University of Miskolc

FACULTY OF MECHANICAL ENGINEERING AND INFORMATICS

PHD DISSERTATION

AUTHOR:

Hussein Ali Ahmed Ghanim
MSc in Information Technology

József Hatvany Doctoral School of

Information Science, Engineering and Technology

Title of the dissertation

Ontology Domain Knowledge Model for E-tutoring Systems

Research Area
Applied Computer Science

Research Group
Data and Knowledge Bases, Knowledge Intensive Systems

HEAD OF DOCTORAL SCHOOL:

Prof. Dr. Jenő SZIGETI

ACADEMIC SUPERVISOR:

Prof. Dr. László KOVÁCS

Miskolc
2024

Declaration of Authorship

The author hereby declares that this thesis has not been submitted, either in the same or in a different form, to this or any other university to obtain a PhD degree.

The author confirms that the submitted work is his own, and the appropriate credit has been given where reference has been made to the work of others.

Author's declaration

I, the undersigned, Hussein Ali Ahmed Ghanim, declare that I have prepared this doctoral dissertation and have used only the sources provided.

All parts that I have taken from another source, either directly or in the same content but paraphrased, are clearly marked with the source.

Miskolc, 2022. September.

[illegible]

Hussein Ali Ahmed Ghanim

Summary

The concept of learning is a technique that differs from each learner to another regarding pedagogical diversity. Therefore, E-learning Systems and their learning materials should support and be tailored according to the diverse differences of individual learners. E-learning is an environment that uses the current technologies, like information and communication technologies, to enhance teaching and learning processes. Technological and pedagogical sides are two core angles of E-learning. The technology side incorporates platforms and infrastructures to provide learners with learning and teaching materials. The pedagogical side manages the learning and teaching contents to develop the knowledge and experience of the learners. The E-learning Framework is a critical support tool for digital learning systems to improve the efficiency of the teaching process, including students and teachers. The existing E-learning environments typically lack the metacognitive awareness levels, personalized tutoring, and time management skills and do not always meet the learners' expectations as required. However, this problem can be solved using education software like an E-tutoring system. In addition, E-learning platforms and their technologies are increasingly evolving and becoming widespread in the educational environment due to different learning benefits achieved through learning without place or time limits.

Nowadays, the use of technologies for enhancing education and learning is rapidly growing, and it can improve the learning process by applying modern learning technologies such as E-tutoring Frameworks. An E-tutoring systems are software applications that deliver direct customized learning using artificial intelligence techniques to provide feedback to the learner. E-tutoring systems can enhance learning and teaching efficiency, mainly for personalized adaptive self-learning. The E-tutoring system has built-in modules to track students' performance and personalize learning according to an adaptation of students' learning styles, knowledge levels, and proper teaching techniques. The E-tutoring system usually has four modules: teaching module, knowledge module, learner module, and learner interface, which they use to design and develop these systems. The teaching module is used to model the behavior of the tutor. The knowledge module includes the semantic definition of the teaching material for the knowledge domain, usually using a semantic network structure. Besides the core knowledge module elements, the working of E-tutoring applications is also based on specific external components, such as fact knowledge bases, ontology databases, reasoning engines, or web services. The learner module is used to model the behavior of the learners, and the interface module is used to implement an intelligent human-oriented interface to the system. E-tutoring platforms have successfully delivered systems efficiently facilitating one-to-one learning and provides adequate aid to the learners and becomes increasingly important for personal and collaborative learning. Thus, there has been significant interest in adopting E-tutoring to facilitate learning processes and enhance learners' performance.

Ontology as a technology has been studied in many areas and is being used in several fields and have used to handle the problems of interoperability in teaching materials, modeling and enriching education resources, personalized learning, and content recommendations in the educational context. For the purpose of enhancing the learning and teaching processes a model is developed in

3 ways. These models are the ontology-based model to support the learning process in LMS, the ontology-based Knowledge domain model for the IT domain in E-tutoring systems, and the ontology-supported domain knowledge module for an E-tutoring system. The main goals of these models are to enhance the learning and teaching process, support recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of new material.

Ontology-based model to support the learning process in LMS is a novel model to improve the E-learning frameworks. In the suggested framework, a domain ontology that describes the learning knowledge is built. and the developed domain ontology can cover the entire learning process. The selected framework concentrated on the ontology model of the learning process for conceptualizing knowledge structures, such as learning courses. It also demonstrated the created course ontology and learning process ontology in detail. In addition, three layers of the learning process are considered. The top layer represents a general framework of the learning process, the conceptual model layer describes the framework of the actual process, and the course ontology model includes the knowledge unit of the learning process. As a result, the model can solve the problems of current E-tutoring Systems. Therefore, the presented model can manage to develop and design E-tutoring frameworks for different domains. In addition, it can achieve the characteristics of standardization, reusability, flexibility, and open knowledge. Using and applying this model can avoid the problem of isolated knowledge bases. The constructed ontology can be used in the future to control adaptive intelligent E-tutoring frameworks.

This research also represented a domain knowledge model for an E-tutoring system that enables knowledge to be stored in a way separated from the domain of interest and assists in keeping transfer and prerequisite knowledge relationships. This innovative technique is helpful for the students in improving their learning progress. In addition, this research introduced a domain knowledge model for an E-tutoring system to support the way of teaching and learning process. In this model, two types of ontologies are introduced, general concepts of domain knowledge ontology and specific domain knowledge ontology. This solution represents the knowledge to be learned, delivers input to the expert model, and eventually provides detailed feedback, selects problems, generates guidance, and supports the student model.

To enhance ongoing research related to the improvement of E-tutoring, this research work present an ontology domain knowledge model that includes a common shared knowledge base besides the standard modules and a knowledge-based background. The shared knowledge bases can improve the behavior model's quality for both the tutor and student models.

Furthermore, this research also introduced a formal logical ontology domain knowledge module for an E-tutoring system. This model supports many features, including reusability, shareability, flexibility, and standardability. In addition, this module also allows knowledge to be stored in a well-defined form and assists the storage of transfer and prerequisite knowledge relationships. The introduced knowledge domain module is designed in two ways the general concepts domain knowledge module and a specific domain knowledge module ontology. This innovative technique is helpful for students in enhancing their learning progress.

Implementing an effective and efficient ontology storage system improves application performance and enables semantic knowledge retrieval. Three model storage systems selected and compared for integrating the ontology model with an E-tutoring framework. The models are OWL ontology model, Jena Fuseki model, and relational model. Comparing different ontology storage models with distinct features allows us to select a proper storage structure for high-performance applications. As a result, the investigation showed that Jena Fuseki and relational model have a minimum execution time which means that Jena Fuseki and relational model are the best model storage systems compared with OWL ontology file. But the relational model storage needs more space and does not provide automatic reasoning. So, Jena Fuseki is the best candidate for model storage and query.

In this research an assessment module is developed for evaluating student knowledge levels in E-tutoring Systems using the intuitionistic fuzzy logic technique. In addition, a prototype E-tutoring framework is created and integrated with the ontology domain knowledge model. The goal is to enhance the learning and teaching process.

Combining the domain knowledge model with E-tutoring systems can enhance the quality of intelligent problem-solving. Also, it will be possible to reuse the knowledge domains. Finally, the domain knowledge model for E-tutoring systems can improve the teaching and learning process, support recommendations, generate hints, and automatically support the generation of problems and solutions.

Acknowledgments

This dissertation becomes a reality because of the help and support of people around me; it was also a result of a lot of effort and challenging work during the past four years.

Above all, I would like to thank my supervisor, Prof. Dr. László Kovács for his continuous support, direction, and encouragement over the past years; this dissertation would not have been possible without him, I would also like to thank everyone in the computer science department.

Furthermore, I am incredibly grateful to many of my colleagues in our department for their help and support in organizing the events associated with the doctorate. Finally, I would like to thank my closest friends, my family, and my children (El Rashid, Ebaid, Nusaiba and Ali). Special thanks to my wife, Afaf Ebaid Mohammed Fadoul for her support and standing beside me step by step to reach my goal.

Hussein Ali Ahmed Ghanim

Table of Contents

Declaration of Authorship I
Author's declaration I
Summary II
Acknowledgments V
Table of Contents VI
List of Abbreviations IX
List of Figures X
List of Tables XII
Chapter 1 Introduction 1
1.1 Problem Domain 1
1.2 Research Goals 2
1.3 Methodology 3
1.4 Dissertation Guide 3
Chapter 2 General Background 5
2.1 Introduction 5
2.2 E-learning and Learning Management Systems 7
2.3 Intelligent Tutoring System/ E-tutoring System 10
2.4 Pedagogy 11
2.5 Ontology 12
2.6 Ontology Model in E-learning Domain 14
2.7 Domain Knowledge Models in E-tutoring systems 15
2.8 Domain knowledge model 16
2.9 E-tutoring Systems Architectures, Models, and Components 16
2.9.1 E-tutoring Systems Architectures 17
2.9.2 E-tutoring Systems Component 22
2.9.3 The current E-tutoring System solutions and models 25
2.10 Summary 28
Chapter 3 Formal Logical Ontology Model for an E-tutoring System 30
3.1 The Ontology Domain Knowledge Model for Teaching Materials 30
3.2 Knowledge Model 31
3.3 Knowledge Representation Scheme 32
3.4 Evaluation of existing knowledge models 37
3.5 The Proposed E-tutoring System Architectures 38
3.5.1 The Extended ITS Architecture Model 38

3.5.2 The Adaptive E-tutoring System Architecture Model 39
3.6 The Proposed Ontology Models 40
3.6.1 Ontology-based Model to Support Learning Process in LMS 40
3.6.2 Ontology-based Domain Knowledge Model for IT Domain in E-tutor Systems............ 43
3.6.3 Ontology Supported Domain Knowledge Module For E-Tutor Systems 45
3.7 Formal Description of the Knowledge Model 47
3.7.1 Knowledge Units Description 48
3.7.2 Knowledge Slots Description 49
3.7.3 Task Units Description 49
3.7.4 Material Units Description 51
3.7.5 Rule Units Description 52
3.7.6 Educational Units Description 53
3.7.7 Specialization/Generalization Relationship 54
3.7.8 Components Relationship 54
3.7.9 Associations Relationship 55
3.7.10 Dependency graph Relationship 56
3.8 The Proposed Ontology Domain Knowledge Model Concepts 57
3.9 Conceptual knowledge representation of the proposed ontology model 57
3.10 Ontological Model of the Domain Knowledge Model 58
3.11 General Form of the Ontology Domain Knowledge Model Components 60
3.12 Summary 62
Chapter 4 Assessment Module 64
4.1 Assessment Module 64
4.2 Domain Model 66
4.3 Student Knowledge Model 66
4.4 Uncertainty of the Knowledge 67
4.5 Student Assessment Using Atanassov's Intuitionistic Fuzzy Technique 68
4.6 Knowledge Level Representation 69
4.6.1 Calculation of the competency levels 69
4.6.2 Example Using Intuitionistic Fuzzy Logic Values 71
4.7 Test Result 73
4.8 Summary 74
Chapter 5 Storage model for the Ontology Model of an E-tutoring System............ 76
5.1 Model Storage Systems 76
5.1.1 Domain knowledge model using ontology OWL file 77

5.1.2 Domain knowledge model using Jena Fuseki model 78
5.1.3 Domain knowledge module using a relational data model 80
5.2 Implementation and Test of the selected model storage systems 81
5.2.1 Test of the domain knowledge module using an ontology OWL file 82
5.2.2 Test the domain knowledge module using Apache Jena Fuseki 83
5.2.3 Test the domain knowledge module using a relational data model 83
5.3 Test Result 84
5.4 Summary 87
Chapter 6 Functionality of the Prototype System 89
6.1 Case Study in the Domain of Teaching Programming 89
6.2 Loop Structures Knowledge Units Case Study 90
6.2.1 Knowledge Units (K-Units) Concepts 90
6.2.2 Knowledge Slots (K-Slots) in loop structures 91
6.2.3 Material Units (M-Units) for loop structures 92
6.2.4 Educational Units (E-Units) Concepts 92
6.2.5 Rule Units (R-Units) in loop structures 92
6.2.6 Task Units (T-Units) in loop structures 94
6.3 Integration of an ontology model with the E-tutoring framework 95
6.4 Implementation of The Ontology Domain Knowledge Model 96
6.4.1 Ontology Construction Using Python and Owlready2 module 98
6.4.2 Application Framework Using Python Flask Framework 101
6.5 Testing and evaluation 103
6.6 Summary 105
General Summary 107
Chapter 7 Conclusion 111
7.1 Contribution 111
7.2 Future work 112
Author’s Publications 114
Publications Related to the Dissertation 114
Other Publications Journal Articles in non-Q Ranking 114
Reference 116

List of Abbreviations

ICTs:	Information and Communication Technologies
ITS:	Intelligent Tutoring Systems
DM:	Domain Model
DK:	Domain Knowledge
LMS:	Learning Management Systems
KE:	knowledge Engineering
KBS:	Knowledge-Based Systems
KRR:	Knowledge Representation and Reasoning
EMES:	E-learning Management Electronic System
IT:	Information Technology
RDF:	Resource Description Framework
RDFS:	Resource Description Framework Scheme
O-BMEMS:	Ontology-Based Model for E-learning Management System
WBES:	Web-Based Educational Systems
TM4L:	Topic Maps for E-Learning
ELMS:	Electronic Learning Management System
SW:	Semantic Web
KR:	Knowledge Representation
DLV:	Deductive DataLog System
ASPIRE:	Authoring Software Platform for Intelligent Resources in Education
xPST:	Extensible Problem-Specific Tutor
CTAT:	Cognitive Tutor Authoring Tools
GIFT:	Generalized Intelligent Framework for Tutoring
ITSB:	Intelligent Tutoring System Builder
LPs:	Learning Processes
AI:	Artificial Intelligence
ORM:	Object-Relational Mapper
API:	Application Programming Interface
OMDKM:	Ontological Model of the Domain Knowledge Model
SPARQL:	SPARQL Protocol and RDF Query Language
API :	Application Programming Interface
ARQ	A SPARQL Processor for Jena
XML:	eXtensible Markup Language
HTTP:	Hypertext Transfer Protocol
TDB	Triple Databased
SWRL:	Semantic Web Rule Language

