Characterization and Analysis with xAPI based Graphs for Adaptive Interactive Learning Environments

Alexander STREICHER
Fraunhofer IOSB, Karlsruhe, Germany
alexander.streicher@iosb.fraunhofer.de

Stefan Wolfgang PICKL
Universität der Bundeswehr München, Neubiberg, Germany
stefan.pickl@unibw.de

ABSTRACT

In e-learning, insights from the analysis of usage tracking data can help improve teaching and learning, e.g., with learning analytics to identify strengths and weaknesses of learners or course material, or for targeted help for individual students. One analysis approach is to examine the graph networks of interaction usages. Adaptive e-learning systems (ALS), which personalize the learning experience to the learners' needs, can make use of relationship information in graph networks to determine the best adaptation strategy. For example, ALS can use graph algorithms to detect central activities that have high influence to the users or to learning objects. This paper shows how to make use of the Experience API (xAPI) protocol and graph networks for its application in adaptive interactive learning environments such as computer simulations and serious games. A prototype implementation hints at the feasibility of the concept and its practical implications.

Keywords: graph algorithms, graph mining, e-learning, learning analytics, adaptivity

1. INTRODUCTION

Characterization and analysis play a central role in building adaptive e-learning systems (ALS). The characterization includes a variety of disciplines, ranging from technical modeling of systems and communication protocols to modeling of the users cognitive or learning states. This modeling can pose the basis for, e.g., adaptive systems like Intelligent Tutoring Systems (ITS) or learning analytics. ALS analyze the observed interaction usage data to determine the best adaptation strategy. In contrast to mere adaptability, which means the possibility to be actually modifiable, the term adaptivity here means, that the process of model learning and analyzing is done automatically, e.g., by using artificial intelligence (AI) techniques. Examples can be cognitive modeling for synthetic reproductions of cognitive user states [1], machine learning to build classification models for user learning styles [2], or data mining techniques like natural language processing to produce semantically relevant learning material recommendations [3].

In this paper we focus on the characterization and analysis of the users' interaction usage data with the utilization of graph network structures and technical application of graph platforms like graph databases. This is motivated by the fact that in our ALS we use the widely applied *Experience API* (xAPI) protocol ([4], [5]), and xAPI's actor-verb-object triple structure makes it suitable to be stored and analyzed in graph platforms. Another important reason for graphs is the possibility for easy and understandable visualization, for example as 2D graphs with nodes and edges. Graphs can communicate relationships, even complex networks, in an understandable way, for developers and users alike. Adaptive systems have to make decisions that are comprehensible to the user. Users can be learners, tutor or teachers, or developers of ALS. Whilst black-box artificial intelligence models like artificial neural networks and its further developments like deep learning can achieve remarkable results, the underlying models are opaque and alien to the human user. Explainable A.I. (XAI) systems try to communicate – in a human understandable way – the data modeling as well as the data analysis. For ALS this could mean, that the reasons for adaptation are being made transparent to the user. Graphs and their comprehensible visualization as 2D networks can provide a solution approach.

For our adaptivity system for interactive learning environments, the research question is: how to characterize and analyze usage interaction data using mathematical graphs?

The contribution of this paper is: we describe how to use the xAPI data as graph structures, and we highlight graph database algorithms which are suitable for learning analytics tasks.

We use graph databases to store the incoming user interaction data, and to make use of graph algorithms for learning analytics tasks. From a mathematical perspective, graph theory offers many well-studied concepts and algorithms, and their application is seen in a variety of fields, one major field of application being social networks and the Social Network Analysis (SNA) [6], [7].

Looking at the state of research, SNA has some applications in e-learning [7], but the application of graph-based SNA approaches for interactive learning environments seems to be very limited. So far, we did not find any relevant literature that is similar to our approach. Regarding gamification, some work can be found which deals with gamified e-learning courses and SNA, as the systematic literature review by Cela et al. shows [8]. Regarding xAPI and its wide applications [5], there are not yet that many applications for computer simulations and serious games. Our work targets that combination of applying graph platforms (i.e., graph databases and graph analysis toolsets) to user interaction tracking data originating from xAPI trackers.

