Gradus Vol 6, No 4 (2019) 128-138
ISSN 2064-8014

GRADUS
GRADUS.KEFO.HU

# THE DEVELOPMENT AND ANALYSIS OF EXTENDED ARCHITECTURE MODEL FOR INTELLIGENT TUTORING SYSTEMS

*Walelign Tewabe Sewunetie [1*], Ghanim Hussein Ali Ahmed [1] and László Kovács[1]*

[1] Department of Information Technology, József Hatvany Doctoral School for Computer Science and Engineering, University of Miskolc, Hungary

***Keywords:***
E-Tutor
ITS architecture model
Ontology
E-Learning.

***Article history:***
Received 14 Nov 2019
Revised 5 Dec 2019
Accepted 10 Dec 2019

***Abstract***

*Intelligent Tutoring Systems (ITS) are computer programs that use learners' knowledge level to providing individualized education. ITS research has successfully delivered systems efficiently supporting one-to-one tutoring. Most of these systems are actively used in real-world settings and have even contributed to changing traditional education curricula. Instructional activities, learning examples, exploring interactive simulations and playing educational games can benefit from individualized computer-based assistance.*

*To enhance ongoing research related to the improvement of tutoring, we present an extended knowledge model including besides the standard modules a common shared database and knowledge-based background, too. The external databases can improve the quality of the behavior models both in tutor and student models. The Python programming language and OWL are efficient tools to combine the ontology management and machine learning functions to develop ITS systems. In this paper, we survey ITS technologies and present a novel extended architecture model for Intelligent e-Tutoring Systems.*

## 1 Introduction

Human beings are continuously attempting to automate mental processes that can be understood and represented by algorithms. The development of IT technologies in education area enabled the widening of the automation of further cognitive activities like Electronic Learning (E-Learning) [1]. E-Learning is the practice of using information and communication technology to generate learning experience that can be formulated, organized and created with full freedom without any boundaries [2]. A new educational phenomenon emerged in the last decade enabled by the E-Learning technology and initiatives known as MOOCs (Massive Open Online Courses) [2]. MOOCs aim at free, massive, online education system comprising open-access and self-learning method courses [2].

* Corresponding author. Tel.: +36 70 584 6503
E-mail address: sewunetie@iit.uni-miskolc.hu

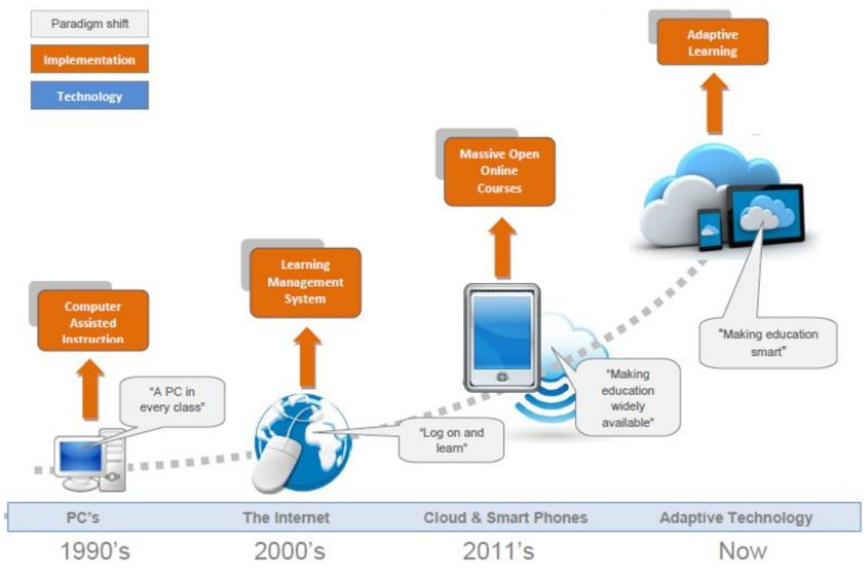


*Figure 1. History of AIED [2]*

Intelligent Tutoring System (ITS) is a computer-based system that aims to offer direct and customized instruction or feedback to learners with personalized guidelines based on their cognitive skills, usually without requiring intervention from a human teacher [3]. Different researchers, designers, and developers define ITSs in different ways. According to Fletcher and Sottilare [4], intelligent tutoring may be viewed as "an effort to capture in computer technology the capabilities and practices of a human instructor who is expert in both the subject matter and one-to-one tutoring". From the earliest days of computers, researchers have struggled to develop ITS that are as effective as human tutors [5]. This paper presents the development and analysis of an extended ITS architectural model.

## 2 Related Techniques

Regarding the ITS architecture, first ITS systems were based on the traditional trinity model containing three basic components: domain, student and tutoring. In 1990, Self John [6] extended the previous model by adding user interface module, Fig. 2 shows the four-component architecture dominating current systems.