List of Figures

Figure 2.1: Historical evolution of education technology
Figure 2.2: The four-component architecture
Figure 2.3: The basic architecture of an E-tutoring System
Figure 2.4: The proposed extension architecture
Figure 2.5: ASPIRE architecture
Figure 2.6: xPST architecture
Figure 2.7: Protus architecture
Figure 2.8: CTAT modular architecture
Figure 2.9: GIFT architecture
Figure 2.10: ITSB architecture
Figure 2.11: SeisTutor architecture
Figure 3.1: Semantic network representation example
Figure 3.2: Frame-based representation example
Figure 3.3: knowledge graph representation example
Figure 3.4: Ontology-based representation example
Figure 3.5: Rule-based representation example
Figure 3.6: Case -based representation example
Figure 3.7: Belief networks representation example
Figure 3.8: Extended ITS Architecture
Figure 3.9: Adaptive Architecture
Figure 3.10: Top layered architecture
Figure 3.11: Conceptual model layers
Figure 3.12: Course ontology model
Figure 3.13: The ontology-based domain knowledge model schema
Figure 3.14: The domain knowledge module for E-tutoring frameworks
Figure 3.15: The suggested domain knowledge module
Figure 3.16: Domain knowledge module sample
Figure 3.17: Knowledge units example
Figure 3.18: Knowledge slots example
Figure 3.19: Task units example
Figure 3.20: Material units example
Figure 3.21: Educational units example
Figure 3.22: Specialization/Generalization relation example
Figure 3.23: Components relation example
Figure 3.24: Association relation example
Figure 3.25: Dependency relation example
Figure 3.26: Ontology model schema of teaching materials

Figure 3.27: Conceptual knowledge model of teaching materials
Figure 3.28: Ontology knowledge model schema
Figure 4.1: knowledge model of the student
Figure 4.2: Student task model in an E-tutoring framework
Figure 4.3: Student knowledge level status for a given knowledge unit
Figure 4.4.a: Student knowledge level progress
Figure 4.4.b: Student results
Figure 4.5.a: knowledge level progress distribution
Figure 4.5.b: The Results of knowledge units
Figure 5.1: Taxonomy of ontology storage approaches
Figure 5.2: Apache Jena Fuseki architecture
Figure 5.3: Triple stores structure
Figure 5.4: The specific domain knowledge module of teaching materials
Figure 5.5: Logical knowledge model of teaching materials
Figure 5.6: Test query performance for small database
Figure 5.7: Test query performance for large database
Figure 5.8: Query execution time for small database
Figure 5.9: Query execution time for large database
Figure 5.10: Distribution of the model storage systems for small database
Figure 5.11: Distribution of the model storage systems for large database
Figure 5.12: Distribution of the storage models for small and large database
Figure 5.13: Size on disk of the model storage systems
Figure 6.1: Loop structures knowledge units
Figure 6.2: Owlready general architecture for Python
Figure 6.3: the consistency of the knowledge domain model ontology
Figure 6.4: The interaction of the student with the prototype framework
Figure 6.5: The main E-tutoring framework interface
Figure 6.6: The E-tutoring framework login interface
Figure 6.7: The E-tutoring framework registration interface
Figure 6.8: The E-tutoring framework dashboard interface
Figure 6.9: Snapshot of the initial progress
Figure 6.10: Snippets of practice task assessment
Figure 6.11: Snippets of the task assessment result
Figure 6.12: Snippets of the result interface
Figure 6.13: Assessment ranking
Figure 6.14: Study aid ranking
Figure 6.15: knowledge level progress

List of Tables

Table 3.1: Comparison of the knowledge representation models
Table 3.2: Relationships of the domain knowledge model Class
Table 4.1: The expected competency level
Table 4.2: knowledge level progress (status)
Table 5.1: Sample triple stores structure
Table 5.2: Comparison of the model storage system
Table 5.3: Test query performance for small database
Table 5.4: Test query performance for large database
Table 6.1: Relationship between terms in object-oriented programming and the formal ontologies

Chapter 1 Introduction

Intelligent Tutoring System (ITS) is a computer system that uses artificial intelligence techniques to improve and customize automation in education. ITS differs from other educational systems because it uses a knowledge base to guide the pedagogical process. This tends to promote the mastery of the student's domain knowledge by managing the generation of new problems, topics, and instruction / feedback. So that the development of an ontology domain knowledge model for intelligent E-learning systems has a great chance to provide high-level services for the learners.

The goal of this research is to develop a novel ontology domain knowledge model for intelligent E-learning systems (E-tutoring systems) and detect the student's knowledge level status when they use the system and prepare the material according to their status.

The research will focus on developing ontology domain knowledge model and student knowledge model for E-tutoring systems to enhance teaching and learning process and make the learning between the learner and the system as one-to-one tutoring. Ontology offers a common vocabulary which can be used to model various domains including the type of object, related concepts, and their properties and relationships. However, understanding the knowledge level status of students is a key feature of their success or failure in their studies. For educational environment, there is a performance-based requirement that verifies student learning.

1.1 Problem Domain

Intelligent Tutoring Systems (ITS) are computer software that utilizes learners' knowledge levels to deliver individualized education. ITS research has successfully provided systems efficiently supporting one-to-one tutoring. These systems are actively used in real-world environments and have even changed traditional teaching curricula.

Most of the intelligent tutoring systems (ITS) designed in the literature are used for a specific domain. It means the solution of one field can't be suitable for another discipline. The developed ITS applies an isolated knowledge base and has some limitations. The limitations of the local knowledge base are lack of reusability, lack of standardization, limited knowledge, lack of flexibility, and Manual control. Most models/methods need manual control, like DITA, and others use ontology and solve specific domains. Regarding the material given to the learners, the current solution can't introduce these materials according to the behavior of the learners.

Due to this problem, this research will use the ontology domain knowledge model as a solution in a general structure that can avoid the local knowledge-based problem. However, considering the main goals, the research is going to answer the following questions:

- How to construct a model that can support the current education models and methods?
- How to allow the constructed model to make e-learning systems more intelligent to meet the learners' requirements?
- Which adaptive control mechanism is suitable for intelligent learning?
- What are the appropriate techniques to manage and cover the learning process?

- How to allow the developed solution to make the domain knowledge module of e-learning environments standardly?

1.2 Research Goals

The main goal of this research is to develop a general and flexible model to facilitate and support the personalization of the learning materials, automatically generating new task assessments and providing hints and recommendations to the student, which can significantly improve the efficiency of the E-learning systems and can enhance the functionalities of teaching and learning processes. This research aims to achieve the following goals:

- Support the current education models and methods and make e-learning systems more intelligent to meet the learners' expectations as required.
- Develop an adaptive control mechanism suitable for building an intelligent learning system to manage and cover the learning process.
- Design and develop a domain knowledge module for teaching materials of E-tutoring in a standard way to enable the learning process in a meaningful and effective manner using various IT technologies. Easy access to every knowledge related to learning we need for the study session and help share the best resources.
- Enhance the learning process using interactive tools and optimize the student's mastery of the domain knowledge module by controlling the generation of new problems, concepts, and instruction/feedback.
- Develop an assessment method to detect the current student status and prepare the material according to the evaluation.
- Develop a general structure of the domain knowledge model, which can support assessment, generation, and suggestion of new material.

To achieve the research goals, I will follow the following approaches: First, I will study and review the current E-learning domain knowledge models and their characteristics; then, based on reviewing, analyzing, and understanding the problems and limitations of these models, I will propose ontology domain knowledge model for intelligent E-learning systems to solve the existing drawbacks of intelligent E-learning. Second, I will discover the existing domain knowledge models of intelligent E-learning systems based on a comprehensive review of current practices and educational research. Moreover, I will try to find the weakness and strengths of each model to make a baseline for the new model. Third, comparing the models will help achieve an efficient and effective mechanism for enhancing the current e-learning systems. Finally, construct a general, flexible, and formal ontology domain knowledge model for intelligent E-learning systems (E-tutoring frameworks) to improve the teaching and learning process, and I will walk through several planning phases to develop a novel ontology domain knowledge model for intelligent E-learning systems to achieve the research goal. In addition, the model will be tested and evaluated to be applied in different environments to get the output and ensure efficiency. I also plan to design and develop a prototype system for an intelligent tutoring system using Python. I chose Python because it has many features supported by ontology, and most programmers now suggest using it.

1.3 Methodology

This research aims to enhance the quality of the teaching and learning process by making it personalized. The process design and development of domain ontology usually encompasses several standard tasks. However, there is no dominating approach for constructing ontologies. The main principle is to define the ontology concepts related to the objects and the relationships for the selected domain. The methodology for creating and building an ontology assumes defining the objectives and domain of applicability. In addition, this methodology must identify the steps in higher-level detail: the purpose of designing the domain ontology, the types of questions that should be answered through it, and how it will be utilized and supported for solving the problem for the selected domain. Several techniques for the design and development of ontology are given, such as [7] and [8]. Though these methods are somewhat different and influenced in varying ways by the technology used, the underlying processes of developing the domain ontology are similar. Therefore, the suggested ontology development process is composed of the following phases:

- Define the domain and purpose of the ontology.
- Discover if there are related ontologies.
- Enumerate essential terms in the domain.
- Define the key classes and their hierarchy.
- Identify the properties of classes.
- Facets attaching to properties.
- Create class instances.

1.4 Dissertation Guide

In this subsection, the researcher outlines the rest of the structure of the dissertation. Chapter 2. provides a general background of the innovative technologies for developing and enhancing learning for E-learning frameworks to make the learning process intelligent. Also, this chapter gives an overview of E-learning and Learning Management Systems and a description of intelligent tutoring systems called E-tutoring systems. Furthermore, the study describes the pedagogy, ontology, ontology models in the E-learning domain, and domain knowledge models in an E-tutoring system. The researcher also gives a general view of E-tutoring systems architectures, components, and current models and describes the four components of an E-tutoring system. In addition, this chapter presents some proposals for a domain knowledge module and an adaptive architecture for an E-tutoring system. Chapter 3. introduces the proposed formal logical ontology model of the domain knowledge model for teaching materials and defines the knowledge model in the E-learning domain. This section also describes the domain knowledge model components and their relationship. And also show the suggested ontology model schema, conceptual knowledge model, and ontology knowledge model for teaching materials. And this chapter also provides Loop Structures Knowledge Units as a modeling example. Chapter 4. introduces the integration of the domain knowledge model with E-tutoring frameworks. Chapter 5 presents the implementation of the domain knowledge model, integrates it with the prototype of an E-tutoring system, and also displays some of the interface designs from the prototype system. In addition, it introduces some of the task tests with correct and incorrect answers. Chapter 6. indicates the functionality and application frameworks.

Chapter 2 General Background

2.1 Introduction

Learning technology has developed rapidly and obtains powerful support from technology development, online learning, and stakeholders with complex requirements applying internet and communications technologies (ICTs) [1]. The use of ICTs has dramatically enhanced the way of reading and learning [2]. This progress develops learning methods by facilitating access to enhance new content and services [2]. Large numbers of learning content are produced continuously on the Web. It is vital to make them easily accessible, usable, and reusable [2]. The design and development of the educational model are essential for the growth and future of ICTs in knowledge management and the teaching/learning process. Universities and businesses need a technique to create scalable and flexible learning systems that simultaneously store and efficiently transmit the large numbers of information required by these educational processes to present this information to their users. This situation motivates and exposes the critical need to establish an effective and timely teaching/learning process based on E-learning platforms that consider student needs and achieve optimum quality.

To achieve this goal a new model/method is needed to standardize the design and implementation of this type of system based on developing a simple domain model that can be used equally well across the various platforms developed. The amount of learning content available in digital form requires using ICT tools that facilitate the creation, reusability, interoperability, and distribution of content through the Web is the most common means of communication. The proposed domain knowledge model must be presented based on a systematic approach for developing E-learning systems considering systematic methods from both E-learning and software development communities. To build the model for the coming E-learning systems, which should be intelligent. In this case, new learning technology comes to enhance the way of teaching and learning, named the intelligent tutoring system. Intelligent tutoring systems (ITS) aim to help learners acquire knowledge and develop skills in a particular domain [3]. To effectively provide tutoring services, these systems must be designed and equipped with a clear representation of the learning activities of the domain knowledge model. Also, this model must be created and fitted with which knowledge representation mechanisms allow the system to reason and solve problems within the domain. ITS is a software system that aims to help students receive direct and personalized instruction or feedback, usually without the intervention of a human teacher. ITS has a common goal of facilitating meaningful and efficient learning through various computing technologies. An ITS usually attempts to increase the demonstrated advantages of one-to-one, personalized tutoring in contexts where students would otherwise have access to one-to-many instruction from a single teacher (e.g., classroom lectures) or no teacher at all (e.g., online homework) [4]. ITS framework is developed and designed to provide access to high-quality education for each learner.

The domain model (DM) is essential for developing E-Learning systems. DM is a representation of objects in a domain and their interrelationships. DM within the intelligent tutoring systems describes the course contents, including the knowledge units and competencies using ICT.

The last dedication of E-learning was managed and controlled by the Learning Management Systems (LMS), such as Moodle, ATutor, or Blackboard; these present integrated systems that enable support for various educational activities. Thus, teachers can use LMS to generate courses and test suites, communicate with the learners, and monitor and evaluate their work. In addition, learners can learn, share, and collaborate through LMS. The problem is that LMS does not offer customized services. Instead, all learners are given access to the same learning content and tools without considering the differences in knowledge level, interests, and goals.

In recent years knowledge Engineering (KE) has become an important topic [5]. KE is the component of system engineering that addresses ambiguous process requirements by emphasizing knowledge acquisition about a process and reflecting that knowledge in a knowledge-based system [5]. KE is a process of designing and developing Knowledge-based systems in any field, whether in the public, private, educational, commercial, or industrial sectors [6]. In earlier years, Artificial Intelligence (AI) research focused on developing formalisms, inference processes, and methods for operationalizing knowledge-based systems (KBS), so the research activities at that time were generally restricted to the realization of small KBSs to test the viability of many approaches [6].

Human beings are better at understanding, reasoning, and interpreting knowledge and using this knowledge [6]. They can perform several actions in the real world. One of the primary purposes of knowledge representation involves modeling intelligent behavior for the agent [6]. Knowledge representation and reasoning (KRR) act as real-world information for a computer to understand and then use that knowledge to solve complex real-life problems, such as communication with human beings in natural language [7]. KR in AI is not only about storing data in database storage systems. It enables the computer to learn from that knowledge and behave intelligently like a human being [6]. KR is the study of knowledge and facts about the world, how it can interpret, and what kinds of reasoning engines for computer applications can manipulate that knowledge. KR is a sub-domain of AI dealing with understanding, designing, and representing knowledge and its implementation [8]. In addition, a well-defined structure of the knowledge representation forms allows scholars to express the knowledge base in computer software applications to be managed and derive new knowledge. However, it can imply rules and convert humans into natural languages, determine what to do next, plan future activities, and solve problems in a domain that usually needs human expertise [8].

Since the early 1990s, ontologies have been the hot and common research topics explored by many AI research groups, including knowledge engineering, representation, and natural language processing. More recently, ontology has become popular in intelligent education and knowledge management research [9]. Ontologies are becoming common, mainly for the promise of a shared understanding of the domain conveyed via people and computers [9]. Recently, the development of information technologies specifications, standards, and tools have guided scholars to develop a new way to interconnect the knowledge base, like ontologies [10]. Applying this technique to education domains requires introducing knowledge models which automatically filter the knowledge to generate a suitable suggestion for each learning objective. Currently, efforts are focused on designing knowledge models, design specs, and standards that increase computational integration using ontologies in the intelligent Web [10].

