jeweiligen Genres

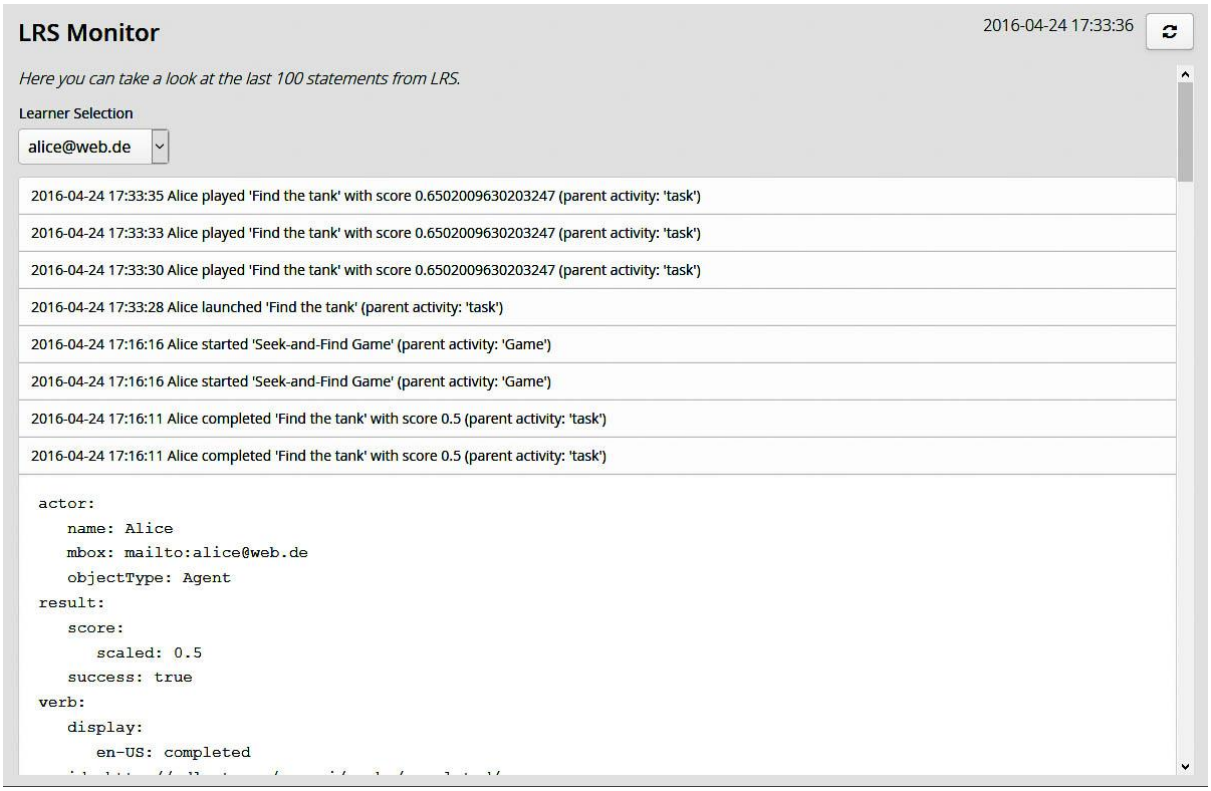


Abbildung 5.5 ELAI-UI LRS Monitor Ansicht. Filterung der letzten Statements nach dem Nutzer alice@web.de

Daraufhin beschließt Bob, den Verlauf der Spielerin weiter auf der Seite **Task Progress** zu beobachten (Abbildung 5.6). Nach Auswahl des Nutzers und der zuletzt gespielten Aufgabe wird ihm der Aufgabenfortschritt über der Zeit als Chart präsentiert. Auffällig dabei ist, dass Alice zuletzt des Öfteren Hilfen angefordert hat und wie erwartet um den Aufgabenfortschritt bei 70 % schwankt. Die Hilfen werden im zeitlichen Verlauf des Charts als Peaks an den jeweiligen Stellen angezeigt. Für eine direkte Hilfe beschließt Bob, ihren Hilfewert zu erhöhen, sodass Alice in ihrem Spiel eine bessere Hilfeleistung erhält. Hierfür wechselt er auf die Ansicht **Adjust Adaptation Level** (Abbildung 5.7). Nachdem er Alice als Lerner ausgewählt hat, werden ihm ihre verschiedenen Genres und deren Fähigkeits- sowie Hilfelevel in Form von festen Schiebereglern dargestellt. Unter diesen befindet sich jeweils ein justierbarer Regler, mit welchem der aktuelle Datenbankeintrag in Prozentwerten zwischen 0 und 100 verändert werden kann. Um Alice eine bessere Hilfe zu bieten, adaptiert der Tutor ihren Hilfelevel des

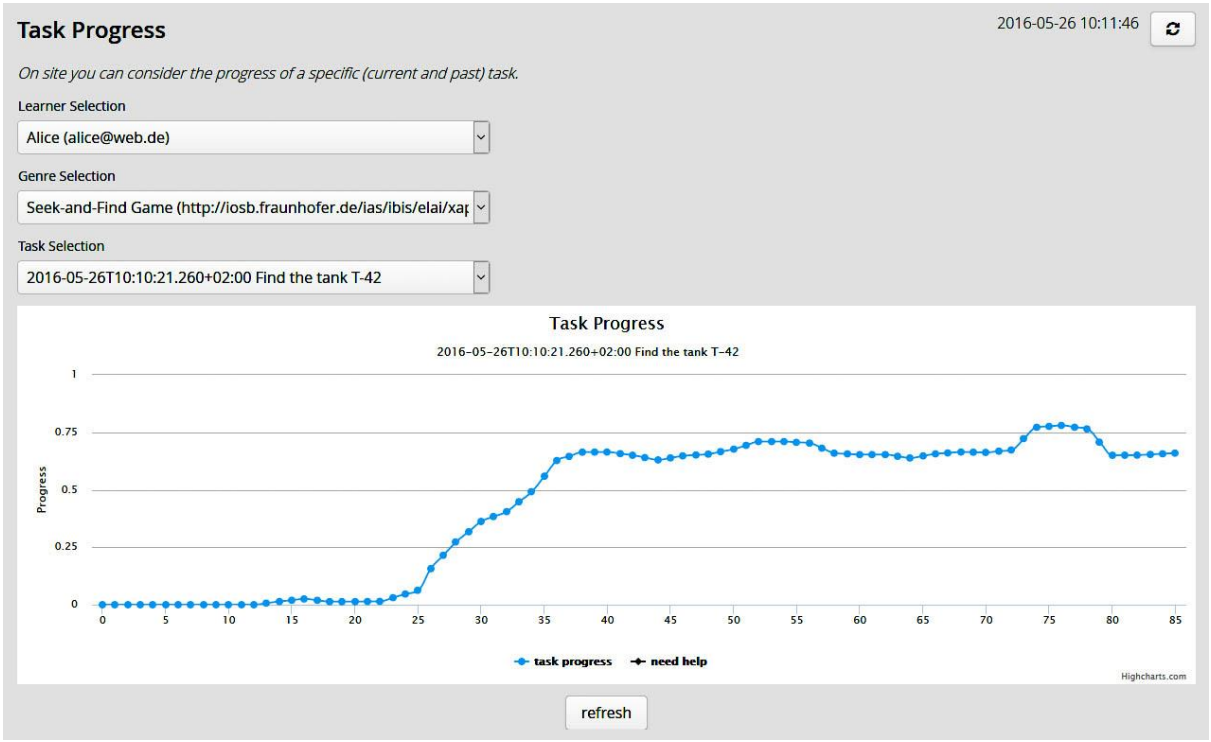


Abbildung 5.6 ELAI-UI Task Progress Ansicht von Alice die momentan eine Aufgabe vom Typ ‚Find the tank T42' spielt und um den Aufgabenfortschritt 70 % schwankt

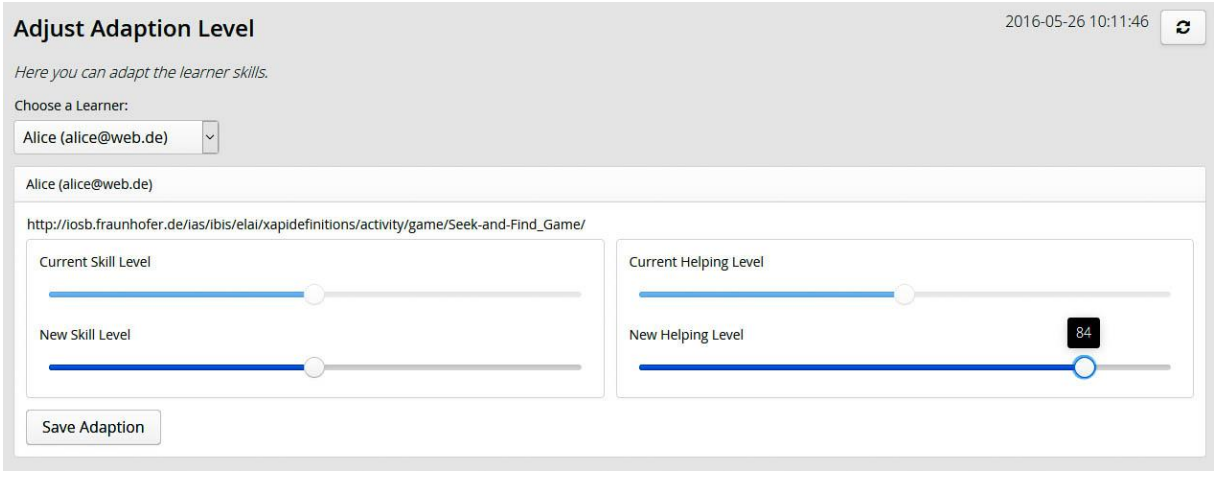


Abbildung 5.7 ELAI-UI Adjust Adaptation Level Ansicht. Alices Helping-Level wird auf den Wert 84 % erhöht

Genre *Seek and Find* auf der Ansicht zu einem höheren Wert und speichert seine Änderung durch betätigen des Save Adaption Buttons.

