IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 8 Issue 8, August 2021
ISSN (Online) 2348 – 7968 | Impact Factor (2020) – 6.72
www.ijiset.com

An Empirical Design Model for an AI-enabled Learning System

Ayse Arslan

Abstract

This paper describes a High-level (Conceptual) Design phase to create an early blueprint of an AI-enabled learning system using User-Centered Design (UCD) approach for higher education. The design of AI-enabled module development system (AIMOD) focuses on two main aspects: Navigation Model and Prototyping. The navigation model will illustrate how all user interface screens should be connected reflecting the user's mental model to facilitate intuitive navigation between screens to accomplish the task of AI-enabled learning system. A high-fidelity prototype will provide details of all key screens and a number of auxiliary screens with appropriate navigation between them. This paper explores the prototype development for AIMOD following a user-centered, rather than a technology-focused, methodology given the high cognitive nature of outcome-based course design tasks, and the high levels of interactions required between the user and the system to help users build an enduring foundation of knowledge, skills and habits of mind about curriculum development.

1. Introduction

Giving the proliferation of AI-based technologies and tools into our everyday lives, there has been a growing demand and interest in using User-Centered Design (UCD) approach to develop an AI-enabled module development system (AIMOD) to facilitate course design, especially for higher education during Covid times. To this end, this paper addresses the following research goal:

- Develop an AI-enabled module development system (AIMOD) for higher education using user-centered design methodology and providing a list of related architectural aspects

The AIMOD will be an open-source web-based course design software that:

- Guides individual or collaborating users, step-by-step, as they define learning objectives, select content to be covered, develop an instruction and assessment plan, and define the learning environment and context for their course(s).
- Generates documentation of course designs similar to an architect's blueprint which articulates the plans for a structure.
- Provides just-in-time help to the user on how to perform course design tasks efficiently and accurately.
- Provides feedback to the user on the fidelity of the course design (whether there is an alignment of the course design components (i.e., content, assessment, and pedagogy) around the defined course objectives.

The development of AIMOD follows a user-centered, as opposed to technology focused, methodology given the high cognitive nature of outcome-based course design tasks, and the high levels of interactions required between the user and the system to help users build an enduring foundation of knowledge, skills and habits of mind about curriculum development.

2. Review of Existing Studies

Constructionism, long before it had a name, was intimately tied to the field of AI. The paper 'Twenty Things to Do with a Computer' (Papert & Solomon, 1971) suggests on the first page that artificial intelligence should deeply affect our thinking about 'computers in education'. Similarly, according to Kahn (1977) AI plays three roles:

(1) learners use AI programming tools in their projects,
(2) learners create artefacts by interacting with AI programmes, and
(3) the use of computational theories of intelligence and learning in the design of learning activities.

During the 1970s Kahn built two microworlds in Logo to support AI projects (Kahn, 1975, 1976). One effort involved supporting the building of projects that could produce simple animations in response to textual instructions. Pattern matching was used to extract the 'meaning' from the instructions. The other microworld provided higher-level support for natural language processing. While much simpler than later natural language toolkits, these efforts were designed to encourage students to think about language and how we generate sentences and extract meanings.

Minsky and Papert spoke of acquiring concepts to make one better at reflection and learning. Yet, using neural nets, learners are not programming at the level of concepts, symbols, and plans, but instead are relying upon crude approximations to how neurons work in brains. As Ames (2018) stated Mindstorms was seen as a potent framework for understanding learning by thinking about brains like we think about computers.

Some real-world examples of AI include (Horizon Report, 2020):

- At Durham University in the UK, staff are using Holly, an AI "student engagement platform," to promote student success through the admissions process.
- Penn State's Spectrum uses natural language processing to analyze transcripts of course sessions, enabling it to "reflect back to teachers different patterns and data points across an entire semester."
- The New Zealand Mind Lab is using AI to develop sentiment analysis tools to investigate the attitudes and emotions of students when they are interacting on social media about their course experience.
- WeLearn, a project originating at the Center for Research and Interdisciplinarity in Paris, enables semantic-localized knowledge sharing through the implementation of AI algorithms so that learning resources are visualized on a "map of concepts".

As Microsoft's Ayoub (2020) asserted, AI can empower teachers and learners of all abilities. These enthusiasms are predicated around three distinct forms of emerging educational technologies:

- Alongside the continued adoption of 'learning management systems' to facilitate the sharing of resources and group communication, are growing institutional enthusiasms for 'blended', 'hybrid' and 'hyflex' approaches that involve teaching to be hosted (at least partially) online.

- Second, is the continued rise of personalized (or more accurately individualized) learning systems designed to direct individual engagement with online learning resources through the use of sophisticated data-driven analytics to guide student decision-making.