You can find many methods and models in the literature that use different knowledge representation forms. Ontology for knowledge representation is a good solution for knowledge management. It can provide a standardized way and reasoning engine for related knowledge management tasks. It is also a commonly used technique to promote understanding, communication, and interoperation between human and software agents. It also allows a clear definition and explicit specification of all the basic concepts and terms of a particular field [7].

A domain model in ontology engineering is a formal representation of the domain knowledge with concepts, roles, individuals, datatypes, and rules, usually based on description logic. Moreover, it contains machine-readable definitions of elementary concepts in the domain and their relationships, allowing sharing of a common understanding of the information structure among people and software agents and the reuse of domain knowledge [11]. Thus, ontologies are considered a fundamental method for knowledge representation, particularly in intelligent E-learning or intelligent tutoring frameworks [11].

Researchers need to consider a common type of architecture and models of intelligent tutoring or E-tutoring systems to match existing standards and support reusability, standardability, flexibility, and interoperability, which remains a challenge for adaptive learning systems [12].

The rapid development of communication and information technologies has offered new possibilities and challenges in multiple disciplines. One of these areas is education domains. Online education provides several advantages: convenience, reduced costs, and ease of taking courses. As a result, many institutions have adopted E-learning platforms in recent years. Furthermore, it has proven effective and efficient for enhancing learners' knowledge. The most appealing feature of an E-learning platform is that it provides flexible learning paths based on learners' skills and abilities. Therefore, teaching materials can be stored and managed more flexibly as learners can choose where and when to learn them based on their individual needs.

Today, new challenges and innovative technologies are faced in the education discipline. However, the vital area of innovative technologies presented lies in the learning process. Therefore, it becomes required for instructors to integrate new directions and methods into the educational process. One such aspect that came into reality is the intelligent tutorial system. The next points deal with details about E-learning and intelligent tutoring systems.

2.2 E-learning and Learning Management Systems

The concept of E-learning has evolved dramatically over the past 40 years. From simple online text to intelligent E-learning, regarding the fast development of the technology on capabilities, software, and hardware [13]. Although many definitions of E-Learning generally focus on the same features. E-learning is an instructional platform that integrates teacher instruction with online, flexible, and electronic resources [14]. It has some limitations on personal learning and a high cost of development. Because E-learning usually contains dominantly static data. Learners cannot obtain what they need in the conventional E-learning solution because of the lack of tutor guidance, feedback, and the massive number of different educational materials. E-learning is a computer-based educational tool or system that enables learning anywhere and anytime [15]. E-learning allows sharing of material in various formats such as videos, slideshows, word documents, and text files. E-learning uses electronic devices and Internet technologies to deliver multiple solutions to support learning and enhance performance [16]. The E-learning field can be represented by a wide range of applications, from virtual classrooms to remote courses or distance learning [17]. A new model for e-learning systems was proposed by [17] using semantic web technology. However, the model consists of various services and tools in the context of a semantic portal, such as course registration, uploading course documents and student assignments, interactive tutorials, announcements, valuable links, assessments, and simple semantic searches [17]. The next generation of E-learning, such as learning management systems, web-based educational systems, and E-learning management electronic systems, appears in reality. Learning Management Systems (LMSs) are web-based systems that enable teachers and students to share materials, submit and return assignments, and communicate online [18]. In addition, LMS is software used to plan, implement and evaluate a specific learning process to increase efficiency [19]. Semantic web technologies are applied to web-based educational systems to better serve the increasing and complicated requirements of the teaching community [20]. Web-Based Educational Systems (WBESs) offer interesting delivery mechanisms to teachers and learners [21]. Governance and accountability are key criteria to consider during the deployment of these WBESs learners [21]. Assessment of WBESs also needs to be done to determine the learner's effectiveness [21]. An E-learning management electronic system (EMES), also called a content management system, is a computer program that enables users to create, edit, collaborate, publish and store digital content [22]. EMES is an integrated computer system that manages the educational process and aims to facilitate the process of interaction between students and faculty members [23]. EMES helps users create, manage, and modify website content without needing specialized technical knowledge. In more straightforward language, EMES is a tool that helps us to build a website without needing any experience [24]. EMES is an application used to manage the learning content, allowing multiple contributors to create, edit and publish the content. EMES is a software application that allows publishing, editing, and modifying content and maintenance from a central interface. In addition, such learning management systems provide procedures to manage workflow in a collaborative environment [25]. EMES is an electronic system that controls the distance learning [23]. In addition, EMES has content creation features to make it possible for users to create and format content smoothly, content storage to store content in a single location consistently, and workflow management to grant privileges and responsibilities according to roles such as authors, editors, administrators, and publishing to plans arranges and pushes content live [24]. Learning is an activity that differs from one person to another in terms of pedagogical diversity. Therefore, E-Learning systems should support such diverse differences [2]. E-learning is a platform that employs information and communication technologies (ICT) to empower teaching and learning activities. The two significant aspects of E-Learning are technological and pedagogical [3]. The technology aspect consists of infrastructures and platforms to deliver learning-teaching content for their users. The pedagogical element deals with learning-teaching content for improving learners' knowledge.

The E-learning platform is becoming increasingly popular in the academic community due to many learning benefits achieved through learning anywhere, anyplace, and anytime. However, it tends to be most used for web-based training to access online courses. One possible reason for lack of success is that it does not practice just by putting lecture notes on the internet. Learning tools such as intelligent tutoring systems (ITS) can improve this situation. ITS incorporates expert systems to monitor a learner's performance and customize instruction based on adaptation to the learner's learning style, existing knowledge level, and appropriate instruction plans in E-tutoring systems. Learning environments rapidly change from traditional to information technology (IT) in the teaching space. In earlier decades, researchers studied how IT technology as a learning tool affects learning performance.

New computer-aided instruction paradigms have emerged during the rapid expansion of internet technologies. These technologies have started from learning management systems, Massive Open Online Courses, and reaching to adaptive learning for intelligent education. Figure 2.1 illustrates the historical evolution of education technologies in E-Learning environments from the 1990s until now [26].

With continuous technological development, different institutions employ and widely exploit many aspects and tools. The tools are blended learning, gamification, microlearning, personalized learning, and continuous learning [27]. Blended learning is the learning process combining two or more learning methods, such as pedagogical and web-based technologies. Gamification is a process of adding some gaming elements to engage learners in the learning process. Microlearning refers to the learning process of teaching materials presented in chunks, sections, or parts in short and measurable periods of time. In microlearning, the learners can proceed to learn the next content or part after completing the previous part. Personalized learning enables learners to choose or customize their learning materials based on pedagogical differences. Finally, continuous learning or lifelong learning refers to the ongoing and voluntary pursuit of knowledge and expertise for personal or professional purposes.

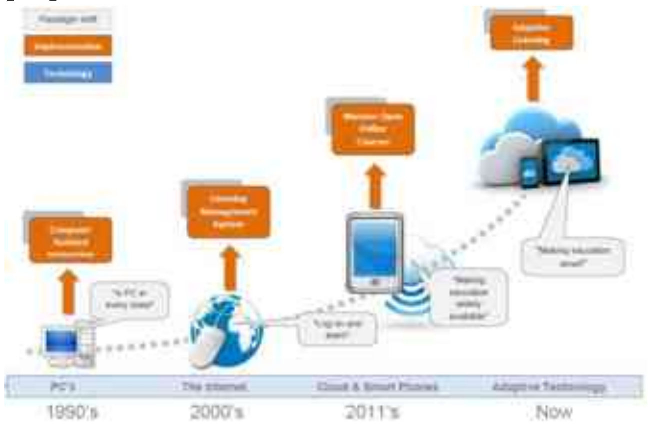


Figure 2.1 Historical evolution of education technology [26]

Nowadays, the use of technology to enhance education and learning is rapidly expanding. Intelligent Tutoring Systems (ITS) is one of the technologies for improving teaching and learning that represent a new way of computer-based training using artificial intelligence techniques. The system uses a knowledge base to provide feedback to the student as the student interacts with the system.

Web 2.0 offered more options to learners and tutors, such as blogs, wikis, and podcasts. Yet, it was not enough. The time has come for Web 3.0 to bring semantic Web to the learning environments for knowledge representation that offer personalization of content to the users. For learning personalization, ontologies have been used recently in learning systems as methods for knowledge representation.

A wide range of studies has shown that students who receive one-to-one tutoring perform substantially better on average than students tested on the same content by traditional classroom instruction.

2.3 Intelligent Tutoring System/ E-tutoring System

In the modern era, technology usage has changed the learning method. Now learners are equipped with various electronic devices. These devices are not only used for entertainment but also learning. Learning materials are obtainable on the internet and can be accessed anytime and anywhere. Such a learning style is considered E-Learning quite useful for all learners in the industry and education sector. The users' interest is continuously growing in this new learning method. It is an essential building block of learning in the twenty-first century. With the improvement of the internet and communication technology, more innovative techniques will support e-learning. With the massive amount of information available. The internet is considered as a best learning material for the learner. The problem with this extensive data is finding relevant material for learning and recommending what to read next. The recommendation becomes more complicated when a different number of learners are available with different knowledge levels and learning styles. The problem with such systems is that they are public, meaning unclear for the individual student. Such systems cannot monitor the learner's performance and cannot personalize classes according to the student's learning style and adapt accordingly. Personalization in the teaching and learning process is a significant issue and needs flexibility in the teaching and learning process so that individual learners can get personal attention. Thus, a tutoring system was required for learners that can intelligently recommend relevant learning materials individually. Artificial Intelligence techniques are added to the learning systems to understand learners' needs and make adaptations. A new generation of e-learning systems has come to solve the current problems known as Intelligent Tutoring Systems (ITS). ITS is an innovative software system that offers a one-to-one student-teacher learning process by enhancing understanding and perception of learning and teaching materials. In other terms, ITS is a computerized education domain that includes computational models from the cognitive sciences, learning sciences, computational linguistics, artificial intelligence, mathematics, and other disciplines. An ITSs tracks the psychological states of students in fine detail, a process named student modeling. The ITS has been followed by education, psychology, and artificial intelligence scholars. ITS goal is to provide the benefits of one-to-one instruction. It allows students to practice their skills by performing the tasks in a highly interactive learning environment.

According to A. Goold et al [28], E-tutoring, also known as the online tutor or e-moderator - is directed to facilitate student activities. The E-tutoring will be different from the teacher in charge in many online classrooms. W. Ma et al in [29] define an Intelligent Tutoring System (ITS) as a computer system designed to support and improve the learning and teaching process in the knowledge domain. ITS is computer science and cognitive science to create computerized tutoring systems that broadly offer immediate feedback and individualized instruction [30]. ITS aims to promote deep learning that persists over time transfers to new domains and accelerates future learning. Smart-Tutoring is a web-based intelligent tutoring system developed and designed for different purposes, such as adaptive teaching techniques, student models based on background knowledge and skills, and teaching methods serving distinct skill sets [31]. Adaptive teaching strategies cover student models, teaching techniques, and instructors' cognitive models. ITS comprises three main models: domain knowledge, user model, and pedagogical model. The domain of knowledge concerns knowledge related to a problem considered for a specific domain, including the teaching contents and the meta-information about the subject to be taught. The user model is concerned with information related to the user's abilities and subject understanding and contains the user's history. The pedagogical model helps the system choose the appropriate explanation style for a particular user based on his history (user model). Two main factors play a significant role in ITSs: knowledge representation and adaptation. Knowledge representation should have some features to be helpful in ITS. Those features might contain simple representation, easy-to-reach information, work with appropriate inference mechanism, be easy to modify and update, provide adaptation, and supports the explanation. Many knowledge representation methods you can apply to represent the knowledge domain in ITS. These techniques are symbolic rules, case-based, neural networks, belief networks, rule-based, knowledge graphs, and ontology. These approaches enhance the way of learning and teaching processes.

2.4 Pedagogy

Pedagogy is the art or science of teaching and educational methods [32]. Pedagogy can refer to the function of a teacher (teaching) and the teaching methods/theories being employed [32]. Pedagogy is the dynamic relationship between learning, teaching, and culture. In terms of education instruction, the classroom activities of tutors are supported by their concepts and beliefs on teaching [33]. Pedagogy interacts with, combines views, and draws beliefs about learners and learning, teacher and teaching, and curriculum [34]. Although there are many tools supporting pedagogy, e-learning tools can support pedagogies to achieve the following goals [35]:

- Effective pedagogies are adjusted to behavior knowledge and understanding of learning.
- Effective pedagogies involve clear thinking about learning outcomes and goals.
- Effective pedagogies are built on students' prior learning and experience.
- Effective pedagogies to improve the efficiency of student learning.
- Effective pedagogies involve a variety of approaches, including whole-class and structured group work, supervised learning, and individual activity.
- Effective pedagogies focus on developing higher-order thinking and metacognition and using dialogue and questioning.

- Effective pedagogies embed assessment methods for learning.
- Effective pedagogies that are comprehensive and meet the various needs of a range of learners.
- Effective pedagogies involving important feedback from students can help improve teaching and learning.

2.5 Ontology

Nowadays, ontologies have become an appropriate representation of formalism, and multiple application domains are considering adopting them. This attention claims techniques for reusing domain knowledge resources for developing domain ontologies.

Ontologies and other Semantic Web technologies have been discussed in the context of E-Learning since early 2004 [36]. Ontologies are utilized differently in E-Learning systems, depending on the E-Learning task they perform. In addition, ontology is a building block of semantic technologies. And it also has a formal description of the relevant knowledge concepts and their relationships. The knowledge which is delivered with a well-defined meaning is more suitable for allowing computer applications and humans to work in collaboration [37]. Ideally, ontologies should describe these well-defined meanings and formalize them in languages such as Ontology Web Language (OWL) [38], Resource Description Framework (RDF) [39], or Resource Description Framework Scheme (RDFS) [40]. OWL is the most commonly used nowadays. OWL ontologies provide classes, properties, individuals, and data values. In addition, the OWL ontologies use files in the forms of RDF/XML format (the most common format) and OWL/XML, N-Triples, TDB, and Turtle to store the knowledge base. Classes are defined using an <owl:Class> element. Regarding properties in OWL, there are two kinds of properties: object properties and data properties. Object properties connect objects (instances of classes that are essential elements for the domain of interest) to other objects. Examples are hasID, isTaughtBy, and teaches. Object properties are defined using <owl:ObjectProperty>. Datatype properties which relate objects to datatype values. Examples are ID, name, and description. OWL does not offer any built-in data types or support specialized definition structures. Instead, it allows us to use XML Schema data types, thus using the Semantic Web layered architecture. Datatype properties are defined using <owl:DatatypeProperty>. User-defined data types are usually stored in an XML schema and then used in an OWL ontology. Besides, OWL provides property restrictions that can be put on the classes using <owl:Restriction> expression. The property restrictions are owl:onProperty, owl:allValuesFrom, owl:hasValue, owl:someValuesFrom, owl:maxCardinality, and owl:minCardinality. Individuals (instances) are the basic building blocks of ontology. The individuals in an ontology include concrete objects such as learners, knowledge units, knowledge slots, rules, materials, and complexity levels. Additionally, OWL offers Special Properties, which means that some property elements can be defined directly:

- owl:TransitiveProperty defines a transitive property: “is less than or equal”, “is ancestor of”, and “has better score than”.
- owl:SymmetricProperty defines a symmetric property: “has same score as”, “is sibling of”.