Zurück auf der **Task Progress** Seite wird ihm angezeigt, dass sich ihr Fortschritt noch nicht merklich verbessert hat. Nach ein paar Sekunden klickt er auf den Refresh-Button, worauf die Ansicht erneut lädt und einen aktualisierten Chart präsentiert. Auf diesem wird deutlich, dass sich der prozentuale Fortschritt, durch die nun verbesserte Hilfeadaption auf 80 % erhöht hat und sich Alice wieder weiter dem Ziel nähert. Somit war die Adaption erfolgreich. Um den Grund für die Fehlklassifikation des Hilfelevels zu bestimmen, betrachtet Bob auf der **Classification Parameter** Ansicht die Klassifikationsparameter (Abbildung 5.8). Dort wird ihm angezeigt, dass die didaktischen Faktoren für die Klassifikationsberechnung alle gleich gewichtet sind. Um seinen Verdacht, dass die Fehlerklassifikation von falschen Gewichtungen herrührt zu prüfen, wechselt er auf die **Classification** Ansicht (Abbildung 5.9). Auf dieser wählt Bob das Genre und die Aufgabe von Alice aus, woraufhin sich ein Chart präsentiert. Auf diesem sieht Bob die erfolgreich gespielten Aufgaben aller Spieler, die durch die Klassifikationsparameter gefiltert wurden. Für jede Aufgabe wurden die didaktischen Faktoren berechnet und im Chart auf die jeweiligen Achsen übertragen. Um die Klassifikation der Aufgaben sichtbar zu machen, sind diese jeweils abhängig vom Klassifikationswert in die farblich getrennten Gruppen *lowLevel*, *mediumLevel* und *highLevel* eingeteilt. Auf dem Chart erkennt er eine ungleichmäßige Verteilung der gespielten Aufgaben. Zurück auf der

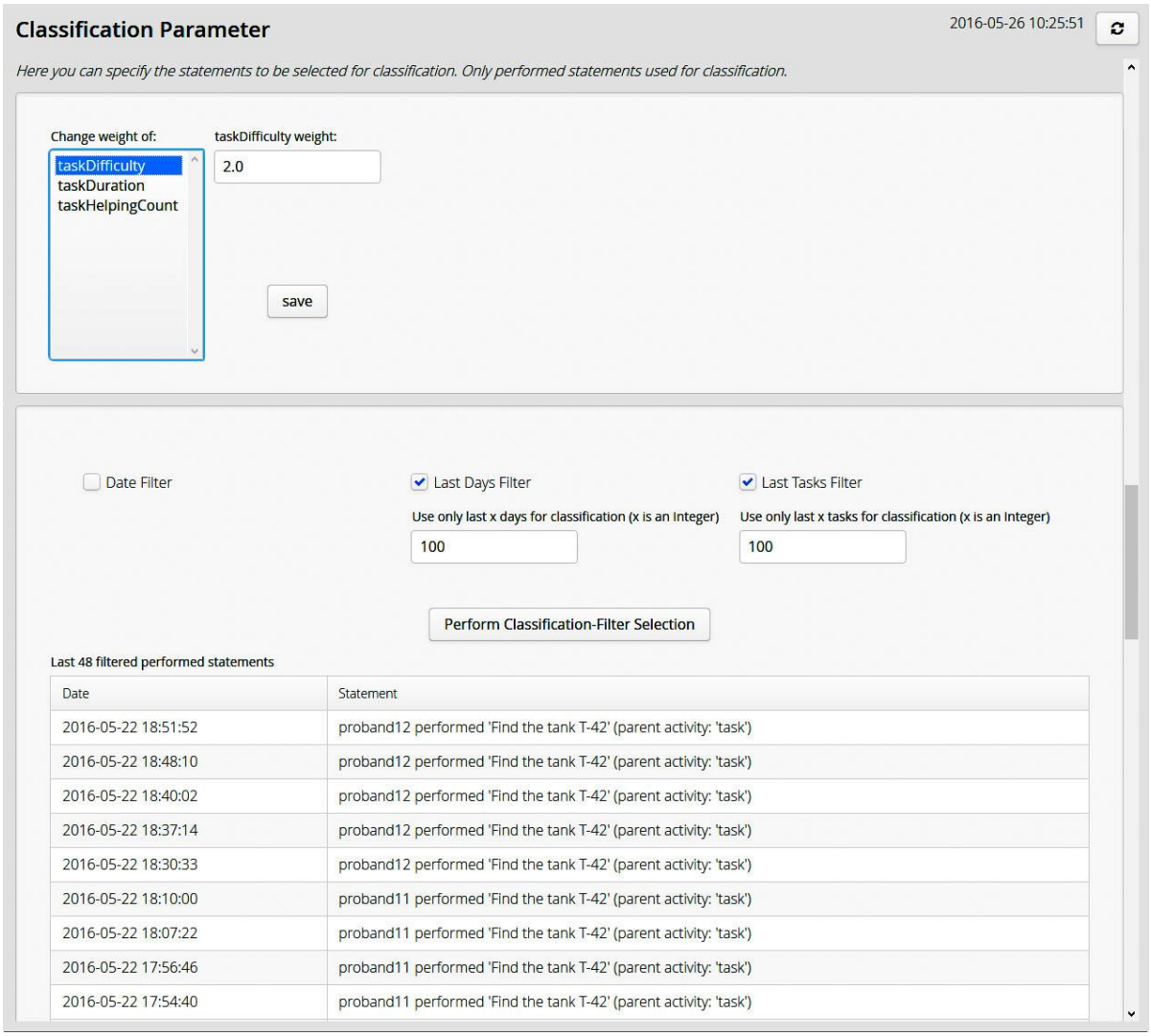


Abbildung 5.8 Classification Parameter; Oben: Einstellung der Gewichte der didaktischen Faktoren; Unten: Zeitlicher Filter der für die Klassifikation relevanten erfolgreich gespielten Aufgaben

Classification Parameter Ansicht setzt er die Variable *taskDifficulty* auf den Wert -2, *taskDuration* auf -4 und den Wert von *taskHelpingCount* auf -2. Weiter fällt ihm auf, dass die Normierungsberechnung der Klassifikation auf die letzten 50 Aufgaben des statischen Zeitraums vom 20.03.2016 bis zum 10.04.2016 beschränkt ist. Aufgrund seiner Mitarbeit in der Entwicklung des Projekts weiß Bob, dass ein dynamischer Wert der letzten 100 Tage und davon jeweils die letzten 100 Aufgaben eine bessere Klassifikation bieten. Nach Speicherung der Werte prüft er die neu eingestellten Klassifikationsparameter auf der **Classification** Ansicht. Auf dieser sieht er nun eine in etwa gleich verteilte Punktemenge auf den 2D-Charts.

Nun wechselt er auf die **Task Progress** Ansicht zurück und erkennt, dass Alice die Aufgabe fast beendet hat. Durch mehrmaliges Aktualisieren des Charts kann er den Anstieg des Aufgabenfortschritts auf 100 % sehen, was den Abschluss der Aufgabe indiziert. Da er weiß, dass mit jeder Beendigung einer Aufgabe die Klassifikation des Nutzers in dem jeweiligen Genre neu berechnet wird, betrachtet er im Anschluss die neu berechnete Klassifikation auf der **Learner Skills Table** Ansicht. Hier wird deutlich, dass Alice nun auf den Fähigkeitswert 0,33 und den Hilfewert 0,66 eingestuft wurde.

Um die Klassifikationsberechnung zu beurteilen, prüft Bob auf der **Classification** Ansicht, wie Alices Werte der letzten Aufgabe im Vergleich zu anderen absolvierten Aufgaben zu evaluieren ist. Für die Identifikation der zuletzt gespielten Aufgabe von Alice wählt er in den Filterelementen links des Charts Alice als Person und die zuletzt gespielte Aufgabe aus, woraufhin sich der Chart aktualisiert. Da sich Alice gut im Mittel des Klassifikationsspektrums der didaktischen Faktoren im Chart positioniert, ist Bob mit der neu eingestellten automatischen Klassifikation der ELAI zufrieden. Um den Einfluss der neuen Nutzerwerte von Alice zu betrachten, wechselt Bob auf die **Task Progress** Seite und wählt Alice als Lerner sowie die von ihr neu begonnene Aufgabe aus. Indem er den neuen Spielverlauf beobachtet, stellt er fest, dass Alice nun weniger Hilfen benötigt und sich der Aufgabenfortschritt fast gleichmäßig erhöht. Zufrieden mit seiner Analyse und den neuen Einstellungen verlässt Bob die Webseite.

Abbildung 5.9 Klassifikation mit den didaktischen Faktoren taskDifficulty und taskDuration; Auswahl der Person Alice mit allen gespielten Aufgaben (orangene Dreieckselemente im Chart)

5.2. Erklärung

Im vorigen Abschnitt wurde anhand eines Szenarios der Einsatz der ELAI-UI dargelegt. Hierbei wurde explizit auf Aktionen mit der ELAI-UI eingegangen. Um diese Betrachtung zu vervollständigen, wird in diesem Unterkapitel die technische Sicht anhand des Szenarios erläutert.

Zu Beginn startet der Tutor Bob mit der **Home** Ansicht der ELAI-UI. Auf dieser soll dem Nutzer ein Überblick über die Daten des LRS gegeben werden. Unter anderem ist hier zu sehen, welche Spieler aktuell an einem angeschlossenen System interagieren und eine Aufgabe spielen. Mit dieser Information kann der Tutor auf weiteren Seiten direkt nach diesen Nutzern und Aufgaben filtern und so die aktuell laufenden Spiele betrachten und die Adaption der ELAI überwachen. Weiter kann der Tutor auf den Datenbankansichten die aktuellen Einträge begutachten und hieraus Schlüsse über Nutzer der Systeme und der Adaption ziehen.

Nachdem Bob beschlossen hat den Verlauf von Alice weiter zu analysieren, wechselt er auf die Ansicht **Task Progress**. Auf dieser wird dem Tutor der Spielfortschritt in Form eines Liniencharts dargestellt. Hierbei ist der Fortschritt auf der Spielseite auf einen Prozentwert zu skalieren und über das play-Statement zu versenden. Da für den Aufgabenfortschritt die angenommen Hilfen ausschlaggebend sind, werden diese ebenfalls in dem Chart anhand von Peaks grafisch dargestellt. Die Task Progress Ansicht ist eine von vier Ansichten mit welchen Daten auf der ELAI-UI analysiert werden können. Weitere Ansichten sind L*earning Progress*, *Statement Selection* und *Classification*. Mithilfe der *Learning Progress* Seite kann der Tutor die Lern-Performance der Nutzer aus den Statements extrahieren und grafisch darstellen lassen. Mit dieser Information können Spiele und deren Lerninhaltspräsentation evaluiert werden. Eine weitere Möglichkeit, eine große Anzahl an Statements zu analysieren, liefert die Seite *Statement Selection*. Diese ermöglicht es, dem Nutzer die Menge an Statements nach Akteuren, Verben, Objekte, Parent-Activities und Grouping-Activities zu filtern. Hierbei wird dann die Mengenanzahl an gefilterten Statements bzgl. des gewählten Akteurs der Durchschnittsanzahl an Statements aller Akteure gegenübergestellt. Somit ist ein Vergleich bestimmter Statements möglich. Für eine genauere Betrachtung der gefilterten Statements können diese auch auf einem weiteren Tab der Seite betrachtet werden.

Auf der Task Progress Ansicht sieht Bob, dass Alice schon seit einiger Zeit um einen bestimmten Fortschrittswert schwankt. Daraufhin beschließt er, ihr mithilfe der Adaptionsmöglichkeit der ELAI-UI zu helfen. Da die ELAI-UI nicht direkt mit dem Spiel gekoppelt ist, kann Bob nur indirekt Einfluss auf das Spiel nehmen. Dies geschieht, indem er über angebotene Servlets der ELAI bestimmte Datenbankeinträge der Nutzerdatenbank überschreibt. Diese Einträge werden zyklisch über das Pull-Servlet der ELAI abgefragt, womit die indirekte Verbindung zur ELAI-UI hergestellt wird. Nachdem Bob die Datenbankeinträge in der **Adjust Adaption Level** Ansicht geändert hat, werden diese bei der nächsten Pull-Abfrage des Spiels in diesem geladen und verarbeitet, sodass Alice aufgrund ihres erhöhten Hilfelevels mehr Hilfe bekommt und somit im Spiel voranschreiten kann.

Bob vermutet, dass der Fehler von einer Fehlklassifikation stammt und somit eine unpassende Einstellung der Schwierigkeit und des Hilfelevels vorgenommen wurde. Um seine Vermutung zu bestätigen, öffnet er die **Classification** Ansicht. Sie stellt eine weitere Möglichkeit für einen Tutor dar, die Daten zu analysieren. Auf dieser wird die Performance der Spieler anhand definierter Faktoren klassifiziert. Die didaktischen Faktoren werden farblich je nach Klassifikationseinteilung auf einem 2D-Chart dargestellt. Hierbei kann der Tutor diejenigen didaktischen Faktoren wählen, welche für die Achsen verwendet werden sollen.