The paper continues with an explanation of the embedding target application scenario of adaptive computer simulations and serious games, in the following more broadly subsumed by the term Adaptive Interactive Learning Environments (AILE). Next follows the description how we transform xAPI to graph structures, following conceptual considerations which graph algorithms work for such kind of graphs. We also describe the technical approach how to implement a xAPI graph-based data processing pipeline and how to design its general software architecture.

2. ADAPTIVE INTERACTIVE LEARNING ENVIRONMENTS (AILE)

A special e-learning type is digital game based learning (DGBL) [9] with its specialized form of educational serious games. Such software systems enable learners to experience learning and training because of high intrinsic motivation in a playful way – they want to play, and by playing, they learn. Intrinsic motivation can sustainably optimize learning outcomes [10], and one approach to increase user motivation is to utilize the principles from DGBL [11]. In general, serious games are games with a characterizing goal [12], in the case of educational serious games being the goal to teach and learn. However, the term serious game does not capture the vast variety of virtual learning environments, ranging from real-world (non-digital) games or learning environments (like classrooms) to computer simulations which also contain game-like playing characteristics. One possible subsuming term is *Interactive Learning Environments* (ILE), which we will use in the following for computer simulations or serious games for education and training. Further learning optimization can be achieved through personalized learning in which adaptive e-learning systems (ALS) modify the software or content to match the individual needs of the user [13]. ALS have their reference in *Intelligent Tutoring Systems* (ITS) [14] and exploit techniques from artificial intelligence to dynamically adapt the systems to the users' context. *Adaptive Interactive Learning Environments* (AILE) therefore embody the principles of ALS to dynamically adapt the interactive learning technologies to the actual needs of the users while considering DGBL principles. This can include strategies to mimic human-like behavior in virtual agents which help or guide the users, or to promote trial-and-error approaches, or to simply not to disturb the users' game flow.

The goal of adaptive knowledge transfer is to promote knowledge acquisition through individual or personalized knowledge management. Our work is focused on virtual e-learning worlds, i.e. computer simulations and digital learning games, so-called serious games [12]. The core thesis is that through adaptivity to the learning progress the learning success can be optimized - analogous to the personal supervision by a human tutor [13]. The concrete goal is to adapt the systems dynamically and iteratively to the individual learning progress of the users. Depending on the level of knowledge and abilities of the learner, the AILE are dynamically modified, different learning materials are offered or situationally adapted assistance is provided, e.g., context-relevant learning material recommendations.

In the following and for process structuring of our adaptivity system we follow an extended 4-phased adaptivity cycle. It basically extends the model from Shute and Zapata-Rivera [15] by combining the analysis phase and the learner models (from the current user and also from other users), cf. Figure 1. In this 4-phased adaptivity cycle, there are the phases (1) capture, (2) analysis and user (learner) modeling, (3) select and (4) present. From a system theoretic point of view, this follows the principles of adaptive control systems with closed feedback control loops [16]. To define the closed loop performance, we need a reference model as basis for deviation estimation, i.e., a model to characterize the users' playing behavior. One approach in adaptive control is Model Reference Adaptive Controller (MRAC) [16]. The graph structures or models work as a basis for such kind of reference models. This concerns the characterization part of this paper: the graph structures encode information on the user interactions, information on the observed systems (games, ILE, etc.) and on the actions of individual users. This is, the graphs' relationships carry information how an ILE is built and how it is used, e.g., which sequence of interaction elements are triggered by whom. Tutors can use this information in learning analytics dashboards to estimate learners' performance or how to improve the courses. Similarly, AILE can use this information to estimate the learners' performance to select suitable adaptation strategies. In this case the graphs together with a previously defined *Ideal Path Model* [17] could act as reference models for MRAC.