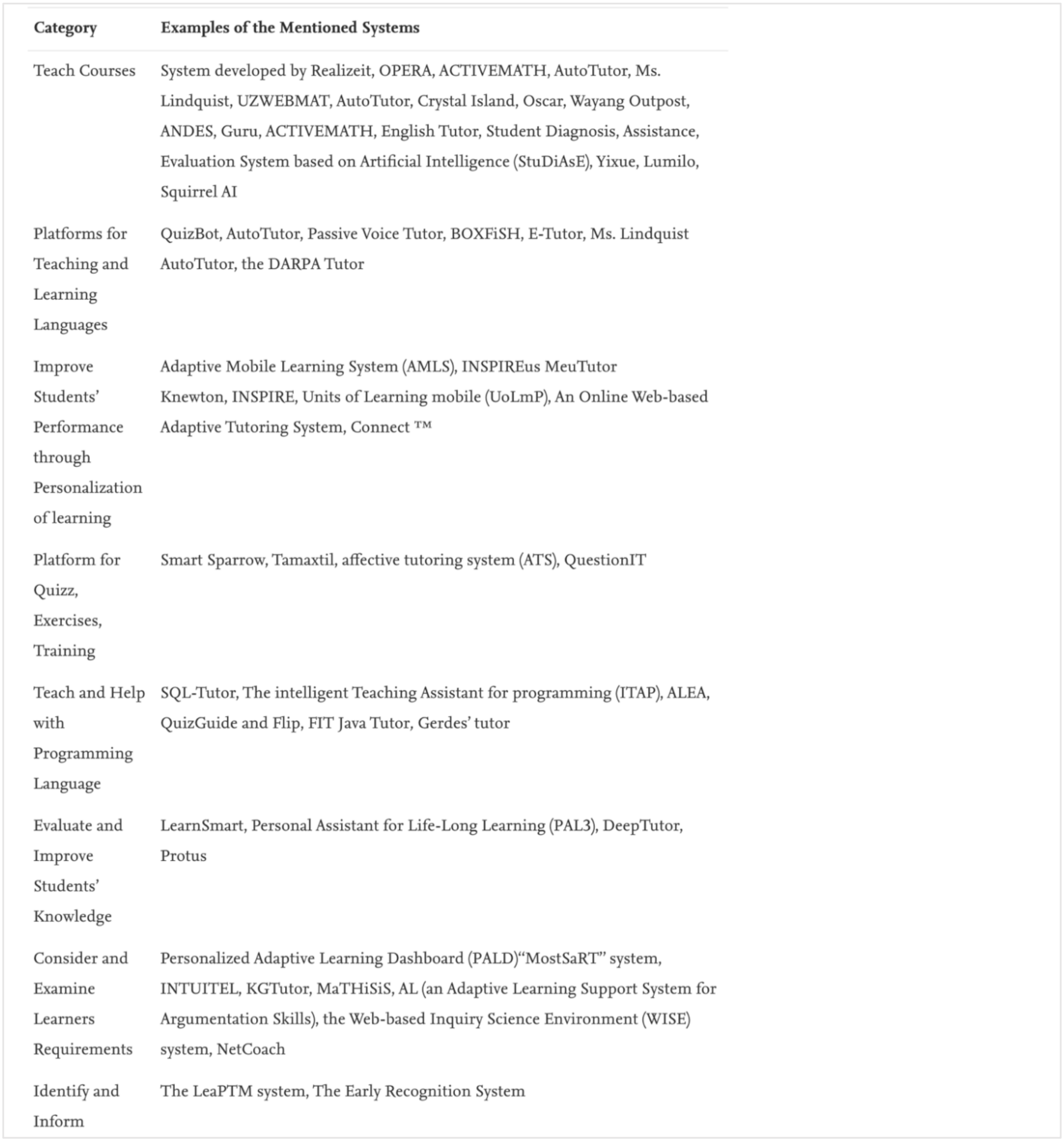
- Third, is a range of other forms of AI-driven technology – mostly designed to support automated decision making for institutions, teachers and learners. This includes system-wide 'automated education governance' based on AI-driven modelling (Gulson and Witzenberger 2020), and institution-specific use of AI-driven recruitment, as well as facial and neurological detection systems to monitor students' attention levels and automated essay assessment – so called 'robo-grading'.

Good examples of AI-enabled learning environments include mainly the following systems (Figure 1):

- Intelligent tutoring systems: An intelligent tutoring system 'uses techniques of AI to model a human tutor in order to improve learning by providing better support for the learner' (Hasanov et al., 2019).

- Adaptive learning systems: Adaptive learning systems are personalized learning platforms that adapt to learners' learning strategies, the sequence and difficulty of the task abilities, the time of feedback and learners' preferences (Pliakos et al., 2019; Xie et al., 2019). These platforms encourage learners to monitor their learning journeys via automated feedback cycles within the systems, allowing them to progress independently of the course instructor.

- Recommender systems: Recommender systems are 'software tools based on machine learning and information retrieval techniques that provide suggestions for potential useful items to someone's interest' (Syed et al., 2017).