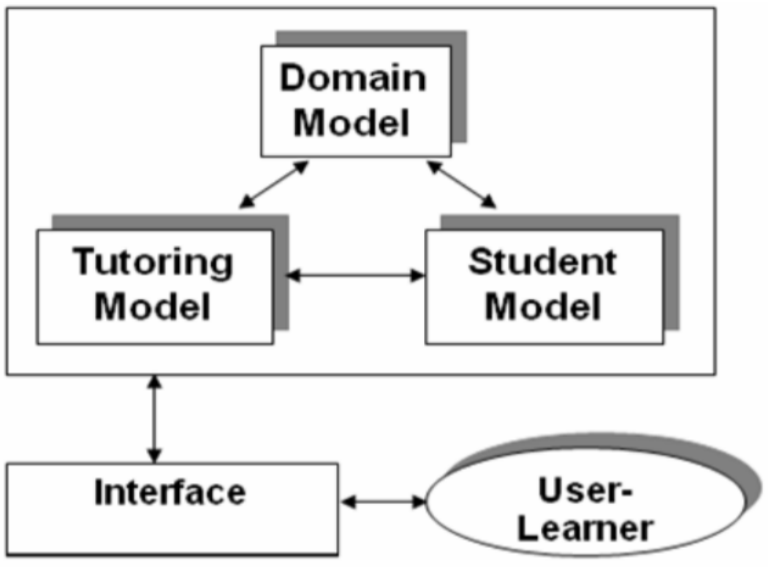


*Figure 2. The four-component architecture [6]*

According to Roger et al [7] each unit of knowledge can be more or less comprehensive and the curriculum can be structured in a complex model according to various structures such as hierarchies, semantic networks, frames, ontology and rules of production. The student's model is to reflect the student's understanding of the subject. It is built up by students ' interaction within an ITS and is used to adapt it and provide sufficient guidance to the individual student.

Regarding technologies, most of the ITS architectures use the Client/Server model. The client / server architecture is also known as a networking computing model, as all requests and services are delivered over a network. There are several techniques to support Client / Server Architecture, including PHP, ASP, Java Servlet and relational databases.

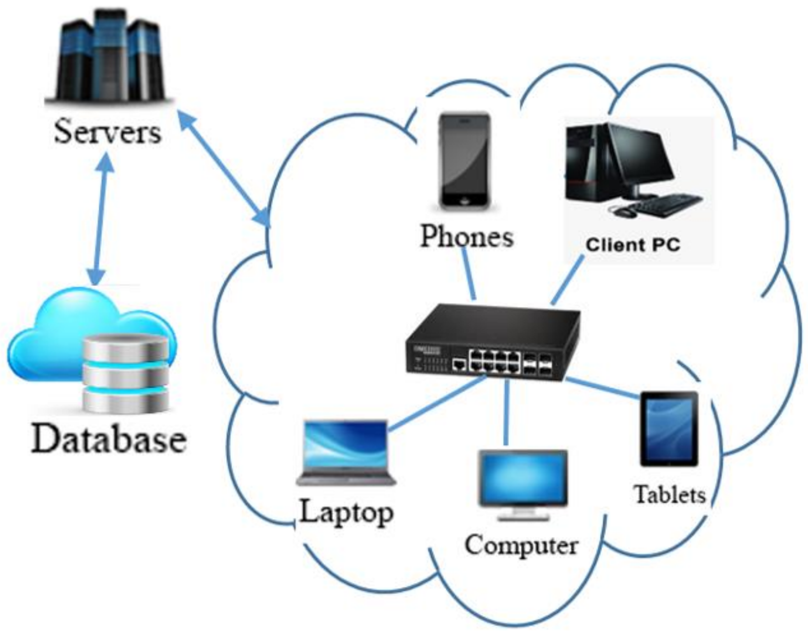


*Figure 3. Client/Server Architecture*

Finally, ITS technologies provides opportunities for experts from countries to collaborate. Collaborations can improve technology transition from more to less developed countries, leading to more widely applied ITSs and to open access for quality education.

## 3 Current ITS Applications

Several publications, technologies (like ontology) and prototype systems (ITS authoring tools) can be found in the literature on current ITS applications. Related to ontology support, Fakoya et al [8] proposed an Ontology-Based Model for E-learning Management System (O-BMEMS), with the objective to increase efficiency and relevancy. They presented ontology including among others course syllabus, teaching methods, learning activities, and learning styles.

In this section, we present some authoring tools having greater influence on development of ITS systems. These tools are widely used ITS communities.

The Authoring Software Platform for Intelligent Resources in Education (ASPIRE) created by the Intelligent Computer Tutoring Group at the University of Canterbury, New Zealand, [9] employs domain experts to create constraint-based tutors by extracting domain knowledge model from device interactions [9].

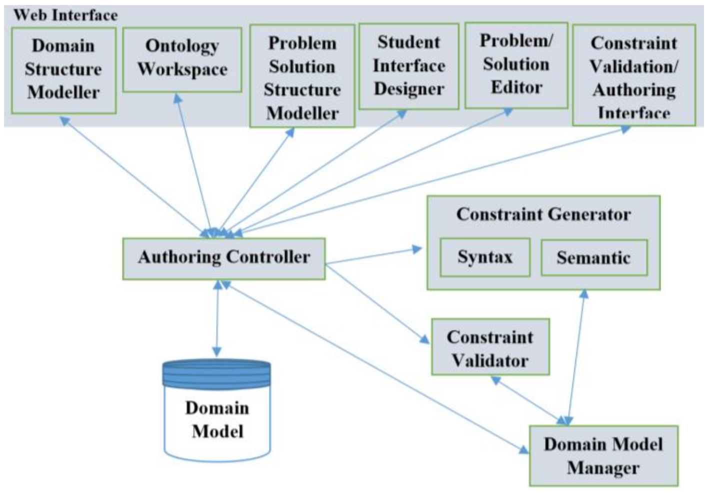


*Figure 4. ASPIRE architecture [9]*

The Extensible Problem Specific Tutor (xPST) [10] is a web authoring tool that facilitates the rapid development of example-tracing tutors on existing interfaces, reducing development time and enabling the interface to be isolated from the tutoring portion. The xPST System consists of the following main components: 1) The xPST Engine, 2) The Presentation Manager, and 3) The Web Authoring Tool.