Nachdem Bob festgestellt hat, dass die vorliegende Klassifikation die Fähigkeitsklassen nur suboptimal trennt, wechselt er auf die **Classification Parameter** Ansicht. Auf dieser können

die Parameter für die Klassifikation sowie die Filter für die Selektion der verwendeten Daten eingestellt werden. Beide Einstellungen betreffen sowohl die ELAI wie auch die Berechnungen der ELAI-UI. Im oberen Bereich der Seite können die Gewichte der didaktischen Faktoren für die lineare Klassifikation verändert werden. Die Gewichte können dabei auch negativ in die Formel eingehen. Betrachtet man die drei didaktischen Faktoren *taskDifficulty*, *taskDuration* und *taskHelpingCount* fällt auf, dass bei einem hohen *taskDuration*- oder *taskHelpingCount*-Wert die Performance schlechter sein muss als bei einem niedrigen. Infolgedessen sollten diese Gewichte negativ in die Formel eingehen. Anders verhält es sich beim Faktor *taskDifficulty*, bei welchem ein hoher Wert auch ein hohes Klassifikationsergebnis zur Folge haben sollte. Im unteren Teil der Ansicht kann der Tutor diejenigen Aufgaben auswählen, die für die Normierung mit einbezogen werden sollen. Dabei kann zum einen ein Zeitintervall und eine maximale Anzahl an Aufgaben angegeben werden. Zum anderen kann das Zeitintervall wiederum als statischer Wert mit einem minimalen und maximalen Datum oder mit einem dynamischen Wert angegeben werden. Der dynamische Wert setzt das maximale auf das derzeitige Datum und das minimale auf das heutige Datum abzüglich der angegebenen Anzahl an Tagen, womit auch im dynamischen Fall ein Zeitintervall entsteht.

Nachdem Bob die Parameteränderungen gespeichert hat, kehrt er zurück auf die Task Progress Ansicht und stellt bei der nächsten Aufgabe von Alice zufrieden fest, dass im Chart ein monoton steigender Aufgabenfortschritt zu sehen ist.

5.3. Fazit

Aufgrund der szenariohaften Durchführung wurden nicht alle Ansichten der Webseite und damit nicht der komplette Funktionsumfang dargestellt. Jedoch haben das Szenario sowie dessen Durchführung gezeigt, dass die Struktur sowie die Implementierung funktionsfähig sind und eine Analyse wie auch eine darauf basierende Adaption eines Serious Games zulassen.

Die ELAI-UI ermöglicht es dem Nutzer, die Daten des LRS und der Nutzerdatenbank zu betrachten und zu filtern, um so einen genaueren Eindruck von diesen zu erhalten. Die implementierten Analysefunktionalitäten, wie die Präsentation des Lernfortschritts in einem Chart und die Klassifikation einer Aufgabenperformance zeigen Möglichkeiten der Darstellung von Nutzungsdaten. Eine Adaption des SaFIR Serious Games wurde anhand der Änderung von Datenbankabfragen über die ELAI-UI realisiert. Dies zeigt, dass Simulationssysteme und Serious Games von einer externen Webkomponente beeinflusst werden können, um eine Interoperabilität zwischen diesen Systemen zu ermöglichen.

Durch die separierte Umsetzung der ELAI und der ELAI-UI kann die prototypische Implementierung der ELAI zu einem späteren Zeitpunkt ausgetauscht werden, ohne auf der Seite der ELAI-UI etwas ändern zu müssen. Es muss lediglich darauf geachtet werden, dass die in dieser Ausarbeitung beschriebenen Grundstrukturen der Verben und Objekte eingehalten sowie die Servlets implementiert werden.

Kapitel 6

Studie Adaptivität

Im vorigen Kapitel wurde auf das Zusammenspiel der ELAI und der ELAI-UI zur Analyse von Daten aus Spielen eingegangen und mithilfe eines Szenarios illustriert. In diesem Kapitel wird die durchgeführte Studie betrachtet, welche die adaptiven Fähigkeiten der ELAI fokussiert. Hierfür war es nötig, ein Serious Game zu erweitern, um in einer realen Situation den Einsatz der ELAI zu evaluieren. Die Studie ordnet sich dabei im ELAI-Konzept bei den Komponenten ELAI und dem entwickelten SaFIR Serious Game ein (Abbildung 6.1). Mit der Studie wurden die folgenden Hypothesen validiert:

- **H1**: Die Adaption des SaFIR Serious Game durch die ELAI wird von Spielern wahrgenommen und führt zu einem besseren Spielfluss.
- **H2**: Die Adaption des SaFIR Serious Game resultiert beim Spieler in einer erhöhten Motivation.
- **H3**: Mithilfe des SaFIR Serious Game kann die Erkennung von unterschiedlichen Panzertypen erlernt sowie der Lerneffekt durch die Adaption gesteigert werden.

Nachfolgend wird das Konzept und die Entwicklung des SaFIR Serious Game sowie die Problemstellung, der Versuchsaufbau sowie die Beobachtung und Auswertung der Studie erläutert.

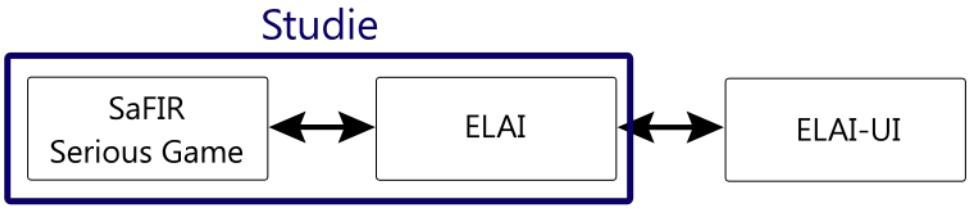


Abbildung 6.1 Einordnung der Studie in das ELAI-Konzept

6.1. SaFIR

Die ELAI dient der systemübergreifenden Analyse und Adaption von Simulationssystemen und Serious Games. Zum Testen und Demonstrieren der ELAI sowie der ELAI-UI wurde ein System benötigt, welches an das Gesamtkonzept gekoppelt werden konnte. Hierfür wurde das im Fraunhofer IOSB entwickelte Serious Game SaFIR verwendet, welches im Rahmen dieser Arbeit grundlegend überarbeitet wurde.

Konzept

In der Bildauswertung nehmen Experten Analysen und Identifizierungen von Strukturen und Objekten anhand von Bildern vor. Dabei können die Bilder von unterschiedlichen Sensortypen stammen, z.B. Elektrooptik, Infrarot, Radar oder von hyperspektralen Abtastungen [82]. Die militärische Bildauswertung steht dabei vor dem Problem, für Training und Ausbildung viele Bildmaterialien zu benötigen aber gleichzeitig den Zugang zu realen Bildern aus Vertraulichkeitsgründen eingeschränkt halten zu müssen [82].

Das Serious Game SaFIR (Seek and Find für Image Reconnaissance) ist eine prototypische Entwicklung eines Suche-und-Finde-Spiels für die Bildauswertung. Das mit der Game-Engine Unity entwickelte Spiel soll dem Spieler ermöglichen, sich auf den Straßen einer beliebigen Stadt, umgeben von 3D-Gebäuden zu bewegen [83]. Durch die Sicht von oben sollen dem Spieler Bilder einer Luftaufklärung suggeriert werden, auf welchen er definierte Objekte finden muss. Weiterhin existiert ein virtueller Assistent, der dem Avatar des Spielers folgt und bei Aktivierung den Aufgabentext oder Hilfen einblendet. Ziel ist es, zum einen automatisch neues Bildmaterial zu erzeugen und zum anderen Bildauswerter in ihren Fähigkeiten zu trainieren und durch den spielerischen Aspekt eine erhöhte Motivation zu erreichen.

Weiterentwicklung

Im Rahmen dieser Arbeit wurde das Spiel um wesentliche Features erweitert. Um dem Spieler verschiedene Aufgaben zu bieten, wurde ein Hauptmenü eingeführt, in dem die Aufgaben beschrieben und gestartet werden können (Abbildung 6.2). Weiterhin wurde das Grundkonzept des Spiels, Daten von OpenStreetMap zur Laufzeit zu laden, aus Performancegründen verworfen. Der Ladevorgang sowie die Erstellung der Gebäude und Straßen sorgten je nach Größe des zu ladenden Gebietes für eine deutlich wahrnehmbare Verzögerung. Zur Lösung wurde der Code derart erweitert, dass es nun möglich ist, Variablen anzupassen und zu bestimmen, ob Offline- oder Onlinedaten verwendet werden. Als Einstellungsmöglichkeiten werden die Eingabe des Standortes mithilfe von Längen- und Breitengraden sowie die zu ladenden Objekte innerhalb eines einstellbaren Radius geboten. Weiterhin wurde eine ausführliche Beschreibung der Funktionalität angefertigt, die dem SaFIR Projekt beiliegt. Als Offline-Datensatz wurden in dieser Ausarbeitung die Daten der Standorte Karlsruhe und

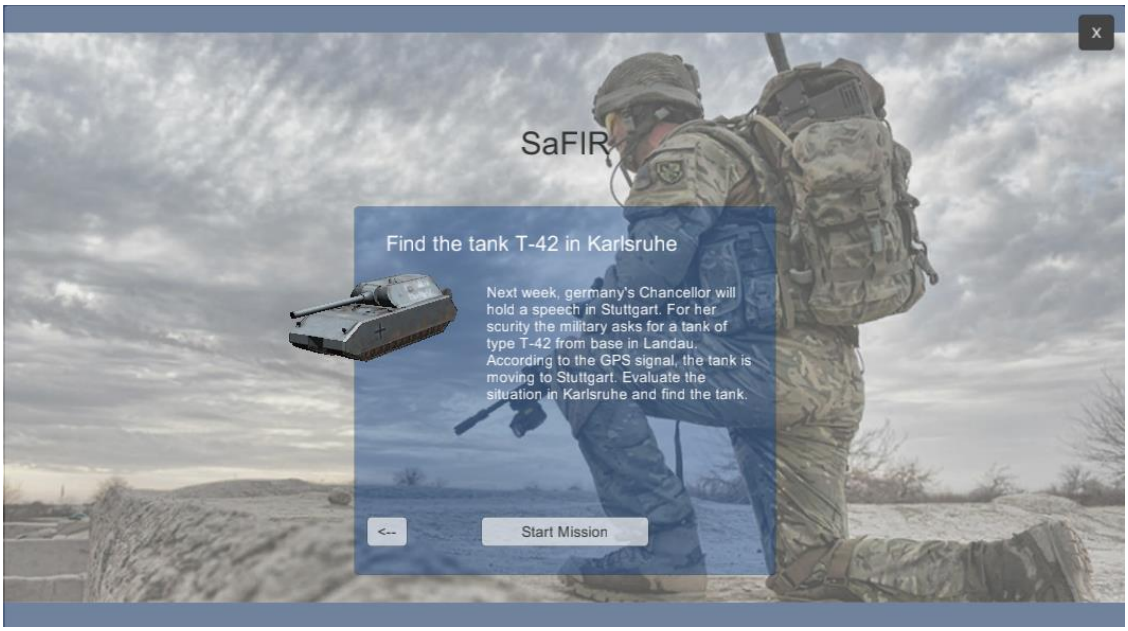


Abbildung 6.2 SaFIR - Hauptmenü - Aufgabenbeschreibung

Ingolstadt verwendet. Durch die Offline-Datensätze werden die Ladezeiten zu Beginn einer Aufgabe um ein Vielfaches verringert und gleichzeitig die Fläche des zu ladenden Gebiets erhöht.