- owl:FunctionalProperty defines a property that has at most one unique value for each object, such as “ID”, “height”, and “directTutor”.
- owl:InverseFunctionalProperty describes a property for which two distinct objects cannot hold similar value, for example, the property “isTheSocialSecurityNumberfor” (a social security number is assigned to one person only).

The formal nature of ontologies allows intelligent devices to interpret the meaning of concepts. Ontologies are developed through consensus among interested groups and provide a vocabulary of concepts that are shared for mutual understanding and communication [41].

Ontology is a standard structure that offers a shared understanding of a specific area. It represents the domain semantically explicitly, allowing intelligent access to knowledge. Since the early nineties, ontologies have been a popular research topic in the artificial intelligence society. The research topics involve knowledge representation, engineering, and natural language processing. In the earlier decade, with the beginning of studying the Semantic Web, ontologies have increased in popularity. More recently, the concept of ontology has been more prevalent in the E-Learning discipline. Ontology and semantic web have been utilized in E-Learning frameworks in various paths, such as representing the knowledge domain, providing metadata for entities and key concepts in the teaching environment, allowing for a more detailed description and retrieval of learning material, promoting exchange and sharing of educational content, personalizing and suggesting teaching material, creating curricula, and assessment of learning [42]. The author in [43] defined the Semantic Web as "*an extension of the current web in which information is given well-defined meaning*." It expects a machine-understandable web with an explicit semantic representation of underlying web pages, data, and other web contents. In this vision, web-based services will assist individuals by understanding more of the content and information on the Web, connecting them in new meaningful ways. This method will result in more accurate filtering, categorization, searching, and reasoning about information resources and data. Ontologies define a standard, shareable view of a domain. They give meaning to knowledge structures exchanged by software systems [44]. The formal definition of ontology introduced by Gruber [45] is that an "*ontology is an explicit specification of a conceptualization*." Ontologies determine or model the domain using concepts, attributes, and relationships, and this explicit formal representation provides meaning for the vocabulary. It is helpful to describe a common vocabulary in which shared knowledge can be represented to formally support the sharing and reuse of defined domain knowledge within intelligent tutoring systems (ITSs). In information science, ontology describes a vocabulary representational for a shared knowledge domain of ITSs by descriptions of classes, relationships, functions, and other entities. In information science, ontology is the behavioral model of the entities and interactions in specific domain knowledge or practices, such as e-learning systems. Generally, in information science and computer technologies domains, we find that ontology is an excellent practice and formal representation of a collection of concepts within a specific domain of interest and their relationships. Many definitions in knowledge engineering, knowledge representation, and intelligent systems are reports about the applications of ontologies in developing and using e-learning systems in intelligent ways.

In computer technology and information science, ontology is the formal description of a specific domain by representing the concepts of a particular domain, their properties, and the relationships among them. The concepts are usually classified regarding a hierarchical relationship of specialization, generalization, and containment among these concepts. In this hierarchical concept, there is the offspring of the broader concept. For example, "learner" is a "human," and "human" is an "Object" (the concept of "learner" is the son of the concept "human"). All concepts are implicitly members of the concept "Thing."

2.6 Ontology Model in E-learning Domain

Several articles in the last dedication are devoted to analyzing ontologies in the education domain, providing overviews of different factors. In many projects, other ontology-based applications dealt with knowledge, structural, and subject domain, which guided two kinds of ontologies [46]. A domain ontology represents the primary concepts of the given domain with their interrelations and essential properties [46]. A structure ontology expresses the logical structure of the content [46]. It is generally subjective and depends greatly on the goals of the ontology application. It typically represents hierarchical and navigational relationships. While a domain ontology can establish a shared understanding of a specific domain, a structured ontology enforces a disciplined technique for authoring, which is mainly essential in collaborative and distributed authoring.

Many researchers proposed e-learning based on semantic Web (SW) technology in the last few years. In addition, some scholars mentioned that SW is a vital technology applied in the e-learning domain. The goal is to solve their problems, starting from representing the student knowledge and assessing knowledge levels of the domain knowledge to develop an efficient association between the learner's needs and adequate teaching resources.

Although there are many studies on ontology and SW technologies, most of these technologies are focused on the same features. Fakoya et al. [47] propose an Ontology-Based Model for E-learning Management System (O-BMEMS), whose primary objective is efficiency and relevancy. They present an ontology for the e-learning process, including course syllabus, teaching methods, learning activities, learning styles, and prevent the student from skipping prerequisite courses. Qingtian et al. [48] introduce a framework for e-learning systems to capture the user's reading behavior. The framework is utilized as a form of an embedded component into any e-learning system as an assistant component to realize user-adaptive or personalized learning or instruction. Mohit et al. [17] propose a new model for e-learning systems using semantic web technology. It consists of various services and tools, such as course registration, uploading course materials and student assignments, interactive tutorials, announcements, useful links, assessments, and simple semantic searches. Their implementation provides students with two kinds of content: learning and assessment, which contain different types of services such as learning services: provide registration, online course, interactive tutorial, course documents, announcements, links, student papers, and semantic search. Assessment services offer exercises and quizzes for evaluation of the student's knowledge. Chung and Kim [49] introduce an ontology model and propose an effective method for enhancing the learning effect by constructing subject ontology for students. The subject of a particular course comprises an ontology made by a teacher and many ontologies created by students. They used the developed ontologies for many aspects, such as discussion, visual presentation, and knowledge sharing between instructors and students. They tested the ontology subject in two lectures in practice. The student feedback analysis found that the ontology subject improves the learner's learning effect. Ioannis P. et al. [50] propose a student model and enhance it with semantics by developing an ontology to be exploitable effectively within an Intelligent Tutoring System (ITS). The ontology scheme consists of two main taxonomies academic and personal student information. Adelsberger et al. [46] discuss the area of Ontologies and Semantic Web technologies in e-Learning. It considers the impact of ontologies on Web-based educational systems (WBES). It then presents an ontology of the area of Ontologies for Education along with a community Web Portal (O4E) driven by that ontology. Finally, it offers a use case of Semantic Web technologies as enabling technologies for building WBES in the case of TM4L. Topic maps for e-Learning (TM4L) is an authoring environment for building ontology-aware standards-based learning materials (objects) repositories.

Adel et al. [51] introduced several websites created by web 2.0, such as blogs, wikis, and web applications, and used in many e-learning systems. Due to the limitations of web 2.0 nowadays, web 3.0 is used for creating ELMS. The SW enables a new generation of intelligent applications by providing programs and software agents to share information and knowledge in rich and effective ways. Using SW for e-learning allows data to be understandable by machines, and the communication between human and machine agents will be based semantically. One of the primary purposes of using SW is to create ELMSs that provide a relationship between user information and the e-learning system content [51]. Accordingly, in the future, the next generation of E-learning systems will be based on SW [51]. The investigation represents an overview of the SW technologies Resource Description Framework (RDF) and Ontology Web Language (OWL) applied to ELMSs [51]. OWL, RDF, and RDFs are the essential representation languages for the semantic modeling of domain knowledge models, with RDF performing as the basis of modeling. RDF manages a primary matter in the semantic modeling of the domain knowledge model: managing distributed knowledge resources. Most of the semantic modeling standards are built on this foundation of distributed resources. RDF relies heavily on the knowledge infrastructure, using many of its standards and demonstrated features while extending them to provide a foundation for distributed knowledge sources.

2.7 Domain Knowledge Models in E-tutoring systems

Nowadays, constructing intelligent systems for education plays a vital role in creating an intelligent tutor. Knowledge engineering offers technology to build an E-tutoring system for supporting teaching and learning processes. E-tutoring frameworks vary from traditional computer-aided techniques because they easily adapt to individual learners' needs. The domain knowledge model description must be well-organized and constructed effectively and efficiently to develop an excellent E-tutoring system and provide relevant and valuable feedback to the learners. The domain knowledge model incorporates the domain knowledge related to the topic and the actual teaching material taught by the learners. It usually consists of two components: (a) knowledge model and (b) study units. The Knowledge model refers to the essential concepts, including the topic and types of relationships between them taught by the students. These relations can cover the dependency, specialization, generalization, or containment relationships. Finally, they are associated with teaching units, including the learning content. Usually, concepts and their structure can be managed and organized in a representation scheme.

2.8 Domain knowledge model

Data Model (DM) is a commonly used notion in many disciplines to deliver an abstract representation of its structure, function, behavior, or others aspects [52]. DM expresses as a conceptual model that organizes knowledge structure, relationship, semantics, and consistency constraints [52]. According to authors [53], DM deals with a collection of knowledge concerning a specific topic, concept, or domain. Artificial intelligence is a tool that can extract and work with this data. The data represented in this case relates to the form of knowledge in a particular domain. As an additional point, the created model is not to replace the instructor's role. According to the knowledge engineering field, the domain model formally represents knowledge with concepts, roles, datatypes, and individuals.

Domain knowledge (DK) defines as specific knowledge related to a given discipline. In other words, The role of DK is to explain specialized knowledge connected to a particular area of interest. DK is powerful and valuable to academic institutions because it is a targeted skill to learn from intelligent learning systems. In addition, the DK can convey several fields, and the most common domains are computer software, education area, software development, and. DK holds all aspects of the knowledge to learn and teach in the education domain. It consists of information on courses, knowledge units, key concepts, topics, activities/tasks, and relationships between them. The domain knowledge also can represent a particular knowledge of the learning system and the necessary inferences, e.g., knowledge about the domain, teaching materials, and updated functions. The domain knowledge model (DKM) introduces a knowledge base for courses, topics, fundamental concepts, teaching units, or knowledge units in the E-tutoring system. In other words, the DKM represents the knowledge base structure of a particular domain in a specific discipline used as a component in E-tutoring systems. DKM is essential and valuable to educational communities because it is usually a targeted skill for developers to build a knowledge base in a well-defined form so that it can use as learning content from the E-tutoring platform [3]. DKM within the E-tutoring system describes the course, topics, knowledge domains, key concepts, teaching units, and knowledge contents and competencies. However, the primary role of developing a DKM is to promote and reuse this knowledge on different E-tutoring frameworks. Domain module is the most significant part of E-tutoring systems providing a base for operational components such as learning material, content recommenders, or collaboration tools.

2.9 E-tutoring Systems Architectures, Models, and Components

With the fast development of technology, computer education has become increasingly incorporated with artificial intelligence approaches to design more personalized educational platforms. These systems are known as E-tutoring Systems. E-tutoring System is a computer application invented to support students with direct and customized education or feedback, usually without human teacher intervention. Several researchers, designers, and developers define E-tutoring systems differently, corresponding to their interests. According to authors in [13], E-tutoring systems are intelligent education techniques applying computer and communication technologies, capabilities, and practices to improve a human instructor who is an expert in the subject matter to enhance personalized learning as one-to-one tutoring. Scholars are struggled since the earliest invention of computers to create intelligent learning systems that are more successful than human instructors [54]. The primary role of using an E-tutoring system is to facilitate and customize student learning and achieve their activities [54].

2.9.1 E-tutoring Systems Architectures

This section presents the main E-tutoring architecture solutions in the domain of intelligent E-learning platform. The architecture developed as authoring tools had a significant impact on the evolution of E-tutoring systems. Commonly the developer communities have used these tools in creating E-tutoring frameworks. Concerning E-tutoring architectures, the first invested architecture model is the basis of the three traditional models, including the elementary components domain model, learner model, and tutoring model. In 1990 Self John [55] developed the earlier architecture model by adding a learner interface module. Figure 2.2 displays the architecture of the four-component managing the current E-tutoring systems.

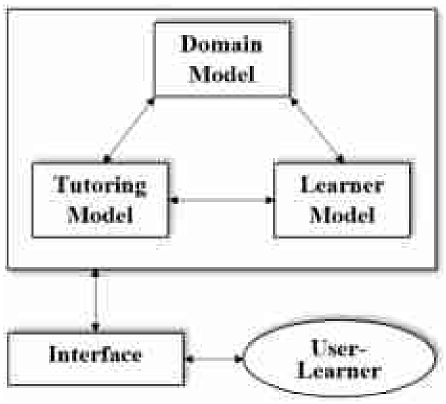


Figure 2.2 The four-component architecture [55]

According to [56], the primary architecture of an E-tutoring system includes four modules: learner module, knowledge module, and tutor module, also known as the teaching methods module. These modules work interactively and intercommunicate through a standard module called the leaner interface. Figure 2.3 displays the primary architecture of an E-tutoring system.

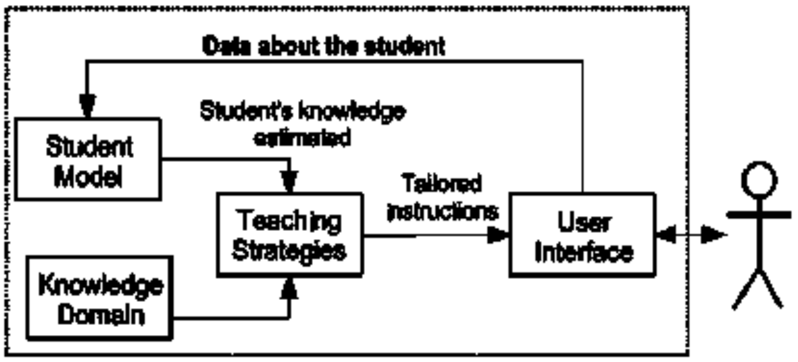


Figure 2.3 The basic architecture of an E-tutoring system [56].

The scholars in [57] suggest an extension of the standard architecture of an E-tutoring System, which contains a selector agent of contents. In its selection process, this selector agent considers the teaching techniques that help the student's learning style. This architecture offers representation innovations in the tutor and knowledge modules; the tutor module includes a selector mechanism, which will determine the content to deliver, considering the teaching processes that help the learners in their learning style [57]. Figure 2.4 indicates this architecture.

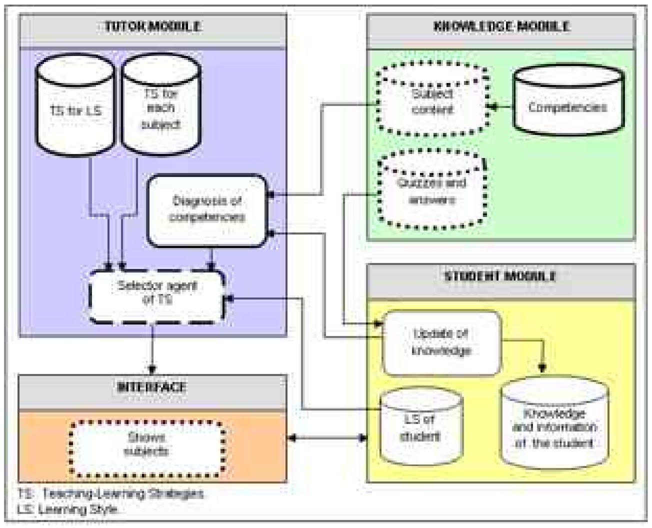


Figure 2.4 The proposed extension architecture [57].