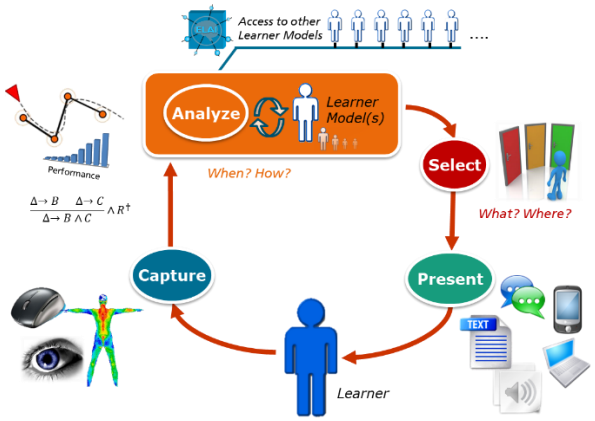

Figure 1: Extended 4-phased adaptivity cycle (based on [15]) with combined analysis-learner-models-phase.

3. GRAPHS & E-LEARNING

Due to the triple nature of xAPI statements (actor, verb, object), these statements can be translated and analyzed very well in graph structures. The observed interaction data with multiple different actors and activities is highly connected, which makes the use of graph platforms relevant. This can also be seen in social networks where similar data triples are used to describe the actions of users. Our approach to use the xAPI's triple structure and graphs is well related to the much bigger domain of social networks and its accompanied discipline of *Social Network Analysis* (SNA). In social networks the W3C *Activity Streams* [18] standard is used to store the users' actions in an interoperable way. The Web frontend typically displays the activity streams as a history, e.g., "Aaron likes John's picture". SNA is used to analyze the highly connected data, for example to search for cliques and neighborhoods, expert finding [19], [20], or to look for so-called "influencers". The possibilities of SNA have also been recognized for e-learning where one is interested in relationships between learners, or between learners and the offered (learning) objects or activities [7].

At first, the xAPI data streams must be stored, and its graph nature with nodes and relationships motivates the use of graph databases [21]. A graph platform which supports both transactional storage and data processing is Neo4j [22]. It supports transactional processing and analytical processing of graph data, i.e., it is a graph database with included graph analytics facilities like commonly used graph algorithms. In contrast to typical relational databases, graph databases belong to the NoSQL databases where the data is not required to adhere to predefined schema [21]. Since we make use of the JSON-formatted xAPI statements, the use of NoSQL databases is apparent. While JSON xAPI statements are typically stored in xAPI *Learning Record Stores* (LRS), which typically use NoSQL document-oriented databases, these databases however

do not provide graph models that promote relationships, nor the possibility to run graph algorithms for analytics purposes.

The underlying theory for graph databases and its graph models is graph theory. Graph theory is the study of mathematical graph structures. A graph is a structure in which the set of objects form pairs having similarity among them. These objects are called vertices or nodes, connected to other vertices by edges. There are two types of graphs: directed and undirected graphs. Directed graphs are asymmetrical since the relationship between the nodes is unidirectional in nature. Undirected graphs, in contrast, are symmetrical because the connecting edges between the vertices represent the reciprocated relationship. In the context of this work, we deal with acyclic directed graphs.

As an example how to apply graph theory to serious games, we look at our field of application, aerial image interpretation, and at a serious game for professional image interpreter training [23]. In such a game the users learn how to systematically plan and task the correct imagery sensor, and how to correctly interpret images towards a standardized report. In the game, the users interact with a virtual weather report console in order to then carry out sensor deployment planning depending on the current or predicted weather conditions. Obviously, the user should not task an optical sensor for clouded weather conditions when only clouds are to be seen; instead, a radar image sensor should be used. One learning objective in the game is to first look at the weather forecast to employ to correct sensor deployment strategy.