Für die Anbindung an die ELAI wurde ein spezifischer ELAI-Adapter entwickelt, der unter anderem die Statements nach dem First-In-First-Out Prinzip über das Push Servlet an die ELAI sendet (Kapitel 4.1). Die Statements werden über eine globale Variable an eine Liste angehängt, welche durchgehend bearbeitet wird. Weiterhin fordert der ELAI-Adapter nach dem Polling-Prinzip alle zwei Sekunden den aktuellen Spieler-Status an. Um ein dauerhaftes Polling bei einer fehlerhaften Kommunikation zu vermeiden, verdoppelt sich die Dauer zwischen den einzelnen Abfragen bis zu maximal einer Minute und wird bei funktionierender Verbindung wieder auf zwei Sekunden gesenkt. Der empfangene Spieler-Status regelt den Schwierigkeitsgrad sowie das Hilfelevel des virtuellen Assistenten. Der Schwierigkeitsgrad korreliert in dieser Arbeit prototypisch mit der Größe des zu findenden Objektes und dem Wolken-Aufkommen. Wie der Schwierigkeitsgrad ist auch der Hilfelevel in drei Stufen unterteilt. Auf der ersten Stufe wird dem Spieler die Himmelsrichtung zum Ziel und in einer weiteren zusätzlich die Entfernung dargestellt (Abbildung 6.3). Bei der letzten und höchsten Hilfestufe wird zusätzlich für drei Sekunden eine Markierungslinie eingeblendet, die direkt zum Ziel weist.

Abbildung 6.3 SaFIR Serious Game - Hilfelevel zwei aktiviert

6.2. Ziel der Studie

Ziel eines Educational Serious Games ist es, trotz des Lerninhaltes den Nutzer im Flow Zustand und so dessen Motivation über die Zeit aufrecht zu halten. Das grundlegend überarbeitete SaFIR Serious Game wurde zusätzlich um einen ELAI-Adapter erweitert, der die Funktionalität einer Adaption des Spiels liefert, um den Nutzer innerhalb eines solchen Flow-Kanals zu manövrieren. Mit der Studie wurde festgestellt, ob die Nutzer eine Änderung der Spiels durch das ELAI-Konzept bemerken (H1), eine erhöhte Motivation durch die Anpassung verspüren (H2) und/ oder mithilfe des Serious Game ein Lerneffekt erzielt wird (H3).

6.3. Versuchsbeschreibung

Zur Untersuchung der Hypothesen H1-H3 wurde ein Fragebogen mit Prä- und Posttest-Fragen erstellt [84]. Die Fragen wurden als geschlossene Fragen formuliert und werden in einer Intervallskala beantwortet. Dabei kommen zwei verschiedene Antwortmöglichkeiten zum Einsatz. Hollenberg empfiehlt in seinem Buch den Einsatz von fünf bis sieben Abstufungen [84] und eine „weiß-nicht“-Antwortkategorie zu vermeiden, was zusammen eine gerade Anzahl an sechs Abstufungen impliziert. Weiterhin beschreibt er, dass für intuitive Antworten des Probanden nur die Pole der Skalen mit Bedeutungen belegt werden sollten. Um eine klare Richtung von den Probanden zu erzwingen, wurden in dem Fragebogen einerseits Ja/Nein Fragen und andererseits Antwortmöglichkeiten mit sechs Skalenpunkten eingesetzt. Weiterhin wurden in den Frageformulierungen allgemein bekannte Worte mit eindeutiger Bedeutung verwendet [84]. Um Fallstricke während der Durchführung der Studie zu vermeiden, wurde zu Beginn ein Prätest durchgeführt, welcher den kompletten Ablauf testete.

Die Studie wurde in der gewohnten Umgebung der Probanden durchgeführt. Sie wurden über die Speicherung der im Spiel entstehenden Daten informiert. Insgesamt nahmen zwölf Probanden an der Studie teil (n = 12). Dabei wurden die Probanden zufällig in zwei Gruppen verteilt, welche sich darin unterscheiden, dass die erste Gruppe mit ausgeschalteter und die zweite Gruppe mit eingeschalteter Adaption das Serious Game spielten. Dabei sind der ersten Gruppe fünf und der zweiten Gruppe sieben Probanden zugeteilt. Dieses Ungleichgewicht wurde gewählt, da für die erste Hypothese teilweise nur Probanden der Gruppe 2 relevant sind. Im Rahmen dieser Studie war es notwendig, alle Teilnehmer der ersten Gruppe denen der zweiten zeitlich vorzuziehen. Dadurch wird eine gleichzeitige Initialisierung des LRS gewährleistet und die notwendigen Daten liegen zur Normierung der Klassifikation beim Durchlauf der zweiten Gruppe vor (Abbildung 6.4). Zur Separierung der Studien- von den Testdaten und der somit entstehenden Möglichkeit nachgehender Analysen, wurde zu Beginn der Studie ein neuer LRS angelegt.

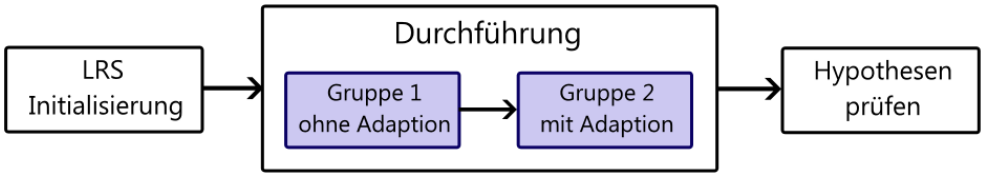


Abbildung 6.4 Studienverlauf Übersicht

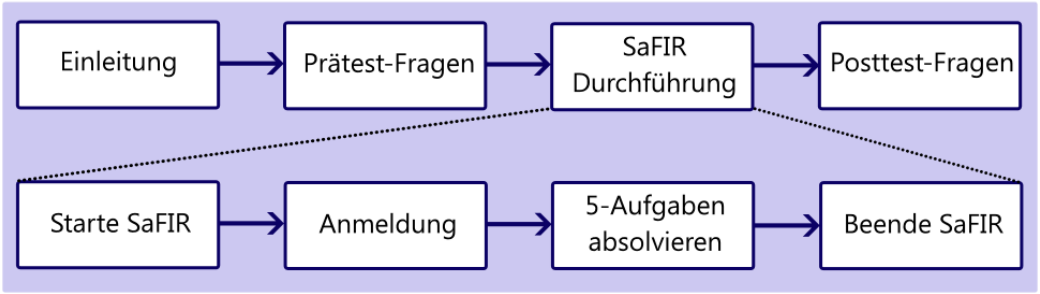


Abbildung 6.5 Durchführung pro Proband

Der Ablauf der Studie wurde mit jedem Probanden wie folgt durchgeführt (Abbildung 6.5):

1. **Einleitung**: Zu Beginn der Durchführung wurde dem Probanden mithilfe von vorbereiteten Folien die Steuerung, Menüführung und Aufgabenstellung erläutert. Hierbei wurde darauf geachtet, jedem Probanden die gleichen Vorkenntnisse zu liefern.
2. **Prätest-Fragen**: Nach der Einleitung wurden dem Probanden ein anonymer Name sowie eine anonyme E-Mail Adresse zugeteilt. Letztere dienten lediglich der eindeutigen Identifikation für die ELAI und wurden nicht als anwählbare E-Mail Adresse umgesetzt. Um Kontextdaten vom Nutzer zu erhalten, wurden ihm anschießend Prätest-Fragen ausgehändigt, welche sein Alter, Geschlecht, Erfahrungen mit Computerspielen und die Kenntnisse in der Bildauswertung sowie in der Unterscheidung von Panzertypen erfassten. Diese Kontextdaten dienten dazu, bei der späteren Auswertung der Ergebnisse mögliche Unterschiede in den einzelnen Gruppen (Altersgruppe, Geschlechtsgruppe, …) zu evaluieren.
3. **SaFIR Durchführung**: Nach der Erfassung der ersten Daten bekamen die Probanden einen Laptop zur Verfügung, auf welchem die ELAI sowie das SaFIR Spiel bereits gestartet waren. Bei Probanden der Gruppe 1 wurde das Spiel ohne Adaption und bei Probanden der Gruppe 2 mit Adaption durchgeführt. Nach Eingabe der anonymen Login-Daten spielten alle Nutzer fünfmal die Aufgabe *Find the tank T42 (Karlsruhe)*. Die Position der Fahrzeuge und des Avatars werden dabei bei jedem Start der Aufgabe randomisiert auf dem Spielfeld platziert. Bei jedem Start wurden somit unterschiedliche Ausgangssituationen erzeugt.
4. **Posttest-Fragen**: Für eine Beurteilung der Hypothesen wurden den Probanden nach Beendigung des Spiels Posttest-Fragen ausgeteilt (Tabelle 4). Dessen Antworten dienen in Verbindung mit den Prätest-Fragen der Untersuchung der Hypothesen. Die Fragen sind dabei in drei Kategorien unterteilt, welche den Probanden nicht ersichtlich waren.

Tabelle 4 Inhalt des Fragebogens: Posttest-Fragen

H1	Q1	Hast du eine Änderung der Schwierigkeit wahrnehmen können?	(Ja/ Nein)
	Q1.1	Falls ja, empfandst du die Anpassung als angemessen?	(Ja/ Nein)
	Q1.2	Falls ja, hast du eine angemessene Anzahl an Hilfestellungen bekommen?	(0-5)
	Q2	Waren die Hilfestellungstypen deinen Fähigkeiten entsprechend?	(0-5)
H2	Q3	Wenn du die Schwierigkeit des Spiels auf einer Skala von 0-5 bewerten müsstest, wie wäre deine Einordnung?	(0-5)
	Q4	Wie gut war die Aufgabe in ein Szenario eingebettet?	(0-5)
	Q5	Wenn du das Spiel öfter spielen würdest, glaubst du, du würdest das Ziel schneller, mit weniger oder genauso vielen Hilfen finden?	(Ja/ Nein)
	Q6	Empfandst du das Spiel als abwechslungsreich?	(Ja/ Nein)
H3	Q7	Auf welcher Abbildung ist der Panzertyp *T42* zu sehen? (Abbildung 6.6)	(F1/ F2/ F3)
	Q8	Auf welcher Abbildung ist der Panzertyp *Tiger 1* zu sehen? (Abbildung 6.6)	(F1/ F2/ F3)

Abbildung 6.6 Folien für Q7-Q8 mit unterschiedlichen Panzertypen; Folie 1 mit Panzertyp T42; Folie 2 mit Panzertyp 38T; Folie 3 mit Panzertyp Tiger 1 (Zuordnung Panzertyp und -name entspricht nicht der Realität)

6.4. Ergebnisse

Nach Abschluss der Studie mit allen zwölf Probanden wurden die Ergebnisse zusammengetragen und evaluiert. In Tabelle 5 sind die ausgewerteten Ergebnisse des Fragebogens dargestellt.

Die Hypothese **H1** (*Die Adaption des SaFIR Serious Game durch die ELAI wird von Spielern wahrgenommen und führt zu einem besseren Spielfluss.*) wird mit den Fragen Q1-Q2 behandelt. Sie prüfen die Fragestellung nach der Wahrnehmung der Adaption sowie dessen positiven Einfluss auf den Spielverlauf. Hierbei erkannten 83 % der Probanden korrekt, ob es sich um ein Spiel mit oder ohne Adaption handelt. Weiter wurden die Probanden, welche angaben, eine Adaption bemerkt zu haben, mit den Fragen Q1.1 und Q1.2 sowie alle Probanden mit Q2 konfrontiert. Mit diesen wurde ermittelt, ob der Einfluss der Adaption zu einem besseren Spielfluss führt. Zwar bewerteten die Probanden, denen die Adaption auffiel, die Anzahl an gegebenen Hilfestellungen positiv bzgl. des Spielflusses, jedoch wurde die Art der Hilfestellung schlechter bewertet als in Gruppe 1. Insgesamt ist die Adaption also bemerkbar, wurde aber in diesem Spiel im Durchschnitt als eher negativen Einfluss auf den Spielfluss gewertet.