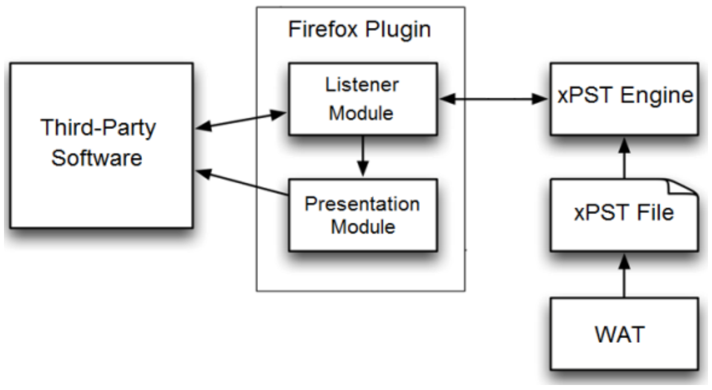


*Figure 5. xPST architecture [10]*

The Generalized Intelligent Framework for Tutoring (GIFT) Authoring Tools are open source, developed by ARL GIFT's user community. GIFT has been created to enhanced self-regulated learning ability and to reduce the time / cost / skill required for ITS authoring [11].

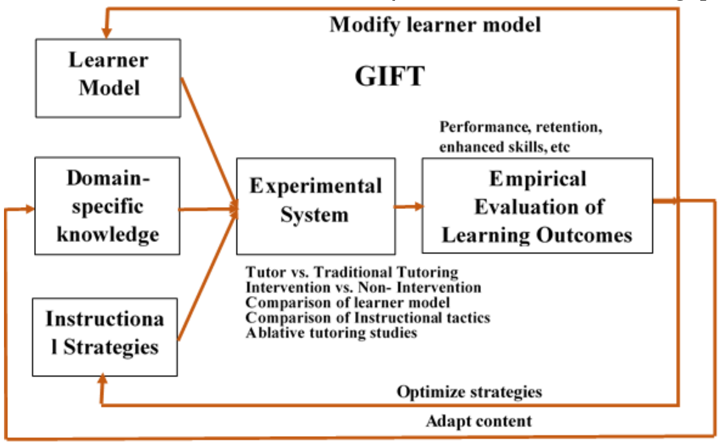

Modify learner model
GIFT
Learner Model
Domain-specific knowledge
Instructional Strategies
Experimental System
Performance, retention, enhanced skills, etc
Empirical Evaluation of Learning Outcomes
Tutor vs. Traditional Tutoring
Intervention vs. Non- Intervention
Comparison of learner model
Comparison of Instructional tactics
Ablative tutoring studies
Optimize strategies
Adapt content

*Figure 6. GIFT architecture [11]*

Intelligent Tutoring System Builder (ITSB) is an authoring tool designed to allow teachers from different fields to develop domain specific ITS modules. The system enables Instructors to add specific smart features to the tutoring system [12].

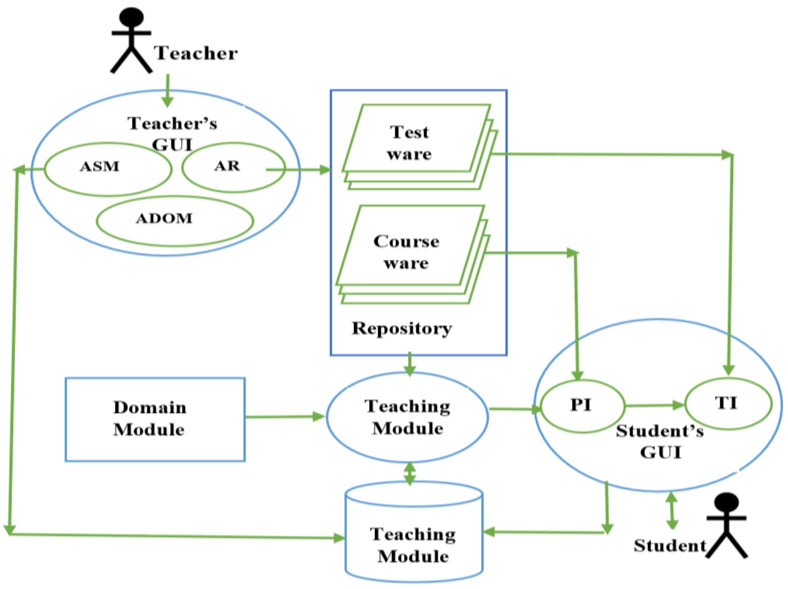


*Figure 7. ITSB Architecture [12]*

The Cognitive Tutor Authoring Tools (CTAT), developed by Carnegie Mellon University, is one of the longest running and most successful toolsets. CTAT allows authors to bind tutoring information elements directly to Graphical User Interface (GUI) elements with little programming effort and to demonstrate template solutions easily [13]. Chris et al [13] defined CTAT as a tool suite for creating online cognitive tutors. CTAT provides personalized support through customized tips and monitors the success of students in providing feedback as they move through a problem.

