Fraunhofer IOSB

HOCHSCHULE SRH HEIDELBERG Intelligence in Learning

User Modeling using Graph Analytics for Adaptivity in E-Learning Systems

Master Thesis

by

Alexander Grissinger

Matriculation number: 11006444

October 30, 2020

SRH University Heidelberg
Faculty of Information, Media and Design
Applied Computer Science (Master of Science)

Reviewers:
Prof. Dr. Barbara Sprick (SRH University Heidelberg)
Dipl.-Inf. Alexander Streicher (Fraunhofer IOSB Karlsruhe)

Affidavit

Herewith I declare:

- that I have composed the chapters for the Master Thesis for which I am named as the author independently,
- that I did not use any other sources or additives than the ones specified,
- that I did not submit this work at any other examination procedure.

Heidelberg, October 30, 2020

Abstract

Diese Arbeit befasst sich mit der Thematik von Adaptivität in E-Learning anhand von Graphen. Die Fragestellung dieser Thesis war die Erstellung von Nutzermodellen basierend aus Daten eines Serious Games mit Hilfe von Graphen und Graph Algorithmen um diese für Adaptivität nutzen zu können. Dabei geht es darum die vorgegebenen Daten korrekt zu interpretieren um diese in einer Graph Datenbank visualisieren zu können. Ebenso sollen die Graphen analysiert werden um Nutzermodelle über bestimmte Nutzerverhalten bestimmen zu können die im Serious Game dafür genutzt werden dem Spieler individuelle Hinweise anzuzeigen. Dafür wurde zunächst in der Konzeptionierung der Rahmen für ein Lernszenario geschaffen, welches es ermöglicht ein Datenmodell abzuleiten aus dem Ansätze für Nutzermodellierung erstellt werden können. Des Weiteren wurde für diesen Zweck ein Tool entwickelt, welches ermöglicht die Nutzerdaten aus dem Serious Game in eine Graph Datenbank zu laden und mit Hilfe des Tools die vorliegenden Nutzerdaten zu analysieren.

This thesis deals with the topic of adaptivity in e-learning using graph analytics. The aim of this thesis was to create user models based on data from a serious game by using graphs and graph algorithms in order to use them for adaptivity. The aim is to interpret the given data correctly in order to visualise them in a graph database. The graphs will also be analysed in order to determine user models about certain user behaviour, which will be used in the Serious Game to display individual hints to the player. For this purpose, the frame for a learning scenario was first created in the conceptual design, which makes it possible to derive a data model. From this model concepts for user modelling can be created. Furthermore, a tool was developed, which allows to load the user data from the Serious Game into a graph database automatically and to analyse the existing user data with the help of the tool.

Contents

1 Introduction

In the paper [Kerres and Preußler, 2012], think of e-learning as different types of learning that make use of digital media. This includes presentations, provisioning of learning material or the communication between teacher and learner.

In addition, [Kerres and Preußler, 2012] describe that the success of e-learning services depends on the didactic preparation of the learning content and does not bring any significant advantages for the learning success itself. According to [Kerres and Preußler, 2012], however, there are successful examples in non-digital learning, where the user without a digital device supports himself through self-directed learning and receives support and feedback within the learning process. The question that arises is, what happens when e-learning contains a system which, as described by [Kerres and Preußler, 2012], gives the user continuous feedback for his learning processes?

According to [Streicher and Roller, 2015] e-learning systems do not, or barely, offer concepts of adaptivity and cannot, in their very nature, react to user activities and his needs. This means that every learner is shown the same learning content, regardless of individual needs. However, there are various research approaches in the field of adaptive learning systems or intelligent tutoring systems. For example, [Streicher and Roller, 2015] present a concept of an external system that works with interoperable standards to exchange information. Also, [Kareal and Klema, 2006] support the intention to link e-learning with an adaptive learning system, because the shown information can be adapted to the interests of the learner and therefore keep up the motivation. One idea to implement an adaptive system for e-learning is to use the Experience API (xAPI), which is used as an e-learning standard. This thesis aims on an approach at the usability of graphs and graph algorithms in an e-learning environment and continuing this question of using user models based on graph analytics for adaptivity.

This work belongs to the research projects on adaptive learning systems that are currently running at the Fraunhofer IOSB Karlsruhe in the Interoperability and Assistance Systems Department [IOSB, 2020].

Contributions:

- Conceptual design for creating a learning scenario and build an appropriate graph data model
- Approach to design user models for the learning scenario

- Architecture and implementation to automatically generate a graph data model and analyze the graph for user models

1.1 Problem Statement

The problem statement for this thesis is that in the game LostEarth 2308 the player does not get recommendations or hints that are personalized on him. In LostEarth 2308 a digital tutor already exists as LISA, but the hints from this system are not adapted to the user. So, there needs to be a connection between the serious game tutor LISA and a system that can model and analyze the players' behaviour. So, the problem that precedes this thesis is the question which kind of data from the collected xAPI statements is needed in order to make decisions about the users' behaviour and how to use them for adaptivity. So, it is to clarify what the xAPI statements should contain and at which points it makes sense to track and store the activities. Another problem is to process the data into a graph database automatically, where the activities need to be modeled in a meanigful view.Furthermore a problem is, how the analyzed data from the graph can be returned to a serious game to be used there.

The problem this thesis deals with is, that the hints and recommendations in the game LostEarth 2308 are just generic and not personalized for the specific user. But depending on the different behaviors of the user those generic hints are not needed. Another problem currently in LostEarth 2308 is, that the learning experience is the same for each player, because the content of the game is static. This can cause a lack in motivation for the players in example. This work aims to generate user individual recommendations to improve the recommendation functionality of LISA and to individualize the learning experience for the player in this approach.

1.2 Solution Approach

The objective of this thesis is to develop a framework that enables the conceptualisation of scenarios for adaptivity with the application of graphs and its visualizing effect by applying data models on the tracked xAPI statements to allow generating user models and gain insights in the different behaviours of the players by analyzing those user models by support of different algorithms. Furthermore, a system needs to be developed, that can transfer this described data in form of xAPI statements from a serious game into a graph database. Solution approach is that the data will first be analyzed and finally used in an serious game for adaptivity in form of recommendations to indivdualize the experience for the players. To analyze the data it needs to be collected from the game in which the adaptivity approach should be applied - in case of this thesis the game is Lost Earth 2308, which is a serious game used for image interpretation assessments. To accomplish this a tracker needs to be implemented. Furthermore, a system which is implemented as the

Graph Analysis Service is needed to communicate with a graph database, by putting the xAPI-statements in it, defined in a specific datamodel. Furthermore, this system needs implementations of pattern matching methods to analyse the graphs in the database about specific patterns and then return the results back into Lost Earth 2308.

2 Fundamentals

Chapter 2, covers the fundamentals to give a solid understanding of terms, required technologies and techniques. This chapter covers the information about the basics of the used graph algorithms for user modeling in graphs and the used graph database Neo4j. Furthermore, this chapter includes different standards and technologies like the Experience-API in the field of e-learning or the at Fraunhofer IOSB assistant ELAI for adaptive learning systems. The sections included in this chapter, with a brief description are listed as following:

- **xAPI Data Retrieval Components:** This section gives an overview on the different used tools and systems to transfer the needed xAPI statements from a serious game into a Learning Record Store (LRS). This section covers the explaination about usage and anatomy of the xAPI statements, how the data is tracked while playing a serious game and where the tracked xAPI statements are stored.
- **E-Learning A.I.:** This section gives an overview on the usage of the E-Learning A.I. (ELAI). In this thesis the ELAI collects adaptivity requests from the serious game LostEarth 2308 and proceeds these to the Graph Analysis Service.
- **Graph Database Neo4j:** Neo4j is used to visualize the xAPI statements in a graph structure and to perform graph analysis methods for user modeling by applying graph algorithms.
- **Graph Algorithms:** In this section the fundamentals for the implemented graph algorithm and the data representation of the graphs are explained. The implementation of the procedures is explained in subsection 5.1.1.

2.1 xAPI Data Retrieval Components

E-Learning Standard: xAPI
The Experience API (xAPI) is used as a technical specification to document and communicate learning experiences. The xAPI offers a standardized grammar consisting of a triplet structure, that always contains an actor, a verb and an object. The listing 2.1 shows an example xAPI-Statement. [ADL.net, 2020]. According to [ADL.net, 2020] the xAPI is used to understand and compare learning experiences and to maximize interoperability of services that want to process information about learning experiences.

Actor: The Actor defines, **who** is doing an action. It distinguishes between an Agent, which is an individual persona or system or a Group, which is a collection of Agents [ADL.net, 2020].

Verb: The Verb defines the action between an Actor and an Activity. The Verbs are not specified by the xAPI, reasoned because a predefined would be limited by defintion and could not be able to capture all the demanded learning experiences [ADL.net, 2020].

Object: The Object defines the thing, that was acted on. An Object can be an Activity, Agent or Group, Sub Statement or Statement Reference [ADL.net, 2020].

```
{
    "actor": {
      "name": "Sally Glider",
      "mbox": "mailto:sally@example.com"
    },
    "verb": {
      "id": "http://adlnet.gov/expapi/verbs/experienced",
      "display": { "en-US": "experienced" }
    },
    "object": {
      "id": "http://example.com/activities/solo-hang-gliding",
      "definition": {
        "name": { "en-US": "Solo Hang Gliding" }
      }
    }
}
```

Listing 2.1: Example xAPI Statement [xAPI Overview, 2020]

In this thesis, the xAPI statements are used to track learning activities and to build graphs based on the defined data model. This is explained in concept section 4.2. The tracking of activities with the xAPI-Statements is explained in this section.

Learning Record Store

A Learning Record Store (LRS) is a server that is responsible for receiving, storing and providing access to the Learning Records. A Learning Record, is an account of a learning experience, formatted to the rules of the xAPI [ADL.net, 2020].

The used LRS in this thesis is Learning Locker[1] and it stores all the xAPI-Statements coming from the serious game Lost Earth 2308. in this storage they are then further processed towards the Graph Analysis service which stores them into the graph database Neo4j (section 2.3).

xAPI-Tracker

The functionality of the xAPI tracker is explained as follows. The xAPI tracker is used to

[1] https://docs.learninglocker.net/welcome/

observe user activities from the serious game LostEarth 2308 and build xAPI statements. Those xAPI statements are then further processed into the LRS. The xAPI-Tracker is directly linked to the serious game Lost Earth 2308 and is used as an interface between the game and the LRS. The six steps do the following:

1. User is playing Lost Earth 2308
2. User triggers event while playing, e.g. starting a mission
3. The code contains an implementation for the xAPI-Tracker for this Event
4. The triggered events is sent to the xAPI-Tracker
5. The xAPI Statement gets saved
6. The xAPI Statement is sent to the Learning Record Store

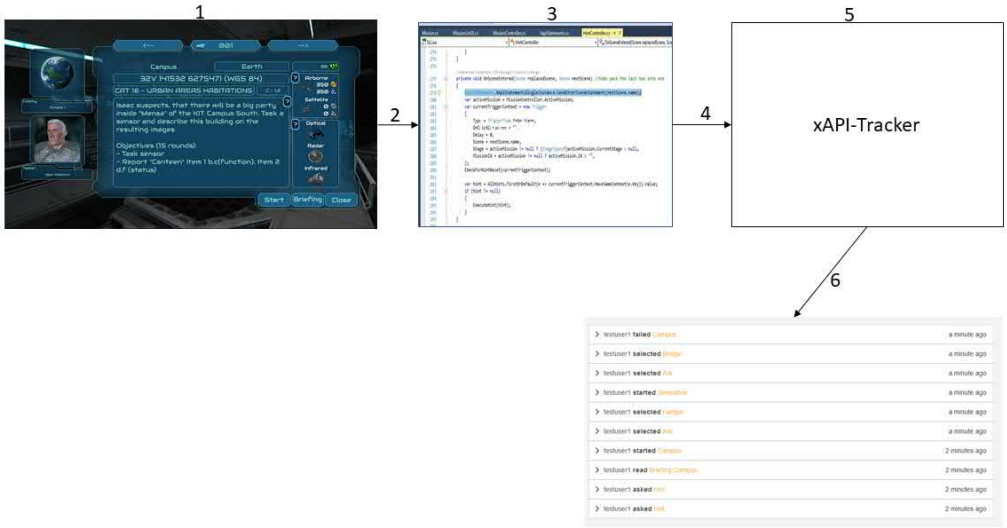

Figure 2.1: xAPI-Tracker Implementation

In this thesis the xAPI-Tracker is used to to track and send the learning activities to the attached LRS while a learner is playing a session. The xAPI-Tracker is part of the system architecture.

2.2 E-Learning AI

The E-Learning A.I. is a framework which allows the use of adaptivity. Therefore, the e-learning games or simulations need to be attached to the ELAI. Generally said, the framework is an intelligent tutor that interprets data from individual games to adapt the game content according to the output from the ELAI [Streicher and Roller, 2017]. In the

context of this thesis the ELAI is used as an interface between the components serious game and the from the graph analysis procedures analysed xAPI statements. The ELAI consists of a communication layer, which is used to mediate the data between the actual game - in this thesis Lost Earth 2308 and the ELAI Controller [Streicher and Roller, 2017]. For this part of the communication, the interoperable E-Learning protocol, xAPI, is used, which got formerly explained in section 2.1. How the communication with the xAPI is working for the example of Lost Earth 2308 is explained in section 5.1. The central component of the ELAI framework is the ELAI controller. The ELAI controller has an interpretation engine for the usage of data analysis and an influence engine for selecting adequate reactions [Streicher and Roller, 2017]. The architectural overview of the ELAI is displayed in 2.2.

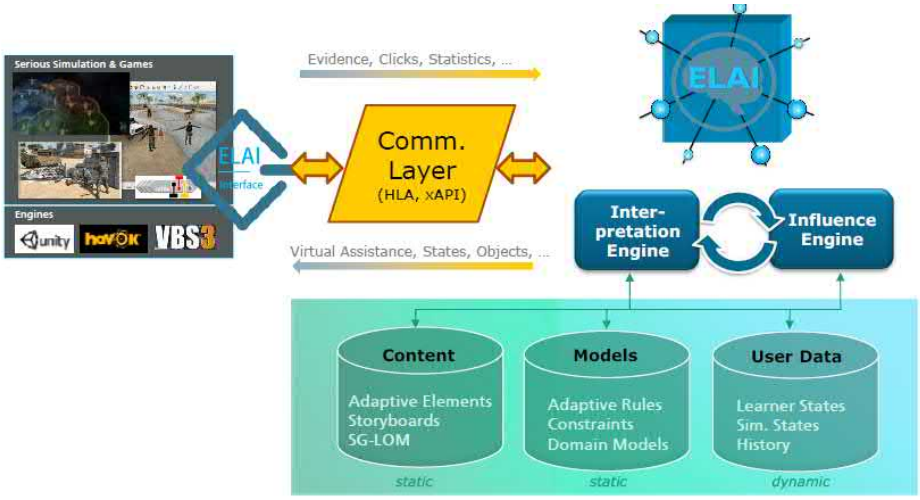

Figure 2.2: ELAI Architecturial Overview [Streicher and Roller, 2015]

In this work, the ELAI is used as a big frame, where the implemented graph service is attached to. It is used to send outgoing adaptivity requests from the attached game to the graph service and also to return the results back to the game.

2.3 Neo4j as Graph Database

Neo4j is a graph database. This graph database or *graph database management system* allows to perform Create, Read, Update and Delete (CRUD) operations. Relationships are the most important point in a graph database and the underlying data model. An important aspect of working with graph databases is data modelling. It aims to structure an abstract target to manipulate it later [Robinson et al., 2015]. Neo4j offers several models, one of those is the labeled property graph model. This consists of *nodes, relationships, properties* and *labels* [Robinson et al., 2015]. The model has various properties that need to be taken into account:

- Nodes contain properties. Nodes should be understood like a document, the properties can be stored in the form of key-value pairs [Robinson et al., 2015].