Mit den Fragen Q3-Q6 wird die Hypothese **H2** geprüft, welche das Thema Motivationssteigerung durch die Adaption behandelt (*Die Adaption des SaFIR Serious Game resultiert beim Spieler in einer erhöhten Motivation.*). Ein grundlegender Aspekt für die Motivation ist das Erreichen des Flow-Zustandes (Kapitel 3.2). Dieser wird unter anderem bei einer ausgewogenen Schwierigkeitseinstellung erreicht. Mithilfe der Frage Q3 konnte gezeigt werden, dass die Adaption eine im Durchschnitt adäquatere Schwierigkeitseinstellung im Vergleich zu einer festen Einstellung erreicht. Ein weiterer wichtiger Punkt für die Erhaltung der Motivation ist die Steigerung der Spielleistung bei mehrfacher Anwendung. Um diese Frage indirekt zu prüfen, wurden die Probanden gebeten, zu bewerten, ob sie glauben, das Ziel nach einer gewissen Spielzeit schneller oder mit weniger Hilfen zu erreichen (Q5). Im Durchschnitt gaben dabei 80 % der Gruppe 1 an, dass sie glauben, das Ziel schneller oder mit weniger Hilfen zu erreichen, wohingegen nur 71 % in Gruppe 2 sich dieser These zusprachen. Zuletzt bewerteten die Probanden die spielerische Abwechslung. Hierbei ergab sich ein durchschnittlicher Unterschied von über 10 %, bei welchem Gruppe 2 die Abwechslung im Spiel besser bewertete als Gruppe 1. Insgesamt kann mithilfe der Fragen Q3-Q6 geschlossen werden, dass die Gruppe 2 eine höhere Motivation verspürte, das Spiel weiter zu spielen als Probanden der Gruppe 1.

Die letzte Hypothese (**H3)** zielt auf den Lerneffekt des Spiels ab (*Mithilfe des SaFIR Serious Game kann die Erkennung von unterschiedlichen Panzertypen erlernt sowie der Lerneffekt durch die Adaption gesteigert werden.*). Im Rahmen der Fragen Q7-Q8 wurden den Probanden drei Bilder in identischer Abfolge gezeigt (Abbildung 6.6). Auf jedem Bild war eine Abbildung eines im Spiel eingesetzten sowie in der Einleitung erläuterten Panzers zu sehen. In Frage Q7 mussten die Probanden den gesuchten Typ *T42* und in Frage Q8 den als Distraktor verwendeten Typ *Tiger 1* in den Bildern erkennen. Die Auswertung ergab, dass die Gruppen den gesuchten Panzer zu 100 % und den Distraktor-Panzer im Durchschnitt zu 58 % wiedererkannten. Mit dieser Auswertung kann geschlossen werden, dass die Probanden die Zuordnung zwischen Namen und Aussehen des gesuchten Objektes erlernt haben, nicht aber diejenigen Objekte, die nicht im Fokus der Suche lagen. Weiter konnte nur eine minimale Differenz des Lerneffekts (3 % zugunsten von Gruppe 1), zwischen den Gruppen bzgl. der Distraktor-Panzer-Erkennung festgestellt werden.

Tabelle 5 Ergebnisse des Fragebogens

Hypothese	Frage	ø ohne Adaption	ø mit Adaption	ø insgesamt	Zielwert
H1	Q1	0,80	0,85	0,83	1,00
	Q1.1	-	1,00	-	1,00
	Q1.2	-	4,17	-	5,00
	Q2	3,80	3,29	3,50	5,00
H2	Q3	3,40	2,40	2,83	2,50
	Q4	3,40	3,43	3,42	5,00
	Q5	0,80	0,71	0,75	1,00
	Q6	0,60	0,71	0,66	1,00
H3	Q7	1,00	1,00	1,00	1,00
	Q8	0,60	0,57	0,58	1,00

Zusätzlich zu den erhobenen Daten aus den Fragebögen mussten die Spieldaten der Probanden in einem separaten LRS gespeichert werden, um die Adaption des Spiels durchführen zu können. Dies ermöglicht auch eine nachfolgende Betrachtung der Spielleistungen und weitere Analysen der Probandendaten mithilfe der ELA-UI. In Abbildung 6.7 ist der Lernfortschritt von Proband 3 über die gespielten fünf Aufgaben zu sehen. In dieser wird deutlich, dass sich mit Ausnahme des vierten Spiels eine Verbesserung der Spielleistung des Probanden einstellte. Weiter zeigt Abbildung 6.8 den Verlauf der dritten gespielten Aufgabe von Proband 6. Dabei stellt die blaue Linie den Aufgabenfortschritt über der Zeit dar. Die schwarzen Peaks ergänzen den Graphen um die Information, zu welchem Zeitpunkt dem Probanden eine Hilfe durch den virtuellen Assistenten angezeigt wurde. Zuletzt werden in Abbildung 6.9, Abbildung 6.10 und Abbildung 6.11 die Klassifizierungen aller Aufgaben, die im Rahmen der Studie mithilfe der Probanden absolviert wurden, mit den drei didaktischen Faktoren in 2D-Grafiken aufgetragen. Die didaktischen Faktoren gehen dabei mit den Gewichten $w_{TaskDuration} = -1$, $w_{TaskDifficulty} = 0{,}6$ und $w_{TaskHelpingCount} = -0{,}3$ in die

Klassifikationsberechnung ein. Auf den Grafiken werden die Klassifikationseinteilungen (*lowLevel*, *mediumLevel* und *highLevel*) der Aufgaben farblich unterschieden. Es ist zu erkennen, dass im Vergleich zu den anderen Gruppen die *highLevel*-Gruppe die höchste Menge an gespielten Aufgaben beinhaltet. Dies bedeutet, dass die Klassifikation der Aufgaben für die Einteilung in die drei Gruppen nur suboptimal gelingt. Diese Fehl-Gewichtung kann durch Einstellung der Gewichte der Parameter verbessert werden.

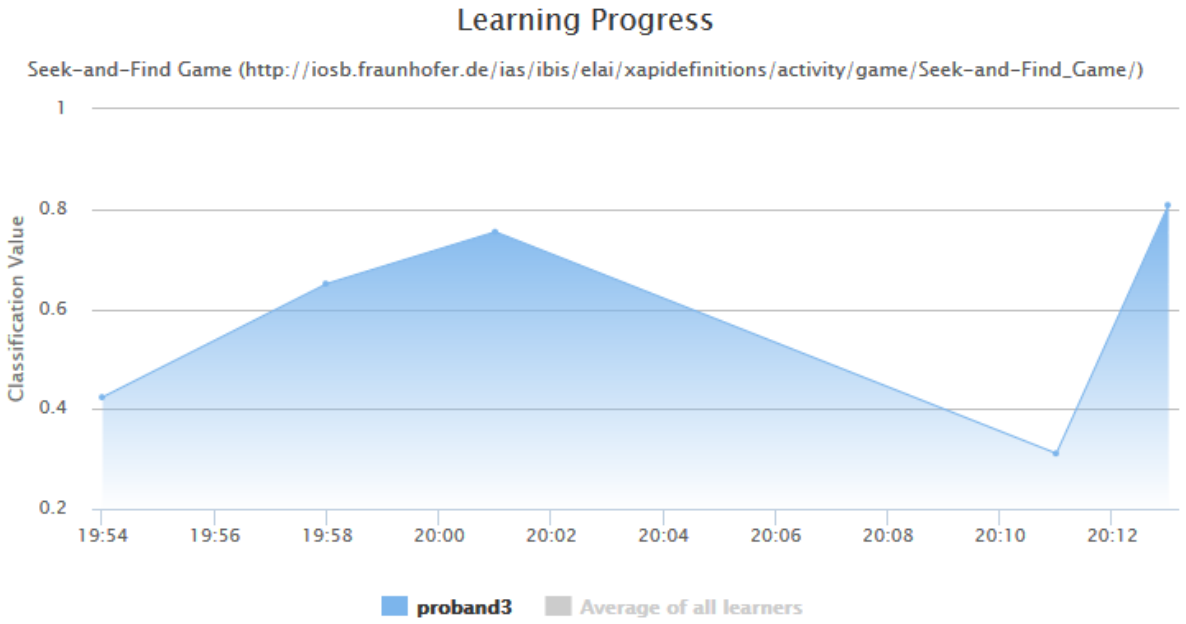


Abbildung 6.7 ELAI-UI – Lernfortschritt von Proband 3

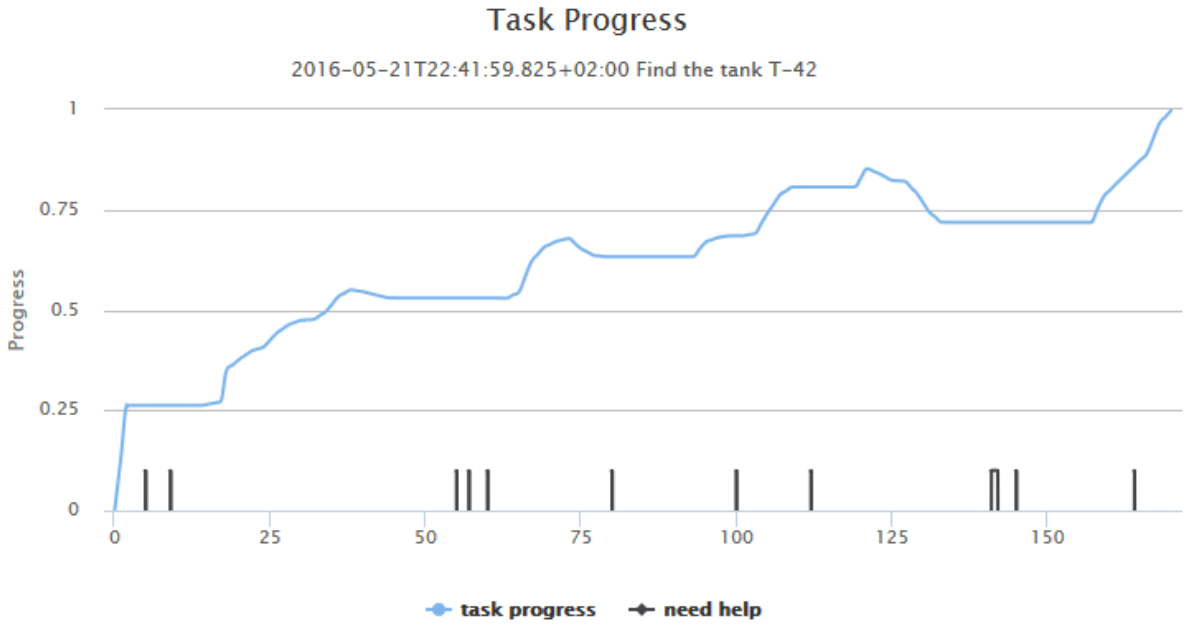


Abbildung 6.8 ELAI-UI – Aufgabenfortschritt der dritten Aufgabe von Proband 6

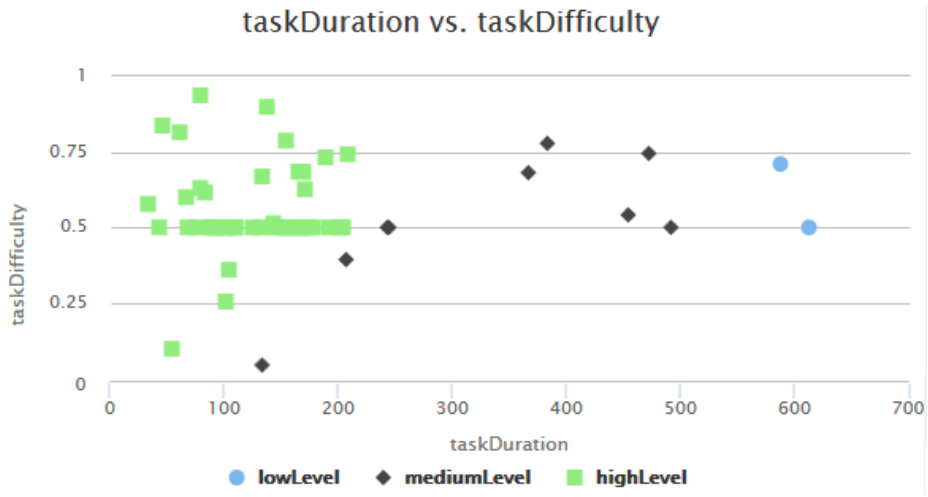


Abbildung 6.9 ELAI-UI – Klassifikation aller Aufgaben der Probanden – TaskDuration vs. TaskDifficulty