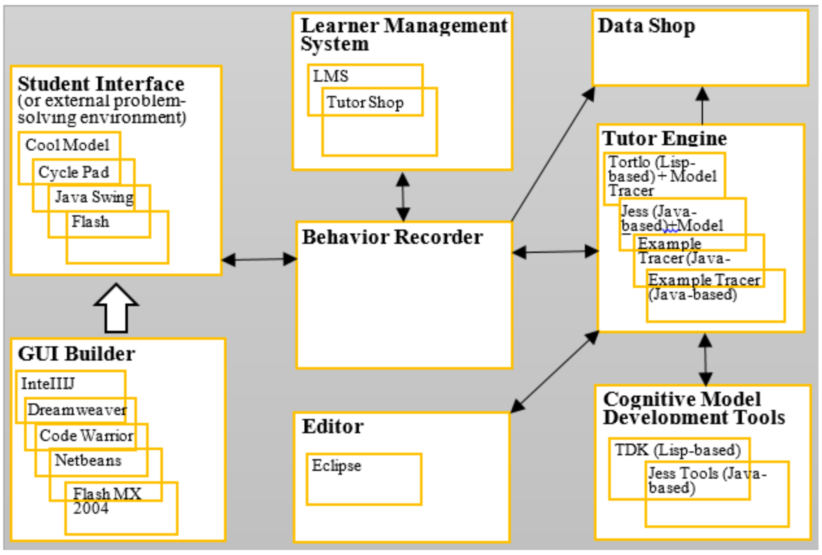


*Figure 8. CTAT architecture [13]*

## 4 The Coming and Challenges of ITS

According to research works [14], only motivated and persistent students generate greater amounts of high-quality data for training of ITS. To increase the performance of the AI educational systems, more realistic student models and better understanding of the pedagogical context are needed. Smart ITS system will increase the possibilities of hard-working students to acquire more knowledge and skills [14].

Most ITS systems developed focus on solving a single domain and they try to create only single course like C++ programming tutor and Java object tutor. The most authoring tools like ITSB, CTAT and GIFT etc use isolated database system, these local databases can provide only limited knowledge background. The limitations of isolated database are: lack of reusability, lack of standardization, lack of flexibility and it has a limited knowledge. In order to overcome the problems mentioned before we are proposing an extended ITS architecture model using shared database.

We expect that the education in the future will use more insensible smart ITS tools. The main building blocks of an extended ITS architecture are shown in Fig 9. The proposed model differs from the standard models in the following aspects.

- ✓ External Knowledge model
- ✓ Explicit knowledge of ontology
- ✓ Extending learner and tutor ML components
- ✓ Using NLP engines
- ✓ Question Generation model.

The proposed extended architecture model includes beside the standard modules a common shared database and knowledge-based background, too.

The benefits of the global database are sharing common understanding of the information structure, reusing the data, and mixing different sources of knowledge. In knowledge management, ontology offers a common vocabulary that can be used to model various domains including the type of object, related concepts, and their properties and relationships. The shared database model may involve external training sets that can be used as input data in different data mining techniques like neural network. The external databases can enhance the quality of the behavior models both in tutor and student models.

Implementation of this architecture is based on a new integration approach that includes existing methodologies and algorithms. The proposed extended ITS architecture is shown in Figure 9.

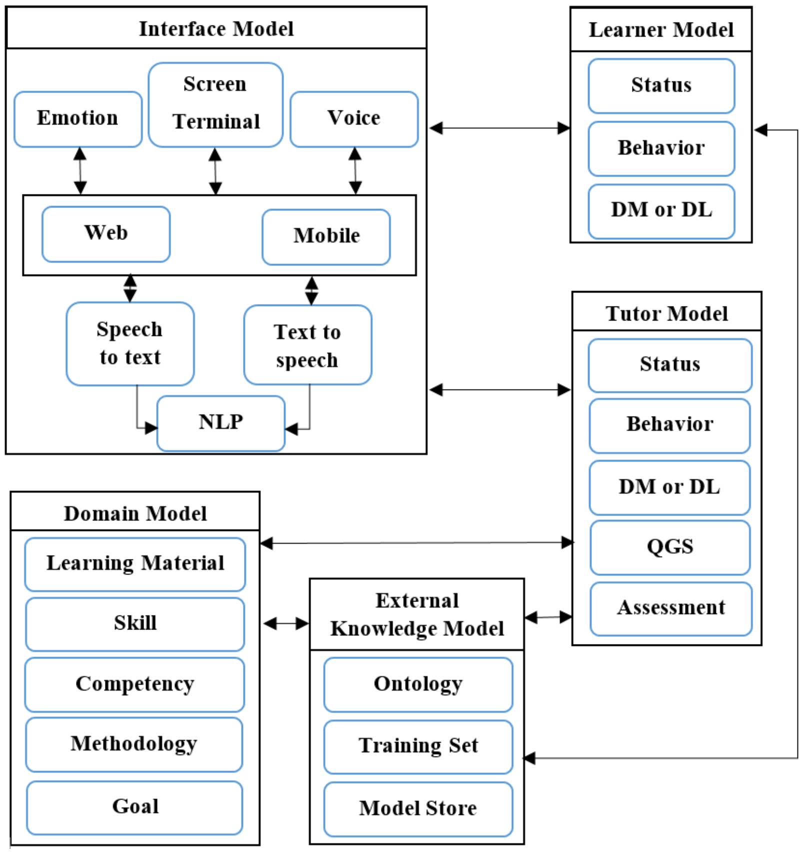


*Figure 9. Extended ITS Architecture*

The status in the student model is to indicate the current academic status of the student which is upgraded in every assessment session. The status can be used to capture the current achievement of a student in a given topic. In the extended ITS architecture we use adaptive behavior model. The knowledge model uses data mining and deep learning techniques for better decision making including classification and clustering the student task.

Considering the functionality of the proposed system we present a sample activity diagram in Fig 10. The figure shows the interaction between students and system during the question generation session. In this system all tutor activities are done by the system, the human teacher might not have direct interaction with the system.