The article in [58] provides an authoring tool called *Authoring Software Platform for Intelligent Resources in Education (ASPIRE)*. ASPIRE was developed at the Canterbury University, New Zealand, by the Group of Intelligent Computer Tutoring. ASPIRE applies domain experts for constructing constraint-based tutors by extracting domain knowledge models from machine interactions. ASPIRE technique also helps domain experts create ITSs and aids in developing E-tutoring systems over the web. ASPIRE offers a partially automated procedure for producing domain models. The instructor must define the domain ontology, determine the form of problems and their solutions, and deliver examples of both. From given knowledge, ASPIRE develops the domain model and delivers an entirely functional ITS web-based platform that learners can then use. ASPIRE also provides extra support for managers to create user accounts and maintain the activities in ASPIRE. It also supports teachers in tailoring ITSs to their classes. Figure 2.5. shows the architecture of ASPIRE.

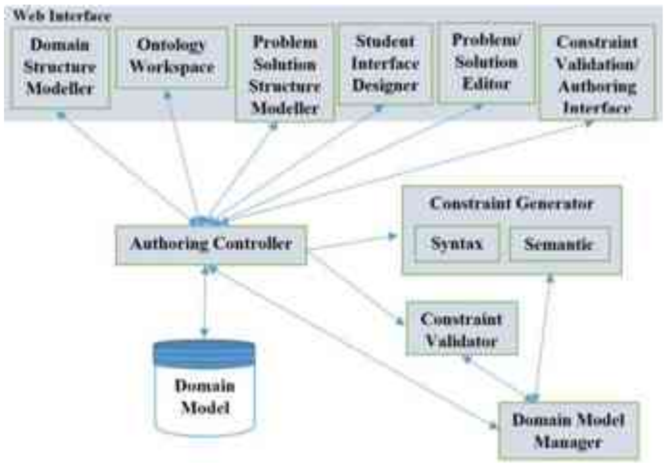


Figure 2.5 ASPIRE architecture [58]

Stephen et al. [59] invented an authoring tool called *Extensible Problem-Specific Tutor (xPST);* the goal is to handle the requirement of rapidly designing a computer software instruction that tracks the behavior of model-tracing tutors. xPST is a web authoring tool that enables the rapid development of example-tracing tutors on existing interfaces, reducing evolution time and separating the interface from the tutoring part. xPST allows instructors who are not programmers and not cognitive scientists to rapidly develop an E-tutoring framework that provides instruction similar to the model-tracing instructor platform. Also, this instruction is covered on current computer software, so that no need to build the learner's interface from scratch. The xPST architecture authorizes extending its capabilities by adding plug-ins that share with other third-party software. The xPST system includes the following primary components: The Listener Module, xPST Tutor Engine, Presentation Module, and Web Authoring Tool. Figure 2.6. Bellow indicates the architecture model.

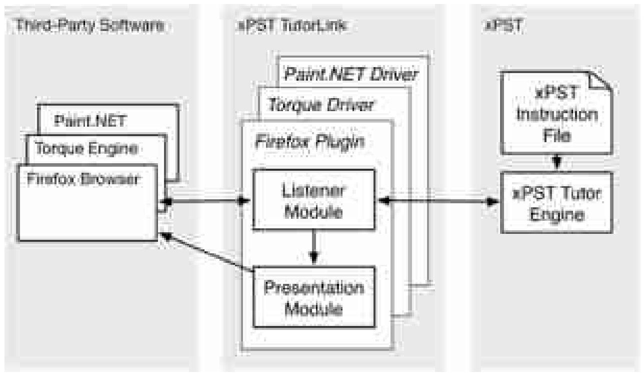


Figure 2.6 xPST architecture [59]

Protus 2.0 was proposed by Boban et al. [60]; Protus 2.0 is an E-tutoring system developed to support education processes in various studies and domains. The system is prepared as a public tutoring system for learning in the context of an essential Java programming domain [21]. It is an interactive system with a primary objective to enable learners to employ instruction material formulated within a proper introductory programming course but also contains a part for testing learners' gained knowledge. Protus 2.0 includes the following educational ontologies:

- offering a domain knowledge (domain knowledge ontology)
- constructing a learner model (learner model ontology)
- proposing exercises in the system (task ontology)
- selecting pedagogical actions and behaviors (education strategy ontology)
- representing the semantics of messages sent among components (communication ontology)
- and determining behaviors and strategies at the student interface level (interface ontology)

The suggested architecture which is displayed in Figure 2.7 is highly modular, enabling adequate flexibility and future replacement of different components as long as they comply with the existing interface.

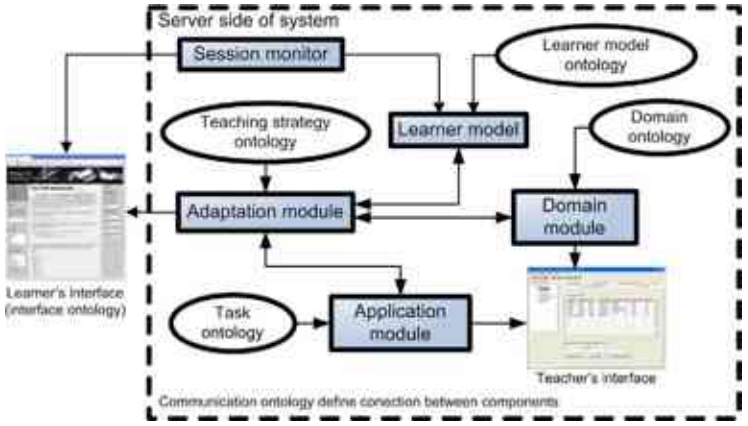


Figure 2.7 Protus architecture

Vincent et al. [61] invented a novel tool called Cognitive Tutor Authoring Tools (CTAT) at the University of Carnegie Mellon. CTAT is one of the extended running and most successful toolsets. CTAT enables instructors to immediately attach tutoring knowledge components to Graphical User Interface (GUI) components with little programming effort and indicate template solutions efficiently [62]. CTAT is a mechanism suite that allows educators to deliver learning by accomplishing online courses. The author [62] expressed CTAT as a mechanism for building online cognitive tutors. CTAT offers personalized help through customized information and observes learners' success in providing feedback as they move through a problem. CTAT helps design two kinds of instructors: 1) cognitive instructors who need AI programming skills to create a cognitive model of learner problem-solving but help to tutor through a range of problems. 2) example-tracing instructors, which can be developed without programming skills but needs problem-specific authoring. CTAT architecture is shown in Figure 2.8.

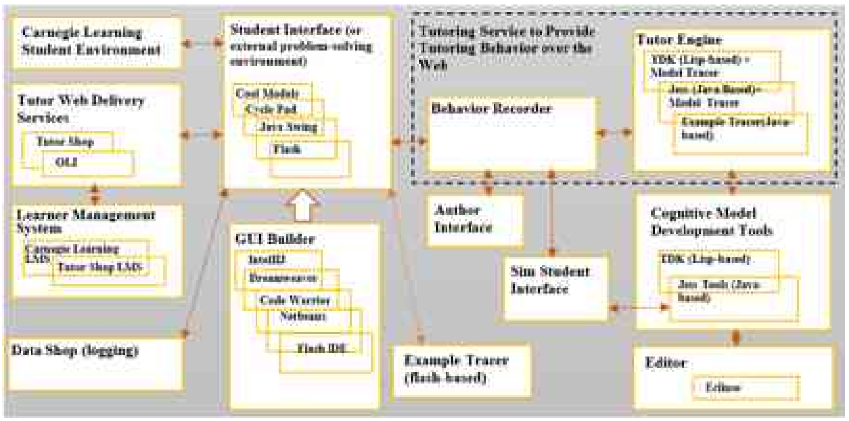

Carnegie Learning Student Environment
Student Interface (or external problem-solving environment)
Tutoring Service to Provide Tutoring Behavior over the Web
Tutor Engine
TDK (Lisp-based) - Model Tracer
Jess (Java-Based)- Model Tracer
Example Tracer(Java-based)
Tutor Web Delivery Services
Tutor Shop
OLI
Cool Modules
Cycle Pad
Java Swing
Flash
Behavior Recorder
Learner Management System
Carnegie Learning LMS
Tutor Shop LMS
GUI Builder
IntelliJ
Dreamweaver
Code Warrior
Netbeans
Flash IDE
Author Interface
Sim Student Interface
Cognitive Model Development Tools
TDK (Lisp-based)
Jess Tools (Java-based)
Data Shop (logging)
Example Tracer (flash-based)
Editor
Eclipse

Figure 2.8 CTAT modular architecture [61]

Generalized Intelligent Framework for Tutoring (GIFT) Mechanisms is an authoring tool developed by the Army Research Laboratory (ARL) community. GIFT is a modular, open-source framework for constructing, deploying and controlling adaptive training content. It can contain external hardware and software components, such as model attributes about the learner, tailor instruction to learners' needs, and allow learners to practice their knowledge in various external applications [63]. GIFT was invented to improve self-regulated education capability and reduce the time/cost/skill required for ITS authoring. GIFT has the following basics functions: GIFT is an authoring ability to design new Computer-Based Tutoring Systems (CBTS) components and all tutoring procedures, GIFT is an instructional manager that merges selected tutoring guides and techniques used in CBTS, and also is an empirical testbed to investigate the efficacy and influence of CBTS components, mechanisms, and processes. GIFT is established on a student-centric technique, a basic driver for tutoring investigation performed by the Education in Intelligent Tutoring Environments (LITE) Lab at Army Research Laboratory (ARL) to enhance connections in the adaptive tutoring education effect chain [63]. Figure 2.9 displays the architecture of GIFT.

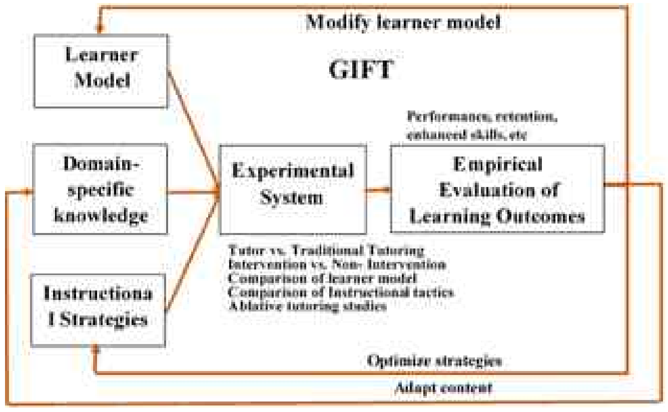

Modify learner model
GIFT
Learner Model
Domain-specific knowledge
Instructional Strategies
Experimental System
Performance, retention, enhanced skills, etc
Empirical Evaluation of Learning Outcomes
Tutor vs. Traditional Tutoring
Intervention vs. Non- Intervention
Comparison of learner model
Comparison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

Figure 2.9 GIFT architecture [63]

The author in [64] developed an authoring tool called Intelligent Tutoring System Builder (ITSB). ITSB is a mechanism that allows instructors from different specializations to design domain specific ITS modules. In addition, the system enables Educators to add specific intelligent features to the E-tutoring system. ITSB authoring tool contains two sub-systems modules. The first is the tutor module, where the instructor adds the teaching materials, questions, answers, etc. The second module is the learners, where the students learn the study material and practice activities. Figure 2.10 demonstrates the ITSB architecture.

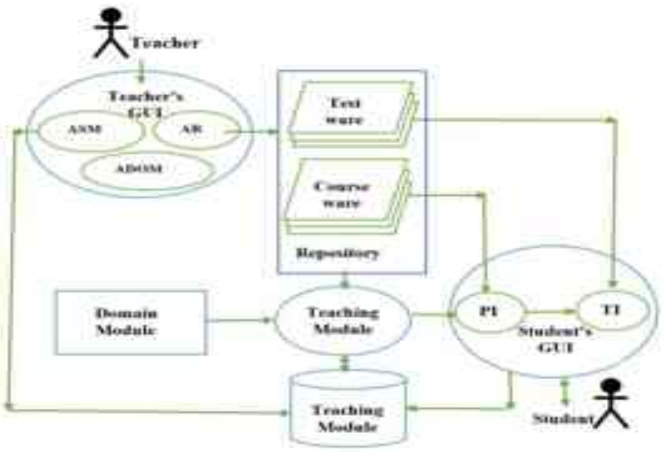


Figure 2.10 ITSB architecture [64]

The scholars in [65] demonstrate a general model architecture for an E-tutoring system known as SeisTutor. SeisTutor customizes its content according to the learner's knowledge preferences by specifying the learning style and prior knowledge level. Furthermore, SeisTutor aims to emulate similar human intelligence by implicitly judging the tutoring strategy before the tutoring session and custom tailoring the tutoring concepts to enhance learning. The implementation of SeisTutor uses the I2A2 index of an education style model. Figure 2.11. indicates the primary architecture of SeisTutor. The architecture contains various modules: Learner Interface, Knowledge organization unit, learner model, pedagogical model, domain knowledge, and inference system. The learner, inference, and pedagogical models include the core of an expert system utilized for creating important decisions within tutoring sessions. In addition, the expert system employs different rules in the form of (if-then) rules [65].

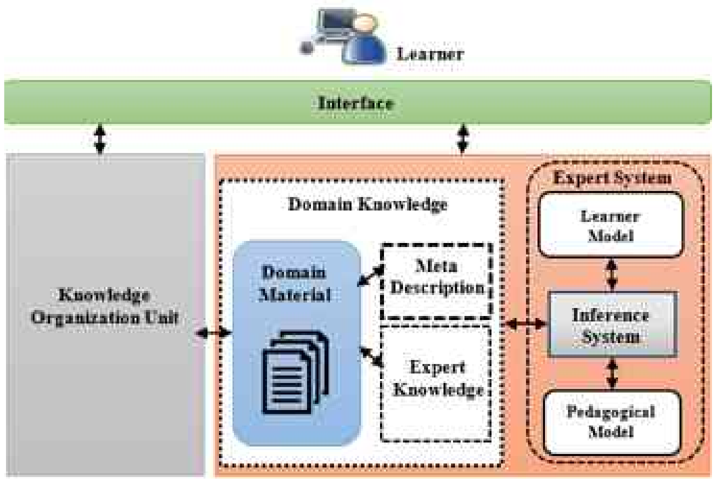


Figure 2.11 SeisTutor architecture [65]

2.9.2 E-tutoring Systems Component

The E-tutoring system has different components that work together to create an educational system that can identify ways of learners' behavior and reply with teaching appropriate to those ways. An E-tutoring system generally includes four significant components, shown in the previous architecture of the E-tutoring system in section 3.1.

2.9.2.1 Learner Model

Learner modeling can be represented as a process of collecting appropriate information to specify and describe the knowledge form of a learner. In other terms, the model of a learner should explain students' knowledge, desired learning techniques, domains of interest besides that of education, desired representation style, and level of attention [66].