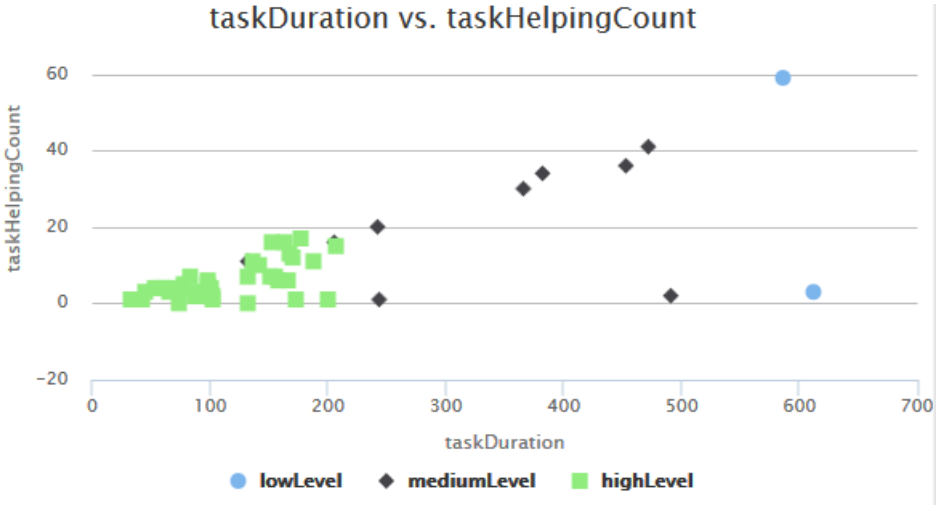


Abbildung 6.10 ELAI-UI – Klassifikation aller Aufgaben der Probanden – TaskDuration vs. TaskHelpingCount

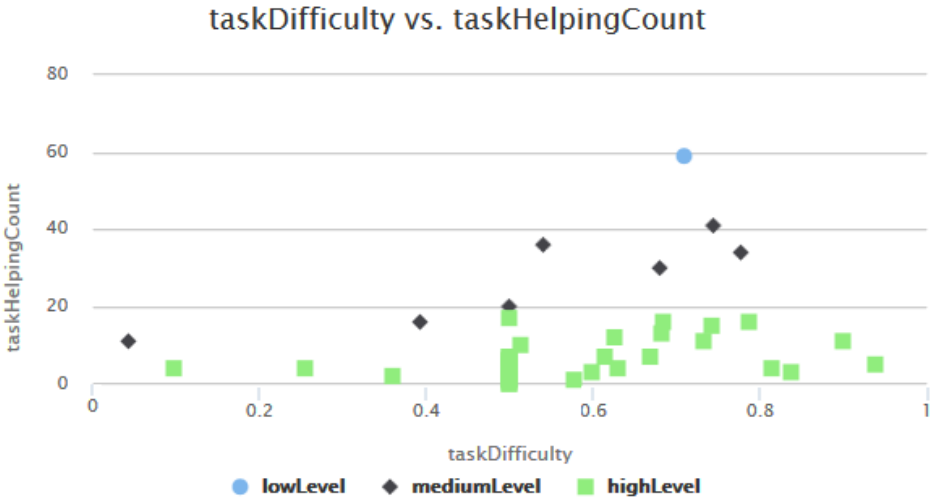


Abbildung 6.11 ELAI-UI – Klassifikation aller Aufgaben der Probanden – TaskDifficulty vs. TaskHelpingCount

6.5. Diskussion

Das vorige Unterkapitel betrachtet die Ergebnisse der Studie und setzt die Daten in den entsprechenden Kontext. In diesem Unterkapitel werden die Entstehung der Daten sowie dessen Kontext diskutiert, um die Schlüsse der Studie weiter zu bewerten.

Die Zielgruppe dieser Studie sind Bildauswerter, die das Spielen des SaFIR Serious Game als Training betrachten. Aus Gründen der Einfachheit wurde auf diese Zielgruppe verzichtet und Probanden aus dem näheren Umkreis des Autors dieser Arbeit gewählt, womit deren Objektivität beeinflusst gewesen sein könnte.

Ein wesentlicher Aspekt der Adaption im Rahmen der Studie ist das durch die ELAI extern gesteuerte Aktivieren des virtuellen Assistenten. In der SaFIR Version, die in der Studie zum Einsatz kam, war die Dauer, in welcher der virtuelle Assistent angezeigt worden ist, auf 30 Sekunden eingestellt. Wird in dieser Zeit der virtuelle Assistent erneut betätigt, ändert sich lediglich der Text in der Sprechblase des Assistenten (Abbildung 6.3). Im Verlauf der Durchführung fiel auf, dass die Probanden eine Änderung des Textes nicht immer bemerkten, da deren Fokus auf der Suche des Panzers lag. Dies bedeutet, dass nicht alle Hilfestellungen der ELAI erkannt wurden, was sich negativ auf das Ergebnis der Hypothese H1 auswirkt. Damit ist festzuhalten, dass der erste Teil der ersten Hypothese (*Die Adaption des SaFIR Serious Game durch die ELAI, wird von Spielern wahrgenommen …*) erfolgreich evaluiert wurde, der zweite Teil jedoch (*… und führt zu einem besseren Spielfluss.*) nicht bestätigt werden konnte.

Wie bereits in Unterkapitel *Ergebnisse* beschrieben, wurde die zur Hypothese 2 gehörende Frage Q5 zugunsten der Gruppe 1 entschieden. Aus den Ergebnissen ist abzuleiten, dass die Probanden im Schnitt glauben, ohne Adaption das Ziel schneller zu erreichen. Hierbei muss allerdings bedacht werden, dass eine Schwierigkeitsadaption des Spiels die Dauer sowie die Anzahl an benötigter Hilfe in einer Aufgabe beeinflusst. Somit war bei den Probanden, die mit einer Adaption spielten, ein niedrigeres Ergebnis bereits zu erwarten. Diese Einsicht resultiert in einer positiven Entwicklung bzgl. der Bestätigung der zweiten Hypothese (*Die Adaption des SaFIR Serious Game resultiert beim Spieler in einer erhöhten Motivation.*).

Im Rahmen der dritten Hypothese (*Mithilfe des SaFIR Serious Game kann die Erkennung von unterschiedlichen Panzertypen erlernt sowie der Lerneffekt durch die Adaption gesteigert werden.*) wurde den Probanden die Frage Q8 gestellt, bei welcher Sie aus einer Auswahl von drei Folien den Distraktor-Panzer des Typs *Tiger 1* bestimmen mussten. Hierbei wurde im Schnitt ein Wert von 58 % erreicht. Bezieht man allerdings mit ein, dass sich die Frage Q7 nach dem Typ *T42* auf die gleichen Folien in derselben Reihenfolge bezieht, stehen nach dem Ausschlussverfahren bei Frage Q8 nur noch zwei Panzertypen zur Auswahl. Diese resultiert in einer Wahrscheinlichkeit von 50 %, den korrekten Panzertypen zu wählen. Betrachtet man nun wieder den Wert 58 % liegt dieser nur acht Prozentpunkte über der Zufallschance, was den Lerneffekt bei den Distraktor-Panzertypen in Frage stellt. Insgesamt ist somit nur ein Lerneffekt bei den jeweils zu findenden Panzertypen erzielt worden.

Kapitel 7

Fazit und Ausblick

Zu Beginn dieser Arbeit stellte sich die Frage, ob eine interoperable Adaptivität von Serious Games möglich ist und wie diese visualisiert und beeinflusst werden kann. Im Rahmen dieser Arbeit wurde die ELAI-Architektur umgesetzt, die es ermöglicht, netzwerkübergreifend Daten von verschiedenen Systemen über Servlets zu empfangen und sie in einer Datenbank zu speichern. Weiter werden die im xAPI-Format gehaltenen Daten von der ELAI analysiert und gegebenenfalls Datenbankeinträge angepasst. Über ein weiteres Servlet ist es den Spielen möglich, Spielerdaten abzufragen und darauf zu reagieren.

Ein wesentlicher Bestandteil der Arbeit war die Entwicklung einer Weboberfläche für die Visualisierung der Daten und Aktionen der ELAI sowie die Möglichkeit der Einflussnahme eines Spiels – ELAI-UI. Die im Vaadin Framework implementierte ELAI-UI bietet verschiedene Webseiten, welche es ermöglichen, die Datenbankeinträge der ELAI darzustellen, Nutzungsdaten zu analysieren und zu klassifizieren sowie die Spielerprofile in der Datenbank zu ändern. Letzteres resultiert durch die zyklische Abfrage des Spiels und der damit einhergehenden Anpassung der Spielelemente in einer Adaption des Spiels. Die Funktionalität der ELAI-UI wurde anhand eines Anwendungsszenarios evaluiert.

Zum Testen der Funktionalität der automatischen Analyse und Adaption der ELAI wurde ein Suche-und-Finde Spiel entwickelt, in welchem der Nutzer einen Panzer auf den Straßen der Städte Karlsruhe oder Ingolstadt finden muss. Ziel ist es, die Fähigkeiten des Spielers in der Bildauswertung durch didaktische Spielelemente zu verbessern. Weiter wird der Spieler von einem virtuellen Assistenten unterstützt, der sowohl die Aufgabenstellung als auch zusätzliche Hilfen anbietet. Über einen implementierten ELAI-Adapter kann das Serious Game mit der ELAI kommunizieren. Zum einen sendet dieser Nutzungsdaten an die ELAI und zum anderen fordert er von der ELAI in regelmäßigen Abständen das Spielerprofil sowie einen Wert für den Hilfegrad an. Letztere zyklische Abfrage sorgt für die Adaption des Spiels durch die ELAI. Hierfür werden für jedes Spielerprofil die Variablen *Schwierigkeitslevel* und *Hilfelevel* angelegt. Diese werden bei Beendigung einer Aufgabe von der ELAI neu berechnet und in der Datenbank gespeichert. Der implementierte ELAI-Adapter verändert abhängig von der Variablen *Schwierigkeitslevel* die Größe des zu findenden Objektes sowie die Sicht beeinflussende Wolkenbildung. Der *Hilfelevel* verändert die Art der Hilfestellung. Diese ist in drei Kategorien unterteilt. Beim niedrigsten Hilfelevel wird dem Spieler nur die Entfernung zum Ziel angezeigt, beim mittlerem zusätzlich die Himmelsrichtung und beim höchsten Hilfelevel wird zudem kurzzeitig die Richtung zum Ziel durch eine Gerade veranschaulicht.