Using graph analysis and machine learning, models of interaction patterns can be learned, e.g. which activities follow each other. In the example, the first step is to first view the weather console, then perform sensor scheduling; see "next" connections in Figure 2. Further, the graph shows which users/actors did not follow typical interaction sequences. In the example, actor A3 did not view the weather report first before scheduling a sensor, which can be problematic and not congruent with the learning objective.

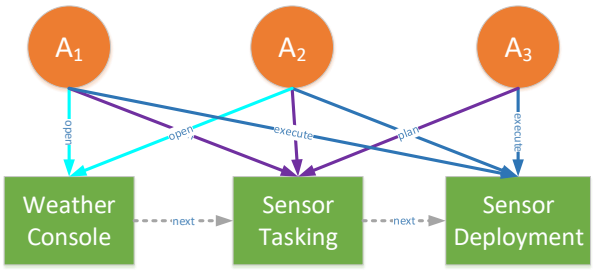

Figure 2: Example of a simple graph structure from observed user interaction data. Three actors (circles A1-A3) use different objects (green rectangles) with different interaction verbs (connections "open", "plan", "execute"). Relationships can also be learned automatically (grey dashed connections, "next").

Next, we consider another serious game example and observed xAPI activity streams. For the image exploitation game "Exercise Trainer" (EXTRA), where the player must produce imagery products, we attached an xAPI adapter on various game actions (activities). Without loss of generality, this approach can also be applied in similar ways to other games, computer simulations or assistance systems in general (xAPI libraries are available for common programming languages).

Figure 3: Serious game "Exercise Trainer" (EXTRA), first level with just a factory, a connection and a market.

In our example game session the user "John Doe" starts the EXTRA game, adds a single connection between an image production "factory" and a distribution point ("market"); he makes multiple sells of generated "raw image" products, and finally completes the level. The observed xAPI statements are (not displaying repeated statements):

```
<John Doe, initialized, EXTRA Game>
<John Doe, added, Connection>
<John Doe, built, raw_image>
...
<John Doe, sold, raw_image>
...
<John Doe, completed, EXTRA Level>
```

Important to note is that in the xAPI statements the user is always the active executing entity (actor). This follows the xAPI specification [4], i.e., not the factories (building raw image products) are the active entities (actors) but always the user/player.

The statements are converted to nodes and relationships, inserted into the graph-database Neo4j (version 3.5), and visualized using Neo4j's browser client using the Cypher query "`MATCH (a:Actor)-[v]-(o:Activity) RETURN a,v,o`". Figure 4 shows the resulting graph.

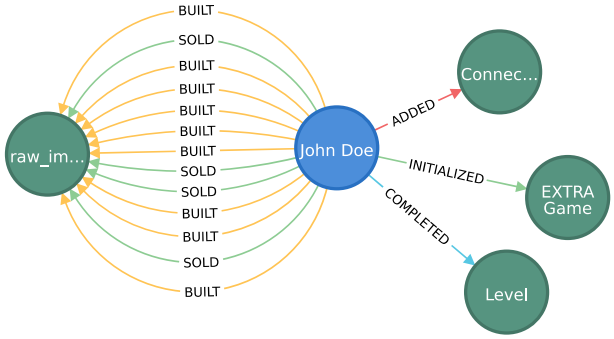

Figure 4: Graph visualization of a simple example for 16 xAPI activity statements with actor node "John Doe" and differently connected activity nodes (colored green). Multiple connections for multiple actions, e.g., 9 x "`BUILT`" with varying timestamp attributes (not displayed here).