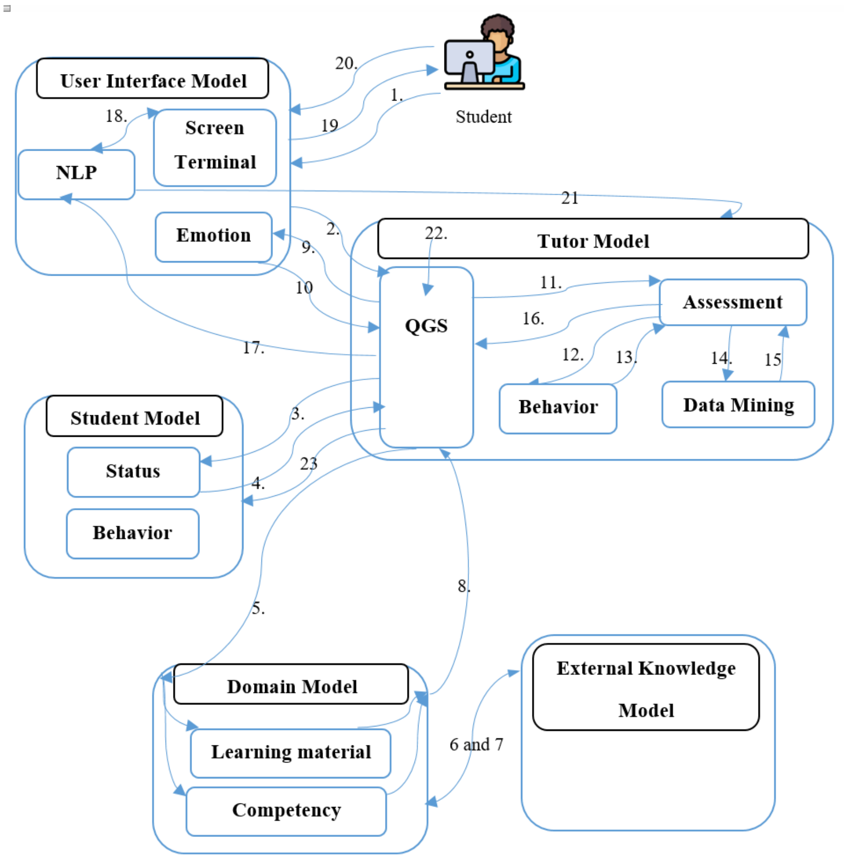


*Figure 10. Sample activity diagram of Extended ITS Architecture*

This section contains the following activity steps:

1. Initiate assessment,
2. Request for generate question,
3. Request Student Status,
4. Response of Student Status,
5. Get the Material and Competency,
6. Retrieve External information if needed,
7. Response learning material and competency,
8. Learning Material,
9. Request Emotion,
10. Response Emotion of Student,
11. Send student model and domain material to the assessment model,
12. Request behavior data for generating abstract questions,
13. Response behavior data for generating abstract questions,
14. Request for abstract question using ML methods,
15. Response abstract question from ML unit to assessment unit,
16. Response abstract questions,
17. Abstract question sends to NLP,

18. Send textual questions to user interface unit,
19. Student receives the question,
20. Student send back the answer,
21. Answer is converted into an abstract form,
22. Validation, evaluation of the answer,
23. Updating the student status and behavioral model.

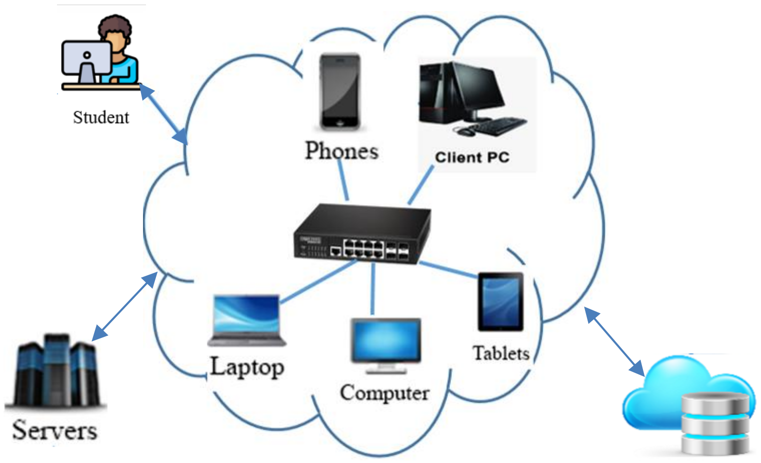


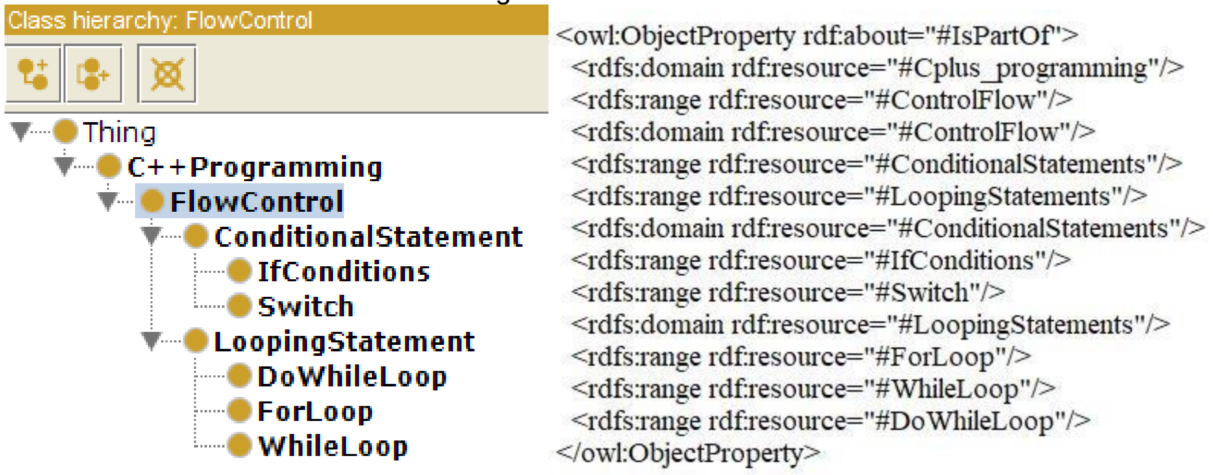
*Figure 11. System Architecture Diagram*