Zur Evaluation der Adaptionsfähigkeit der ELAI wurde eine Studie mithilfe des entwickelten Spiels mit n = 12 Probanden durchgeführt. Die Probanden mussten fünf Runden des Serious Game absolvieren und einen Prätest- sowie einen Posttest-Fragebogen ausfüllen. Die Auswertung des Fragebogens ergab unter anderem, dass die Probanden die Anpassung des

Serious Games an ihre Fähigkeiten insbesondere durch Schwierigkeitsänderung bemerkten, diese jedoch nicht als positiven Effekt wahrgenommen haben.

Die gesetzten Anforderungen der Arbeit konnten innerhalb des zeitlichen Rahmens umgesetzt und durch die Studie evaluiert werden. Jedoch ist zu erwähnen, dass die ELAI nur prototypisch mithilfe eines linearen Klassifikators implementiert wurde und für weitere Arbeiten Machine-Learning-Algorithmen eingesetzt werden könnten. Mithilfe dieser Algorithmen wäre es möglich, die Klassengrenzen den Parameterwerten der gespielten Aufgaben anzupassen und so eine dynamischere Klassifizierung zu erreichen. Denkbar wäre der Einsatz der Methoden Support Vector Machine, Neuronale Netze oder abhängig von den Daten der Clustering Algorithmus k-Means. Auch ist die Erweiterung der Parametermenge von Schwierigkeitslevel und Hilfelevel um weitere Parameter und detailliertere Berechnungsmethoden möglich. Außerdem könnten die automatisch berechneten Werte der ELAI in einer gesonderten Spalte der Datenbank gespeichert werden, sodass bei manueller Anpassung diese nicht überschrieben werden. Ein wichtiger Aspekt der ELAI-Architektur, der in dieser Arbeit nicht fokussiert wurde, ist die Interoperabilität der Nutzerfähigkeiten zwischen verschiedenen Serious Games oder Simulationssystemen. Das Augenmerk der Arbeit lag auf der Adaptivität der ELAI und der Analyse und Einflussnahme der ELAI-UI, weshalb nur ein Serious Game zum Einsatz kam. In weiteren Arbeiten wäre es daher sinnvoll, mehrere Serious Games und Simulationssysteme an die ELAI anzukoppeln und die Interoperabilität zu fokussieren.

I. Literaturangaben

[1] Streicher, A. u. Roller, W.: Towards an Interoperable Adaptive Tutoring Agent for Simulations and Serious Games. In International Conference on Theory and Practice in Modern Computing, Bd. 4. 2015 - 2015, S. 194–197

[2] Keller, J., Schäfer, J. u. Weber, J.: Die Gamesbranche: ein ernstzunehmender Wachstumsmarkt. HA, Hessen-Agentur, Hessen-IT 2010

[3] Global digital and physical gaming market value 2012-2016 | Forecast. http://www.statista.com/statistics/255196/global-digital-and-physical-gaming-market-value/, abgerufen am: 08.01.2016

[4] Michael, D. R. u. Chen, S. L.: Serious games: Games that educate, train and inform. Muska & Lipman/Premier-Trade 2005

[5] Doujak, G.: Serious Games und Digital Game Based Learning. Spielebasierte E-Learning Trends der Zukunft (2015)

[6] Woolf, B. P.: Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann 2010

[7] Josef Brozek, Bhakti Stephan Onggo u. Antonin Kavicka: High Level Architecture - Virtual Assistant Framework, University of Pardubice u. University of Lancaster 2014

[8] Poltrack, J., Hruska, N., Johnson, A. u. Haag, J. (Hrsg.): The next generation of SCORM: Innovation for the global force, Bd. 2012. 2012

[9] Google Analytics: Webanalyse und Berichte, 2013. https://www.google.com/intl/de_ALL/analytics/index.html, abgerufen am: 21.04.2016

[10] Neuhold, B.: Learning Analytics-Mathematik Lernen neu gedacht. BoD–Books on Demand 2013

[11] J.P. Medved: Best LMS (Learning Management System) Software | 2016 Reviews of the Most Popular Systems, 2015. http://www.capterra.com/learning-management-system-software/#infographic, abgerufen am: 17.04.2016

[12] Siemens, G. u. d Baker, R. S. J. (Hrsg.): Learning analytics and educational data mining: towards communication and collaboration. ACM 2012

[13] Dyckhoff, A. L., Zielke, D., Bültmann, M., Chatti, M. A. u. Schroeder, U.: (eLAT) Design and Implementation of a Learning Analytics Toolkit for Teachers. Educational Technology & Society 15 (2012) 3, S. 58–76

[14] Bader-Natal, A. u. Lotze, T. (Hrsg.): (Grockit) Evolving a learning analytics platform. ACM 2011

[15] Learning Record Store (iLRS) - Tin Can Experience API | Instancy, 2016. http://www.instancy.com/learning-record-store.html, abgerufen am: 21.04.2016

[16] Hooshyar, D., Ahmad, R. B., Yousefi, M., Yusop, F. D. u. Horng, S.: A flowchart-based intelligent tutoring system for improving problem-solving skills of novice programmers. Journal of Computer Assisted Learning (2015)

[17] Doleck, T., Basnet, R. B., Poitras, E. u. Lajoie, S. (Hrsg.): Towards examining learner behaviors in a medical intelligent tutoring system: A Hidden Markov Model approach. IEEE 2015

[18] Bakkes, S., Spronck, P. u. van den Herik, J.: Rapid and reliable adaptation of video game AI. Computational Intelligence and AI in Games, IEEE Transactions on 1 (2009) 2, S. 93–104

[19] Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I. u. Postma, E.: Adaptive game AI with dynamic scripting. Machine Learning 63 (2006) 3, S. 217–248

[20] Auslander, B.: Recognizing the Enemy: Combining Reinforcement Learning with Case Based Reasoning in Domination Games. ProQuest 2009

[21] Ashmore, C. u. Nitsche, M. (Hrsg.): The quest in a generated world. 2007

[22] Pita, J., Magerko, B. u. Brodie, S. (Hrsg.): True story: dynamically generated, contextually linked quests in persistent systems. ACM 2007

[23] Mott, B. W. u. Lester, J. C. (Hrsg.): U-director: a decision-theoretic narrative planning architecture for storytelling environments. ACM 2006

[24] Barber, H. u. Kudenko, D.: Dynamic Generation of Dilemma-based Interactive Narratives. AIIDE 7 (2007), S. 2–7

[25] Shaker, N., Yannakakis, G. N. u. Togelius, J. (Hrsg.): Towards Automatic Personalized Content Generation for Platform Games. 2010

[26] Togelius, J., Nardi, R. de u. Lucas, S. M. (Hrsg.): Towards automatic personalised content creation for racing games. IEEE 2007

[27] Kazmi, S. u. Palmer, I. J.: Action recognition for support of adaptive gameplay: A case study of a first person shooter. International Journal of Computer Games Technology 2010 (2010), S. 1

[28] Lopes, R. u. Bidarra, R.: Adaptivity challenges in games and simulations: a survey. Computational Intelligence and AI in Games, IEEE Transactions on 3 (2011) 2, S. 85–99

[29] Left 4 Dead. http://www.valvesoftware.com/games/l4d.html, abgerufen am: 02.02.2016

[30] Heavy Rain. http://www.quanticdream.com/en/#!/en/category/heavy-rain, abgerufen am: 02.02.2016

[31] Max Payne. http://www.rockstargames.com/maxpayne/main.html, abgerufen am: 02.02.2016

[32] Johnson, W. L., Marsella, S. u. Vilhjalmsson, H. (Hrsg.): The DARWARS tactical language training system. Citeseer 2004

[33] Peirce, N., Conlan, O. u. Wade, V. (Hrsg.): Adaptive educational games: Providing non-invasive personalised learning experiences. IEEE 2008

[34] Markram, H., Meier, K., Lippert, T., Grillner, S., Frackowiak, R., Dehaene, S., Knoll, A., Sompolinsky, H., Verstreken, K., DeFelipe, J., Grant, S., Changeux, J.-P. u. Saria, A.: Introducing the Human Brain Project. Procedia Computer Science 7 (2011), S. 39–42

[35] Magerko, B., Stensrud, B. S. u. Holt, L. S.: Bringing the schoolhouse inside the box-A tool for engaging, individualized training (2006)

[36] Noseworthy, J. R. (Hrsg.): The test and training enabling architecture (TENA) supporting the decentralized development of distributed applications and LVC simulations. IEEE 2008

[37] van Oijen, J., Vanhée, L. u. Dignum, F.: CIGA: A middleware for intelligent agents in virtual environments. In: Agents for Educational Games and Simulations. Springer 2011, S. 22–37

[38] Herrlich, M., Wenig, D., Walther-Franks, B., Smeddinck, J. D. u. Malaka, R.: ,,Raus aus dem Sessel“ – Computerspiele für mehr Gesundheit. Informatik Spektrum (2014) 6, S. 558–566

[39] Lampert, C., Schwinge, C. u. Tolks, D.: Der gespielte Ernst des Lebens: Bestandsaufnahme und Potenziale von Serious Games (for Health). Medien Pädagigik - Zeitschrift für Theorie und Praxis der Medienbildung, 15/16 (2009)

[40] Stokes, B. G.: Videogames have changed: time to consider Serious Games? Development Education Journal 11 (2005) 3, S. 12

[41] Alvarez, J. u. Michaud, L.: Serious games: Advergaming, edugaming, training and more. Montpellier, France, IDATE (2008)

[42] Susi, T., Johannesson, M. u. Backlund, P.: Serious games: An overview (2007)

[43] Tate, R., Haritatos, J. u. Cole, S.: HopeLab's Approach to Re-Mission. International Journal of Learning and Media 1 (2009) 1, S. 29–35

[44] Ricciardi, F. u. Paolis, L. T. de: A Comprehensive Review of Serious Games in Health Professions. International Journal of Computer Games Technology (2014) 108, S. 1–11

[45] Goertz, L.: Einsatzmöglichkeiten für Serious Games in Unternehmen. Spielerisch lernen und Zusammenhänge erkunden (2011)

[46] Shute, V. J. u. Zapata-Rivera, D.: Adaptive technologies. Handbook of research on educational communications and technology (2008), S. 277–294

[47] Schein, A. I., Popescul, A., Ungar, L. H. u. Pennock, D. M. (Hrsg.): Methods and metrics for cold-start recommendations. ACM 2002

[48] Motyka, M. (Hrsg.): Persuasion und Wissenserwerb durch Serious Games im Politikunterricht. Reihe Studium und Forschung, Bd. 21. Kassel: Kassel Univ. Press 2012

[49] Krömker, I. D. u. Sacher, P.: Integration der Experience API in Autorensysteme für Web Based Trainings

[50] Tin Can API. https://tincanapi.com/, abgerufen am: 16.01.2016

[51] Fujimoto, R. M.: Time management in the high level architecture. Simulation 71 (1998) 6, S. 388–400

[52] SCORM Explained, 2016. http://scorm.com/scorm-explained/, abgerufen am: 03.06.2016

[53] Hoermann, S., Schneider, S., Glowalla, U. u. Steinmetz, R.: Erstellung von SCORM-kompatiblen Kursen im Projekt k-MED. Puppe, F.; Albert, J.; Bernauer, J.; Fischer, M (2002)

[54] Walther, P., Giebler, P. u. Spilke, J. (Hrsg.): Aktueller Entwicklungsstand der Standardisierung von Lernobjekten und deren Softwareunterstützung. 2004

[55] Baumgartner, P., Häfele, H. u. Maier-Häfele, K.: E-Learning Standards aus didaktischer Perspektive. 2002

[56] Roth, A. u. Suhl, L.: Plattformübergreifende Architekturen in föderativen E-Learning-Umgebungen, S. 143–152

[57] Marko Grönroos: Book of Vaadin 7, 2014. https://vaadin.com/book/vaadin7/-/page/intro.html, abgerufen am: 16.01.2016