The learner model records knowledge about the student. This knowledge reflects the system's confidence in the learner's existing knowledge form and assists in guiding the learner through the domain. For example, the pedagogical module uses the diagnosis done by the student modeler to recognize errors, generate feedback messages, generate problems, and control progress through the curriculum. The ability of E-tutoring to provide proper personalized instruction to a learner depends on the type of knowledge held about the student in the learner model [66]. It relies on the style and difficulty level of the knowledge representation used in the system and the effectiveness of the methods used to extract new information about the student and integrate the new information into the learner model [66].

A proper intelligent tutor has a good sense of what the learner understands, learns, and can do. A better learner model will result if this knowledge is employed to sequence the education materials. Creating a more effective learner model will also impact the instructional model, making it the most critical component of the E-tutoring system [66].

The learner model is particularly complicated. Building a learner model is quite challenging and severe due to the vast investigation areas. Different techniques for dealing with the intractability of learner modeling have been introduced. Self suggests such design of the interactions that knowledge required for building a learner model is delivered by the student and not inferred by the system. Also, it is not valuable to specify misunderstandings in the learner's knowledge that the tutor cannot deal with the learner's needs. An E-tutoring system should also model only what it can utilize to develop corrective or other pedagogical activities.

2.9.2.2 Pedagogical/Tutoring Model

The pedagogical instructional, tutor, or teaching models include information for making decisions about tutoring techniques. This model is responsible for enforcing an E-tutoring system's teaching techniques and allows the system to train by encoding the teaching techniques used by the tutoring system [66]. In addition, the model contains the correct teaching styles for learning goals based on the weaknesses and strengths of learners' experiences. Consequently, this model depends on the knowledge obtained from the learner's technique and compares it to the expert's knowledge to make the proper decision in the next learning step.

The teaching model is the core unit of an E-tutoring system because it addresses vital education decisions through tutoring sessions. It adjudges the tutoring tactics, determines the exclusive material, tracks the learner's Affection (Psychological) state, tracks performance parameters, takes the pedagogy flipping decision, and computes the learner's measure of assignment tutoring performance (Degree of understandability, Degree of engagement, and Learning gain) [4]. In addition, it allows the knowledge construction for tailoring the explanation of tutoring material through the information gathered in the learner model [65].

Comparing the tutor model with a human tutor who can adopt various techniques and methods, most instructional models depend on a group of tutoring plans. In the current research, pedagogical models utilized are implemented correctness to ensure that the system is in complete control, computer trainer knows that the student is in control. Dialogic teaching ensures that the system leads students to form general principles by posing questions and providing examples. Collaborative learning provides where more than one student is involved [66].

2.9.2.3 Domain/Knowledge Model

The domain model which is mentioned in section 2.7 and 2.8 is usually holds perfect expert knowledge and may also have the problems and misunderstandings that students sometimes indicate. Data Model (DM) is a commonly used notion in many disciplines to deliver an abstract representation of its structure, function, behavior, or others [52]. DM expresses as a conceptual model that organizes knowledge structure, relationship, semantics, and consistency constraints [52]. According to authors [53] Domain Model deals with a collection of knowledge concerning a specific topic, concept, or domain. Artificial intelligence is a tool that can extract and work with this data. The data represented in this case relates to the form of knowledge in a given domain. As an additional point, this model is not designed to replace the role of the instructor.

The purpose of an E-tutoring system is to replicate these knowledge structures in the mind of learners. The domain model is connected closely with the learner model; the system must search for domain knowledge as it compares the model of a student's learning with that of the domain knowledge. We can utilize domain ontologies for the domain model: Developing domain knowledge is a task that needs a primary part of the time and effort when building an E-tutoring system. Scholars have been exploring methods for automatically implementing the knowledge acquisition technique since the beginning of E-tutoring systems with limited success [66]. Likewise, research attempts at automatically acquiring knowledge for E-tutoring systems have limited success [66].

As an idea, a modern technology can be used in the domain model this technology called ontology. Ontology provides OWL language based on an XML syntax to represent information about properties, classes, and relationships. A benefit of ontology is that it enables a better inference engine than works with RDF Schemas. All components/attributes offered in RDF and RDF Schema can be applied when building an OWL document. Using OWL to describe an ontology model has some benefits: 1) Extensible: much easier to add new properties. In contrast with a database to add a new column may break many applications 2) Portable: much easier to move an OWL document than to move a database. The concept ontology which is discussed in section 2.5 and 2.6 can enhance adaptivity and the functionality of the E-tutoring systems providing more widely available and maintainable.

2.9.2.4 Interface Model

The interface Model refers to communication between the user and the system. The interface is a graphic design that enables the learners to interact with the learning content and receive the program's feedback. In addition, the interface is designed to satisfy the learners' requirements, through which questions can be asked and information can be saved.

In the human-computer interaction discipline, interface design has continued to be a primary study field in Computer Science. A good interface will predict learner activities to be consistent, deliver a basic level of interaction, arrange learners' thinking, and use metaphors. In addition, the user is learning the interface along with the content, so any additional cognitive load should be minimal. There are many types of interfaces. A particular style may depend on the learner's capability and the knowledge to be learned. The ongoing investigation studies how well an artificial dialog models the instructor-learner relationship. The interface is essential as a communication medium, a problem-solving environment that helps the student in the task activity, and an external representation of the system's domain and instructional models.

The communication problem means the association between the learner's cognitive states and the perceptible actions captured by the interface model. As computer software becomes more powerful and complicated, it will be possible to deliver interface models that improve performance. The result will be a better diagnosis of the learner's level and the subsequent actions by the pedagogical models. The learner interface should be well-designed so that humans (learners) can easily interact with the computer software. It can contain both hardware and software components.

2.9.3 The current E-tutoring System solutions and models

This section is an overview of E-tutoring system solutions which are currently found in the literature, and it covers the related knowledge formats used for representing the domain knowledge model. Therefore, several works for E-tutoring system solutions can be found in the literature. Section 3.3.1 gives some of the current model solutions in recent studies regarding the continuous development of modern technology, which can change the way of learning and teaching process. To enhance sharing and reusing of the domain knowledge model in E-tutoring systems, ontologies used to represent and organize domain knowledge I will give the most common of the current domain knowledge models.

In all computer courses, SQL content is essential for teaching and learning. In [67], W. Yathongchai et al proposed an ontology model for personalized education as an intelligent learning system for the SQL domain called SQL-PITS. SQL-PITS is Intelligence Tutoring System that teaches SQL and delivers adaptive content according to individual student information. It is considered the preferred paradigm for tutoring students with educational materials, including examples, exercises, content, and tutoring materials customized for personalized learners according to their profiles, capabilities, preferences, and background. The SQL-PITS describe units of knowledge domain to be learned, in the form of topics, as modular units isolated from material and tutoring techniques. Ontology and learning objects are used to improve the SQL-PITS capabilities for effective SQL instruction. The proposed model demonstrates the SQL ontology model in three parts: SQL Knowledge, Learner Model, and Tutoring Techniques ontology.

The scholars in [56] introduce a new way for an intelligent tutoring system using Bayesian networks named BITS. BITS is an intelligent web-based tutoring system designed for teaching computer programming. There are two tasks discussed for helping a learner navigate a personalized Web-based education platform. Firstly, the design of the problem part must be modeled. Secondly, learner knowledge regarding every concept in the problem part must be tracked. The decision method guided in the proposed system is conducted by one of the formal frameworks for uncertainty management in Artificial Intelligence which is based on probability theory called Bayesian networks. The proposed system can support a student navigate through the online course materials, recommending learning objectives, and generating proper reading sequences.

Ayturk et al. [68] designed an E-tutoring System for Mathematics Education called ZOSMAT. The system tries to emulate the behavior of an intelligent human tutor and a domain expert [24]. ZOSMAT can be utilized for either personal learning or a real classroom platform with the suggestion of a human teacher during a formal teaching process [68]. ZOSMAT was developed to provide some features: demonstrate the key concept knowledge on the topic, an idea of which types of knowledge can be implemented for solving problems in an offered area, provide examples of problem-solving with examples module, recommend proper test reports from past performance, and suggest the next most efficient activity for the learner to adopt. ZOSMAT achieves flexibility and generality in ways that adapt to individual students' needs and abilities.

AutoTutor is an E-tutoring technique that allows students to learn diverse technical knowledge domains of computer literacy, Newtonian physics, and scientific methods by (a) having a discussion in a natural language, (b) simulating human tutors' pedagogical and motivational techniques, (c)modeling cognitive states of students, (d) using the student model to dynamically customize experiences with individual students, (e) answering students' questions, (f) identifying and correcting misunderstanding, and (g) keeping students engaged with images, animations, and simulations [69]. AutoTutor helps students form descriptions of complex concepts of the domains in natural language with adaptive dialog moves similar to those of human tutors [69]. Most AutoTutor represent world knowledge as spaces like Latent Semantic Analysis (LSA). Still, AutoTutor or its progeny have incorporated other forms of knowledge representation, such as textbooks, glossaries, and conceptual graph structures [69].

Latham et al. [70] represent a novel adaptive online conversational E-tutoring system that implicitly adapts to a personalized learning style and offers an individual natural language tutorial called Oscar-CITS. Oscar-CITS predicts and adapts dynamically to learner education styles. The goal of Oscar-CITS is to simulate a human instructor by using learning style knowledge to adapt its tutoring style and enhance the learning experience's significance [70]. In natural language, learners can automatically explore and address topics, allowing develop a deeper understanding of the topic and increasing confidence [27]. The Oscar Adaptive CITS mimics a human educator by conversing with students and providing learning material adapted to their learning styles, individualized problem-solving procedure, and smart answer analysis [70]. The investigation [71] suggests an E-tutoring System model utilizing the Bayesian Network to define students' capabilities and instruct skills competency material forms based on every learner's knowledge level. The presented BN module at E-tutoring System provides material form recommendations and determines students' ability levels into low, medium, and high [71]. BN in the pedagogical module can conclude students' capabilities separately. It employs the input to answer questions from the ability quiz. In addition, BN can infer grades quantitatively by individually describing the learners' abilities by getting quantitative information from the students [71].

The study [72] presents an Adaptive learning tool applying the Mastery Learning Technique in the form of a Web-Based E-tutoring System. According to the status of learner knowledge, this system adapted rule-based approaches and learner profiling to generate classes and quizzes for them. The suggested technique was automatically adjusted to properly adaptive lessons and quizzes for learners who can pass a specified test. This prototype is developed to improve Java Programming domains at Suan Sunandha Rajabhat University, Thailand, for active students. The rule-based and learner profiling methods can support student learning and reduce time-consuming study.

The article [73] introduces an intelligent web-based tutoring system in adult education named SmartTutor. SmartTutor is an E-tutoring system designed for distance education in Hong Kong. SmartTutor is incorporated into an electronic learning platform, which supports courses offered in Hong Kong University environment. SmartTutor integrated the educational theory and artificial intelligence (AI) approaches into a singular intelligent E-tutoring system. The aim is to deliver personalized guidance to learners based on the experience and background of singular learners. SmartTutor is developed to demonstrate the key knowledge on the subject, suggest which types of knowledge can be involved to give a solution for the problems in a given domain, recommend appropriate test reports from past performance, and propose the next considerable efficient exercise for the student to adopt. SmartTutor performs the generality and flexibility of tutoring in ways that adapt to individual learner requirements and capabilities. SmartTutor includes the following components: Learner Model, Content Structure, Knowledge Model, Course controller, Question Bank, and Learner Interface.

EduTutor is a smart tutor system developed by Joel et al. [74]. EduTutor is designed for the Aulanet learning management web-based system. EduTutor concentrates on topics for the Portuguese primary teaching system studies between the first to fourth years. Its goal is to promote the perception of an education process of each learner, separately, in a virtual environment and as a study guide. Also, the proposed system was created, and its architecture was organized for being easily incorporated into high levels of education, various subjects, and different languages. Furthermore, the student will easily access requests for updated information delivered by EduTutor, avoiding the requirement to return to the main page [74].

SeisTutor is a personalized intelligent tutoring system developed by Ninni et al. [65]. The aim is to customize the content according to the learner's knowledge preferences by specifying the learning style and prior knowledge level, to emulate similar human intelligence by implicitly judging the tutoring strategy before the tutoring session and personalize tailoring the tutoring concepts for enhancing learning. SeisTutor contains various components: Learner Interface, Knowledge organization unit, learner model, pedagogical model, domain knowledge, and inference system. The learner model, inference system, and pedagogical model include the core of an expert system utilized for creating important decisions within tutoring sessions. The domain knowledge includes three types of knowledge i) knowledge concepts, ii) meta-description, and iii) teaching topics. Knowledge concept is the essential part of the knowledge of a domain [65]. Knowledge concepts contain multiple attributes such as the name of the concepts, difficulty levels, mode of presentation, and detail or explanation level. Every knowledge concept is given into a concept group. The concept group comprises topics that are closely related to the concepts [65].

Ninni S. et al. [75] proposed a customized-tailored intelligent tutoring platform for sustainable education called SeisTutor. SeisTutor was explicitly created for the "Seismic Data Interpretation" topic domain. SeisTutor suggests learning materials for personalized learning to detect individual learners' performance in a pre-test called a prior knowledge assessment test. Furthermore, SeisTutor maintains track of the learner's behavior as a psychological state throughout the whole learning process and performs a test to select the degree of understanding of the topic domain. SeisTutor tracks the learner's performance through the tests and customizes the learning material accordingly. SeisTutor is an adaptive tutoring system to adapt the material and link-level. Adaptive material level demonstrates that the learners with different performances in the pre-test receive different learning material. Before involving learners in the learning session, SeisTutor allows learners to go via a pre-test. There are two styles of assessments conducted in pre-test: a learning style test and a prior knowledge test. The main emphasis of the study is on the prior knowledge test.

Most of the current knowledge model representation forms used in the existing solutions are inappropriate for developing and designing a common domain knowledge for E-tutoring systems. Of course, most knowledge representation techniques employed in the current E-tutoring framework to build domain knowledge models are standard and have many benefits. On the other hand, also they have some limitations. The present solution limitations are a lack of flexibility, sharing a common understanding of the knowledge base structure, reusing the knowledge base, merging different knowledge base sources, and Standardability. These limitations motivate us to develop a novel domain knowledge model using the most powerful techniques like the ontology-Based approach. Table 1. shows the comparison of these different approaches.

2.10 Summary

In this chapter, I give a general introduction and background of the current investigation on E-learning domains and the historical evolution of education technologies from the 1990s to now.

Intelligent E-Learning and Tutoring Systems are the key technologies supporting educational activities and processes. In this chapter also, I present the most common E-tutoring architectures, models, and components that significantly impact the evolution of E-tutoring frameworks. Concerning the E-tutoring architecture, the first E-tutoring systems were based on the traditional trinity model, including three elementary components: domain, learner, and tutoring module. the scholars developed and extended the earlier architecture model by adding a learner interface module, and also I give a brief description of pedagogy, ontology, Ontology model in an E-learning environment, and the domain knowledge models in E-tutoring system their architecture, components and models.