- Nodes can be tagged with one or more labels. Labels group nodes together and are an indicator of their role in the data set [Robinson et al., 2015].
- Relationships are used to connect nodes together and structure the graph. A relationship always has a direction, a name, a start and an end node. The direction and the name of a relationship provide semantic clarity when structuring nodes [Robinson et al., 2015].

An example for the labeled property graph model is shown in figure 2.3. To create graphs

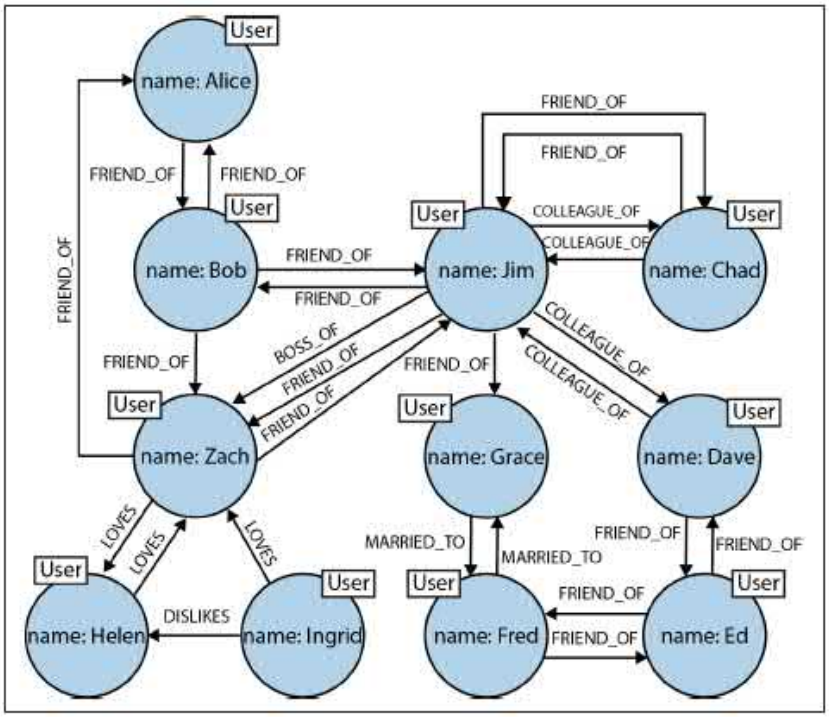

Figure 2.3: Labeled Property Graph Model Example with Nodes [Robinson et al., 2015]

or to manipulate a graph, a query language is required. In Neo4j this is called Cypher. With this programming language graphs can be easily described [Robinson et al., 2015]. Cypher is designed to be easily read and understood. Cypher helps to query the database for data in the graph or for specific patterns [Robinson et al., 2015].

In [Needham and Hodler, 2019] the applicability of Neo4j graph algorithms is described. It shows that Neo4j offers many different pre-implemented graph algorithms that can be used for different approaches in graph analysis. The categories from the implementations in Neo4j are as follows:

- Pathfinding and Graph Search Algorithms
- Centrality Algorithms
- Community Detection Algorithms

Neo4j also offers libraries for various programming languages and frameworks that allow Cypher to write queries within the programming language and connect to Neo4j [Neo4j Graph Database Platform, 2020]. For the .NET framework, that is used for implementing the graph analysis service, Neo4j offers a library to compute Cypher Querys in C♯ [DotNet4Neo4j, 2020].

In the context of this thesis Neo4j is used as a database to store the data from the xAPI statements as a graph in a certain data model. Also, storing the xAPI statements in a graph strucutre allows the application of graph analysis operations and graph algorithms to build user models that can be used for adaptivity purposes. More details about the usage of Neo4j and graphs can be viewed in section 5.2.

[Miller, 2013] explains that the strength of graph databases is the way of effectiviely retrieving information out of graphs by performing a traversal. A traversal means walking along the elements of a graph and is a fundamental operation for data retrieval [Miller, 2013].

2.4 Graph Algorithms

This section describes the used methods and algorithms according to graphs that have been used in this thesis.

Yen's k-Shortest Paths

[Yen, 1971] describes the Yen's k-shortest paths algorithms as an algorithm that is used to find the lenghts of al shortest paths from a fixed start node to all other nodes in a non-negative distance network. [Needham and Hodler, 2019] explain that this algorithm is useful to find alternative paths in a network. According to [Yen, 1971] the following steps are needed to find the K shortest paths:

[Yen, 1971] explains that the Yen's k-shortest paths algorithm gets terminated when there are negative loops in the network. If there are only non-negative loops at least one path with the shortest lenght should be found [Yen, 1971]. Important notations and definitions according to [Yen, 1971] are, that in an N-node network $i = 1, 2, ..., N$, 1 is the source and N is the sink.

In iteration 1 A^1 is determined: [Yen, 1971] explains that A^1 is determined by an efficient shortest path algorithm, like Yen's algorithm that finds the lenghts of all shortest paths from a fixed node to all other nodes in an N-node non-negative distance network [Yen, 1971]. If the result is less than K and more than one path, any arbitary of these paths is stored in List A as A^1, which is the list of the k-shortest paths [Yen, 1971].

In iteration $k(k = 2, 3, ..., K)$ A^k is determined: The author [Yen, 1971] shows that to find A^k , the shortest paths $A^1, A^2, ...A^{k-1}$ must have been previously determined and A^k is found in the following procedure [Yen, 1971]:

Algorithm 1: Determine A^k [Yen, 1971]

foreach *i = 1,2,...,* Q_{k-1} **do**
  for $j = 1, 2, ..., k-1$ **do**
    if *subpath of the first i nodes of* A^{k-1} *in sequence = with the subpath consisting of the first i nodes of* A^j **then**
      $d_{iq} = \infty$ where q is the $i+1$st node of A^j

[Yen, 1971] explains that in the next step a shortest-path algorithm is applied to find the shortest path from i to N, by allowing it to pass nodes that are not yet included in the path.

Graph Representation

For the graph $G = (V, E)$ with a collection of nodes V and edges E two different ways for representation exist [Cormen, 2009]. [Cormen, 2009] say that one way is the representation with *adjacency lists* and the other is with *adjacency matrices*. Both ways apply to directed and undirected graphs. To represent a graph $G = (V, E)$ as an *adjacency matrix*, the nodes from the graph need to be numbered in any form from 1 to $|V|$. The representation of the graph G is in a $|V|$ x $|V|$ adjacency-matrix $A = a_{ij}$ with the elements [Cormen, 2009]:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

In this thesis the graphs are represented as *adjacency matrices*, after Neo4j returned the path of the player activities to multiply those for retrieving the visited nodes within two or more paths. The implementation of this approach is shown in subsection 5.1.1.

3 State of the Art

According to the frame of this thesis, this chapter covers the state of the art in the required research fields. The chapter contains the following sections:

- **Adaptive E-Learning:** This section provides the state of the art about existing projects and research approaches in the field of adaptive e-learning- and tutoring systems in their derivations and results.
- **User Modeling in E-Learning:** This section gives the reader an insight into the different approaches how and for which reasons users are modeled and which conclusions can be drawn.
- **Graph Databases and Analytics for E-Learning:** In this section different fields of the usage of graph databases are described and it shows approaches from e-learning analytics in which graphs and graph databases were used.
- **Application of Graph Pattern Matching:** This section shows, beside the approaches from graph theory, practical examples in which graph pattern matching approaches for different domains are used. It focuses on the procedure of the methods in the individual application examples.

3.1 Adaptive E-Learning

In [Streicher et al., 2019] the authors write about the systematic development of adaptive e-learning systems. This paper presents a concept for an interoperable, flexible testing tool for adaptive e-learning system development. The concept contains a prototype implementation in a serious game that contains an xAPI recording functionality combined with a visualization of the usage flow.

The paper from [Streicher et al., 2018] focusing on the problem statement when an adaptive serious game needs to adapt, that means to automatically personalize/customize the learning experience, with the additional question how to effectively assess user progress. [Streicher et al., 2018] say, that one important parameter is the correct timing, to really match the players' needs. Adaptive serious games are trying to personalize gaming and learning experience to maximize the learning outcome. An effective adaptivity is based on user or learner models that contain all demanded information to guide the user through the game in an adaptive manner [Streicher et al., 2018]. The idea in [Streicher et al., 2018]

is to measure the progress of the user by looking at the purposefulness or goal orientedness of the actions the user is taking. So for example, a user who is lost in the game and is moving in a wrong direction should be assisted by an adaptive system, whilst a user who is working efficiently towards a goal does not need to be assisted by an adaptive system [Streicher et al., 2018].

[Streicher et al., 2018] provide an approach that measures a users's goal-orientation. The attempt is a definiton of a metric that is measuring the distance between an ideal path and the observed actions of the user. The paper contains the concept and a work-in-progress of the Ideal Path Model (IPM) and the Ideal Path Score (IPS). The IPS is meant to improve the adaptivity of serious games by more accurately estimating performance and need for help, based on the interactions of the player that is supported by the movements of the users' eyes. To evaluate the attention level, regarding to the goal-orientedness, the authors created a reference model for the IPM which can be seen in figure 3.1. The IPM builds on several blocks, which are [Streicher et al., 2018]:

- **Scene Manifestations:** Captures the current state of the scene
- **Interaction Elements:** All game elements a player can interact with
- **Ideal Path:** Going through the sequene of all secene manifestations and interaction elements
- **Actual Path:** Reflects the actual sequence a player is taking

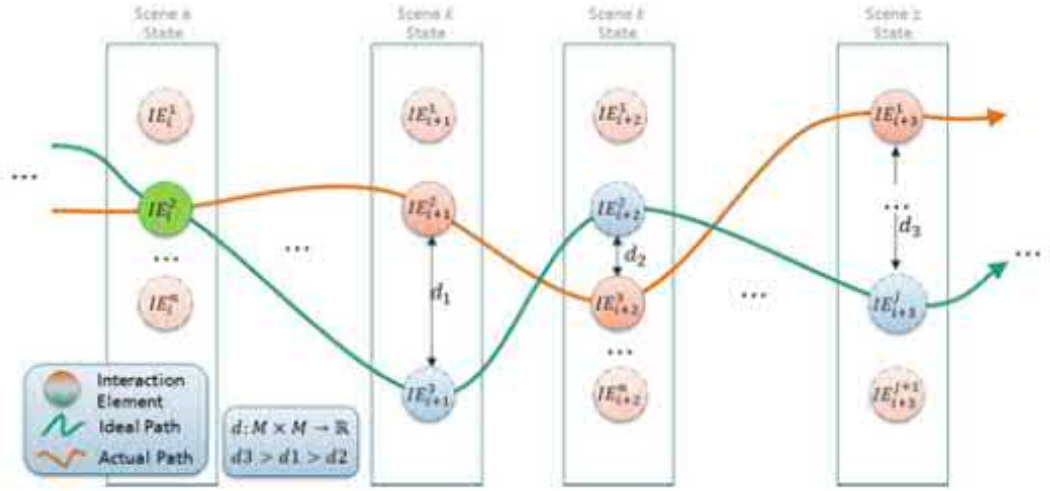

Figure 3.1: Reference Model IPM [Streicher et al., 2018]

The IPM is meant to describe all necessary steps to reach the goals inside the scene by not taking unnecessary detours. The IPS is a normalized score between $[-1; 1]$ where a $IPS = 1$ means a perfect game move, equivalent to the ideal path, a score from $IPS = 0$

means that there is no impact on a learning process and a score from $IPS = -1$ means that the user is moving on a wrong path [Streicher et al., 2018].

[Streicher and Smeddinck, 2016] explain their idea that personalization and adaptivity can promote motivated usage, increase the user acceptance and the identification with the serious game. They explain, that in the context of games, adaptivity describes an automatic adaption of different game elements which can be:

- Content
- User interfaces
- Game mechanics
- game difficulty

Furthermore, [Streicher and Smeddinck, 2016] introduce the adaptive cycle which is shown in figure 3.2. The adaptive cycle is based on the idea from [Shute and Zapata-Rivera, 2012]. In general, the typical 4-phased adaptive cycle consists out of

- **Capture phase:** Capturing the interaction data from the user and gathering information about the learner. This phase is used to update internal models that are maintained by the system [Streicher and Smeddinck, 2016, Shute and Zapata-Rivera, 2012]
- **Analyze phase:** Part where e.g. an adaption engine attempts to read the captured data. This can include qualitative or quantitative analysis techniques, machine learning, data mining or other techniques from artificial intelligence. Based on the analysis of the data, a user model is created. This phase typically is representing information in terms of inferences on current states [Streicher and Smeddinck, 2016, Shute and Zapata-Rivera, 2012].
- **Selection phase:** Based on the collected and analyzed data, the system makes a selection on which content should be provided to the user and also it chooses when and how to adapt. So this is the actual adaptation process [Streicher and Smeddinck, 2016, Shute and Zapata-Rivera, 2012].
- **Presentation phase:** After the selection, a response needs to be presented. This could be new content or more subtle system adjustments [Streicher and Smeddinck, 2016, Shute and Zapata-Rivera, 2012].

Also, [Streicher and Smeddinck, 2016] are discussing the problem of a cold start of the analysis, because initially a system can only act on the captured data of the first run.One introduced possibility is to provide an adaption engine to gain more information about the user model by in example asking questions before the actual assessment or serious game starts.

In the paper [Streicher and Smeddinck, 2016] the authors are discussing the questions about when, what and how to adapt. In the explained adaptive cycle, the user model

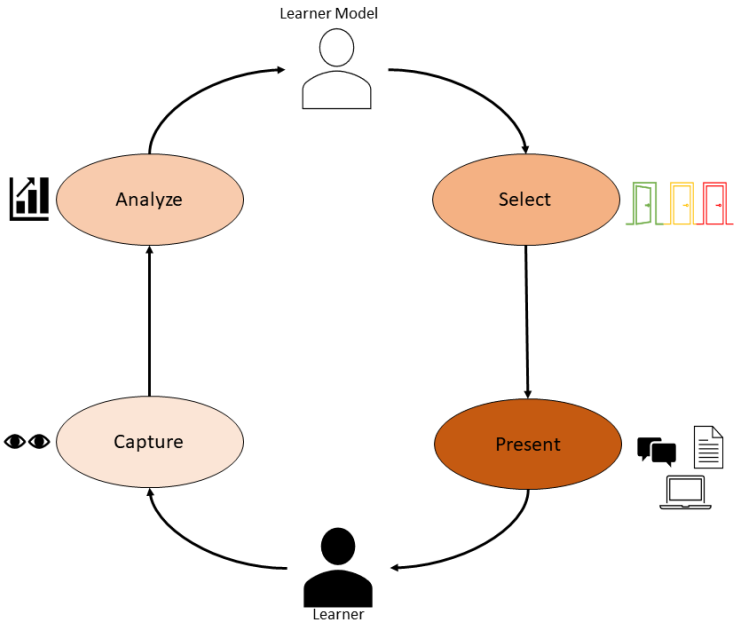

Figure 3.2: Adaptive Cycle For Adaptive Learning Systems (Adapted from [Streicher and Smeddinck, 2016, Shute and Zapata-Rivera, 2012])

gets more concrete the more cycles were processed. When to adapt is the key question according to [Streicher and Smeddinck, 2016], because adaptivity needs to be justified at each step when a user is playing a game that is adaptive. The authors state that adaptive software needs a reaction model that determines when to start the process. An example for when to adapt is when there is a measure of a decrease of motivation or the learning progress is below the threshold. [Streicher and Smeddinck, 2016] explain that what to adapt is in the first step depending on which media, rules or game mechanics are used, because by taking a serious game as an example, it cannot easily be split down into atomic parts without the risk of destroying specific mechanics. How to adapt means according to [Streicher and Smeddinck, 2016] the involvment of the mechanics behind the adaption. This part relies on the given software architecture or the techniques from A.I..