Regarding the implementation of an ontology and Natural Language Processing (NLP) engine Python is one of the most common language used. It is an interpreted, object-oriented, extensible programming language [15], which provides an excellent combination of clarity and versatility in different disciplines. In information science, it offers many modules and package for management and implementing the ontology, data mining and NLP.

Many tools are available for building or managing an ontology. Regarding editing of the ontology by humans Protégé is the most commonly used editor framework which is created at Stanford University [16]. Protégé is a free, open-source software to construct and to update the ontology knowledge base. The tool has features for building, editing and visualizing ontologies, importing and exporting capabilities of different ontology formats.

To build an ontology, first, we use protégé 4.3 environments to define the concepts, which are treated as classes and arrange them in a taxonomic, superclass-subclass hierarchy. Fig 12 shows part of the arrangement of the classes and subclasses of the FlowControl domain in C++ Programming.

The knowledge base domain, considered here, is "C++ Programming", the ontology created consisted of subclasses Flow Control. Figure 13 shows the classes and the subclasses.



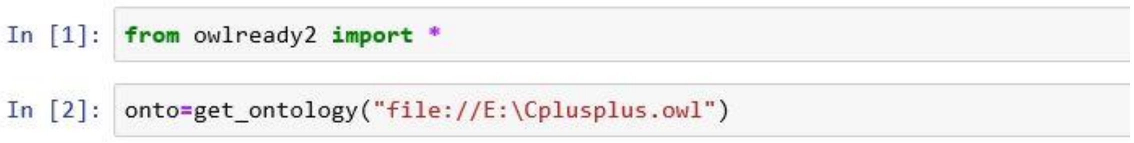
```
Thing
  C++Programming
    FlowControl
      ConditionalStatement
        IfConditions
        Switch
      LoopingStatement
        DoWhileLoop
        ForLoop
        WhileLoop
```

```
<owl:ObjectProperty rdf:about="#IsPartOf">
  <rdfs:domain rdf:resource="#Cplus_programming"/>
  <rdfs:range rdf:resource="#ControlFlow"/>
  <rdfs:domain rdf:resource="#ControlFlow"/>
  <rdfs:range rdf:resource="#ConditionalStatements"/>
  <rdfs:range rdf:resource="#LoopingStatements"/>
  <rdfs:domain rdf:resource="#ConditionalStatements"/>
  <rdfs:range rdf:resource="#IfConditions"/>
  <rdfs:range rdf:resource="#Switch"/>
  <rdfs:domain rdf:resource="#LoopingStatements"/>
  <rdfs:range rdf:resource="#ForLoop"/>
  <rdfs:range rdf:resource="#WhileLoop"/>
  <rdfs:range rdf:resource="#DoWhileLoop"/>
</owl:ObjectProperty>
```

Figure 12. Class hierarchies for 'C++ Programming' in Protégé

There are many languages used for building ontology domain model like XML, RDF, RDF(s), OWL, and DAML+OIL, the most common used is Web Ontology Language (OWL) [17]. According to W3C [18], OWL is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs, e.g., to verify the consistency of that knowledge or to make implicit knowledge explicit

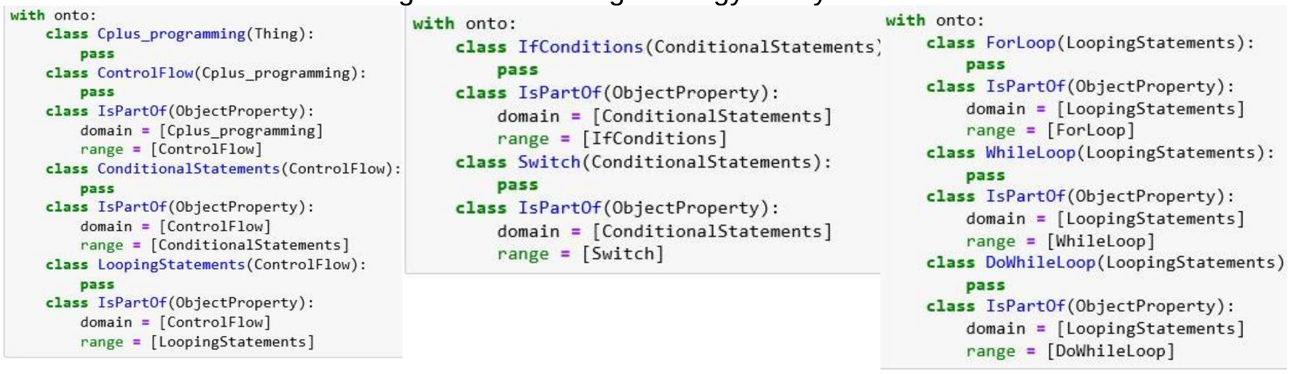
Python object model can be integrated with an OWL ontologies using python_module called Owlready2. Owlready2 is used to obtain transparent access to ontologies, manipulate the classes and the individuals object properties, data properties, annotations, property domains and ranges, constrained datatypes, disjoints and class expressions (such as intersections, unions, property value restrictions etc). Fig 13 shows a simple example, to create an ontology.



```
In [1]: from owlready2 import *

In [2]: onto=get_ontology("file://E:\Cplusplus.owl")
```