Problematic with such kind of graphs is that they can become very big and hardly comprehensible, least understandable. As depicted in our small example the amount of visible relationships is identical to the number of xAPI statements

because for each action (verb) a new relationship with a different timestamp attribute is created. For easier and more understandable visualizations, identical relationships (verbs) should be drawn as just a single connection type. This reflects better the basic idea behind graphs to primarily focus on just nodes and relationships.
Further observations will be added to the graph network. In our example, a different user "Aaron" also plays the game in a similar fashion but with a different product "video" and without completing the level (no observed completed-statement). The observed xAPI statements are:

```
<Aaron, initialized, EXTRA Game>
<Aaron, add, Connection>
<Aaron, built, raw_image>
<Aaron, built, video>
<Aaron, sold, video>
```

Figure 5 displays the resulting graph; existing nodes are reused and a new nodes for "video" is created. However, in this visualization identical actions (e.g., "BUILT") have been joined into a single dedicated relationship type; the information about the amount of observed actions is stored as an attribute.
One observation is that such graphs typically follow a power law distribution where one quantity varies as a power of another. This is especially true for real-world networks with an uneven distribution of nodes and relationships [22]. In our case the number of actor and activity nodes grows less than the number of relationships; new actor and activity nodes produce lot's of new connections, as can be seen in our example. At some point the number of activity nodes is saturated, because typically an interactive software has a finite number of interaction activities – even for adaptive systems with dynamically modified environments and modified activity sets. Similarly this is also true for the number of actor nodes.

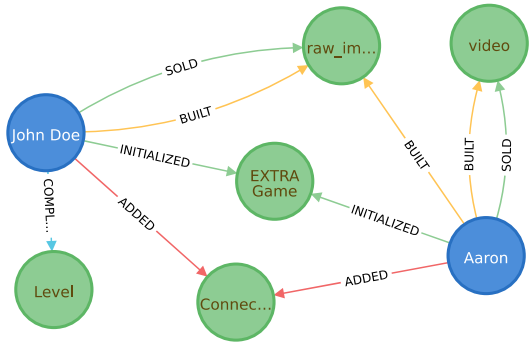

Figure 5: Graph visualization of additional game session from user "Aaron". Existing activity nodes (colored green) are reused, e.g., "Connection" and "EXTRA Game". Identical actions are joined into dedicated relationship types.

An important question is how to model the time dimension. For e-learning analytics we are interested in questions which incorporate time. For example, how long did a user work on a certain task, or how fast is the users' learning pace. In the above simple example, we chose to encode time in an attribute field of a connection. However, another approach could be to have multiple graphs per timestamp, i.e., time would add another dimension to the 2D graph. For each observation a new identical graph of nodes and connections would be created. However, typical graph platforms and graph visualization tools are oriented at 2D graph network structures, and time is typically encoded as attributes to nodes or connections.

4. GRAPH ALGORITHMS & E-LEARNING

This section deals with the algorithmic aspects of the initially stated research question, i.e., our interest is what graph algorithms are suitable for the analysis of our xAPI interaction data graphs. Graph algorithms are used to compute metrics for graphs, nodes, or relationships. They can provide insights on relevant entities in the graph (e.g., centralities, ranking), or inherent structures like communities (e.g., community-detection, graph-partitioning, clustering). Considering classical graph algorithm classes, possible learning analytics for the xAPI based observations questions could be:

- What are central learning objects or interaction elements, or what are central users? Or inversely which users or objects are less connected?
- What are typical interaction patterns and how do they develop over time?
- What are costs, capacities or control points?
- What interactions happen between users?
- What influence do certain users have on other users?
- …

In our type of graphs, we observe mainly actor and activity type nodes; each user is represented by an actor node, each activity or object by an activity node (cf. previous section). For computer simulations and serious games the activities can be interaction elements like interactive consoles, other player avatars or those of non-player characters (NPC), etc.

Node type	Source
Actor	User/learner
Activity	Learning object (ILE) Interaction element

The type of graph algorithms is oriented at our approach's implementation strategy to use the graph platform Neo4j [22]. The interesting algorithm classes for the described field of application are (not exhaustive):

- Centrality [22]
- Expert Finding [20] [19]
- Community Detection
- Social Influence Detection [6] [24]
- Pattern Detection [25]

Centrality
Centrality is one of the most important algorithm used in SNA in general [22], and also for e-learning motivated SNA [7]. Centrality helps to find the most important nodes by evaluating the roles and impact of the node in the network, and to understand the group dynamics such as credibility, accessibility, outreach and links or connections [22].