[Shute and Zapata-Rivera, 2012] present different scenarios that can be represented in the 4-process adaptive cycle, because the model accomodates alternative types and levels of adaption. The adaptive cycle allows a full-adaptive scenario, where all processes of the cycle are exercised. The cycle will continue until all the goals have been met. A less adaptive scenario is where the learner is allowed to interact with the learner model. The next scenario is a diagnosis-only-scenario in which the learner is monitored and the learner profiles are updated, but in which no adapted content is. A short-memory cycle is a further presented scenario, where adaption is performed using information gathered from the latest interaction. The last introduced scenario is short memory, no-selection cycle, in which a predefined path strucutre is followed and no learner model is maintained. The predefined path dictates the educational resources that are presented to the learner

[Shute and Zapata-Rivera, 2012].

[Lim, 2015] gives an example of tracking learning progress in serious games by using the xAPI and LRS to keep on track with the players' activities and interactions.

[Blatsios and Refanidis, 2019] present an adaption and personalisation methodology for serious game design. First of all they present the general approach, the Task Based Learning (TBL). This theory is based on tasks that derive from the characteristic of having the ability to arouse the interest of the learner. The simplest way to illustrate such tasks is to develop a linear story in a game. According to [Blatsios and Refanidis, 2019] this approach is not feasible for adaptive serious games, because the linear story is too inflexible. The authors [Blatsios and Refanidis, 2019] describe in their paper a modular, non-linear design approach to enable adaptivity and personalisation in serious games. It is implied that the game design has to be as modular as possible to allow concurrent and continuous development processes. For this purpose the designer of the game should create different tasks that can be freely linked to each other and thus offer plenty of possibilities for adaptation when linking possible storylines and paths. Figure 3.3 shows the approach of the methodology for modular game design. A task can have different properties, which are typically defined by the author of the task. A task can be a puzzle or similar. A task should contain various additional parameters like expected duration or entertainment value. These should be used to evaluate the current gameplay and to adjust the difficulty of the task during runtime [Blatsios and Refanidis, 2019].

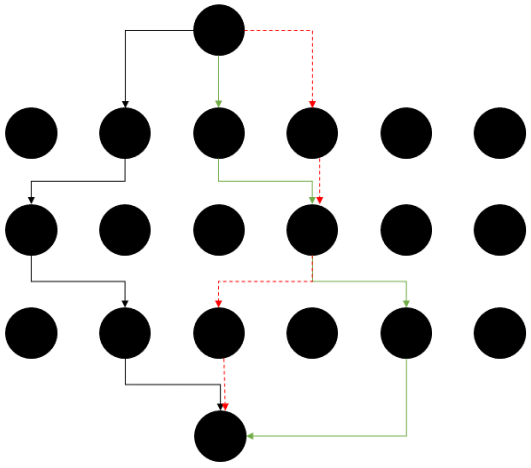

Figure 3.3: Modular Game Design (Adapted from [Blatsios and Refanidis, 2019])

[Khribi et al., 2008] describe in their paper "automatic recommendations for E-Learning personalization based on web usage mining techniques and information retrival" an approach that automatically makes recommendations to active learners without the need of explicit feedback from the learners. The recommended learning content is based on the navigation history of the learner and as well on the disclosure of similarities and differences between the learner's preferences and the learning content. [Khribi et al., 2008] address the problem that personalised e-learning applications are still mostly developed in research

institutions and that other e-learning systems still present standardised content for each user despite the different user learning-profiles involved. [Khribi et al., 2008] distinguish between different knowledge types and gives some examples like learners' knowledge, learning material knowledge, learning process knowledge. [Khribi et al., 2008] also distinguishes between general points for personalised e-learning, such as *adaptive course delivery* which is the most common and used adaption technique in e-learning, like dynamic course restructuring or adaptive selction of learning objects [Khribi et al., 2008].

According to [Khribi et al., 2008] using web mining techniques in e-learning for personalization has increased, because of the broad availability of web based e-learning systems and the need to personalize the content for the students. In [Khribi et al., 2008] the authors explain that the personalisation and recommendation process can be approached in different ways. For example, with classification, clustering, prediction, association rule or a sequential pattern. Clustering, for example, allows students to be differentiated according to their learning characteristics, which makes it possible to provide individual paper recommendations [Khribi et al., 2008].

[Khribi et al., 2008] provides the architecture and functionality of a framework for building automatic recommendations in e-learning platforms, which consists of two components, the *modeling phase* and the *recommendation phase*. The framework consists of two modules. One is used to pre-process data, which is used to create learner and content models. The other one is used to apply the created learner models and to identify the students' goals and send a recommendation to the users. After creating the user models, all existing learning content is examined and an *inverted index* is created, where each keyword is mapped to a list of pages containing these keywords. In order for the recommendations to take place for the learner, the user preferences during the session are determined based on the names of URLs. These are then processed using various methods [Khribi et al., 2008]. In the *recommendation phase* the first step is to create a query that contains all recently visited learning content. This is represented by a vector. The recommendation phase is shown in figure 3.4. The learning objects are visited URL's of the learner, as only the last visited websites are tracked, a timestamp is used. The individual terms of the visited website are stored in the vectors [Khribi et al., 2008].

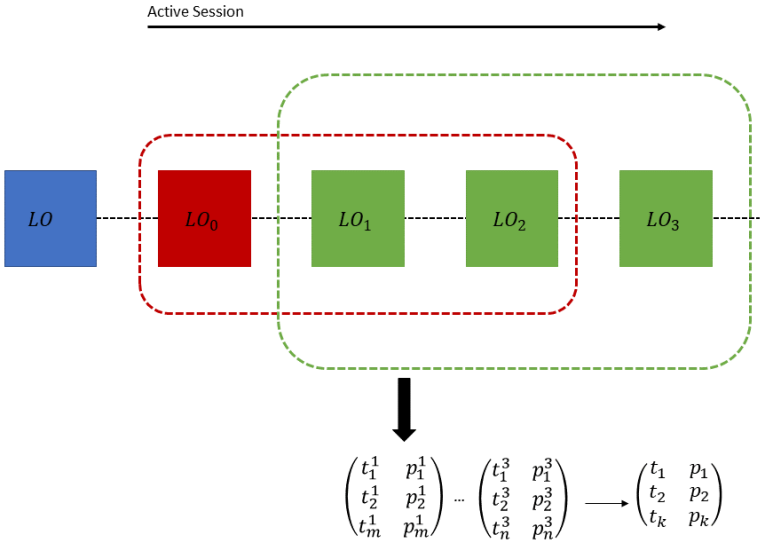

Figure 3.4: Term Vector Building Process (Adapted from [Khribi et al., 2008])

The *recommendation process* uses methods from content based filtering (CBF) and collaborative filtering (CF). The results are ranked according to the cosine similarity of their content [Khribi et al., 2008].

3.2 User Modeling in E-Learning

[Brusilovsky, 1998] describes a user model as a representation of information about an individual person, which is the basis for an adaptive system to create an *adaption effect*. [Brusilovsky, 1998] implies with *adaption effect* that the system behaves differently, but adapts to the user. This adaptive behaviour can have different characteristics, e.g. when a user searches for information, the system can adaptively prioritise the items according to relevance [Brusilovsky, 1998]. [Brusilovsky, 1998] also explains that adaptivity and user modeling are two sides of the same coin and the amount of information about the user is reflected in the adaptivity effect. [Brusilovsky, 1998] says, that different aspects of the user can be modelled, especially in adaptive educational systems, user models represent the knowledge of the user, but also overlaps with user models about personal backgrounds or goals of the persons [Brusilovsky, 1998].

[Streicher et al., 2018] describe user modeling as a model that contains information about the user, like his abilities that can be measured explicit or implicit. By measuring explicit, e.g. by directly asking questions to a user, it could have a negative impact on the flow eperience on the user [Streicher et al., 2018]. On other hand, implicit measuring means the approach to estimate the current learning progresses or cognitive states [Streicher et al., 2018].

[Sweta and Lal, 2016] provide a paper that introduces an automatic student modeling

approach for identifiying learning styles in learning management systems. To identify the learning style of each student, the paper proposes the use of fuzzy cognitive maps (FCM). The authors state that with the usage of FCM to model the students, the results show that the different learning styles can be detected for the students. [Sweta and Lal, 2016] explain that by knowing different learning styles, the system can be optimized for learning processes and improve the effectiveness. The authors of [Sweta and Lal, 2016] describe different learning types according to the *Felder-Silverman Learning Style Model* (FSLSM) which table 3.1 shows.

Table 3.1: Different Learning Styles According to FSLSM (Adapted from [Sweta and Lal, 2016])

Preferred Learning Style	**Dimension**
Sensory-concrete material, more practical, std procedure	Perception
Intuitive-abstract material, innovative, challenges	Perception
Visual-learning from picture	Input
Verbal-learning from words	Input
Inductive- It has proved that engg. students are Inductive	Organization
Deductive	Organization
Active-Learning by doing, group work	Processing
Reflective-learning by thinking work alone	Processing
Sequential-learn in linear steps, use partial knowledge	Understanding
Global-learn in large steps, need big picture	Understanding

FCM are fuzzy signed graphs with feedback [Sweta and Lal, 2016]. A FCM forms a dynamic model that consists of collection of concepts and there are cause and effect relationships among them. FCM consist of nodes represented by concepts C_i and interconnections strength e_{ij} between a concept C_i and C_j. The interconnection's strength e_{ij} among concepts is characterized by a weight w_{ij} [Sweta and Lal, 2016]. Figure 3.5 shows an example of a FCM.

With the FCM and FSLSM [Sweta and Lal, 2016] try to model learning styles, mapped between the actions from a student in the system and then try to find a best fitting learning style. To accomplish this, the inputs for the network are required to be identified and the outputs with the possible values need to be decrypted. As input layer, the automatic captured activity data from the user is taken and stored as a log file. The data consists of

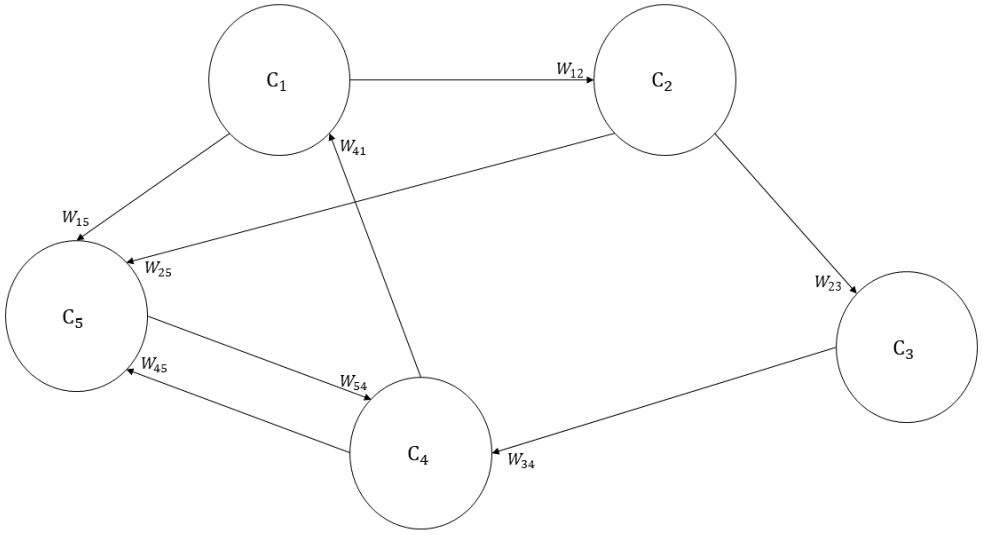

Figure 3.5: Simple FCM Structure (Adapted from [Sweta and Lal, 2016])

information like number of visits, time period and order of visiting the learning material. [Sweta and Lal, 2016] explain that this amount of data has measuable patterns. To analyze the data they use a learning style diagnostic module. The output are fuzzy variables which are low, medium and high to measure the learning style.

[Beel et al., 2015] deal with the question of the potential by using mind maps for user modeling. The approach is based on the premise that mind maps generally contain a lot of information and can possibly be used in recommender systems. The user modelling approaches were applied and evaluated by using the reference management software *Docear*. Based on 4700 user mind maps, 430893 research paper recommendations were displayed. According to [Beel et al., 2015], the use of mind maps has gained attention in a variety of domains, for example, they are used to implement lambda calculators, filter search results from google, present software requirements or as a learning tool. Examples for user modelling with mind maps in recommender systems, [Beel et al., 2015] names the mind-mapping and reference-management software *Docear*, furthermore there have been approaches for personalised advertising by *MindMeister* and *Mindomo*. The functionality of *MindMeister* is that terms are extracted from nodes with which a user model can be created and this user model is sent as a query to Amazon Web Services, for example [Beel et al., 2015]. The return values are book recommendations from Amazon [Beel et al., 2015]. *Mindomo*, seemed to use a similar approach for advertisement. However, it was abandoned like *Mindmeister* due to lack of interest [Beel et al., 2015]. [Beel et al., 2015] decided not to analyse all of the users mind maps, because some, especially old data sets, could bring noise into the user models. To evaluate the user models the Click-Through Rate (CTR) was calculated. [Beel et al., 2015] follows different approaches, for example only the x most frequently accessed mind maps or the most frequently processed nodes of the users were analysed. The most effective way for user modeling in [Beel et al., 2015] was selecting random nodes and extending those with their siblings and

children. [Beel et al., 2015] considered node weighting for the mind map user models. The approach was to weight the nodes according to their depth related to the root node. To calculate the weight of a node the authors used to multiply the default value 1

a) with the absolute depth of the node,

b) by the natural logarithm of the depth,

c) by the common logarithm of the depth or

d) if resulting weight < 1 by the $\sqrt{depth}$

[Khribi et al., 2008] describe the user modelling approach using clustering for learner sessions. Each cluster contains similar user sessions and similar interests. Each cluster can be seen as a single learner model. [Khribi et al., 2008] notes that it is difficult to decide which learning objects best fit a student in a given situation. The learner model must contain two different types of information. One is domain specific information and the other is domain independent information [Khribi et al., 2008]. In [Khribi et al., 2008] the learner model represents a sequence of weighted learning objects and thus users can be grouped by common interests.

[Khribi et al., 2008] distinguish between two approaches of user modeling, namely the *collaborative user model* and the *automatic learner model*. The *collaborative user model* requires explicit information about learners' preferences and needs, whereas in the *automatic learner model* the information retrieval is done automatically, based on the online behaviour and activities of learners.

In [Birk et al., 2015] data from over 3400 users were collected for the creation of user models from the game *Pot Farm* from 2010. *Pot Farm* is a social network game that benefits from the use of social media services by the players. Player-centric and personality-traits along with validated measures of player experience and and logs of ingame behaviours of the players were stored. In order to collect data from players, various surveys were conducted to find out about player gender, age, player centric traits and personality traits. The aim of [Birk et al., 2015] was to find out how the link between satisfaction and motivation is when these are moderated by personality or player traits. For this purpose moderated multiple regression analysis was applied.

3.3 Graph Databases and Analytics for E-Learning

According to the review from [Angles, 2012] graph databases have a broad usage in many different areas, for example chemistry, biology, web mining or the semantic web. [Angles, 2012] explains, that one important element of a database is the data model, which is a toolset used to model representations of real-world entities. [Angles, 2012] characterizes graph database models as a combination of data structures for the schema and instances that are modeled as graphs. Furthermore, analytics are performed by graph-oriented expressions.

In his work [Miller, 2013] writes about the usability of graph databases. He tells that graph databases are an effective tool for modeling data, when the focus is on relationships between the entities and so almost everything can be represented in a corresponding graph [Miller, 2013]. [Miller, 2013] presents different application areas of graph databases:

- **Social Graph:** A social graph is a network that is used to qunatify relationships between individuals. He tells about the friend of a friend problem, that would require a lot of work in a relational database, because of the recursive data structure which would require a expensive join tables in a relational database whilst in a graph database traversing across an edge has very little costs[Miller, 2013].

- **Recommender Systems:** [Miller, 2013] says that recommender systems are used to advise a user on relevant products and information by predicting the intersests based on various data. Naturally, two correlation methods apply to the graph data model [Miller, 2013]. One is the item-to-item correlation that recommends products based on associations to other products (content based filtering) [Miller, 2013]. And the user-to-user recommender systems that make predictions based on correlations made according to how the user interacts with a system (collaborative filtering).