I found that the problem is that most ITS systems focus on solving a single domain and try to create only a single domain knowledge like C++ programming tutor and Java object tutor. Most of the solutions use isolated knowledge base systems, and these local databases can provide only limited knowledge background. The limitations of the isolated knowledge bases are lack of standardability, lack of shareability, lack of flexibility, lack of reusability, and a limited knowledge base.

Chapter 3 Formal Logical Ontology Model for an E-tutoring System

Many investigations in artificial intelligence (AI) and information science disciplines have focused on ontology. Ontology has emerged as a critical component of the semantic web, including many knowledge domains. Although many domain fields require ontologies, creating domain ontologies remains a big problem in the forms of design and execution. Information Science (IS) is one field in which a unified ontology model is required to simplify information access across diverse data resources and share a common understanding of domain knowledge.

The primary purpose of this research is to construct a general and flexible ontology domain knowledge model, which can be integrated with an E-tutoring system platform. Flexibility means allowing for extensions and modifications of the core model. Flexibility also refers to using the knowledge model in various domains and a dynamic and automated framework. Moreover, flexibility refers to the ability of the system to respond to any potential developments without affecting its value performance.

3.1 The Ontology Domain Knowledge Model for Teaching Materials

The domain knowledge model requires information about the study domain and teaching materials associated with the knowledge topics covered through the tutoring session [65]. The purpose of the domain knowledge model is to represent the knowledge content of an E-tutoring system. The domain knowledge model is used, among others, for storage, construction, and description of the learning material. The representation of the domain knowledge model is related to the semantic model of the teaching materials. A semantic model (SM) is a higher level of semantic-based representation and structuring formalism for the ontology database. SM specification describes an ontology database regarding the types of knowledge units in an E-tutoring framework, the classifications and groupings of those knowledge units, and the structural relations among them. SM provides a group of higher-level modeling principles to understand knowledge unit relations semantics in an E-tutoring application framework. In ontology engineering, semantic modeling has been implemented to create different learning knowledge units [76]. Semantic refers to the representation of knowledge unit meaning with symbols. Semantic modeling has been used to make sense of knowledge concepts, relationships, and what they represent in an E-tutoring system knowledge model [76]. The semantic model differs from conceptual data modeling, physical data modeling, logical data modeling, and modeling process rules in many aspects. The difference extensively depends upon the specific knowledge unit use case and the desired objectives from each knowledge unit use case [76]. Semantic modeling is applied to describe the relationships between specific types of data. Modeling is the method of categorizing knowledge for community use. Modeling keeps this in three forms: It provides a framework for human communication, a means for demonstrating conclusions, and a structure for controlling different types of knowledge. Semantic modeling is a continuous process for representing knowledge. The knowledge should be well-structured and understandable, and these structures can be represented in the modeling language.

Teaching materials are one of the essential educational and learning activities and components. In addition, the instructor can use teaching materials to support students in learning a knowledge domain through textbooks and visual or audio knowledge. Therefore, well-formed teaching materials will significantly strengthen and provide the primary purpose of students for learning the knowledge unit and their interest in the domain knowledge model.

Tutors promote the development of distinctive styles of teaching materials according to further requirements to support individualized and adaptive learning. However, traditional methodologies for designing teaching materials are time-consuming. To speed up the teaching materials' development process, we can use a modern approach based on the knowledge model's automatic generation and generalize and make it flexible. Consequently, due to the ongoing development of the technologies and knowledge domain requirements, scholars must be robust and try to produce a vast knowledge model for the teaching materials and make them available on the E-tutoring system.

3.2 Knowledge Model

A human's mental knowledge commonly begins with notifying and recognizing, and mental representation is contacted upon to build knowledge. In artificial intelligence, solving user needs requires a knowledge model consisting of all facts related to the problem domain and a way to manage the knowledge representation for obtaining the solution. For better results, knowledge should systematize sufficiently. There are two fundamental forms of knowledge: procedural knowledge and declarative knowledge.

Declarative knowledge (ontology), sometimes known as formal or descriptive knowledge, refers to information or facts held in the memory [77]. Declarative knowledge manages the actual or conceptual knowledge related to a human. In ontology, declarative knowledge relates to the aspects of material and strategy for the knowledge domains. It is a static description that captures the real world by interpreting concepts. Declarative knowledge, also called explicit knowledge, represents and explains textual, graphical, or verbal representation structures [78]. Because declarative knowledge deals with the study of facts, methods, techniques, and practices, it can be displayed, reported, and reproduced in the form of information to become clear knowledge assets. Declarative knowledge represents facts, events, operations, and relationships to a given domain. The combination of procedural knowledge (behavioural model) and declarative knowledge (ontology) provides access to existing domain-specific glossaries, taxonomies, and ontologies preferred to build the domain knowledge model.

Procedural knowledge (behavioural model) is a type of knowledge covered in different scientific disciplines and requires different techniques for its enhancement. This knowledge indicates production and is formed by preparing for the problem-solving approach [79]. In addition, procedural knowledge can be a procedure for a sequence of steps performing some activities to reach the target task [79]. Procedures are recognized by applying skills, strategies, results, and internal actions [79]. These procedures can perform in the way of algorithms, a predefined flow of actions, which leads to a successful response when done correctly, and potential actions that must sequence the problem-solving procedure (like equation-solution steps).

A well-designed structure for the knowledge domain requires adopting a proper methodology, especially in ontology representation. There are comprehensive study guidelines and frameworks for ontology building, development, and maintenance. I suggested an ontology knowledge domain model for an E-tutoring platform, which provides a suitable form for learning material representation. The presented model can be employed effectively for intelligent processing of teaching material and can form a base for interaction and collaboration supporting E-tutoring framework components. For knowledge representation, the ontology approach can used. Ontology is a good representation schema because it is common and describes the concept, knowledge, facts, properties, and relationship standardly. Using an ontology can reuse the knowledge domain stored on it. The suggested model can support solving the limitations of existing models. I will present the proposed model with a details description of the formal definition and structures in the following section.

3.3 Knowledge Representation Scheme

Knowledge Representation uses symbols to represent a collection of facts inside a knowledge domain to facilitate inferring facts to construct a new knowledge element. Knowledge representation plays an essential part in the artificial intelligence discipline. Knowledge representation is a technique applied in a computer software structure to define the knowledge base and allow AI mechanisms to perform well. In software design, representation of knowledge and manipulation has drawn an excellent deal of concentration since the earlier days of the computer science domain, resulting in various Knowledge Representation schemes. In addition, the Knowledge Representation scheme should be able to represent structural and relational knowledge naturally.

Although many kinds of knowledge representation (KR) schemes are found in the literature, two types are widely used, including single and hybrid KR schemes. The single schemes deal with the structured knowledge representations, and hybrid schemes integrate two or more single KR schemes. Single representation schemes represent the knowledge in the form of a graph. Accordingly, they are suitable for representing structural and relational knowledge. Single schemes involve several KR techniques: semantic networks, frames, knowledge graphs, ontologies, rule-based, case-based, logic-based and belief networks.

A semantic network defines a knowledge representation scheme that utilizes a graph notation to describe knowledge structure according to its concepts, instances, and properties corresponding to graph nodes. At the same time, the relationships between them (i.e., taxonomy) are determined via directed arcs. Formally, the *semantic network* is described as a tuple $< I, R_1 , R_2 , \dots , R_n , M >$, where I stands for information entities, $R_1 , R_2 , \dots , R_n$ Indicates types of relationships between knowledge entities, and M corresponds to a map that specifies the relationships between knowledge entities [80]. Figure 3.1 shows an example of semantic network knowledge representation scheme.

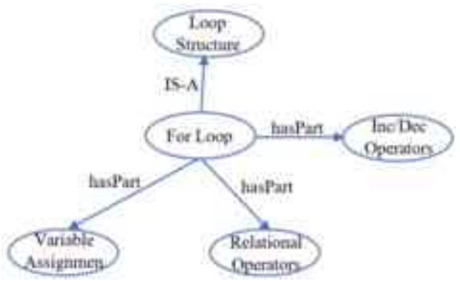


Figure 3.1 Semantic network representation example

A frame is an object-centered knowledge representation structure that builds its ontological commitment to objects, properties, values, and relationships among them. As a result, it is often called an "object-property-value" representation. Objects are a standard way to organize knowledge about physical items and situations. What is of great importance is that objects allow a collection of strategies for specifying their properties and the properties of other objects. The object-centered representation can be achieved by merging many properties of the object of the same type into one structure. In frame-based systems, an object that contains properties filled by values corresponds to a "frame" that contains "slots" filled by "fillers"; therefore, the frame can be defined as a structural and relational representation of an object [80]. Each frame has a name and includes a set of slots with the associated fillers. For example, the filler could take a value of the primitive types (number, string, date, etc.), complex types (set, list), or serve as a reference to another frame. Based on the nature of a description object, two types of frames are distinguished individual and generic. The individual frame represents a single object, an instance of some category, while a generic frame can define an abstract category [80]. Consequently, individual frames usually have a particular slot called "INSTANCE-OF," The filler is a title of (or a pointer to) an appropriate generic frame. In contrast, generic frames may have a particular slot called "IS-A" with a filler that points to a more generic frame. These unique slots are crucial since they enable an inheritance mechanism. Figure 3.2 displays an example of frame knowledge representation scheme.

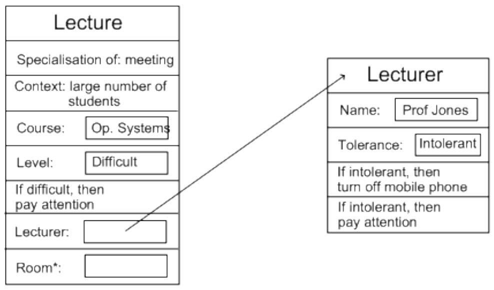

Lecture
Specialisation of: meeting
Context: large number of students
Course: Op. Systems
Level: Difficult
If difficult, then pay attention
Lecturer:
Room*:
Lecturer
Name: Prof Jones
Tolerance: Intolerant
If intolerant, then turn off mobile phone
If intolerant, then pay attention

Figure 3.2 frame-based representation example

A knowledge graph considers a type of graph-based data model used to capture the knowledge in application systems, such as the management, integration, and extraction of information from various sources of knowledge at a large scale [81]. A graph-based abstraction of the knowledge base offers some benefits over alternatives like NoSQL or relational model. Graphs provide a brief and intuitive abstraction of diverse domains, where edges and paths represent various, potentially complex relationships among entities [82]. Graphs allow developers to delay the description of a scheme while the information grows more flexibly. A knowledge graph classifies as a type of structured knowledge representation that can represent knowledge, facts or specific domains, and this graph includes entities, semantic descriptions, and relationships. In the graph, entities can be real-world objects and abstract concepts. Relationships describe the relationship among entities and the semantic descriptions of entities, and their relationships include classes and properties with well-defined meanings. The knowledge graph is considered a knowledge base with minor differences, and it can be used as an interpretation and inferences over facts or given domains [83]. Figure 3.3 illustrates an example of knowledge graph representation scheme.

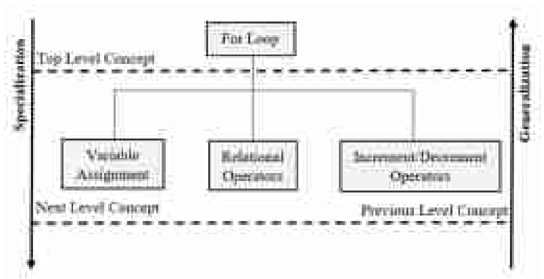


Figure 3.3 knowledge graph representation example

Ontology is an important technology that enables the collaboration of appropriate content, sharing knowledge, and avoiding irrelevant knowledge. Ontology is a modern knowledge representation scheme basis on a conceptualization. Conceptualization is a simplified and abstract view of a specified domain, including the concepts, objects, and other entities in the domain of interest and their relationships. Ontology can be used for reasoning in the target domain and validating the developed semantic model. Specialization, generalization, association, and containment are essential concepts in knowledge representation and modeling, which help to develop a relationship between superclass and subclass. Ontology can be classified according to the levels of representation into; Catalog, Glossary, Taxonomy, Thesaurus, DB/OO scheme, and Axiomatic theory. Ontology is considered the primary approach for knowledge representation, especially in E-learning systems. Figure 3.4 explains an example of the ontology-based knowledge representation scheme.

Many logic languages offered by ontology are utilized for knowledge representation, and reasoning, the most considered are Description Logic, ALC DL, SROIQ, DLV, and SHOIN). Description logic is a formal language developed to represent and reason knowledge. Description logic is used to model the roles, concepts, and individuals and their relationships. ALC DL language, which stands for Attributive Language with Concept negation, includes concepts, roles, constructors, and individuals [84]. SROIQ is one of the DL languages which are the most expressive and commonly considered for knowledge description. However, it also primarily compromises expressivity with OWL 2 DL ontology language. DLV stands for Deductive DataLog system. DLV was introduced as disjunctive and logic programming, widely considered a new way of implementation and managed several characteristics. DLV usage as a mechanism for representation and reasoning the knowledge and describes a unique declarative programming procedure that encodes complex problems. Finally, SHOIN is a description logic language underlying OWL-DL's web ontology language [84]. An ontology with a collection of individual class instances creates a knowledge base. In the following demonstration of the essential elements of the commonly used notation in the ontology:

- Class (concept) – stands for a concept in the domain of interest.
- Instance (individual) – represents an individual instance of a specific class.
- Slot (property, role) – describes a feature or an attribute of a concept or individual.
- Facet (role restriction) – represents a restriction on a slot.
- Relations – describes how classes and instances can be related to each other.
- Rules – "if-then" sentences define possible logical inferences from an assertion.
- Axioms – are assertions in logical forms that comprise an overall theory of a domain of interest.

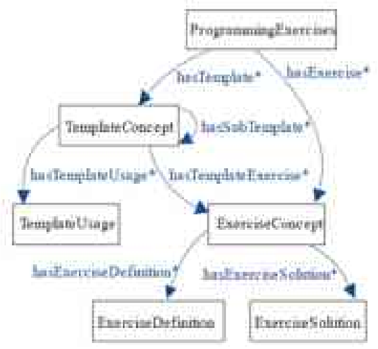


Figure 3.4 Ontology-based representation example

The rule-based technique is a knowledge for a particular discipline represented in the Production rule structure, sometimes referred to as a rule, that includes two-part statements that incorporate small parts of knowledge [85]. The first component of the rule is named antecedent, which describes a situation or assumption. In contrast, the second component, called consequent, displays a specific action or conclusion if the situation or assumption is clear. The first or left-hand component of the rule is a statement (sometimes called action or conclusion). The second or right-hand part of the rule is the statements (sometimes called the expressions or conditions), a clause structure in the First-Order Predicate Logic. The *rules-based* is one of the most popular KR methods. They represent the given knowledge domain in the form of rules using the if-then format: if <conditions> then <conclusion>, where the term <conditions> represents the conditions of a rule, whereas the term <conclusion> represents its conclusion. Intelligent Tutoring Systems adopt rule-based reasoning applying logical connectives such as AND, OR, NOT, and so on to form logical functions. The rule-based supports the tutor in ways of what strategy is used in tutoring a different style of student level. It diagnoses the learner's progress and specifies the level of each learner. It will provide information to the tutor for producing the questions given to the learner for further tutoring. Rule-based has some drawbacks that it depends on the used inference mechanism, and its structure does not help in allowing the system to work either in forwarding chaining or backward chaining [86]. Therefore, if the current rule fails, the standard rule structure does not help guide the system that rules to be tried [86]. Researchers attempted to solve these problems by inventing many solutions like the ripple down rule. If a rule fails, the system can know its exception rule, and if it is fired, the system can know the next rule. Another solution is the Hierarchical Rule (HR) structure, which can help knowledge representation in many systems, especially in ITS [86]. The following example in Figure 3.5 denotes the rule-based knowledge representation scheme.