Category	Examples of the Mentioned Systems
Teach Courses	System developed by Realizeit, OPERA, ACTIVEMATH, AutoTutor, Ms. Lindquist, UZWEBMAT, AutoTutor, Crystal Island, Oscar, Wayang Outpost, ANDES, Guru, ACTIVEMATH, English Tutor, Student Diagnosis, Assistance, Evaluation System based on Artificial Intelligence (StuDiAsE), Yixue, Lumilo, Squirrel AI
Platforms for Teaching and Learning Languages	QuizBot, AutoTutor, Passive Voice Tutor, BOXFiSH, E-Tutor, Ms. Lindquist AutoTutor, the DARPA Tutor
Improve Students' Performance through Personalization of learning	Adaptive Mobile Learning System (AMLS), INSPIREus MeuTutor Knewton, INSPIRE, Units of Learning mobile (UoLmP), An Online Web-based Adaptive Tutoring System, Connect ™
Platform for Quizz, Exercises, Training	Smart Sparrow, Tamaxtil, affective tutoring system (ATS), QuestionIT
Teach and Help with Programming Language	SQL-Tutor, The intelligent Teaching Assistant for programming (ITAP), ALEA, QuizGuide and Flip, FIT Java Tutor, Gerdes' tutor
Evaluate and Improve Students' Knowledge	LearnSmart, Personal Assistant for Life-Long Learning (PAL3), DeepTutor, Protus
Consider and Examine Learners Requirements	Personalized Adaptive Learning Dashboard (PALD)"MostSaRT" system, INTUITEL, KGTutor, MaTHiSiS, AL (an Adaptive Learning Support System for Argumentation Skills), the Web-based Inquiry Science Environment (WISE) system, NetCoach
Identify and Inform	The LeaPTM system, The Early Recognition System

Figure 1. Review of existing studies

As Verdú et al. (2015) stated, 'Many of these learning systems as well as Intelligent Tutoring Systems are described in the literature, and their effectiveness has been proven. However, these systems are rarely used in real educational settings practices in ordinary courses.' The problem remains, and recent studies (Figure 1.) highlight the lack of successful AI-enabled learning systems, such as adaptive systems, implemented in practice (Cavanagh et al., 2020; Imhof et al., 2020; Somyürek, 2015). These studies, had a notable lack of evidence concerning the potential association between certain problems faced by learners and lecturers and AI-enabled learning interventions that solve these problems.

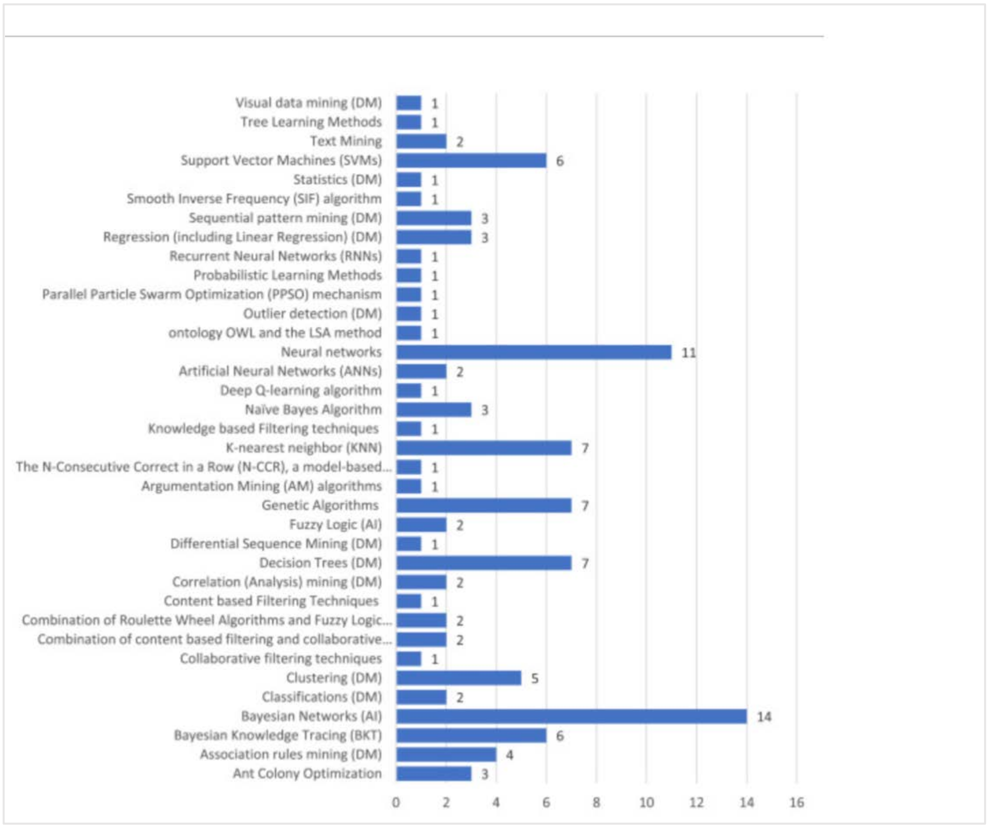


Figure 2. Techniques for AI-enabled learning systems

As displayed in Figure 2, existing studies can be placed in four categories based on implemented interventions and solutions applied in AI learning environments:

- Systems: Many of the published documents used a system (adaptive learning system, intelligent mechanism, or adaptive learning platform) as an intervention.