The paper [Aguilar et al., 2013] the authors reveal the relationship between participation of student's in an e-learning environment and the achieved results. The authors call their approach a visual analysis for measuring accesses from students to an e-learning environment. In their work [Aguilar et al., 2013] use a social graph representation of a visual analytic system that can display interactions in the online environment, they conclude that there is a pattern between behavior of the students according to participation and their regarding grades. The data for their study were collected through an online course on moodle. [Aguilar et al., 2013] explain that visualizing this data in a social network is not just about creating pictures, it is also about creating learning situations. Furthermore, [Aguilar et al., 2013] state that images of social networks provide new insights about a network structure and support to communicate insights to others. In their study [Aguilar et al., 2013] prove the existing correlation between quantitative varibles of the student participation. This extends the contribution of a visual analytic tool, that can be used with other existing metrics like centrality or degree of separation [Aguilar et al., 2013]. To answer the hypothesis of finding relationships between the student performance and the participation, [Aguilar et al., 2013] analyse three different cases:

1. Relationship among the frequency of access to fora and the frequency of access to reading resources with the grade of the student [Aguilar et al., 2013].

2. Relationship among the frequency of access to fora, the frequency of access to discussions and the grade of the student [Aguilar et al., 2013].

3. Relationship among the frequency of access to fora, frequency of access to reading

resources, frequency of access to reading discussions and the grades of the student [Aguilar et al., 2013].

[Aguilar et al., 2013] conclude that analysing those data is directly relevant to the student engagement and as well to evaluate the implemented learning activities. Also, they answered question types related to how and when students are engaged and what activities promote engagement [Aguilar et al., 2013].

[Beuster et al., 2013] introduce the development of a tool that monitors and analyzes learning processes. It aims to support teachers or researchers which analyze user data from online or blended-learning learning scenarios. The platform allows to evaluate data from different e-learning platforms like Moodle or Clix, for example user activities, the intensity of user activities over a period of time, the average use of the platform or to detect frequent visited paths between learning objects in a graph [Beuster et al., 2013]. [Beuster et al., 2013] explain to specify the requirements for the analysis tool, a survey with teachers and provider of e-learning content was conducted. This survey led to a catalogue of more then 80 different question types that are used as an entry point for the analysis process [Beuster et al., 2013]. [Beuster et al., 2013] show different question types, for example:

- At which point in time did students access the learning platform?
- How often did students access single learning materials?
- Are there any frequent visited paths?

Based on these question types different procedures for analysis and visualisation are used like a network presentation or a path visualization [Beuster et al., 2013].

3.4 Application of Graph Pattern Matching

[Myers et al., 2000] describe a framework which is used to compare corrupted relational graphs and search for matches. The paper builds on the original ideas of editing distances, which were originally described by [Sanfeliu and Fu, 1983]. [Myers et al., 2000] identify two main problems in graph-matching:

1. How to measure similarity.
2. How to search efficiently for the best match.

An approach describe [Sanfeliu and Fu, 1983]. An edit-distance that counts node and edge relabelings together with the number of node and edge deletions or insertions that are needed to transform one graph to another [Sanfeliu and Fu, 1983].

[Jouili et al., 2009] evaluate different graph matching measures which are used for document retrieval. The authors deal with the representation of images in an attributed

graph-based form. [Jouili et al., 2009] describe different scenarios in which graph-based representations are used. Some of them are for example the representation of circuit diagrams, shape recognition, image matching or old document analysis. However, different methods are often used to incorporate features of the document image. This is based on the varying characteristics of the data and the different representations [Jouili et al., 2009]. Thus [Jouili et al., 2009] state that in the field of structural pattern recognition a representation of documents by a graph data structure is useful. This structuring leads to the fact that graphs can be compared with each other, which is generally called graph-matching. Graph-matching is divided into two major categories if two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ exist. The first category contains exact graph-matching methods, which are also called graph isomorphism [Jouili et al., 2009]. These require a one-to-one mapping. The second category contains inexact matching methods. Here a strict correspondence between the nodes or edges of the two graphs cannot be determined [Jouili et al., 2009].

In the paper [Jouili et al., 2009] present different methods for pattern-matching in graphs, which are evaluated by the authors using different graph databases. Each of the four graph databases contains images of a certain category. The categories are images from the GREC database, an ornamental letters database, a shape database and a logo database [Jouili et al., 2009]. One method [Jouili et al., 2009] present for graph pattern matching, is graph edit distance from spectral seriation, which represents graphs as strings. The similarity of the graphs is the measured edit distance of the strings in a probabilistic framework. The graph edit distance in this case is the cost of the shortest edit path in an edit lattice for transofrming the data graph into the model [Jouili et al., 2009]. Problem with this method is computing the edit distance to find the least expensive path through the edit lattice based on the Levenshtein distance. Each state of the edit path is a cartesian pair [Jouili et al., 2009].

Another method of evaluation that [Jouili et al., 2009] present in their paper is graph matching based on node signatures. The point here is that each node is associated with a vector [Jouili et al., 2009]. The node signature is used to determine the cost matrix C. The results of the experiments leading to the evaluation of the different methods resulted in the following. The performance of the graph-matching measures depends on the databases [Jouili et al., 2009]. Graphs with a low edge and node density the string-based representation is not discriminant [Jouili et al., 2009].

The paper from [Huang and Shiu, 2012] is about an approach to extend e-learning to a user centric learning environment. In this paper a user-centric adaptive system is presented which uses sequential pattern mining to create adaptive learning paths based on users' collective intelligence and employs item response theory (IRT) to assess learners' skills in order to provide adapted learning materials. [Huang and Shiu, 2012] criticise the fact that most adaptive systems are not user-centric and content is provided by only a few experts, which means that learners themselves cannot learn efficiently. In order to select the content [Huang and Shiu, 2012] use the implementation of sequential pattern analysis.

This is normally used in the business sector and helps to analyse which items have been purchased after another item has been purchased. The output of the algorithm is all maximum sequences. Maximum sequences are frequent sequences and are not included in other sequences [Huang and Shiu, 2012].

[Gallagher, 2006] offers in his paper various insights into the research field of graph pattern matching, the problems involved and the possible solutions. The basic graph matching problem is to find matches in a graph according to a defined pattern. On the one hand there is always a *data graph* and a *pattern graph*, which sets the structural and semantic requirements for a subgraph of the *data graph* to fit into the pattern of the *pattern graph* [Gallagher, 2006]. The task is to find a set of subgraphs. The problem is to define precisely what is a match [Gallagher, 2006]. Formally spoken [Gallagher, 2006] explains that a matching is a form of pairs of bijections, which consists of nodes and edges. A semantic graph is described by [Gallagher, 2006] as a data-representation in which the nodes of the graph contain concepts (for example *actor* or *movie*) and the edges represent the relationships between the concepts (for example *appeared-in*). The nodes and edges of the graph have different attributes and the semantic graph always follows an associated ontology. [Gallagher, 2006] also presents in his paper some approaches in the field of semantic matching. OntoSeek is an information retrieval system that matches documents to queries and considers each of them as conceptual graphs and measures the semantic similarity of the graphs. In order to achieve a matching in this case an exact structural correspondence between the query and the subgraph of a document is required. The system proceeds with a heuristic model. Another model presented in [Gallagher, 2006] is TRAKS, where pattern matching is performed in typed, directed graphs. The matches are ranked according to their similarity to the original pattern. In which the ontological distance between the types is taken into account. The algorithm for the pattern matching, is performed in a kind of depth search [Gallagher, 2006].

[Robles-Kelly and Hancock, 2004] show in their paper how the inherent structure of adjacency matrices can be used for robust pattern matching. The nodes in a graph are converted to strings, the order of the nodes is the result of the random walk. String-matching operations are used to find matches. [Robles-Kelly and Hancock, 2004] explain that the steady state random walk is not sufficient for a string representation of the graph, because the result from the application is not a path linked by edges. To avoid this problem [Robles-Kelly and Hancock, 2004] use a constraint filtering technique to search for an edge connected path using the steady site probabilities. The steady state random walk uses the eigenvectors of the adjacency matrix. In the first step, adjacency matrices are created according to the existing graphs [Robles-Kelly and Hancock, 2004]. The edges of the graphs are arranged in a sequence and then a random walk is performed on the nodes. In order to be able to carry out the analyses on the matrices, they must be symmetrical according to [Robles-Kelly and Hancock, 2004]. Therefore a diagonal matrix must be calculated first.

4 Conceptual Design

This chapter covers the conceptual design of the solution approach to implement adaptivity in an e-learning environment such as the serious game LostEarth 2308, that is taken as example in the sections of this chapter. Apart from the conception of the learning scenario, this chapter provides an approach to build a data model which derives from the defined learning scenario. The data model contains Figure 4.1 illustrates those dependencies. This chapter therefore, provides the following sections:

- **Concept for Adaptivity in Serious Games:** This section gives an introduction on defining a learning scenario for the serious game LostEarth 2308. It focuses to answer the three questions when, what and how to adapt for a described learning scenario. A learning scenario describes a situation that occurs in LostEarth 2308 in which adaptivity can be used. This section describes a learning scenario about failing a mission in LostEarth 2308.

- **Concept for xAPI Statement Based Graph Data Model:** The data model is created based on the elaborated adaptivity scenario. It builds a graph structure that represents the learning activities in a sequence based on the xAPI statements in the LRS. The section 4.2 explains the concept of the data model and the results in the graph database Neo4j are in section 5.2. A graph based data model allows to visualize user profiles. The application of graph algorithms on the data models allows creating user models for adaptivity scenarios.

- **Concept for User Model and Analysis:** This section describes a concept how user models are created by applying algorithms on the graph data model. The example in this section explains a pattern matching algorithm to create a user model for visited and not visited nodes to build a sequence of recommendations for the player. Subsection 5.1.1 describes the implementation for LostEarth 2308. A user model is the representatition of a user profile according to specific criteria such as interactions on a platform or achieved results to gain knowledge about a user and to define assertions about a user.

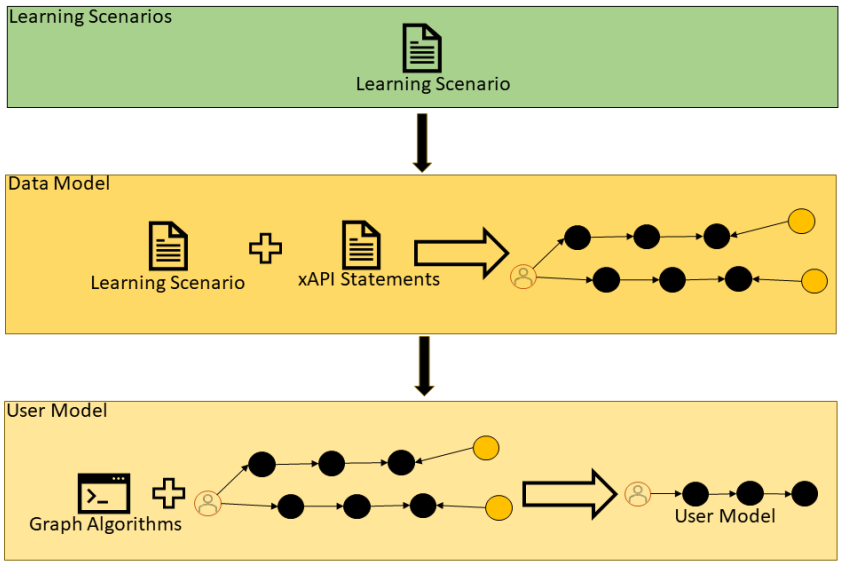

Figure 4.1: Conceptualisation Process

Figure 4.1 illustrates the interaction between the conception of learning scenarios, data modeling and the application of user modeling. This chapter explains this figure by breaking it into its components and describing the conceptualisation process. The general approach in this thesis is to define an adaptivity scenario in context of LostEarth 2308 and then build the graph data model based on the structure of the xAPI statements and the defined adaptivity scenario. The frame *Data Model* in figure 4.1 shows this process. The output contains an actor node, the activity nodes (black) and result nodes (orange). The frame *User Model* shows the application of graph algorithms on the graph user model to create the user model.

4.1 Concept for Adaptivity in LostEarth 2308

This section introduces a possible learning scenario for the game LostEarth 2308. This learning scenario sets the created xAPI statements in a useful context to further process those into the respective data model. The learning scenario allows the application of different graph analysis methods based on defined questions. Those questions serve to build user models according to answer the respective question. The user models are used for adaptivity inside the defined learning scenario. Section 4.3 describes the application of graph algorithms in a conceptual style. The definition of adaptivity includes the structure of *When to adapt?*, *What to adapt?*, *How to adapt?*. Figure 4.2 shows an example of possible activities a player can perform during a learning session. The grey nodes with the content *Act* represent activities a player can perform in a specif scene. For example in the right scene in the figure the player is able to interact with a character. The yellow star indexes when adaptivity is possible in LostEarth 2308, for example when the player selects

LISA or switches a scene. Those activities captures the xAPI-Tracker.

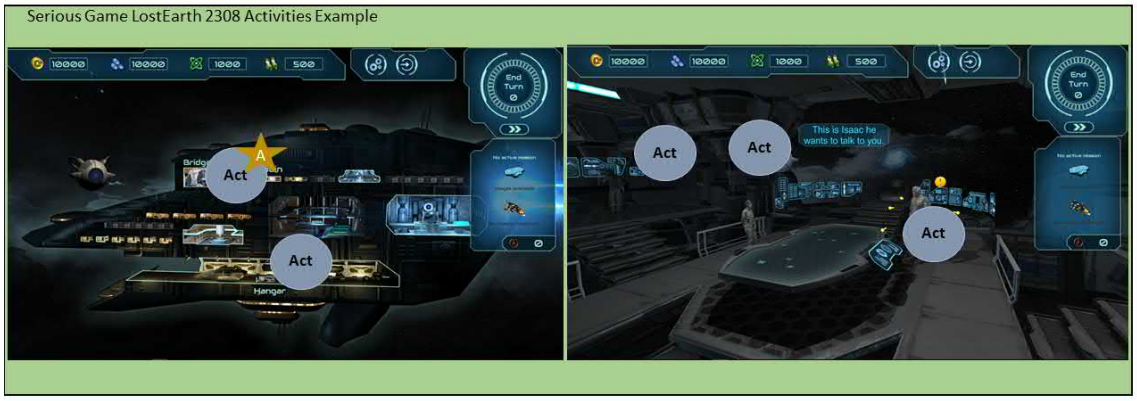

Figure 4.2: User Activities Example in LostEarth 2308

Learning scenario: Player fails a mission

In this learning scenario the learner plays a mission in LostEarth 2308. For each activity the player performs in LostEarth 2308 the graph database creates an activity node. In this scenario the learner fails the played mission. This scenario aims to adapt the activities of the user while playing the game. The concept of adaptivity introduces an approach that examines the activities of the users and, based on the results, makes recommendations for further activities. Further characteristics that need to be defined for this learning scenario are the three questions of *When to adapt?*,*What to adapt?* and *How to adapt?*.

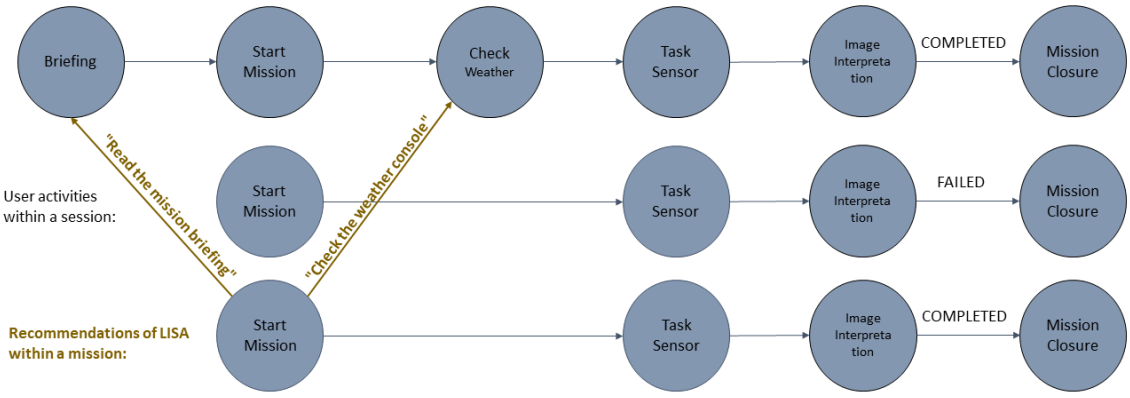

Figure 4.3: Adaptivity Scenario Activity Recommendation LostEarth 2308

Taking LostEarth2308 as an example, there are various activity nodes within a mission that a player can follow until the mission is completed. These include reading the mission description or checking sensor settings. Figure 4.3 shows these possible activities at the top. In the previously described scenario, the user has now played a mission and finished this with a failure. This activity sequence including the mission failure the middle node sequence shows, described in the picture as "User activities within a session". At the next start of a learning session, the player receives recommendations via an assistance system, which activities should be executed to get more information about e.g. the mission content.