Degree Centrality: measures the number of relationships of a node. Examples:

- Which actor is interacting with many activities? What is the in-degree or out-degree of central actor nodes?
- Which activities are used frequently, or vice versa, which are unused and possibly obsolete?

Closeness Centrality: measures which nodes have the shortest path to all other nodes. According to Needham and Hodler [22] Closeness Centrality should be used when interested in information spreading, i.e., to find nodes which disseminate things the fastest. Example: which learning activities typically happen together?

Betweenness Centrality: measures the control flow between nodes and groups. Examples:

- Which activity has the most influence on the flow between actors and activities of the neighborhood group?
- Which activities must not be missed, because they are part of important sequences (sub-graphs)?

PageRank: measures a node's influence by summing up scores of the node's neighbors, and their neighbors. Examples:

- What are influencing activities and what is their ranking?
- Which actors play an important role in the network, which actors can be seen as experts?

Community Detection

Most networks form groups or sub-graphs which can be identified as patterns. Member nodes of such sub-graph have more relationships within the group than with nodes outside their group [22]. Important community detection algorithms are triangle count, clustering coefficient, connected components, or label propagation.

Triangle Count: measure how many nodes form triangles. Example: which activities typically appear together, which form a group?

5. APPLICATION EXAMPLE

Our prototype implementation follows the described approach: a serious game is attached with an xAPI adapter which sends selected activities to an xAPI compliant endpoint (cf. Figure 6). Typically, this endpoint is a standard xAPI database, a Learning Record Store (LRS), which client systems can periodically query for new data. However, this would not allow for instant, live or online analysis. We propose to use the wiretap enterprise integration pattern which, placed between the sending xAPI adapter and the receiving LRS. The wiretap can be implemented as a Java servlet proxy in our "E-Learning A.I." (ELAI) architecture which targets adaptivity for ILE [26]. The proxy approach allows to react to new usage data without a time offset which. In our ELAI architecture the xAPI JSON data is converted to nodes and relations, and then stored into the graph database (cf. Figure 6). The adaptive learning system (ALS) gets the output of the above described graph algorithms, e.g., selection of the closest nodes (closeness centrality) or most influencing node (betweenness centrality) for the currently observed user. The adaptivity system sends the interpreted data to a controller component which adapts the interactive environment [26]. This data could be scalar values which carry information on the users' performance, skill level or helping level, or recommendations in form of modified content, text, hyperlinks, etc. [26] For learning analytics purposes the reporting output of the LRS, of the graph platform and the ALS is made available with a Web-based dashboard (example in Figure 7).

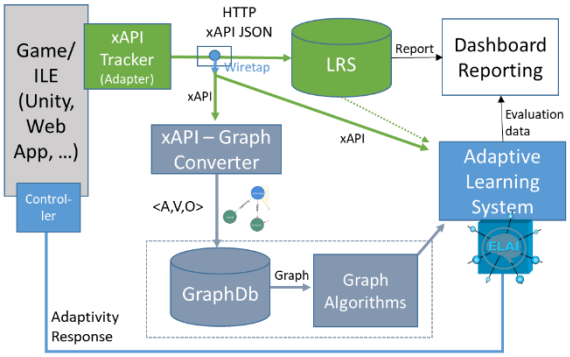

Figure 6: Data processing architecture.

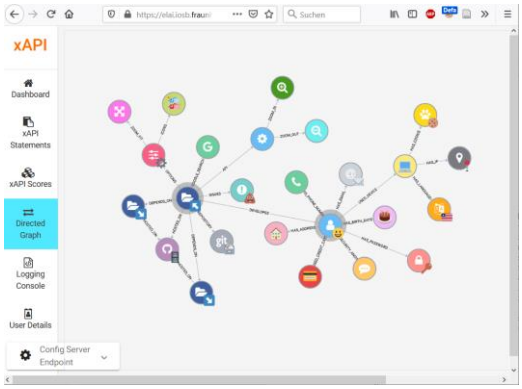

Figure 7: Example for a graph visualization in a learning analytics dashboard, segment for a selected user.