- Frameworks: Frameworks are constructs that define concepts, practices, values and assumptions as well as provide a set of guidelines on how to implement the frameworks. Most of the frameworks recommended as solutions in these papers comprised essential elements and features for implementation in learning environments. The proposed items are in the form of AI techniques, user (learner) models and other adaptive techniques (Figure 2.). The frameworks highlighted and described the relationships amongst the suggested elements.

- The coded frameworks provided numerous vital steps and actions for achieving a better adaptive learning experience.

- Models: A model is 'a pattern of something to be made, a description or an analogy used to visualize and reason about the system to be developed and its likely effects' (Stoica et al., 2015, p. 45). The models were either a problem-solving tool, experiment or abstract narrative of a component or system to be designed.

- Approaches: An approach refers to a set of viewpoints or theoretical concepts applied to understand, explain and solve a problem observed in a particular phenomenon.

Iten et al. published in January 2020 an article, on neural networks about physical concepts, where the authors were exploring whether the laws of quantum physics, and other physical theories can explain data from experiments if one assumes no prior knowledge of physics. They claim to achieve this goal by implementing a neural network, called SciNet, that mimics a physicist asking questions about the future behavior of a system. In other words, SciNet aims at mimicking a physicist who deduces from observations the corresponding equation, e.g. in a simple case of constant motion at speed. The functioning of SciNet is schematized in Fig. 3, reproduced from Iten et al. (2018).

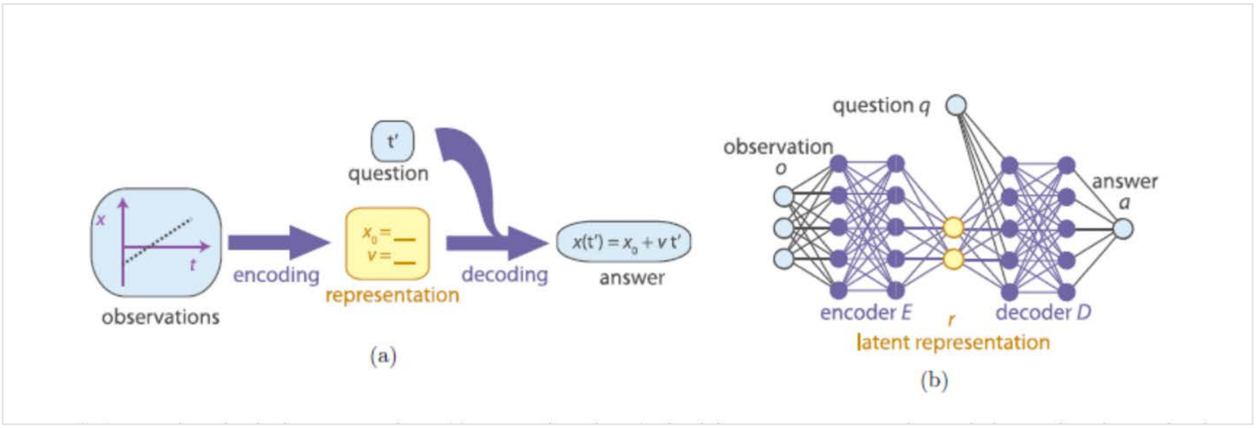


Fig. 3. Learning physical representations

As seen in Figure 3, part (a) provides an overview of human learning in which a physicist compresses experimental observations into a simple representation (encoding). When later asked any question about the physical setting, the physicist should be able to produce a correct answer using only the representation and not the original data. This process of producing the answer from the representation is referred to as “decoding.” For example, the observations may be the first few seconds of the trajectory of a particle moving with constant speed; the representation could be the parameters “speed v” and “initial position” and the question could be “where will the particle be at a later time?”

However, there are still several problems that have yet to be addressed by AI-enabled learning interventions such as designing and assessing adaptive courses, high instructor workload, lack of a specific framework for implementing intelligent agents in the systems and high levels of attention among learners in the execution of the proposed tasks. Most of the models, as noted by Dargue and Biddle (2014, p. 1), ‘are quite complex to enable just about any learner to get the optimum tailored experience possible’. Several studies have applied AI-enabled learning interventions to address different concerns such as poor feedback or quality of the learning process. One good example is a personalized adaptive online learning analysis model that analyses the structure of a learning process using big data analysis (Liang & Hainan, 2019). This proposed novel adaptive e-learning model can improve the quality of the learning process by providing the most suitable learning content for each learner. This model was designed to address inaccurate and incorrect learning material selection processes in adaptive learning systems.

Another major criticism is that AI is based upon an obsolete and limited conception of cognition (Rescorla, 2020; Strube, 2001). The argument is that the old view of cognition underlying AI ignores the situated and social nature of cognition. The new view is that ‘cognitive systems are [to be] conceived [of] as autonomous social agents, situated in a complex dynamic environment’ (Strube, 2001). Most of the AI programming tools are capable of controlling robotic artefacts and communicating with other agents.