The assistance system in LE2308 is called LISA. The recommendations of the system are shown in the graphic in the lower part, with the description, "Recommendations of LISA within a mission".

Hence the serious game LostEarth 2308 multiple players generate multiple graphs that grow together to a network of activities. Section 4.2 explains the according data model to this learning scenario. The final results of the data model describes section 5.2.

Table 4.1: When, What, How to Adapt?

Scenario	**When**	**What**	**How**
Player fails a mission	While learner is playing a session in LostEarth 2308 and failed the mission in the previous session	Personalized hints and recommendations about activities the player should perform in this session	Analyzes graphs to generate user models based on defined questions (further details in section 4.3).

4.2 Concept for xAPI Statement Based Graph Data Model

After the development of a concept for the usage of adaptivity, this section deals with the requirements of the data model to create the needed graph. In general, the following specifications apply to the graph. Since the adaptivity approach is based on events and the activities of the user must be available in a sequence, it is recommended to model the graph in such a way that the xAPI statements are arranged in a sequence that visually looks like a chain. The concept idea can be seen in picture 4.4. Note that the idea of this graph is to store the users' results from a session in a separate node, with its own label. The reason for this is to make a distinction between the final node and the corresponding result.

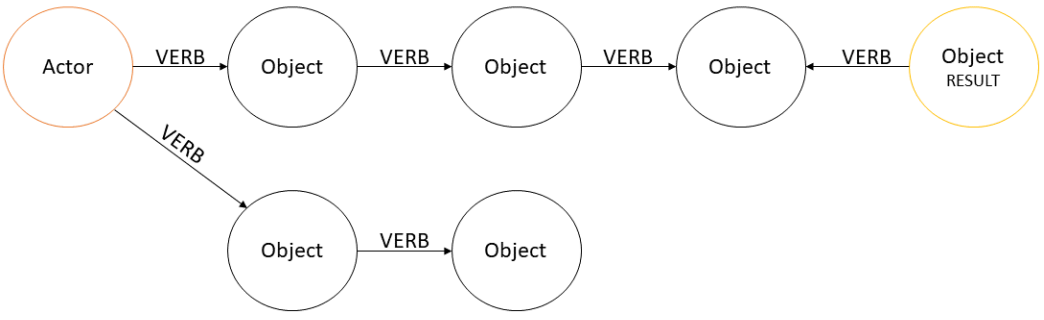

Figure 4.4: Overview Concept Session Datamodel with Result Node

```
{
    "actor": {
        "objectType": "Agent",
        "name": "Example@mail.com",
        "mbox": "mailto:Example@mail.com"
    },
    "verb": {
        "id": "http://adlnet.gov/expapi/verbs/failed",
        "display": {
            "en-US": "failed"
        }
    },
    "object": {
        "objectType": "Activity",
        "id": "http://www.iosb.fraunhofer.de/schema/xapi/
lostearth/missionfailure",
        "definition": {
            "type": "http://curatr3.com/define/type/level",
            "name": {
                "en-US": "Mission Campus"
            }
        }
    },
    "result": {
        "completion": true,
        "success": false
    }
}
```

Listing 4.1: xAPI-Statement Data Model from Lost Earth 2308 with Result

The xAPI statement in listing 4.1 contains the data that is needed to create the graph data model in a sequence of activities for each individual user. The property *result* extends the data model, which defines the user's rating for a single session. If a user ends a session early, this can have an impact on the learning scenario because no evaluation of the session result is available. Another property for this data model is the *actor*, which represents the user. In the xAPI statement, this actor has a name and a unique assignment via a mail address. The next property represents the *object* and is a description of the activity the user performs in Lost Earth 2308. The activity is identified by a unique ID, which is defined as a URI. In addition, each *object* property has a definition of the activity, which includes a name and a type. However, the type of the activity is not uniquely defined and

can also be used in other xAPI statements. The different types[1] define the activity in order to what the activity should represent. The relationship between *actor* and *object* property in the graph is the *verb*. It contains a name and a unique ID as URI. The structure of the data model within a Cypher query and the result of the data model explains section 5.2. These properties of the xAPI statement are used to create the data model in the graph database Neo4j. At this point, however, the data model cannot show which activities were carried out commonly by several users. It isolates the different sequence data models between the different users. As section 4.1 explains, multiple users play LostEarth 2308 and each of them creates an individual sequence data model, but this approach of the data model does not provide insight into the frequency of visited activities. For this purpose, the data model from figure 4.4 is extended with a further *object* property from the xAPI statement. This *object* property links each individual actor with this object node, if a user performed this activity. This brings the advantage that the learners can be linked to each other via common activity nodes and thus they and their individual activities in the chain data models are no longer loosely available in the network. The concept for the complete network, consisting of the different sessions of all users, can be seen in figure 4.5. This also allows finding activities for the user as a recommendation with regard to the learning scenario for LostEarth 2308.

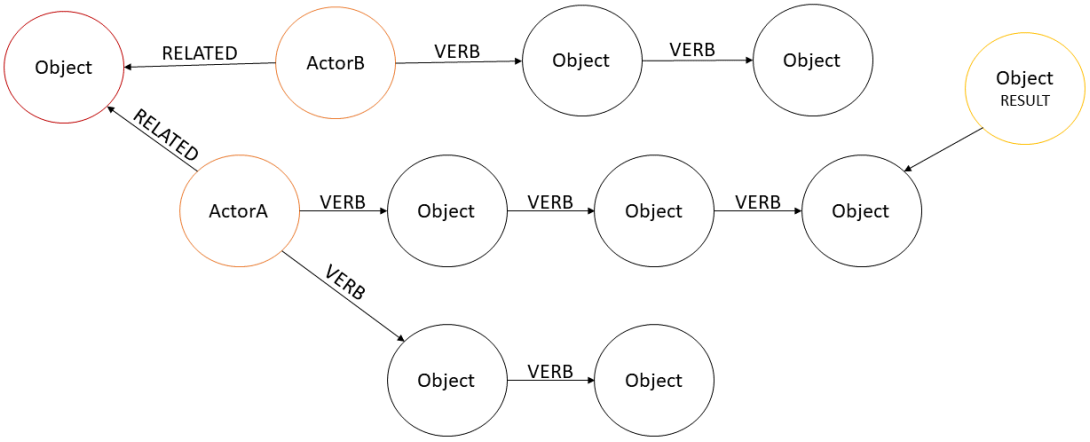

Figure 4.5: Extended Concept for Graph Data Model

4.3 Concept for User Models and Analysis

The data model gets created through the tracked player activities in LostEarth 2308 and the structure that figure 4.5 shows, based on the learning scenario that section 4.1 describes. This section shows how the user models for adaptivity are conceptually created by applying graph algorithms to answer the question *How to adapt?*. Implementation of the algorithms subsection 5.1.1 describes. Results section 5.2 and 5.3 explain.

[1] https://registry.tincanapi.com/: Collection of different activity types with description

User model creation based on possible questions for analysis

According to the defined learning scenario in section 4.1, a user model is needed that describes different user interactions. Those are related to the given learning scenario, which intends to show recommendations (*What to adapt?*). For the question *How to adapt?* the user models are needed, because they show how the user acts in LostEarth 2308. The question *How to adapt?* this chapter explains.

First of all, by looking at the learning scenario, different questions can be applied that support the creation of a user model with graph algorithms. Possible questions for analysis related to the learning scenario are:

- Which path did the player follow in the session?
- How many times did the player start a session?
- Are there frequent visited activities?
- What is the score of the user compared to other players?
- Which activities did a player perform that was successful in a mission?
- What is the last activity a player performed?

As section 2.3 describes, the data in the graph database can be manipulated via queries. To do so, operations need to be implemented that allow to gain information from the graph. These operations are localized in the network, because the query always has a starting point.

With the analysis of the question types on the data model from section 4.2 the user model is composed. More information about the generated graph data model in the graph database section 5.2 provides. As a derivation for the creation of the user models lets take a look back at the conceptual data model from figure 4.5 which shows a segment of the whole network for LostEarth 2308.

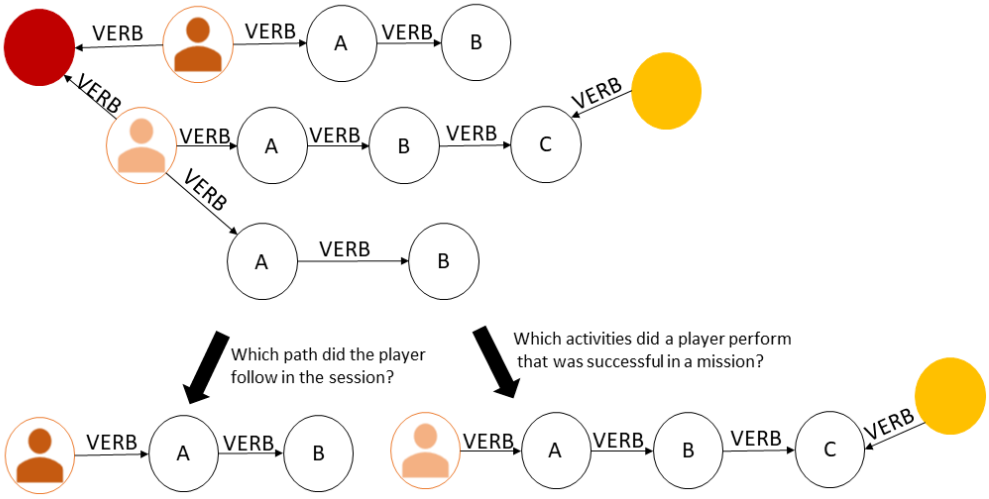

Figure 4.6: User Model Conceptual Design

Figure 4.6 shows how the user model in a graph is accomplished based on the question types for the learning scenario and the data model. The node with the person inside represents the actor (player), the node with a letter represents an activity (object) the player performed, the dark red node is a single activity several players performed and the yellow node visualizes the result of a session. The questions that this figure visualizes are *"Which path did the player follow in the session?"* and *"Which activities did a player perform that was succesful in a mission?"*. To generate the user model, behind each of those question types is an algorithm. These algorithms need to contain the properties of the nodes to traverse the graph according to the intended result of the query. For example the query for the question *"Which path did the player follow in the session?"* contains the properties of the player name and a timestamp for the activity. Futher information about the used properties are listed in section 5.1. To find the path of the session, the algorithm needs to be a pathfinding algorithm like Yen's-k shortest paths algorithm. This algorithm and further methods section 2.4 shows. Those user models already allow the selection of results that can be presented to the player according to his activities. The different user models also allow analysis approaches between each other, for example to find specific patterns in different user models. This section also introduces a concept to use the results from an applied graph algorithm for a pattern matching procedure to generate activity recommendations.

Applying pattern matching on user models

Figure 4.7 shows such a activity model from a user that gets generated after applying a shortest path algorithm on the data model. The shown sequence of activities belongs to a specifc player and is a part of the pattern matching approach to serve as *pattern graph*. The user model for the *pattern graph* is a sequence of activities with that has passed the mission. It is important to note that the pattern matching does not aim towards the structure of the graph, but at the semantics in the nodes. To make this possible, a string-matching

procedure is needed. It is important that the pattern matching concept can be applied to the described learning scenario from section 4.1.

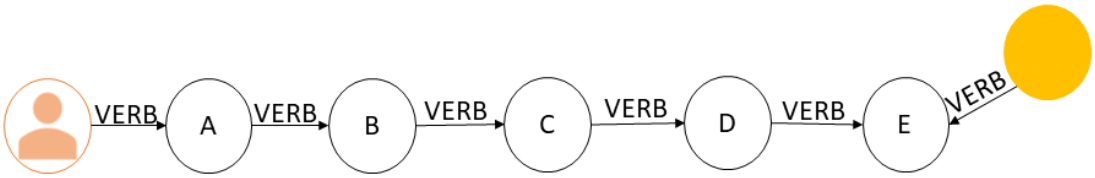

Figure 4.7: Pattern Matching Concept

The first approach for graph pattern matching was to compare graphs based on their structure. However, findings from this approach showed that it is not practical to analyze the user graphs from LostEarth 2308 for their structure, because the structure of the graph follows the sequential data model. More important, however, is to analyze the various nodes in the graphs for their semantics, which are the individual names of the activities according to the defined data model. To explain this concept, figure 4.7 is presented, which shows the different possible activities a user can do during a session. The activities (node names) are identified by the letters $V = "A","B","C","D","E"$. In the example of LostEarth 2308, these node names would be the activities from the xAPI statements. The user model for a sequence of a session is described in a chain in which each performed activity follows another activity until the session is closed. Those chains are calculated from the data model with a suitable graph algorithm like Yen's-k shortest paths or a single source shortest path. The flow graph itself has no multiple connected edges but has a source (starting point) which is the name of the actor and a sink (ending point) which is the last activity from the player in this specific session. This graph is a *pattern graph*. According to the conceptualised learning scenario, the pattern graph needs to fulfill the criteria to be a session with a successful mission enclosure.

It is important to note that using the example of LostEarth 2308, no activities are possible that make it impossible to complete a mission. Only the mission result changes to failed or passed. To get enough information about the current mission the player can read certain hints, but is not forced to do so.

Similar conditions apply to the user model of the current player. It is possible to either use one, such as the user model from the last played session or up to all graphs of the user for the pattern matching analysis. This graph to be analysed is called *data graph* and contains the names of the activities in the nodes just like the *pattern graph*. Now the question arises how the individual graphs of the *possible interaction sequence* and the *data graph* can be queried. The answer to this question is to create a query that requests the corresponding data from Neo4j. The best strategy is to use a query that executes a path finding algorithm, because this can also help to retrieve the amount of taken steps of the session. More details about this implementation step can be found in section 5.1.1. A further question is how to find matches between the graphs and how

to use these matches for a recommendation according to the learning scenario in section 4.1. The chosen approach to this solution is that the subgraph of the *data graph* from the *pattern graph* must be found. In order to be able to draw conclusions about which activities were not carried out in the subgraph and to send the recommendation for single or several activities back to the user. To make this possible, the idea in this concept is to create matrices as a representation of the individual graphs and then multiply them among themselves. Reference for this approach describes section 2.4. The concept for the solution shows figure 4.8. Information about the implementation process of the *Pattern Matching* which is visualized as an arrow in that figure, is provided by subsection 5.1.1.