6. CONCLUSION

Adaptive interactive learning environments (AILE) need dynamic and flexible data structures for user or learner modeling, and to model the interactive learning environments (ILE). Graph structures can provide a basis for this modeling. In this paper we describe how to make use of graph databases and their algorithms for the characterization and analysis of ILE. In respect to standard input-processing-output, we use the e-learning standard xAPI as input; it is converted to graph structures which a graph platform like Neo4j can process; the output from graph algorithms, like degree centrality or expert finding, can be visualized for the user (e.g., in a learning analytics dashboard), or it can be used in the adaptivity logic of AILE to determine the next best adaptation strategy, e.g., recommendation of the closest connected and highest ranked activity node. On basis of our xAPI based graph structures the paper presents a selection of graph algorithms and their possible application examples in the e-learning analysis context.

7. DISCUSSION

The graph algorithm selection is preliminary and further research studies must show how the algorithms react to real data. Also, further research should study the influence of different game genres on the resulting graph structures, e.g., what graph similarities (patterns) exist between different genres and how do they differ. This is important for automatic analysis of games, because such graphs could act as classification features to automatically determine the type (genre) of a game or as a basis for a metric for retrieval tasks.

The paper currently just describes a syntactic approach, i.e., how to transform incoming data into graph structures. There is no semantic information. A semantic level should be considered, what – in a semantic sense – did the users do? How is the mapping to learning objectives, and how to determine if a user has achieved the learning objectives? Subgraph and pattern matching according to an additional reference model graph could be done here. A study is going to be conducted to examine what the actual effect of the approach to real e-learning courses is.

8. ACKNOWLEDGEMENTS

The underlying project to this article is funded by the Federal Office of Bundeswehr Equipment, Information Technology and In-Service Support under promotional references. The authors are responsible for the content of this article.

9. REFERENCES

[1] N. Matsuda, W. W. Cohen, J. Sewall, G. Lacerda, and K. R. Koedinger, "Evaluating a Simulated Student using Real Students Data for Training and Testing," *Proceeding Int. Conf. User Model.*, vol. 2007, pp. 1–10, 2007, doi: 10.1007/978-3-540-73078-1_14.

[2] E. J. Brown, T. J. Brailsford, T. Fisher, and A. Moore, "Evaluating Learning Style Personalization in Adaptive Systems: Quantitative Methods and Approaches," *IEEE Trans. Learn. Technol.*, vol. 2, no. 1, pp. 10–22, Jan. 2009, doi: 10.1109/TLT.2009.11.

[3] A. Streicher, N. Dambier, and W. Roller, "Semantic Search for Context-Aware Learning," *7th Int. Conf. Gener. Web Serv. Pract.*, pp. 346–351, 2011, doi: 10.1109/NWeSP.2011.6088203.

[4] Advanced Distributed Learning (ADL), "Experience API (xAPI) Specification, Version 1.0.1," Advanced Distributed Learning (ADL) Initiative, U.S. Department of Defense, 2013.

[5] K. C. Lim, "Case Studies of xAPI Applications to E-Learning," p. 12, 2015.

[6] D. J. Cook and L. B. Holder, *Mining Graph Data*. 2006.

[7] K. L. Cela, M. Á. Sicilia, and S. Sánchez, "Social Network Analysis in E-Learning Environments: A Preliminary Systematic Review," *Educ. Psychol. Rev.*, vol. 27, no. 1, pp. 219–246, Mar. 2015.

[8] L. de-Marcos *et al.*, "Social network analysis of a gamified e-learning course: Small-world phenomenon and network metrics as predictors of academic performance," *Comput. Hum. Behav.*, vol. 60, pp. 312–321, Jul. 2016, doi: 10/f8nsbw.