Figure 13. Creating ontology on Python

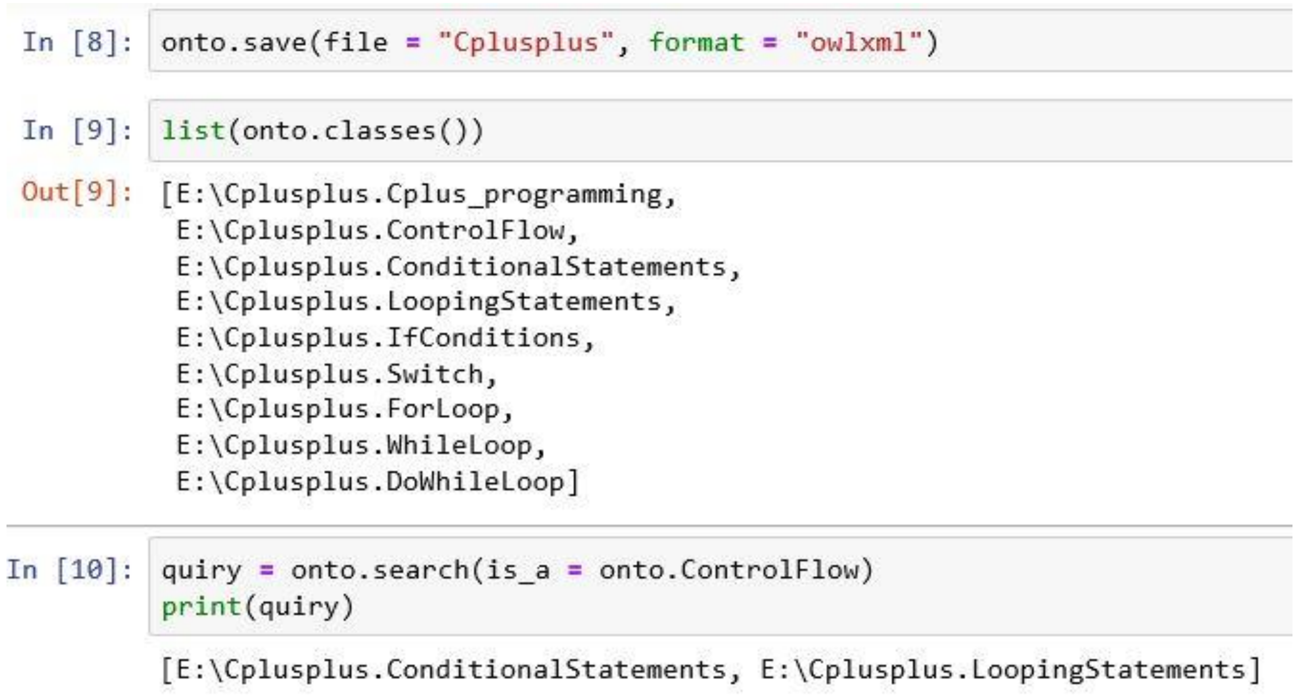


```
with onto:
    class Cplus_programming(Thing):
        pass
    class ControlFlow(Cplus_programming):
        pass
    class IsPartOf(ObjectProperty):
        domain = [Cplus_programming]
        range = [ControlFlow]
    class ConditionalStatements(ControlFlow):
        pass
    class IsPartOf(ObjectProperty):
        domain = [ControlFlow]
        range = [ConditionalStatements]
    class LoopingStatements(ControlFlow):
        pass
    class IsPartOf(ObjectProperty):
        domain = [ControlFlow]
        range = [LoopingStatements]
```

```
with onto:
    class IfConditions(ConditionalStatements)
        pass
    class IsPartOf(ObjectProperty):
        domain = [ConditionalStatements]
        range = [IfConditions]
    class Switch(ConditionalStatements):
        pass
    class IsPartOf(ObjectProperty):
        domain = [ConditionalStatements]
        range = [Switch]
```

```
with onto:
    class ForLoop(LoopingStatements):
        pass
    class IsPartOf(ObjectProperty):
        domain = [LoopingStatements]
        range = [ForLoop]
    class WhileLoop(LoopingStatements):
        pass
    class IsPartOf(ObjectProperty):
        domain = [LoopingStatements]
        range = [WhileLoop]
    class DoWhileLoop(LoopingStatements)
        pass
    class IsPartOf(ObjectProperty):
        domain = [LoopingStatements]
        range = [DoWhileLoop]
```

Figure 14. Snapshot of for building classes and subclasses



```
In [8]: onto.save(file = "Cplusplus", format = "owlxml")

In [9]: list(onto.classes())

Out[9]: [E:\Cplusplus.Cplus_programming,
         E:\Cplusplus.ControlFlow,
         E:\Cplusplus.ConditionalStatements,
         E:\Cplusplus.LoopingStatements,
         E:\Cplusplus.IfConditions,
         E:\Cplusplus.Switch,
         E:\Cplusplus.ForLoop,
         E:\Cplusplus.WhileLoop,
         E:\Cplusplus.DoWhileLoop]

In [10]: quiry = onto.search(is_a = onto.ControlFlow)
         print(quiry)

         [E:\Cplusplus.ConditionalStatements, E:\Cplusplus.LoopingStatements]
```

Figure 15. Snapshot of for displaying classes and subclasses

The python programming language is efficient tool to combine the ontology management and machine learning functions to develop ITS systems.