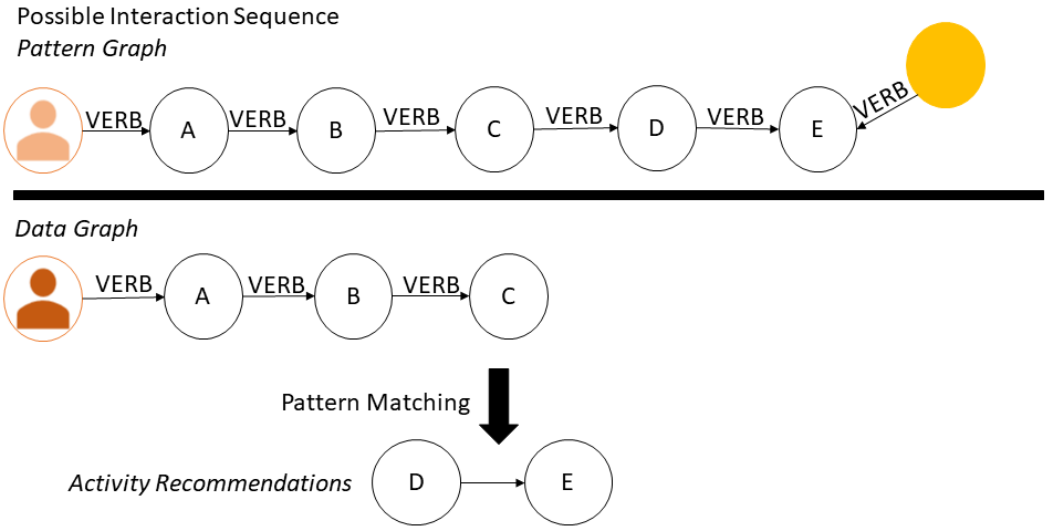

Figure 4.8: Pattern Matching Procedure Example for Activity Recommendation *(Based on concept from Alexander Streicher)*

5 System Architecture Introduction

This chapter provides an overview of the entire system architecture. The individual components that are required are described in this chapter. The graph analysis service gets a more detailed description in section 5.1 and the graph database Neo4j in section 5.2. Section 5.3 shows the adaptivity results that are displayed to the user in LostEarth 2308. Figure 5.1 shows the architecture of the system with its different components.

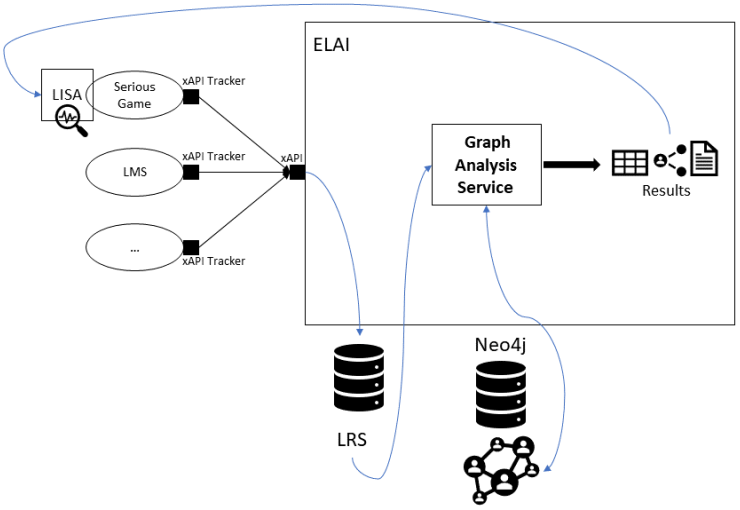

Figure 5.1: Overview Concept System Architecture

Serious Game

The serious game is an external system docked to the architecture. By playing this serious game, the user interacts with different game mechanics and plays different missions for learning purposes. In this system, the results of the adaptivity analysis process are displayed to the player. These results can be presented in textual form. In this thesis the serious game is represented by LostEarth 2308.

xAPI-Tracker

The functionality of the xAPI tracker has already been described in section 2.1 and is used to track the individual activities of the user and to generate data in the form of xAPI statements.

LRS

The LRS stores the xAPI statements which the xAPI-Tracker generates. Individual xAPI-Statements are requested via an xAPI-Endpoint. Further details about the endpoint and the functionality are in in the section 5.1.

ELAI

ELAI forms the main framework into which the implementation for graph analysis is integrated as a separate service. Details are mentioned in section 2.2. The ELAI provides several endpoints that are addressed to transfer adaptivity requests and responses between the serious game LostEarth 2308 and the Graph Analysis Service.

Graph Analysis Service (GAS)

The GAS is implemented as a service and contains procedures to communicate with the graph database, execute algorithms to analyse the graph and to send results back to the ELAI. Further details about the architecture and functionality of the GAS are explained in section 5.1. The algorithm implementation as example for the purpose of graphs and graph algorithms subsection 5.1.1 describes.

Neo4j

Graphs can be created and analysed using the Neo4j graph database. The GAS sends the corresponding requests to Neo4j, while Neo4j sends the corresponding results about the analysed graphs back to the GAS. The architecture and functionality of Neo4j is explained in section 5.2.

The communication between the individual components is a particular challenge and is shown in this picture indicated by arrows. The important individual (communication) steps are shown in the sequence diagram 5.2.

The sequence diagram in 5.2 shows the communication between the game LostEarth 2308 and the GAS. Important to notice is the direct transfer from tracked xAPI statements into Neo4j. This process allows a flexible and dynamic development of the learning environment based on the data that gets created by the xAPI tracker. Therefore, Neo4j creates a network representation of this learning environment that allows user data analyis with graph algorithms depending on the learning scenario and which information is needed. Responsible for the communication of adaptivity requests between LostEarth 2308 and the GAS is the ELAI. The ELAI proceeds the adaptivity request to the GAS. The GAS executes a query that contains a graph algorithm to build the needed user model. Neo4j returns the user models and the GAS returns the adaptivity response back to the ELAI. The ELAI procceeds the results back into LostEarth 2308 where they are displayed to the user.

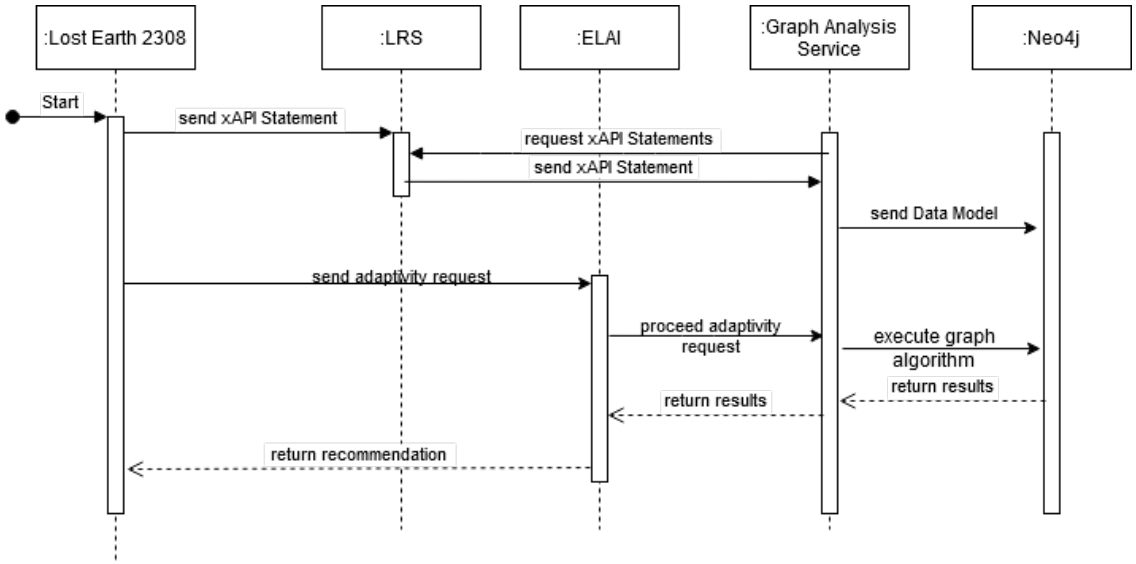

Figure 5.2: Sequence Diagram System Architecture

5.1 Component: Graph Analysis Service

The service is a web application developed in C♯ in ASP.Net Core and the .NET Core Framework[1]. Important for the choice of the environment is that the system is versatile in functionality regarding the development interfaces, because in this thesis different systems need to communicate with each other.

As previously described in chapter 4, the solution approach is used to prepare the xAPI statements from LostEarth 2308 to arrange them in a certain data model in order to analyse the network to build different user models. The first question that occurs is at which point the implementation of the data model and the algorithms should take place. As mentioned in chapter 5 the whole system contains different components. One of these components is the Graph Analysis Service (GAS), which is described in this section. The GAS contains all interfaces to communicate with the graph database and with the ELAI to send results to a docked serious game after analysing graphs. Each component of the Graph Analysis Service figure 5.3 shows.

The procedure in GAS provides for the following procedures.

- Creation of graphs based on the defined data model (section 4.2).
- Application of graph analysis methods, requested by the ELAI (subsection 5.1.1).

Graph Creation Based On The Data Model

The creation of the user models proceeds as follows. First the xAPI statements of the player are transferred to the corresponding LRS client via the xAPI tracker docked to LostEarth 2308. At the same time the xAPI Tracker sends a *HTTP POST* request to the GAS to

[1].NET Core Version Number 3.0

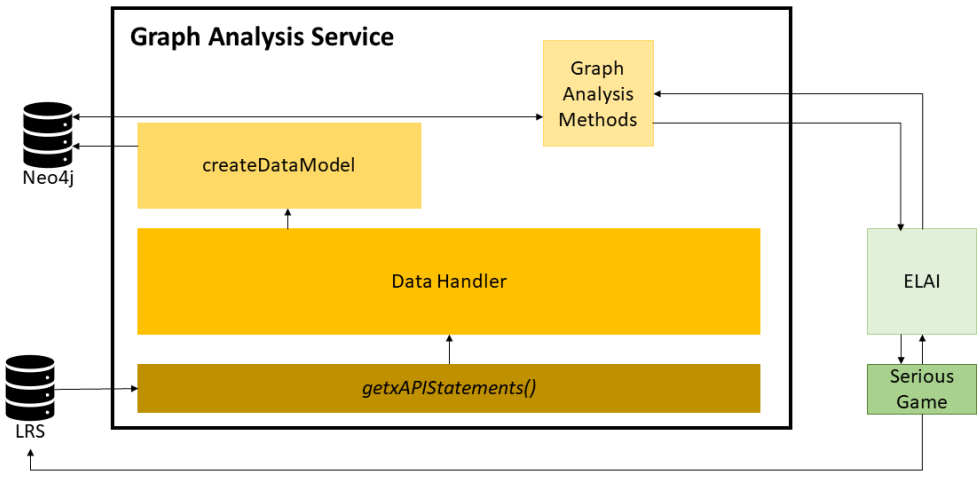

Figure 5.3: Overview Processing System Architecture for Graph Analysis Application

process the last statement stored in the LRS. The GAS starts the method *getxAPIStatements()*, which queries the corresponding statement from the LRS using the existing ID of the xAPI statement and saves it as *JSON Object* inside the service. A detailed list of the requests can be viewed in 5.1. The received data, which the service stores as *JSON Object*, must now be prepared for the creation of the data model. *DataHandler* does this. Here the individual statements are stored in their single components, that are *actor*, *verb*, *object* and *result*. The individual parts of the xAPI statement are stored in lists within the service. This also ends the procedure of the *Data Handler*. In order to load the corresponding statement into Neo4j according to the data model, the method *createDataModel* is started, which processes the individual entries from the previously created lists in a single Cypher query and stores them in Neo4j. A sample query can be viewed in listing 5.1.In this listing a node is created with the label *LEPlayer* and the properties *name* and *id*. This node is only created once in Neo4j. This process of posting individual statements in Neo4j from the GAS happens parallel to the game. This is a significant time improvement compared to posting all xAPI statements in the LRS to the Neo4j graph database.

```
MERGE (n:LEPlayer{name:$p0, id:$p1})
SET (n.name = $p0)
WITH PARAMS(id = $p1, name = $p0)
```

Listing 5.1: Query Example in GAS for Node Creation

After creating only single nodes of actors and activities, the shown Cypher query from listing 5.2 shows the creation of relationships between two activities and the corresponding verb as edge. For this purpose, two nodes with the label *LEActivity* are taken from Neo4j

and have matches in their properties with the parameters from the previously created lists (*name* and *timestamp*). The creation of relationships between actors and activities follows the same schema except that one activity node is replaced by an actor node.

```
MATCH (f:LEActivity), (g:LEActivity)
WHERE (f.name = $p0)
AND (g.name = $p1)
AND (f.timestamp = $p2)
AND (g.timestamp = $p3)
CREATE (f)-[r:SELECTED]->(g)
```

Listing 5.2: Query Example in GAS for Relationship Creation

The actor properties are:

- ID: the e-mail address of the player
- Name

The activity properties are:

- Activity ID
- Activity name
- Actorname
- Timestamp

The verb properties are:

- Verb ID
- Verb name
- Timestamp

Neo4j results of those data model queries are in section 5.2.

Table 5.1: Needed HTTP REST Endpoints In GAS

Request	**Result**
GET xAPIStatement	Returns the last xAPI statement in the LRS
POST xAPIStatement	New node in Neo4j based on the used data model
GET path recommendation	returns result from pattern matching

5.1.1 Implementation: Graph Analysis Methods

This subsection describes the implementation steps to answer the questions to build user models for adaptivity based on the learning scenario. And it explains the implementation example for the pattern matching algorithm that shows how user models are further processed. Chapter 4 anf its sections show the conceptual procedure from a learning scenario up to suitable user models.

- Building user models for adaptivity based on questions for analysis.
- Algorithm to detect patterns for recommendations based on user model.

In the component *Graph Analysis Methods* (figure 5.3) , the different graph algorithms to build the user models, for example Yen's-k shortest paths, or the graph pattern matching algorithm are implemented. The component *Graph Analysis Methods* couples with the Neo4j database to execute adaptivity requests from the ELAI. Also, it communicates with the ELAI to send back results from the graph database into the serious game visible for the player. This results shows section 5.3. The following parts of this subsection present the workflow inside the *Graph Analysis Methods* to generate the user models and further use them in example for pattern matching.

Building user models for adaptivity based on the questions for analysis: An approach for answering an analysis questions like *Are there frequent visited activities from the players?* is to cluster the graph for specific properties in the nodes to find the frequent visited activities. The query in listing 5.3 shows the approach to cluster the players according to the activities. In this query, the player are clustered in the activity *BriefingCampus*. The query returns a table in Neo4j of all players that visited those activities.

```
MATCH (activity)-[:IN_RELATION]-(player)
WHERE activity.name = 'BriefingCampus'
RETURN player.name
```

Listing 5.3: Query Example for Clustering

In this part of the subsection a user model is created based on a question by showing a suitable implementation. Results of this presents section 5.2. The shortest path algorithm shown in listing 5.4 traverses the data model in Neo4j to create user model for the session according to the visitied nodes and the achieved result. The output of the query depends on the current player. The returned nodes have different properties which are the names and IDs of the nodes, the achieved result and the timestamps of the different nodes. Those nodes are ordered by the timestamp and only one path gets returned, which is the last path a player took to finish a mission.

```
.Match("(first:LEPlayer {name:'" + $"{name}" +
    "'}),(last:LEActivity{name: 'MissionCampus'}),(g:LEResult)")
.Where("last.timestamp = g.timestamp")
```

```
.AndWhere("first.name = last.actorname")
.Call("algo.kShortestPaths.stream(first,last,1, 'cost',{})")
.Yield("index, nodeIds, path, costs")
.Return(() => new {
places = Return.As<List<string>>("[node in
    algo.getNodesById(nodeIds) | node.name]"),
ids = Return.As<List<string>>("[node in algo.getNodesById(nodeIds)
    | node.verbid]"),
score = Return.As<double>("g.raw"),
timestamp = Return.As<string>("last.timestamp")})
.OrderByDescending("last.timestamp")
.Limit(1);
yensKQuery.ExecuteWithoutResults();
```

Listing 5.4: Cypher Query in C♯ to apply Yen's k-shortest paths

The algorithm in listing 5.4 is used to answer the analysis questions:

- Which path did the player follow in the session?
- Which activities did a player perform that was successful in a mission?

Algorithm to detect patterns for recommendations based on user model: This part of the subchapter presents a method which allows to compare the different sessions of one player (G_{data}) with one or more sessions of another player ($G_{pattern}$). Based on this, this leads to suggestions as to which activities a student is able to perform in his game session in order to potentially obtain a better result in the assessment. The algorithm itself is an extension of the procedure presented in the concept 4.3. In the general case, the algorithm therefore examines the user's individual nodes and creates a sequence of interactions that the user has or has not performed, compared to the sessions of other users.