[9] M. Prensky, "Computer games and learning: Digital-based games," in *Handbook of Computer Game Studies*, 2005, pp. 97–124.

[10] S. Sampayo-Vargas, C. J. Cope, Z. He, and G. J. Byrne, "The effectiveness of adaptive difficulty adjustments on students' motivation and learning in an educational computer game," *Comput. Educ.*, vol. 69, pp. 452–462, Nov. 2013, doi: 10.1016/j.compedu.2013.07.004.

[11] M. Prensky, *"Don't bother me Mom, I'm learning!": how computer and video games are preparing your kids for twenty-first century success and how you can help!* 2006.

[12] R. Dörner, S. Göbel, W. Effelsberg, and J. Wiemeyer, Eds., *Serious Games - Foundations, Concepts and Practice*. Cham: Springer International Publishing, 2016.

[13] A. Streicher and J. D. Smeddinck, "Personalized and Adaptive Serious Games," in *Entertainment Computing and Serious Games: International GI-Dagstuhl Seminar 15283, Dagstuhl Castle, Germany, July 5-10, 2015, Revised Selected Papers*, R. Dörner et al., Ed. Cham: Springer International Publishing, 2016, pp. 332–377.

[14] B. P. Woolf, *Building Intelligent Interactive Tutors*. Morgan Kaufmann, 2009.

[15] V. Shute and D. Zapata-Rivera, "Adaptive educational systems," *Adapt. Technol. Train. Educ.*, vol. 7, no. 1, pp. 1–35, 2012, doi: 10.1017/CBO9781139049580.004.

[16] K. J. Åström and B. Wittenmark, *Adaptive control: second edition*. Dover Publications, 2013.

[17] A. Streicher, S. Leidig, and W. Roller, "Eye-Tracking for User Attention Evaluation in Adaptive Serious Games," in *13th European Conference on Technology Enhanced Learning, EC-TEL 2018*, Leeds, UK, 2018, pp. 583–586, doi: 10.1007/978-3-319-98572-5_50.

[18] J. Snell and E. Prodromou, "Activity Streams 2.0," W3C, W3C Recommendation, 2017.

[19] A. Kardan, A. Omidvar, and F. Farahmandnia, "Expert finding on social network with link analysis approach," in *2011 19th Iranian Conference on Electrical Engineering*, 2011, pp. 1–6.

[20] Y. Fu, R. Xiang, Y. Liu, M. Zhang, and S. Ma, "Finding Experts Using Social Network Analysis," in *IEEE/WIC/ACM International Conference on Web Intelligence (WI'07)*, Fremont, CA, USA, 2007, pp. 77–80, doi: 10/b4s9q7.

[21] I. Robinson, J. Webber, and E. Eifrem, *Graph Databases*. O'Reilly Media, Inc., 2013.

[22] M. Needham and A. E. Hodler, *Graph Algorithms: Practical Examples in Apache Spark and Neo4j*. O'Reilly Media, 2019.

[23] D. Atorf, E. Kannegieser, and W. Roller, "Balancing Realism and Engagement for a Serious Game in the Domain of Remote Sensing," *Games and Learning Alliance*, vol. 11385. Springer International Publishing, Cham, pp. 146–156, 2019, doi: 10.1007/978-3-030-11548-7_14.

[24] R. Aviv, Z. Erlich, G. Ravid, and A. Geva, "Network analysis of knowledge construction in asynchronous learning networks," *Online Learn.*, vol. 7, no. 3, Mar. 2019, doi: 10/ggf9bp.

[25] T. R. Coffman and S. E. Marcus, "Pattern classification in social network analysis: a case study," in *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720)*, 2004, vol. 5, pp. 3162-3175 Vol.5, doi: 10/bndgn3.

[26] A. Streicher, "Interoperable Adaptivity and Learning Analytics for Serious Games in Image Interpretation," in *International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, Funchal, Portugal, 2017, pp. 709–710.