## 5 Conclusion

Education is UNESCO’s top priority because it is a basic human right and the foundation on which to build peace and drive sustainable development. In education, AI has begun producing new teaching and learning solutions that are now undergoing testing in different contexts. Using ITSs for education sector is essential to achieve all these goals, which aims to “ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.”

Intelligent E-Learning and Tutoring Systems are the key technologies in supporting educational activities. In this paper, we present a survey on current status, challenges and opportunities of ITS architectures. Based on the common requirements we have developed an extended Intelligent Tutoring System architecture (ITS). The proposed architecture includes beside the standard modules a common shared database and knowledge-based background, too. Benefits of the shared database are to share common understanding of the structure of information, to reuse the data and to mix different knowledge sources. Instead of the traditional human generated behavior models machine learning can provide more flexible behavior models using the available training bases. Another key component in ITS architectures is the module for automatize assessment activities. Data mining tool can be used to generate the questions and to evaluate the correctness of results.

The benefit of using ontology in ITS is reusability, standardization, flexibility and knowledge sharing. For the implementation of proposed ITS architecture we used OWL and Python.

## Acknowledgment

The described study was carried out as part of the EFOP-3.6.1-16-00011 “Younger and Renewing University – Innovative Knowledge City – institutional development of the University of Miskolc aiming at intelligent specialization” project implemented in the framework of the Szechenyi 2020 program. The realization of this project is supported by the European Union, co-financed by the European Social Fund.

## References

[1] L. Samuelis, "Notes on the Components for Intelligent Tutoring Systems Tutoring Systems," *Acta Polytechnica Hungarica* , vol. 4, no. 2, 2007.

[2] Zlatko Bezhovski, Subitcha Poorani, "The Evolution of E-Learning and New Trends," *Information and Knowledge Management,* vol. 6, no. 3, 2016.

[3] E. b. R. A. S. e. a. Sinatra, Design Recommendations for Intelligent Tutoring Systems Volume 6 Team Tutoring, Orlando, Florida: US Army Research Laboratory (ARL), United States of America First Printing, August 2018.

[4] Robert Sottilare, et al, Design Recommendations for Intelligent Tutoring Systems, Florida.: U.S. Army Research Laboratory, 2015.

[5] K. VanLehn, "The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems, Educational Psychologist," *Routledge* , vol. 46, no. 4, pp. 197-221, 2011.

[6] J. Self, "Theoretical Foundations for Intelligent Tutoring Systems," *Journal of Artificial Intelligence in Education,* vol. 4, no. 1, pp. 3-14, 1990.

[7] J. B. R. M. Roger Nkambou, "Introduction: What Are Intelligent Tutoring Systems, and Why This Book?," *Studies in Computational Intelligence.,* 2010.

[8] A. S. a. O. P. Fakoya Tunde, "ONTOLOGY-BASED MODEL FOR E-LEARNING MANAGEMENT SYSTEM," *IJCSI International Journal of Computer Science Issues,* vol. 12, no. 3, pp. 118-126, 2015.

[9] Mitrovic, A., Martin, et al, "An authoring system and deployment environment for constraint-based tutors," *International Journal of Artificial Intelligence in Education,* vol. 19, no. 2, pp. 155-188, 2009.

[10] S. B. S. B. &. K. S. Gilbert, "The Extensible Problem-Specific Tutor (xPST): Evaluation of an API for tutoring on existing interfaces.," *Artificial Intelligence in Education.* , 2009.

[11] R. Sottilare, "Adaptive Intelligent Tutoring System (ITS) Research in Support of the Army Learning Model," Army Research Laboratory, 2013.

[12] S. S. A. Naser., "ITSB: An Intelligent Tutoring System Authoring Tool.," *Journal of Scientific and Engineering Research. Research Article,* 2016.

[13] G. G. S. J. a. C. Z. Chris Feng, "Authoring Tools for Easy Creation of Adaptive Tutoring.," *BHCI Capstone Project. Spring ,* 2018.

[14] V. Kolchenko, "Can Modern AI replace teachers? Not so fast! Artificial Intelligence and Adaptive Learning: Personalized Education in the AI age," *HAPS Educator,* vol. 22, no. 3, pp. 249-252, December 2018.

[15] "What is Python? Executive Summary," Python, [Online]. Available: https://www.python.org/doc/essays/blurb/. [Accessed 11 11 2019].

[16] R. K. E. P. T. A. K. A. D. J. Emeka Oguejiofor, "Intelligent tutorin System" an ontology-based approach," *International Journal of IT in Architecture, Engineering and Construction,* vol. 2, no. 2, pp. 115-128, 2004.

[17] L. Razmerita, "An Ontology-Based Framewrok for Modeling User Behavior: A Case Study in Kowledge Management," *System and Human, IEEE,* vol. 41, no. 4, pp. 772-783, 2011.

[18] A. R. S. H. I. H. Ian Horrocks, "Web Ontology Language (OWL)," W3C, 11 12 2012. [Online]. Available: https://www.w3.org/2001/sw/wiki/OWL. [Accessed 13 11 2019].

[19] L. H. Marian Babik, "Deep Integration of Python with Web Ontology Language".