In this context, we have the following conditions. The graphs present in the database are all directed and are in their form a flow graph $G(V, E)$ without multiple edges, with a source $Q = playername$ and a sink $S = lastvisitednode$. These conditions were ensured during the conception of the data model in section 4.2 and apply on the user model after traversing the data model with a shortest path algorithm.

The first operation that is performed for the implementation of *Graph Analysis Methods* is the request to Neo4j for specified user sequences. The player issues data whose behaviour is to be analysed. The data is loaded into the service via the procedure, mapped in algorithm 2 and stored there as a *Graph-Element*.

Algorithm 2: Data Retrieval From Neo4j To GAS

Input: Neo4j Query results

foreach *item in* $G_{pattern}$ **do**
  Add Graph-Element

foreach *item in* G_{data} **do**
  Add Graph-Element

Algorithm 3: Creating Data Structures

Input: $G_{patternGraphElements}$

Input: $G_{dataGraphElements}$

foreach *GraphElement in* $G_{patternGraphElements}$ **do**
  if $GraphElement = E$ **then**
    Add E
  else
    Add V

foreach *GraphElement in* $G_{dataGraphElements}$ **do**
  if $GraphElement = E$ **then**
    Add E
  else
    Add V

In the following step, the obtained graphs have to be transformed into data structures for further use inside the GAS, which represent the retrieved graphs. For this principle, a graph G contains a collection of nodes V and edges E. This arrangement is performed as shown in algorithm 3.

After mapping the graphs into data structures, the graphs are matched using a string-matching procedure to find out if they are similar. This creates a subgraph as a matrix from the graphs of $G_{patternGraphElements}$ and $G_{dataGraphElements}$. The procedure is explained in Algorithm 4. The obtained matrices are now multiplied with each other. This process takes place recursively until all matrices have been multiplied. The product of this multiplication is the smallest possible subgraph of a user and represents the nodes that a user has always visited in each session. It is noticeable that the subgraph becomes smaller, when a lot of different graphs with varying activities are analysed. The subgraph contains more nodes if only single sessions were played among each other or if the user had many similar activities within several sessions.

Algorithm 4: String-Matching To Calculate Activity Matrices

Input: $G_{patternGraphElements}$

Input: $G_{dataGraphElements}$

for $i \leftarrow 0 < G_{patternGraphElements} \leftarrow len$ **do**

  for $j \leftarrow 0 < G_{dataGraphElements} \leftarrow len$ **do**

    if $G_{patternGraphElements}[i] = G_{dataGraphElements}[j]$ **then**

      $ActivityMatrix_n[i,j] = 1$

    else

      $ActivityMatrix_n[i,j] = 0$

Output: $ActivityMatrix_n$

The advantage of creating the matrices by using a string-matching operation on the individual nodes and using adjacency matrices only for the pattern graph is that the information is not lost where the nodes have matched and are therefore used for recommendations based on the activities. The multiplication of the matrices is possible because each is an NxN matrix, based on the size of the pattern graph ($G_{patternGraphElement}$).

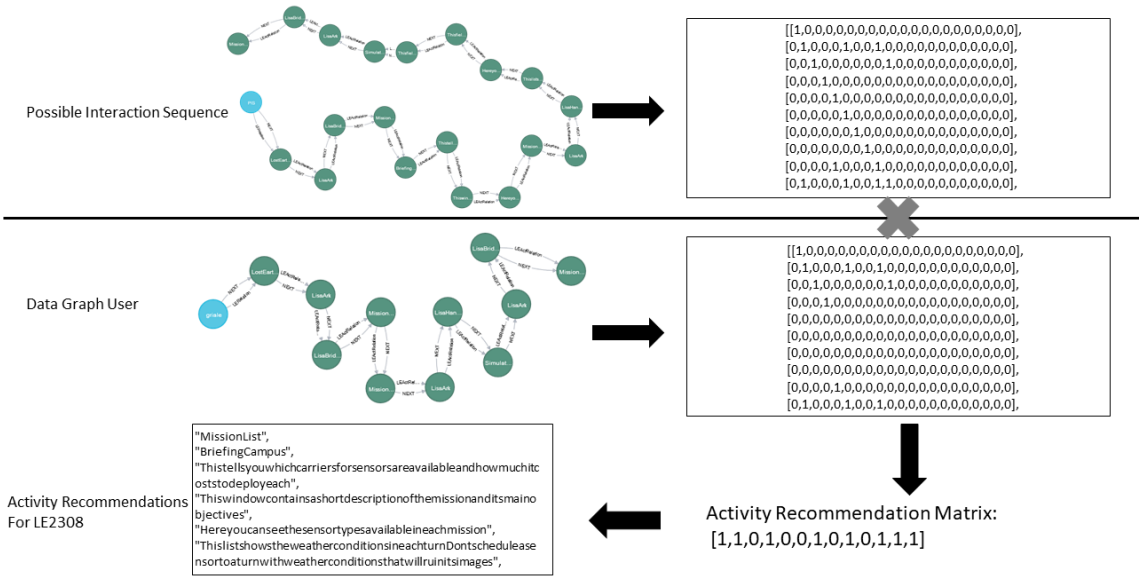

Figure 5.4: Overview Pattern Matching Algorithm Implementation

In figure 5.4 you can see an overview, which visualizes how the pattern matching algorithm works and how the output looks like at the end - namely with activity recommendations for the player. The graphs on the left side are the user models based on the analysis question in the learning scenario and represent a session from the player. In the first step, the *possible interaction sequence* gets transfered into a NxN diagonal matrix. Also, for the *data graph* of the user a matrix with the size of the possible interaction sequence is created. The multiplication of graph matrices has the advantage that this system is robustly built and also allows to model and compare the different user model and thus to

get a picture of the user's activities on the platform. The result of this implementation is a set of recommendations the current player can use as a sort of checklist or guideline in his session. The *activity recommendations for LE2308* are the activities the player in the *data graph* did not perform unlike to the player in the *pattern graph*. Based on the results from the *pattern graph* the player from the *data graph* has the opportunity to achieve a succesful mission enclosure.

5.2 Component: Neo4j as Graph Database

Neo4j is used to store and visualize the defined data model from section 4.2 and based on the query implementation that are shown in section 5.1. Furthermore, it allows to perform user modeling approaches with the query language Cypher that subsection 5.1.1 further explained. This section serves to display the results that are related to the graph database Neo4j. Neo4j serves as a virtualization tool and to deal with the request from the GAS.

Figure 5.5 shows the general procedure. The input comes from the GAS, which is either an adaptivity request to Neo4j or a request to set an activtity or actor node and the corresponding relationships regarding to the data model. Depending on the request, the output in Neo4j represents a result of the adaptivity request that gets returned back into the game or a new node for the data model, which is a representation of learning activities.

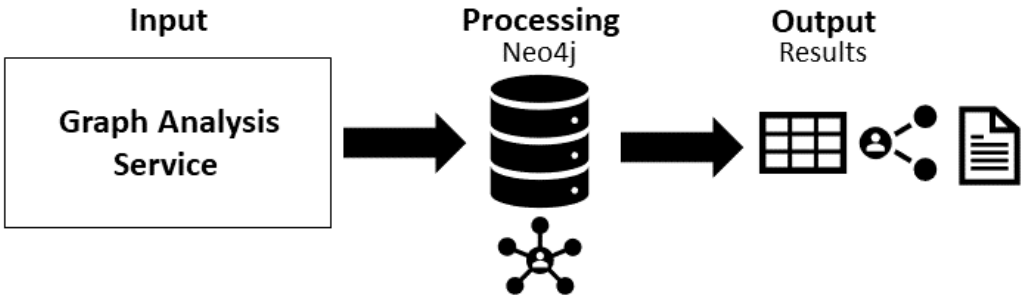

Figure 5.5: Overview Processing System Architecture Neo4j

Beside the visual aspect of the graph data model, that allows an insight on xAPI statement based activity network, like exemplified in figure 5.6. An other advantage is, that in this approach the data model gets generated dynamically and the system can grow based on the incoming data. Even without applying any graph algorithm for generating user models, it is possible to see a general overview of activities in the data model as seen in figure 5.6. This data model gets generated through the shown queries in listing 5.1 and 5.2. The gray node represents the actor node, which is unique in the network. The green nodes represent the performed activities and the yellow node is the achieved result in the session. This figure shows that one of player graphs is larger than the others. But by just looking at the data model, useful statements for adaptivity are hardly to define, because the data model just provides an overview about the whole environment.

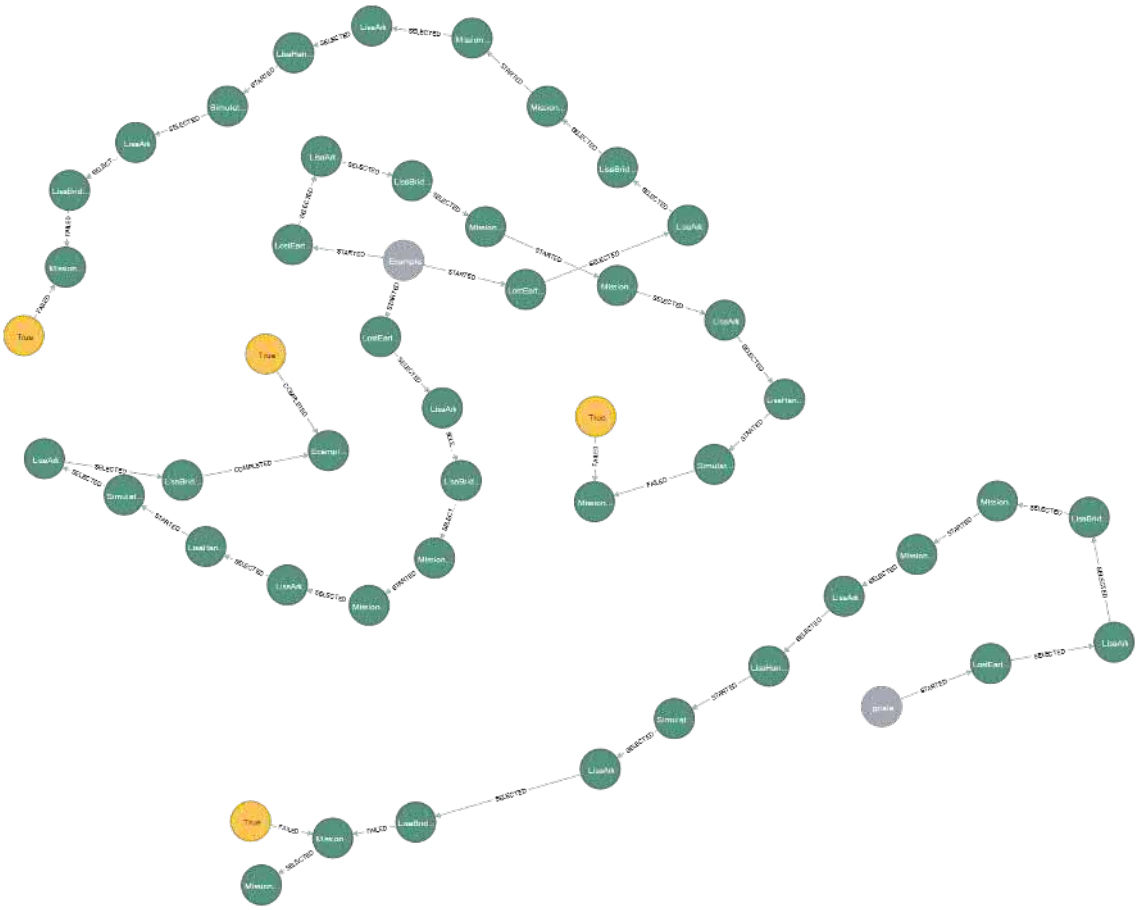

Figure 5.6: Graph Output Sequential Data Model with Results

To achieve this, the questions for analysis with the corresponding graph algorithms are needed to create user models for the learning scenario. These allow to traverse the graph in an efficient manner to generate the user models for adaptivity.

As previously figure 4.5 showed it is possible to extend the sequential data model with a further *object* node that allows relating the different players with each other. The *object* node represents a single activity which a player is related to, if he visited this *object* during a session in LostEarth 2308. This extra object node allows performing further analysis questions for user modeling on the data model, like centrality measurments or applying a page rank algorithm. The result of this extended data model is visualized in figure 5.7. This data model contains the sequential graphs as figure 5.6 shows. The actor is represented as the blue node in figure 5.7 and the grey nodes are the added object-nodes. Each chain still represents a single session, whilst the green nodes are the performed activities and the yellow node is the result within the session. All graphs in Neo4j are directed.

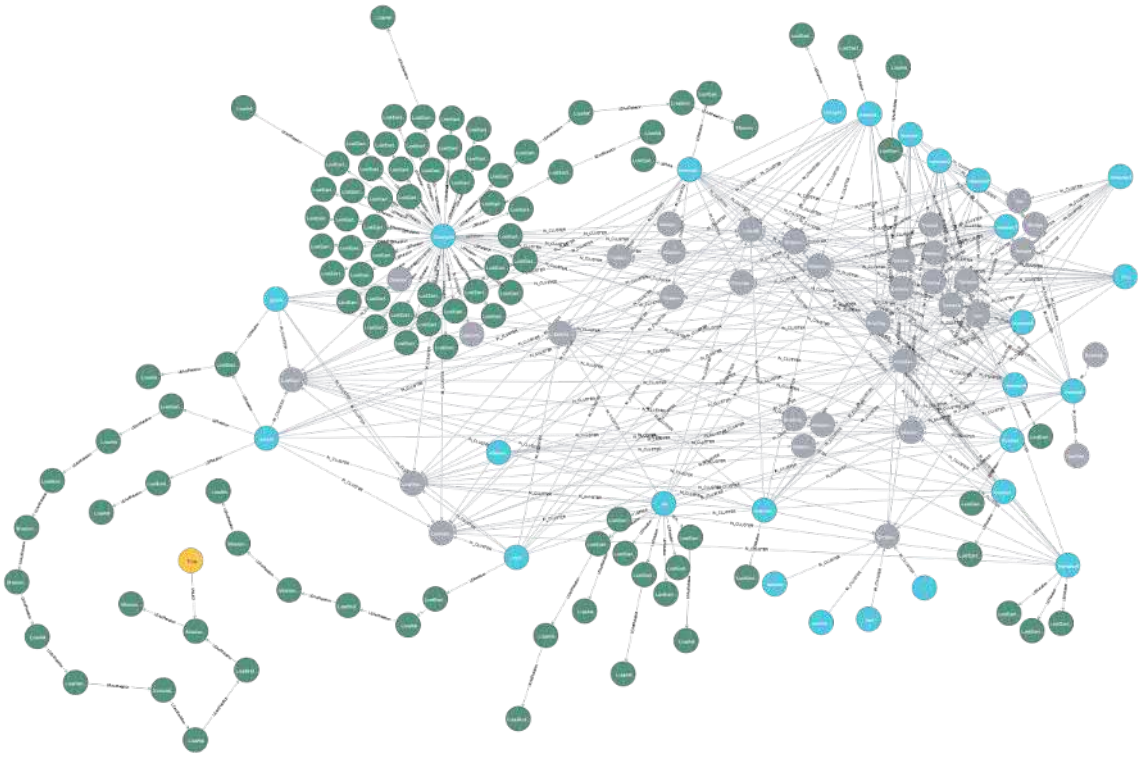

Figure 5.7: Graph Output Sequential Data Model with Added Object-Node

Figure 4.5 represents all activities and according relationships for each user in LostEarth 2308 and is a visualized representation of the interactions with the game, because this data model is based on the tracked xAPI statements. This data model allows the application of graph algorithms that got presented in subsection 5.1.1 for user modeling.

Figure 5.8 shows a possible user model of a player after exectung the shortest path algorithm that got described in listing 5.4. The algorithm traverses the data model that is shown in figure 5.7 with the given parameters in the query and returns this subgraph in Neo4j. This shortest path is also used for the pattern matching algorithm for activity recommendation.