The reviewed articles leave no doubt that some of the most advanced research efforts in AI precisely aim at constructing ANNs that can learn, discover, and master (use) concepts, laws, and ultimately theories – in the cases studied physical concepts, laws, theories. This is clear in Iten et al. (2018, 2020): here the networks are trained to discover concepts and law statements, and use these to answer questions about the future evolution of specific systems.

In sum, a human agent uses on a daily basis a vast array of theories; in one specific free act a whole web-of-theories (physical, sociological, psychological, axiological, etc.) is usually involved; and this web needs to form a coherent whole. An AI-enabled system will have to monitor its acts by, notably, theories about what is a sensible act in a given (physical, social, ethical,…) environment. Clearly, the ‘theories’ that state-of-the-art ANNs can handle are a far cry from this requirement and much work needs to be done to even dissect the various conceptual ingredients and logical steps of how such a web-of-theories could work, before trying to implement it in computer code.

3. User-Centered Design Methodology

Traditional software design methodologies focus on a typical requirements-design-development approach in which user interface and user interactions are deferred to later stages and are typically added near the end or in a parallel track to the development process. While this approach allows developers to tackle large and complex software applications to the completion point, they often result in less usable applications. One of the main reasons is that the internal structure and organization of software features is determined by the developers. While this might not be a visible problem in simpler software applications with few features – the type of applications developed in small-scale projects, or simple mobile apps-, it becomes a significant concern with larger, more complex applications such as AI- enabled systems that involves hundreds of features and complex tasks to execute in software engineering (SE).

User centered design focuses on software development using a top-down holistic approach. User-Centered Design (UCD) is typically divided into 5 main phases:

- Phase 1 – User Research

- Phase 2 – High-level design
- Phase 3 – Detailed design
- Phase 4 – Development and development support
- Phase 5 – Testing and Installation support

The main goal of UCD is to start by building a user navigation model first using multiple UCD tools and techniques. The second major step will be to use this navigation model to define and design the application structure using multiple prototyping techniques (Figure 3).

From a user perspective, an application is only what it offers as its user interface (UI). A perfect user mental model is an accurate representation of what the application actually is; an ideal, but rare case. In reality, the user mental model has some resemblance of the actual application model, and it gets better with continuous use, marking the difference between a beginner and an expert user.

Several tools have been devised by the UCD methodologists to execute their development steps. Tools in general allow individuals to manage more complex amount of information, providing leverage over complexity.

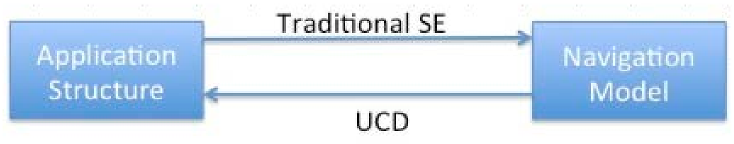


Figure 4. Traditional SE (software engineering) vs UCD

As it can be seen in Figure 4, given the main departure from the traditional software development starting with application structure, the main focus of UCD is on the following different points:

1) Who are the users?
2) What do the users currently do?
3) What do the users want from the new application?

Who are the users?

Two main aspects need to be considered:

i. All related stakeholders ranging from teachers to administrative workers, learners, and technical support should be included when it comes to specifying target audience. These other types of users are researched and identified as “User Roles”.
ii. In a typical situation, teachers have different backgrounds, different experience, and different context to teach in. These differences are looked at using “User Personas” as a common UCD tool that is growing in popularity.

What do the users do?
Several users can be met during brainstorming sessions or design workshops to get their input on their daily work practices. A series of interviews and questionnaires could also supplement the work resulting in potential categories for data analysis.

What do the users want?
This is where the navigation modeling is used as an innovative technique to envision the user needs in terms of a navigation structure that can be translated into an actual application structure.

After successful completion of the User Research phase, the high-level and detailed design phases can start. The following sections provide an overview of the various phases of UCD.

3.1 Phase 1 – User Research

During the User Research phase, focus groups with engineering and computing systems faculty can be organized to understand the course design process used by the participants. At the beginning of each session all participants can be asked to fill an electronic background survey that collect demographic information, primary areas of interest in teaching and research, time spent on teaching, number of courses taught per year (at both undergraduate and graduate levels), and number of new courses developed (both at undergraduate and graduate levels). Participants can also be asked to fill an electronic questionnaire about curriculum design tools that they currently use to create and manage their courses (e.g. preparing syllabi; communicating with learners; developing teaching materials; preparing, assigning, and delivering grades, etc.).

Data collected from the focus groups about course design process can be categorized as inputs, processing and decision-making, and output artifacts.