[58] Hunter, J. u. Crawford, W.: Java servlet programming. " O'Reilly Media, Inc." 2001

[59] jmcconnell: Jetty - Servlet Engine and Http Server. http://www.eclipse.org/jetty/, abgerufen am: 18.01.2016

[60] GlassFish Server, 2015. https://glassfish.java.net/, abgerufen am: 17.01.2016

[61] Apache Tomcat, 2016. http://tomcat.apache.org/, abgerufen am: 17.01.2016

[62] Yi Lu: Großer Beleg. AJAX-basiertes Workflowmanagementsystem für das Framework CANDY (2008)

[63] Könnecke, S. u. Hacker, A.: Proseminar Webtechnologien WS2011/12 Ajax und Webentwicklung mit Prototype (2012)

[64] Thiel, N.: Konzeption und Entwicklung einer Web-Applikation für kollaboratives Checklisten-Management. Ulm University 2013

[65] Smeets, B., Boness, U. u. Bankras, R.: Beginning Google Web Toolkit. Springer 2008

[66] Chaganti, P.: Google Web Toolkit. Packt, Birmingham (2007)

[67] Vaadin Website. https://vaadin.com/, abgerufen am: 16.01.2016

[68] Marko Grönroos: Model-View-Presenter Pattern with Vaadin. https://vaadin.com/web/magi/home/-/blogs/model-view-presenter-pattern-with-vaadin, abgerufen am: 16.01.2016

[69] Event Bus - Vaadin-MVP-Lite. https://github.com/sockeqwe/Vaadin-MVP-Lite/wiki/It-all-starts-with-the-EventBus, abgerufen am: 03.01.2016

[70] Szentes, D.: Lost Earth 2307. https://www.iosb.fraunhofer.de/servlet/is/55534/, abgerufen am: 29.05.2016

[71] Rustici Software: SCORM Cloud, 2016. http://scorm.com/scorm-solved/scorm-cloud-features/, abgerufen am: 16.04.2016

[72] Rustici Software: TinCanJava Library, 2014. http://rusticisoftware.github.io/TinCanJava/, abgerufen am: 16.01.2016

[73] Kazmier, P., Poeschl, M., van Zyl, J., Pugh, E., O'Brien, T. u. Goers, R.: Commons Configuration – Java Configuration API, 2016. https://commons.apache.org/proper/commons-configuration/, abgerufen am: 16.04.2016

[74] Hibernate, 2016. http://hibernate.org/orm/, abgerufen am: 16.04.2016

[75] Blumauer, A. u. Pellegrini, T.: Semantic Web und semantische Technologien: Zentrale Begriffe und Unterscheidungen. Springer 2006

[76] Vaadin Compare. https://vaadin.com/comparison, abgerufen am: 16.01.2016

[77] Die 5 wichtigsten Web Usability-Regeln | a coding project. http://www.a-coding-project.de/ratgeber/sonstiges/design/die-5-wichtigsten-web-usability-regeln, abgerufen am: 27.05.2016

[78] Manhartsberger, M. u. Musil, S.: Web usability. Das Prinzip des Vertrauens. Galileo Design. Bonn: Galileo Press 2002

[79] Huber, M.: Vaadin HighChartsApi-add-on. https://vaadin.com/directory#!addon/highchartsapi-add-on, abgerufen am: 17.01.2016

[80] Highcharts. http://www.highcharts.com/, abgerufen am: 26.05.2016

[81] Balzert, H., Klug, U. u. Pampuch, A.: Webdesign & Web-Usability: Basiswissen für Web-Entwickler. W3l GmbH 2009

[82] Roller, W., Berger, A. u. Szentes, D.: Technology based training for radar image interpreters. 2013 6th International Conference on Recent Advances in Space Technologies (RAST), S. 1173–1177

[83] Unity Technologies: Unity - Game Engine. https://unity3d.com/, abgerufen am: 16.04.2016

[84] Hollenberg, S.: Fragebögen. Fundierte Konstruktion, sachgerechte Anwendung und aussagekräftige Auswertung. essentials. 2016

[85] Experience API Working Group: Experience API (xAPI) Specification, 2013. https://github.com/adlnet/xAPI-Spec/blob/master/xAPI.md

II. Abbildungsverzeichnis

III. Anhang

Komponenten eines xAPI-Statements

Experience Application Programming Interface (xAPI) ist eine Spezifikation, welche es ermöglicht, Lerndaten zu speichern und wieder abzurufen (Kapitel 3.3). Die Daten sind dabei im JSON-Format spezifiziert und werden als Statement bezeichnet. Dieses besteht aus den Grundelementen *Actor Verb Object* und kann um Metadaten erweitert werden. In Tabelle 6 sind alle möglichen Elemente eines xAPI-Statements aufgeführt und die Bedeutung ihrer Eigenschaften erläutert.

Tabelle 6 Elemente eines Statements [50][85]

Element	Typ	Beschreibung	Erforderlich	Eigenschaften
Id	UUID	UUID wird gesetzt vom LRS, wenn nicht bereits vom Activity Provider geschehen	Empfohlen	
Actor	Objekt	Gibt an, über wen das Statement handelt, kann Agent oder Gruppe sein	Erforderlich	Erforderlich: - Inverser Identifier Optional: - name: languageMap - objectType - (member: [Actor])
Verb	Objekt	Aktion zwischen Actor und Objekt/ Aktivität	Erforderlich	Erforderlich: - id: UUID Empfohlen: - display: LanguageMap
Object	Objekt	Kann eine Agent, Gruppe, Aktivität oder ein neues Statement sein; falls nicht spezifiziert, wird es als Aktivität angenommen	Erforderlich	Aktivität: Erforderlich: - id: UUID Empfohlen: - objectType: String - definition: Object Agent, Gruppe oder Statement: - siehe jeweilige Eigenschaften - objectType
Result	Objekt	Gemessene Details, welche das Verb weiter spezifiziert	Optional	Optional: - score: Object - success: Boolean - completion: Boolean - response: String - duration: Timestamp - extensions: Object

Context	Objekt	Kontext des Statements, was zum besseren Verständnis beiträgt	Optional	Optional: - registration: UUID - instructor: Actor - team: Group - contextActivities: Activity - revision: String - platform: String - language: String - statement: Statement - extensions: Object
Timestamp	Datum/ Zeit	Zeitstempel (nach ISO 8601) wann Statement stattgefunden hat; wenn nicht angegeben, sollte LRS die Zeit im Element „stored“ setzen	Optional	
Stored	Datum/ Zeit	Zeitstempel (nach ISO 8601) wann Statement stattgefunden hat; gesetzt von LRS	Gesetzt durch LRS	
Authority	Objekt	Agent, der behauptet das Statement sei wahr; verifiziert von LRS basierend auf Authentifikation	Optional	
Version	Version	Statement assoziiert mit xAPI Version; formatiert gemäß Semantic-Versioning-1.0.0	Nicht empfohlen	
Attachments	Array von Attachment Objekten	Kopf der Anhänge/ Attachments bezogen auf das Statement	Optional	Erforderlich: - usageType: IRI - display: languageMap - contentType: InternetMediaType - length: Integer - sha2: String Optional: - description: languageMap - fileUrl: IRL

Konfigurationsparameter der ELAI-Architektur

Diese Ausarbeitung basiert auf vier wesentlichen Komponenten: die ELAI-UI, die ELAI sowie das SaFIR Serious Game und dessen ELAI-Adapter. Für die Kommunikation zwischen den entkoppelten Komponenten werden Servlets verwendet, deren URL sich abhängig vom Server der ELAI-Anwendung ändert. Um solche und weitere Konfigurationselemente leicht veränderbar zu halten, besitzen die Komponenten Konfigurationsdateien, die vom Nutzer vor dem Start des Systems modifiziert werden können (Abbildung III.1).

Die ELAI-UI bezieht die Daten für die Analyse und Adaption über die Servlets der ELAI. Hierfür muss die vom Anwendungsserver abhängige Domäne festgelegt werden. Diese Einstellung

ist in eine config Datei ausgelagert. Durch Entwicklung der ELAI-UI als Webanwendung ist es erforderlich, die Konfigurationsdatei in den WEB-INF Ordner zu verlagern, damit diese vom Deployment Descriptor ausgeliefert und von der Anwendung zur Laufzeit gelesen werden kann. Zum Auslesen der Parameter wurde die Apache Common Configuration Library verwendet. Das Package stellt die Möglichkeit bereit, Parameter im XML-Format oder als Textdatei zu bearbeiten. Verwendet wurde in dieser Ausarbeit eine normale Textdatei, welche als Key das Konfigurationselement und als Value den Wert besitzt. Da der Pfad des Servlets konstant ist, wurde dieser statisch implementiert. Somit beinhaltet die Konfigurationsdatei der ELAI-UI nur die Domäne der ELAI.

Eine weitere Konfigurationsdatei, die für eine Anpassung genutzt werden kann, befindet sich im ELAI Projekt. Wie bei der ELAI-UI wurde auch hier die Apache Commons Configuration Library verwendet, welche eine Datei im WEB-INF Ordner ausliest. Da die ELAI die Verbindung zum LRS bereitstellt, kann in dessen config Datei unter anderem der Endpunkt, der Nutzername und das Passwort zur Registrierung des LRS gesetzt werden. Weiterhin bietet sich dem Tutor die Möglichkeit, den Host sowie den Port eines Proxys einzustellen. Die Verwendung eines Proxys wurde dabei mithilfe von Default-Werten optional gehalten. Bei der Evaluation des Konzeptes ergab sich das Problem, dass die ELAI-UI alle Daten des LRS benötigt. Da das LRS jedoch mehrere tausend Statements beinhalten kann, würde dies zu einer enormen Ladezeit beim Start der ELAU-UI führen. Als Lösung wurde ein weiterer Parameter auf Seiten der ELAI eingeführt, der die maximale Anzahl zu ladender Statements begrenzt. Weiterhin implementiert die ELAI die Schnittstelle zur Kommunikation mit den Nutzerdatenbanken. Da diese Verbindung mithilfe von Hibernate realisiert wurde, befindet sich deren Konfiguration in der benötigten hibernate.cfg.xml Datei. In dieser können unter anderem die Datenbank-URL, der Nutzername sowie dessen Passwort gesetzt werden.

Wie bereits in Kapitel 4.1 und Kapitel 4.2 beschrieben, nutzt ein angeschlossenes Serious Game oder Simulationssystem einen ELAI-Adapter, der die Kommunikation mit der ELAI realisiert und über die Push- und Pull-Servlets Daten austauscht. Ähnlich der ELAI-UI benötigt dieser die Einstellung der ELAI-Domäne, um auf dessen Servlets zugreifen zu können. Die einstellbaren Parameter der Datei sind dabei die Domänen Adresse, der Hostname, der Port sowie die Namen der beiden Servlets.

Bei allen Konfigurationsdateien in dieser Ausarbeitung sind statische Default-Werte hinterlegt, die bei falscher Verwendung oder Verlust der config-Datei zum Einsatz kommen.

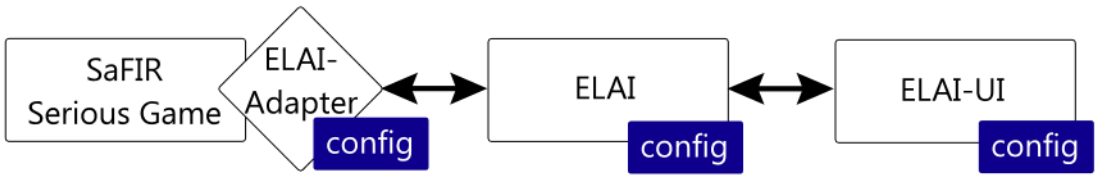


Abbildung III.1 Konfigurationsdateien Überblick