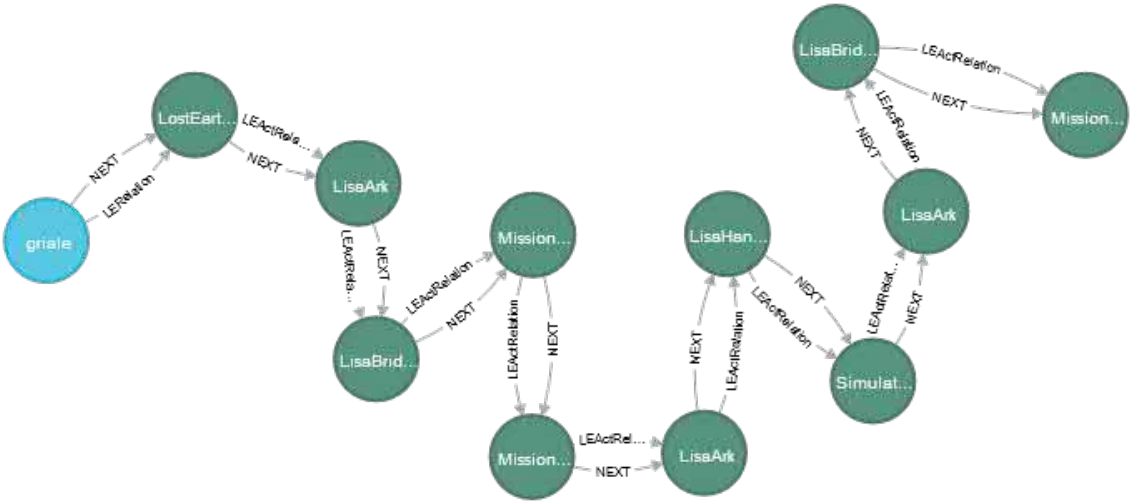

Figure 5.8: Overview Pattern Matching Algorithm Implementation

Table 5.2: Frequent Visited Nodes by Players on Example *"BriefingCampus"* Extract

Player Name
"newuser20"
"PIS"
"Example"
"newuser4StartMode"
...

The table 5.2 shows an extract which players visited the node *"BriefingCampus"* according to the query approach from listing 5.3. In total 15 players went to this node. Other nodes like the *"MissionCampus"* node was visited 44 times.

5.3 Component: LISA Recommendations in LostEarth 2308

This section describes the results of the user modeling and pattern matching process that is sent back to the serious game LostEarth 2308. Different results are possible. Those depend on the general progress of the player, which means in first place to return the message to the player that shows figure 5.9, in case the player started LostEarth 2308 for the first time or did not finish a mission before. So, this message occurs in LostEarth 2308, if the pattern matching algorithm could not find a path for the current player until the excpected end node.

Figure 5.9: Result Recommended Activities for the Current Player

Activity recommendations are possible, if the player already achieved some progress. So, figure 5.10 shows the result of the pattern matching process, in which LISA recommends the player different activities that can be performed in this session. This result is based on the taken path from a player with a positive mission outcome and the last taken path from the current player, if the result of the mission was a failure. By following this analysis approach the system is able to regulate itself based on the content of graph data model that gets updated frequently as long the players proceed in playing which allows to analyse always which paths lead to good or better results. *"BriefingCampus"* is a recommendation based on the pattern matching process and represents a single activity the user can perform in this session.

Figure 5.10: Result Recommended Activities for the Current Player

In case, the player executes those recommended activities and encloses the mission, this activity won't be recommended in the next session. So, this system allows a steady

adjustment of recommendations for the player based on the activities in the sessions.

6 Verification

This chapter summarizes the contributions of this thesis. The chapter is divided into the following sections:

- **Scenario:** This section explains the results to the reader of the thesis, using an example according to the defined learning scenario in section 4.1.
- **Discussion:** This subchapter is used to compare and discuss the achieved results with the target state previously mentioned in section 1.2.

6.1 Scenario

This section shows the results of the thesis from a players perspective, which means it explains what the player sees in LostEarth 2308. Furthermore, this section refers to the according implementation steps to show what happens in the background while the player plays a session. In this scenario a learner starts the serious game LostEarth 2308 for the first time, which means that no user model of this player exists. Figure 6.1 shows the first session the player plays. In this session the adaptivity response from the GAS for the player shows *It seems you did not finish a single mission yet. Finish ona and I will try to assist you then.*. This message occurs, because in the user model of the player no path is found that includes a node for a mission result, this process is explained in subsection 5.1.1. The user plays LostEarth 2308 by navigating to the different scenes and clicking through the menus. The player fails the image interpretation task and then closes the game. By doing this, the xAPI-Tracker tracks each activity the player performs and sends them to the LRS.

Figure 6.1: First Session with Failure

Figure 6.2: Second Session with Recommendation

Figure 6.2 shows the recommendation to the user according to graph from the previous

session of the current player matched with the graph of another player. In this example, the player gets the recommendation to read the briefing of the mission first and then proceed with the mission.

This scenario shows the application of the conceptuated learning scenario in section 4.1 in the serious game LostEarth 2308. In this scenario the tracked xAPI statements are directly loaded into the graph database Neo4j as soon as the GAS starts a request to the LRS. The implementation of the GAS allows to build the needed data model for this learning scenario and allows the graph analysis procedures to generate hints for the individual player in this learning scenario. So this processes allow a steady growth of the e-learning environment of LostEarth 2308 with graph data models and graph user models working in the background, that can be used for adaptivity purposes.

6.2 Discussion

One topic of this master thesis was the application of adaptivity for serious games. Various approaches regarding adaptivity have been examined and evaluated. This gathered information have been used for the conceptual approach of the learning scenario that has been worked with in this thesis.

The approach in this thesis for adaptivity was to generate graphs that can represent the user activities. The data basis for this graph were tracked xAPI statements from the serious game LostEarth 2308. Based on the learning scenario and the structure of the xAPI statements, it was possible to build a suitable graph data model, that got further enhanced during the development of this thesis.

For the creation of the user models, different approaches to solutions from existing research work were considered. The chosen approach in the context of this thesis was to derive the user models from the existing data model with the help of graph algorithms and thus to transfer the usability from the field of social network analysis to the field of e-learning. The advantage of this approach is that a wide range of graph analysis options can be used. In order to limit those, questions about users were defined for the serious game LostEarth 2308, which reflect different analysis possibilities with the help of graph algorithms. In this approach, the aim was to define a user model in order to give recommendations for the learning scenario. In this case the focus was on the distinction between *What to adapt?* and *How to adapt?*. The enclosing between *What to adapt?* and *How to adapt?* is, that *What to adapt?* stands for the recommendation of activities for the user and *How to adapt?* is answered with the creation of user models that are based on applied graph algorithms.

At the beginning of the thesis the preparation of the data was a difficult process. The first approach by loading the xAPI statements manually into the graph database was not manageable over time, because the manual loading of the data into the graph database had the disadvantages that the loading of the xAPI statements from the LRS became

time-consuming and with the time unclear. This approach became impractical for the scope of this thesis, because the data had to be loaded into the graph database without time detours so that it could be analysed directly. To solve this problem a service has been developed which allows to load single and multiple xAPI statements from the LRS into the graph database Neo4j. Examples of different analysis tools have been considered. The service also allows the implementation of different graph algorithms and procedures that are useful for user modeling. For the interpretation of the xAPI statements to create them as nodes in the desired data model in Neo4j a lot of time was invested until a robust implementation of this work was available. As well as for the pattern matching algorithm, which uses the user models created from the data model, a lot of time and research was invested to enable a viable implementation for the learning scenario.

A relation between serious games and adaptivity through user modelling based on graph analytics could be established, which contains the corresponding knowledge about problems and the potential for further (scientific) use.

There was a setback during the thesis regarding the clustering of the individual users regarding different parameters, but the current data model offers the possibility to cluster users according to their activities. However, clustering on the basis of different learning types would be a far-reaching approach, which could not be implemented because the data basis from the xAPI statements of LostEarth 2308 was not sufficient enough.

7 Conclusion and Outlook

In conclusion, it can be said that a system for user modelling with the help of Graph Analytics could be realised and also integrated in the existing ELAI. Which, in addition to the analysis methods, includes an implementation that can load xAPI statements directly into a corresponding data model in the graph database Neo4j. Based on this thesis, further steps towards adaptivity with the help of graph analytics in connection with serious games or other e-learning applications can be taken.

Looking forward, the realized service can be well developed in terms of implementation of new data models, for example if other learning games or learning scenarios are used as a starting point. It is also possible to implement further algorithms for other analysis purposes, such as clustering. Similarly, with regard to graphs, there should be a stronger focus on the weighting of edges, because not every step is equally difficult for every user. The edge weighting could be calculated dynamically within the implemented service. Furthermore, it would make sense to continue working on the game LostEarth 2308 in order to have more content suitable for adaptivity. As a further point for the outlook it is possible to automate the path of the possible interaction sequence consisting of the existing user models.

Appendix

The attached CD contains the following items:

- Thesis document as PDF-file
- LaTeX files of the thesis document
- Figures and concepts as individual files
- BibTeX-file for literature resources
- BibTeX-file for online resources
- Fundamentals and State of the Art literature as PDF-files

List of Figures

Listings

List of Tables

List of Algorithms

Bibliography

[ADL.net, 2020] ADL.net (2020). xAPI Specification 1.0.3. `https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-Data.md`. Last checked on 2020-08-19.

[Aguilar et al., 2013] Aguilar, D. A. G., Theron, R., and Penalvo, F. J. G. (15.07.2013 - 18.07.2013). Reveal the relationships among students participation and their outcomes on e-learning environments: Case study. In *2013 IEEE 13th International Conference on Advanced Learning Technologies*, pages 443–447. IEEE.

[Angles, 2012] Angles, R. (2012). A comparison of current graph database models. In *2012 IEEE 28th International Conference on Data Engineering Workshops*, pages 171–177. IEEE.

[Beel et al., 2015] Beel, J., Langer, S., Kapitsaki, G., Breitinger, C., and Gipp, B. (2015). Exploring the potential of user modeling based on mind maps. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 3–17. Springer.

[Beuster et al., 2013] Beuster, L., Elkina, M., Fortenbacher, A., Kappe, L., Merceron, A., Pursian, A., Schwarzrock, S., and Wenzlaff, B. (2013). Learning analytics und visualisierung mit dem lemo-tool. *DeLFI 2013: Die 11 e-Learning Fachtagung Informatik*.

[Birk et al., 2015] Birk, M. V., Toker, D., Mandryk, R. L., and Conati, C. (2015). Modeling motivation in a social network game using player-centric traits and personality traits. In Ricci, F., Bontcheva, K., Conlan, O., and Lawless, S., editors, *User Modeling, Adaptation and Personalization*, volume 9146 of *Lecture Notes in Computer Science*, pages 18–30. Springer International Publishing, Cham.

[Blatsios and Refanidis, 2019] Blatsios, S. and Refanidis, I. (2019). An adaptation and personalisation methodology for serious games design. In *European Conference on Games Based Learning*, pages 991–XIII. Academic Conferences International Limited.

[Brusilovsky, 1998] Brusilovsky, P. (1998). Methods and techniques of adaptive hypermedia. In *Adaptive hypertext and hypermedia*, pages 1–43. Springer.

[Cormen, 2009] Cormen, T. H. (2009). *Introduction to algorithms*. MIT Press, Cambridge Mass., 3rd ed. edition.

[DotNet4Neo4j, 2020] DotNet4Neo4j, G. (2020). Dotnet4neo4j/neo4jclient. `https://github.com/DotNet4Neo4j/Neo4jClient`. Last checked on 2020-09-07.

[Gallagher, 2006] Gallagher, B. (2006). Matching structure and semantics: A survey on graph-based pattern matching. In *AAAI Fall Symposium: Capturing and Using Patterns for Evidence Detection*, pages 45–53.

[Huang and Shiu, 2012] Huang, S.-L. and Shiu, J.-H. (2012). A user-centric adaptive learning system for e-learning 2.0. *Journal of Educational Technology & Society*, 15(3):214–225.

[IOSB, 2020] IOSB (2020). Adaptive lernsysteme fraunhofer iosb karlsruhe. `https://www.iosb.fraunhofer.de/servlet/is/8918/`. Last checked on 2020-09-07.

[Jouili et al., 2009] Jouili, S., Tabbone, S., and Valveny, E. (2009). Evaluation of graph matching measures for documents retrieval.

[Kareal and Klema, 2006] Kareal, F. and Klema, J. (2006). Adaptivity in e-learning. *Current Developments in Technology-Assisted Education*, 1:260–264.

[Kerres and Preußler, 2012] Kerres, M. and Preußler, A. (2012). *Mediendidaktik*. Oldenbourg Wissenschaftsverlag.

[Khribi et al., 2008] Khribi, M. K., Jemni, M., and Nasraoui, O. (2008). Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval. In *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, pages 241–245. IEEE.

[Lim, 2015] Lim, K. C. (2015). Case studies of xapi applications to e-learning. In *The Twelfth International Conference on eLearning for Knowledge-Based Society*, pages 3–1.

[Miller, 2013] Miller, J. J. (2013). Graph database applications and concepts with neo4j. In *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA*, volume 2324.

[Myers et al., 2000] Myers, R., Wison, R., and Hancock, E. R. (2000). Bayesian graph edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):628–635.

[Needham and Hodler, 2019] Needham, M. and Hodler, A. E. (2019). *Graph algorithms: Practical examples in Apache Spark and Neo4j*. O'Reilly Media, Sebastopol CA, first edition edition.

[Neo4j Graph Database Platform, 2020] Neo4j Graph Database Platform (2020). Drivers & language guides - neo4j graph database platform. `https://neo4j.com/developer/language-guides/`. Last checked on 2020-10-11.

[Robinson et al., 2015] Robinson, I., Webber, J., and Eifrem, E. (2015). *Graph databases*. O'Reilly, Beijing, second edition edition.

[Robles-Kelly and Hancock, 2004] Robles-Kelly, A. and Hancock, E. R. (2004). String edit distance, random walks and graph matching. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(03):315–327.

[Sanfeliu and Fu, 1983] Sanfeliu, A. and Fu, K.-S. (1983). A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics*, SMC-13(3):353–362.

[Shute and Zapata-Rivera, 2012] Shute, V. J. and Zapata-Rivera, D. (2012). Adaptive educational systems. *Adaptive technologies for training and education*, 7(27):1–35.

[Streicher et al., 2019] Streicher, A., Bach, L., and Roller, W. (2019). Usage simulation and testing with xapi for adaptive e-learning. In *European Conference on Technology Enhanced Learning*, pages 692–695. Springer.

[Streicher et al., 2018] Streicher, A., Leidig, S., and Roller, W. (2018). Eye-tracking for user attention evaluation in adaptive serious games. In *European Conference on Technology Enhanced Learning*, pages 583–586. Springer.

[Streicher and Roller, 2015] Streicher, A. and Roller, W. (2015). Towards an interoperable adaptive tutoring agent for simulations and serious games. In *International Conference on Theory and Practice in Modern Computing, MCCSIS*, pages 194–197.

[Streicher and Roller, 2017] Streicher, A. and Roller, W. (2017). Interoperable adaptivity and learning analytics for serious games in image interpretation. In *European Conference on Technology Enhanced Learning*, pages 598–601. Springer.

[Streicher and Smeddinck, 2016] Streicher, A. and Smeddinck, J. D. (2016). Personalized and adaptive serious games. In *Entertainment Computing and Serious Games*, pages 332–377. Springer.

[Sweta and Lal, 2016] Sweta, S. and Lal, K. (2016). Learner model for automatic detection of learning style using fcm in adaptive e-learning system. *IOSR J.(IOSR J. Comput. Eng.)*, 18(2):18–24.

[xAPI Overview, 2020] xAPI Overview (2020). xapi statements: The anatomy of an xapi statement. `https://xapi.com/statements-101/?utm_source=google&utm_medium=natural_search`. Last checked on 2020-08-19.

[Yen, 1971] Yen, J. Y. (1971). Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716.