3.2 Phase 2 – High-level Design

The main goal of high-level design is to plot down schematic ideas and steps into visual graphs and models; an early blueprint. This can be done by investigating different options and provide design alternatives to ensure a broad view before identifying a good design. Doing this early on, at high-level, sketchy, paper-based only, and without going into details could help provide several solution alternatives at a very low cost. The high- level design sketches can be discussed with the users to make sure what they said in unstructured dialogs and vague ideas and imaginations can now be concretely captured in design artifacts for further validation and clarifications. At this stage, there are 2 tools that are most suitable for the development stage:

a) **Navigation Model** is one of the essential methods of design. A significant challenge in complex software is not the contents of each screen, but how the user mentally builds a mental view of how all screens are connected (like a city road map), and how to navigate between hundreds of screens to accomplish their task. In this regard, an effective technique- elastic prototyping- can be used an implementation of a participatory design to help designers and users build a navigation model together, greatly reducing time and effort needed. Figure 2 shows the navigation model for the primary application.

b) **Prototyping (PT)** is extensively used in UCD to visualize and validate all otherwise vague ideas and unclear expectations at low cost and high effectiveness. There exist three main categories of prototyping: Paper (low-level) PT, low-fidelity electronic (medium level) PT, and high-fidelity, detailed PT. Paper prototypes are very inexpensive and help capture several initial ideas and concepts, and validate them. After explaining their needs, users often change their minds when they see them on paper. Therefore, multiple paper PT sessions gives a head start in validating what users actually mean and need. After initial concepts, once design ideas and directions were identified, a medium fidelity prototyping stage can start where a sketchy visualization of key screens without contents are provided to be gradually validated them and added with initial contents.

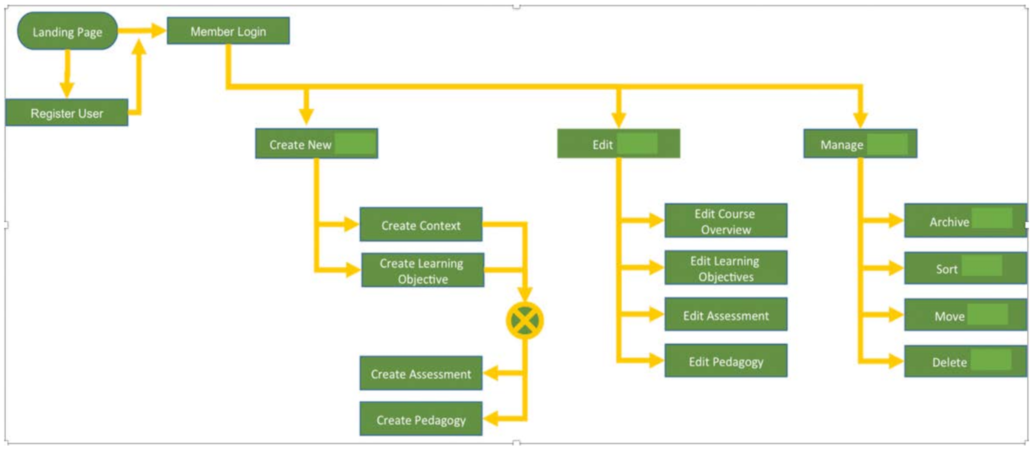


Figure 5: Primary Application Navigation Model

3.3 Phase 3 – Detailed Design

At this stage, the focus is on the main high-level solution, including details from different perspectives such as main application features, auxiliary features, concrete navigation models, detailed screens (all screens, with all contents), menu options, visual and interaction consistency across all screens, exceptions and error messages and recovery, reliability assurances and, help. This phase can proceed in parallel with development phase as more details are uncovered and technical problems arise. User interface mockups can be created with details of various user inputs that will be solicited through the course design process.

As discussed in the literature review, in order to personalize learning contexts and provide adaptive navigation, different models of the adaptive system can be proposed, which are, technically speaking, based on methods of correspondence analysis, linear regression and classification algorithms of learners across a neural network [44-47].

The auto updating system can provide an adaptive navigation, based at the beginning, on static data concerning the learner's profile, evolving with dynamic data including pedagogical activities, learning styles and preferences, learning objectives, use and interaction in social environments, etc. Dynamic data can always be updated automatically, detecting the learner's behavior, activities and performance in his adaptive learning environment. The architecture of this approach with its various elementary components is described in the following Figure 6:

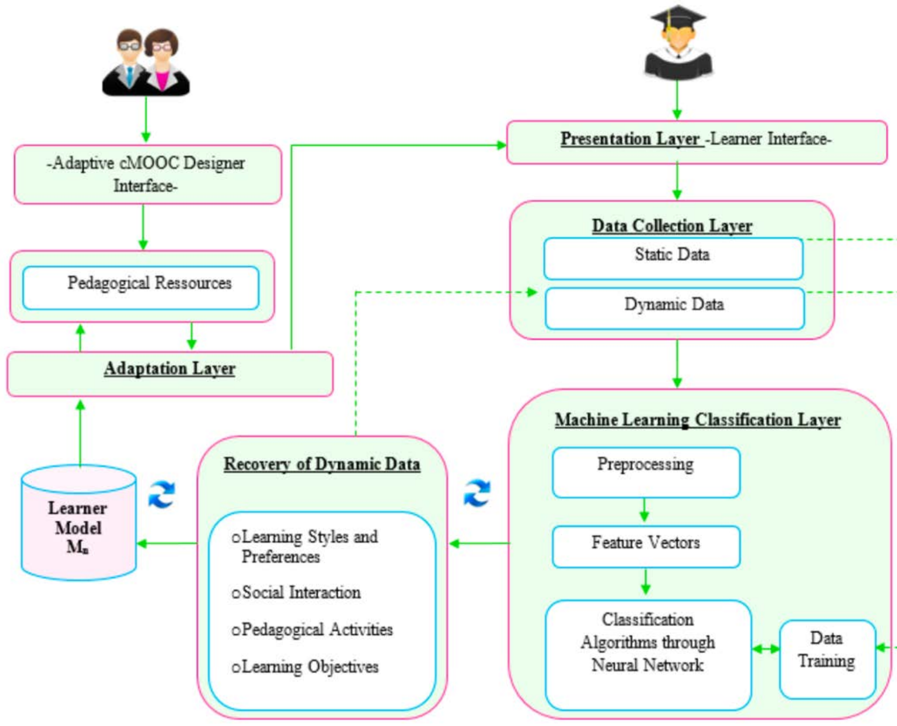


Fig. 6. Overview of the intelligent adaptive system

It includes six layers: a presentation layer (user interface), a data collection layer, a classification layer through a neural network algorithm, recovery of dynamic data and learner's model building and an adaptation layer. After being registered in the platform and filling all the necessary input data, an initial learner profile can be built.

Concerning the classification layer, the system will start with the pre-processing of the collected data, in order to keep the right information for the application of the neural network algorithm. Then the system will extract the characteristics which mostly reflect the learning styles and preferences, learning objectives, and degree of social interaction. On the basis of this data, the system will create vectors for each learner's model in this step. These vectors represent the input of this model which will be automatically updated to the current learner's model. This is very important for the adaptation process, where each learner's profile will be provided by navigation, resources and pedagogical activities adaptive to itself. Also, from this phase, the intelligent system will identify the classes of learners who demand the same preferences, styles and learning objectives to create groups of learners who share the same characteristics (see Figure 6).

3.4 Phase 4 – Development and Development Support

At this stage, different technology options can be examined to experiment with platforms and initial software architecture. As implementation of essential features starts, close collaboration between designers and software engineers (software architects and developers) is essential to ensure the consistency of design and to prevent any deviations.

The Agile software development methodology- Scrum- can be used for the development which entails an iterative and incremental framework for managing product development. A sprint (or iteration) is the basic unit of development in Scrum. The sprint should ideally be restricted to a specific duration, two weeks in case of the AIMOD project. Each sprint can start with a planning meeting where the aim is to define a sprint backlog where the tasks for the sprint are identified and an estimated commitment for the sprint goal is made. Each sprint ends with a sprint review-and- retrospective meeting, where the progress is reviewed and shown to stakeholders and improvements for the next sprints are identified.

4. Analysis of Technologies

A Model–View–Controller (MVC) architecture is suggested as the underlying web application framework. MVC is a software architecture pattern which separates the representation of information from the user's interaction with it. The recommended architecture can be described as follows:

- The foundation is the Java Virtual Machine (JVM).
- There is a separation between the Java language and the JVM.
- The final layer of the architecture is the application layer. This layer follows the Model-View-Controller (MVC) pattern.
- A controller handles requests and creates or prepares the response. A controller can generate the response directly or delegate to a view.
- A controller can have multiple public action methods, each of which maps to a URI.
- A model is a Map that the view uses when rendering information on the web page. The keys within that Map correspond to variable names accessible by the view.

The purpose of analyzing various technologies during this phase of the project is to ensure rapid development for AIMOD and use open-source technologies wherever feasible. Towards this end, an analysis of web application frameworks, version control systems, server-side technologies and client side technologies can be performed.

Table 1. shows some exemplary technologies to inform the final selection of these open-source technologies for development of AIMOD.

Key Architecture Functions	**Possible Solutions**	**Analysis Comments**	**Final Solutions**
Framework	Django, Grails, WebApp2	• Django and WebApp2 are written in Python and have Google App Engine support	

		• Django has request handler, template engine and form processor • WebAp 2 has request handler	
Version Control	Bitbucket, Github, Gitlab, Gitlolite, SVN	• Bitbucket - free private repositories • Github - free public repos + paid private repos • SVN is centralized, Git is decentralized • All work with Unix, Linux and Windows systems	

Databases	SQL, NoSQL, JSON, Google Datastore, MySQL, PostGreSQL	• App Engine Datastore provides a NoSQL schema-less object datastore, with a query engine and atomic transactions • AIMOD data is expected to have numerous relations and hence schema-less store is not being chosen	
Client side scripting	ExtJS, jQuery, JavaScript, CoffeeScript, AngularJS, BackboneJS, HTML5, CSS3, Twitter Bootstrap	• Backbone.js requires more Boilerplate code, but is smaller than Angular.js • ExtJS does not provide good support	
Semantic Web Technologies	OWLite, Protege, Apache Jena	Apache Jena • API for reading, processing and writing RDF data in XML, N-triples and Turtle formats • Rule-based inference engine for reasoning with RDF and OWL data sources • Stores to allow large numbers of RDF triples to be efficiently stored on disk • Query engine compliant with the latest SPARQL	

Licensing		• GPL, MIT, BSD - It is not permissible under the GPL to use GPL in proprietary software while keeping that software closed source • MIT and BSD: Because you cannot restrict others from simply obtaining the source code, selling open-source	

		licensed software as is makes for a difficult proposition	
Cloud/web technologies	Google App Engine, Amazon Web Services (AWS)	• Google App Engine: Free within quota, help and tutorial • AWS: Free usage for a year	

Table 1: Analysis of Technologies for AIMOD development

5. Conclusion

In this paper, a development framework for a AI-enabled adaptive learning systems was presented based on existing studies. The main aim of the adaptive approach is to ‘allow to generate learning paths adapted to the profiles of the learners and according to the pedagogical objectives fixed by the teacher’. (Hssina & Erritali, 2019).

Most of the systems and frameworks that have been designed and proposed are currently in their experimental phases. They have not been tested in practice or adopted in real learning settings. Therefore, the following questions might be worth to reflect upon:

- How might these emerging technologies interact with the existing social contexts of learning?
- What forms of learning will be valued or ignored?
- What evidence is there to support their impact on learning?

Last, but not least, regardless of the type of technology used, the following lessons as learnt in the past should be taken into account:

- AI alone does not transform education or improve learning.
- There can be unintended consequences of AI use in education that are impossible to predict and that stretch far beyond matters of learning.
- Any ‘impacts’ are context specific and tied with socio-technical factors.

REFERENCES

[1] G. C. Furman, “Outcome-Based Education and Accountability.,” *Education and Urban Society*, vol. 26, no. 4, pp. 417–437, 1994.
[2] S. Bansal, O. Dalrymple, A. Gaffar, and R. Taylor, “User Research for the Instructional Module Development (IMOD) System,” in *American Society for Engineering Education Annual Conference (ASEE)*, Indianapolis, IN, 2014.
[3] O. Dalrymple, S. Bansal, A. Gaffar, and R. Taylor, “Instructional Module Development (IMOD) System: A User Study on Curriculum Design Process,” in *Frontiers in Education (FIE)*, Madrid, Spain, 2014.
[4] J. Lazar, J. H. Feng, and H. Hochheiser, *Research methods in human-computer interaction*. John Wiley & Sons Inc, 2009.
[5] W. Lidwell, K. Holden, and J. Butler, *Universal principles of design: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design*. Rockport Pub, 2010.

[6] K. Andhare, O. Dalrymple, and S. Bansal, "Learning Objectives Feature for Instructional Module Development System," presented at the PSW American Society for Engineering Education Conference, San Luis Obispo, California, 2012.
[7] Doshi-Velez F. and B. Kim (2017). "Towards a rigorous science of interpretable machine learning," arXiv:1702.08608.
[8] Dunjko, V., Briegel, H. J. (2017). "Machine learning and artificial intelligence in the quantum domain" arXiv preprint arXiv:1709.02779.
[9] Dzeroski, S., Langley, P., Todorovski, L. (2007). Computational Discovery of Scientific Knowledge, Berlin: Springer.
[10] Popper, K. (1959). The Logic of Scientific Discovery, London: Hutchinson.
[11] Roscher, R., Bohn, B., Duarte, M. F., Garcke, J. (2020), "Explainable Machine Learning for Scientific Insights and Discoveries," in IEEE Access, vol. 8, pp. 42200-42216, and arXiv preprint arXiv: 1905.08883.
[12] Rosenthal, D. (2005). Consciousness and Mind, Oxford: Oxford University Press Salmon, W. (1984). Scientific Explanation and the Causal Structure of the World, Princeton: Princeton University Press. Searle, J. (1980). "Minds, Brains and Programs", Behavioral and Brain Sciences, 3 (3): 417–457, doi:10.1017/S0140525X00005756.
[13] Shevlin, H. (2020, forthcoming). "General intelligence: an ecumenical heuristic for artificial consciousness research?", Journal of Artificial Intelligence & Consciousness.
[14] Vervoort, L., and Blusiewicz, T. (2020b). "Free will and (in)determinism in the brain: a case for naturalized philosophy", THEORIA : International Journal for Theory, History and Foundations of Science 35:3, 345-364.
[15] Wu, T. and M. Tegmark (2019). "Toward an AI Physicist for Unsupervised Learning", Physical Review E 100, 033311, with an open-access version on arXiv: arXiv:1810.10525v4 [physics.comp-ph]
[16] Zheng, D., Luo, V., Wu, J., Tenenbaum, J. B. (2018). "Unsupervised Learning of Latent Physical Properties Using Perception-Prediction Networks", arXiv preprint arXiv:1807.09244.