(Rule name <rule name>	{each rule has a unique name}
IF <condition>	{precondition (AND) condition}
THEN <action/process/direction>	{concept/decision/head}
GENERALITY [G]	{general information/parent rule}
SPECIFICITY [S]	{specific information/more specific rule}
EXCEPTION [E]	{the rule to tried in case of failure})

Figure 3.5 Rule-based representation example

The case-based scheme is the knowledge of a specific domain representing the collections of cases, each case describing the problem and its solution [85]. The approach used for the representation of any case is via the common characteristics for all problems; these characteristics are Initial state (s), Goal state (s), Processes: The process of transforming from one state to another in the problem area, and Problem space: Usually, the start consists of the Initial state (s), goal state (s) and all the assertions given.

Many techniques can be used to represent the problem space, such as a directed graph. The problem solution will represent the path from an initial state (s) to the goal state (s). Therefore, any case can represent a problem space, which consists of the initial state (s), goal state (s), path solution and the directed graph together with all processes that have been used to find the solution. Case-based representations store a massive set of past cases with their solutions in the case base and use them to solve a similar new case. Figure 3.6 illustrates the case-based knowledge representation scheme for different cases.

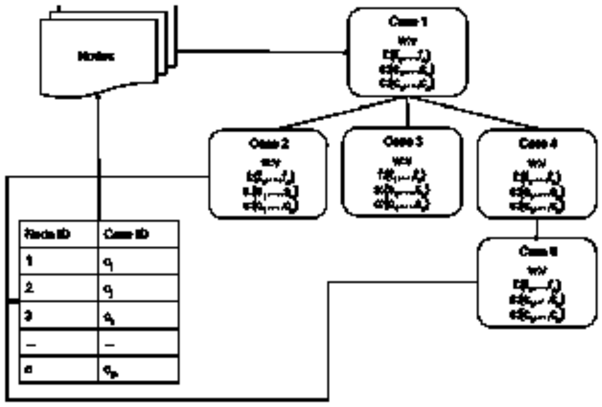


Figure 3.6 Case -based representation example

Belief networks (or probabilistic nets) are graphs where nodes represent statistical concepts and links mainly represent causal relations. Bayesian Belief, also called Bayesian Network, is introduced as a graphical model based on probabilistic that defines conditional dependencies among random variables via a so-called Graph with a Directed Acyclic. The Belief networks (BN) are a highly flexible graphical and probabilistic modeling framework. It is explicitly developed to describe conditional independence among arbitrary variables of the domain interest and manipulate this knowledge to facilitate the complexity of probabilistic inference [87]. In addition, BN is a controlled model of conditional dependency among a collection of arbitrary variables. The conditional independence takes in a belief network by arranging the variables and outcomes in a directed graph. Formally, BNs are represented using a directed cyclical graph where variables appear as nodes and probabilities appear as links between nodes. In practice, a priori and conditional probabilities are defined using heuristically defined rules by the experts, and they can be derived from the empirical data. If each node K; in the network is associated with a conditional probability table, it can specify the probability distribution of the associated random variable, given its direct parent nodes Parents(Ki). Then the BN delivers a close representation of the combined probability distribution over all network variables that are represented by the following equation [87]: $P(K_1, \dots, K_n) = \prod_{i=1}^{n} P(K_i | Parents(K_i))$. Figure 3.7 shows an example of the belief networks knowledge representation scheme.

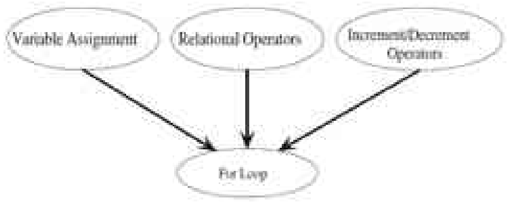


Figure 3.7 Belief networks representation example

Integrating two or more different knowledge representation techniques is an active investigation area in Artificial Intelligence. It generally accepts that complex problems can more easily be solved with hybrid approaches. The purpose is to construct hybrid formalisms satisfying each of their components. Hybrid KR schemes include several methods such as fuzzy rules (Fuzzy logic) and description logic, neurules, rules and case based.

3.4 Evaluation of existing knowledge models

Table 3.1 displays a comparison according to some criteria or features for the suggested model with others in the literature. These features covered reusability, standardability, flexibility, open knowledge, simplicity, reasoning engine, and uncertainty.

These features are the main requirements in implementing an efficient ITS systems, I used these features in the comparison because they are the most commonly used features in developing a standard domain knowledge model. I focus on solving these issues and propose a general model to cover the goals of supporting the teaching and learning process. These goals include understanding specific domain facts and solving the task activities in learning, obtaining a conceptual and intuitive understanding of the material in the selected domain, and learning general problem-solving and metacognitive skills. A significant feature of our model is that the knowledge representation techniques have a standard structure. However, this standard structure allows us to apply general representational inference tools and control mechanisms in order to facilitate the pedagogical analysis of the knowledge domain.

Table 3.1 Comparison of the knowledge representation models

	Reusability	Standardability	Flexibility	Open knowledge	Simplicity	Reasoning engine	Uncertainty
Knowledge graph	√	√	×	√	×	√	√
Semantic networks	√	√	×	√	×	√	√
Rule-base	×	√	×	×	×	×	√
Case-based	×	√	×	×	×	√	√
Bayesian Network	×	√	×	×	×	×	√
Frame-based	√	√	×	√	×	×	×
Logic-based	√	√	×	×	×	√	√
Ontology-based	√	√	√	√	√	√	√

√ Means the feature is allowed, and × means the feature is not allowed.

Most ITS systems developed in the literature focus on solving a single domain. They try to create only single courses like introduction to computer programming tutor and Java object tutor. Most of them use isolated knowledge bases, and these local knowledge bases can provide only limited knowledge background. The limitations of the local knowledge bases are lack of reusability, lack of standardization, lack of flexibility, simplicity, reasoning engine, uncertainty, and it has limited knowledge base.

3.5 The Proposed E-tutoring System Architectures

In this research, I contributed to the development of the extended ITS architecture and based on this experience I later developed the adaptive ITS architecture based on the current research directions and trends presented in the previous work. Besides the standard models, the proposed architecture also includes a common shared database and knowledge-based background. The advantages of the shared database are to share a common understanding of the knowledge structure, reuse the knowledge, and mix different knowledge bases.

3.5.1 The Extended ITS Architecture Model

The extended ITS architecture model proposed by Walelign et al. [26] utilises a shared database. The main building blocks of an extended ITS architecture are shown in Figure 3.8. The suggested model varies from the traditional models regarding the factors: external Knowledge model, extending learner and tutor ML components, Explicit knowledge of ontology, Using NLP engines, and Question Generation model. The proposed extended architecture model incorporates a common shared database and knowledge-based background besides the standard modules. The advantages of the global database are sharing a common understanding of the knowledge structure, reusing the data, and mixing different sources of knowledge. In knowledge management, ontology offers a common vocabulary that can model various domains involving the type of object, related concepts, and properties and their relations. The shared knowledge base model may involve external training sets that can be used as input data in different data mining techniques like a neural network. The external knowledge bases can enhance the behavior models' quality in both tutor and student models.

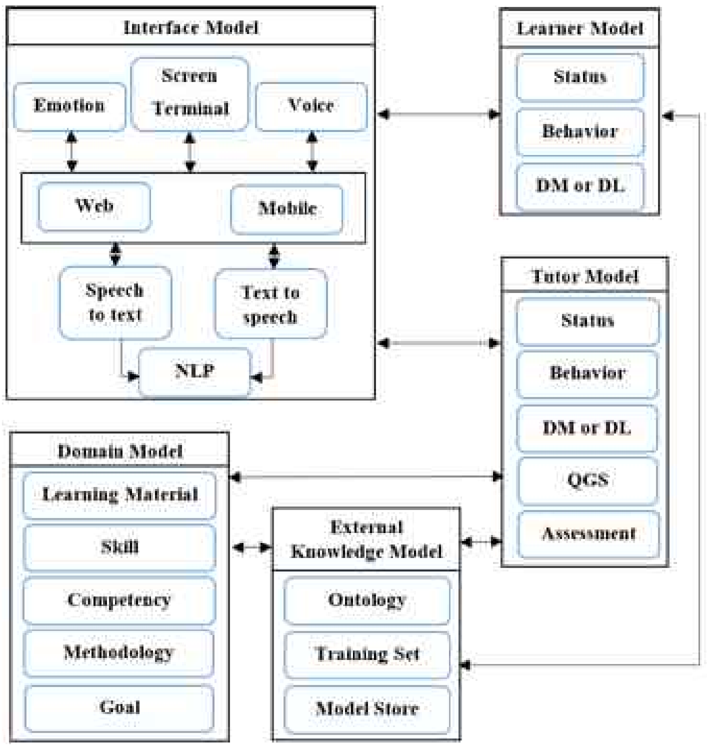


Figure 3.8 Extended ITS Architecture [26]

3.5.2 The Adaptive E-tutoring System Architecture Model

Based on the current research directions and trends presented in the previous work, I propose an adaptive E-tutoring system architecture shown in Figure 3.9. Beyond the standard models, the proposed architecture also includes a common shared database and knowledge-based background, too. The advantages of the shared database are to share a common understanding of the knowledge structure, reuse the knowledge, and mix different knowledge bases. Furthermore, ontology delivers a shared language in knowledge databases, which can model different domains, including the type of entities, related concepts, their properties, and relationships.

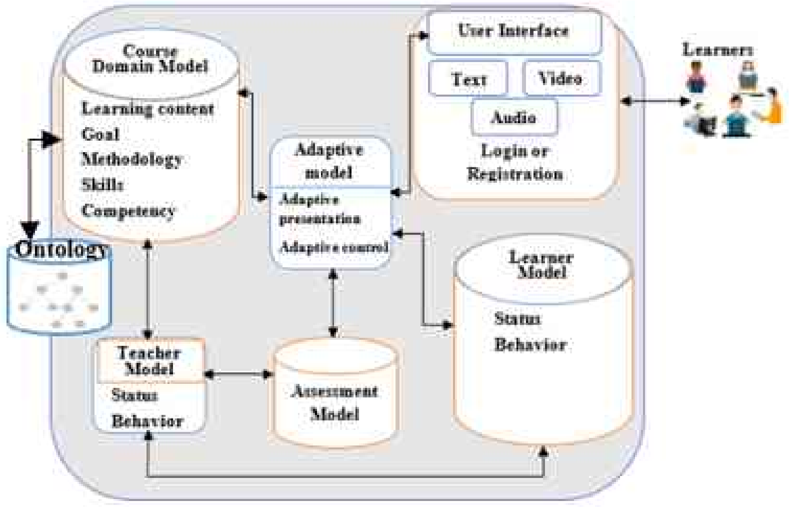


Figure 3.9 Adaptive Architecture

The main differences between the extended ITS architecture and adaptive E-tutoring architecture mentiond in sections 3.5.1 and 3.5.2 is that the adaptive architecture includes a common shared database and knowledge-based background and also it has adaptive model to mange the learning process and assessment model for evaluating the learner's knowledge level.

3.6 The Proposed Ontology Models

In this research, I have developed a model for the purpose of enhancing the learning and teaching processes in 3 ways. These models are the ontology-based model to support the learning process in LMS, the ontology-based Knowledge domain model for the IT domain in E-tutoring systems, and the ontology-supported domain knowledge module for an E-tutoring system. The main goals of these models are to enhance the learning and teaching process, support recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of new material.

3.6.1 Ontology-based Model to Support Learning Process in LMS

In this work, an ontology-based model is introduced to facilitate the Learning Management System (LMS) in the learning processes (LPs) [88]. The suggested domain ontology model is a general framework, and the main goal is to discover the behavior of learners and cover the entire learning process (LPs) and its components. To support the LPs, I developed an ontology model to define a set of relationships that would be sufficient and clear to represent all relationships for building the ontology model. The presented model structure contains three layers of LPs, the top layer architecture, the conceptual layer architecture, and the course ontology model architecture.

The power of the learning platform is to recognize and adjust the learner's context plays an essential part in personalized learning. The ontology-based model describes the learner's properties, including essential details regarding domain knowledge, learning performance, interests, preferences, goals, tasks, and personal characteristics. It can identify the learner's context and modify the learning process accordingly. It will also be helpful for cognitive awareness and adaptive learning.

The top layer architecture to define a general framework of LPs to the learners is shown in Figure 3.10 In addition, this framework consists of selected objects:

- The learners who want to learn about a particular subject or how to do something.
- Interface: should be well-designed so that individuals (users) efficiently interact with the computer. It can include both hardware and software components [89].
- Learning media refer to forms that can deliver a message, promote thought-feeling, and facilitate learning. Learning media is a means for channeling learning messages and information. Well-designed learning media will hugely help learners achieve learning objectives [90].
- Knowledge unit means the model of concepts, principles, theories, and practices related to a course to learn or study. It is described as theoretical or concept knowledge.
- A presentation unit is a technique that teachers could use to improve student's learning and help better regulate their learning through effective learning techniques [91].

- The presentation form refers to the material resources used in the learning processes and to assist the learners in obtaining knowledge and profiling different capabilities and values.
- The evaluation form represents the last stage of working with teaching and learning resources. The purpose of an evaluation is to evaluate the effectiveness of particular teaching and learning resources in attaining the goals and objectives of teaching; in other words, its contribution to understanding, connecting and interpreting, developing desirable skills, and adopting specific values of students.
- The learning model is to interpret student behaviors and then distinguish students according to motivations, habits, forms, skills, and competencies [92].
- Motivation refers to the factors that stimulate the desire and energy of learners to be continuously interested and dedicated to a learning role or topic or to construct an effort to achieve a goal.
- Habits refer to a typical way of behaving or a tendency that a learner has settled into, as in "good learning habits."
- Forms refer to a supporting frame model of the student figure or part of the learner formation used for learning material.
- Skills refer to the power of learners to use one's knowledge effectively and readily in execution or performance in a learning process.
- Competences refer to the knowledge that enables a student to learn and understand a subject.

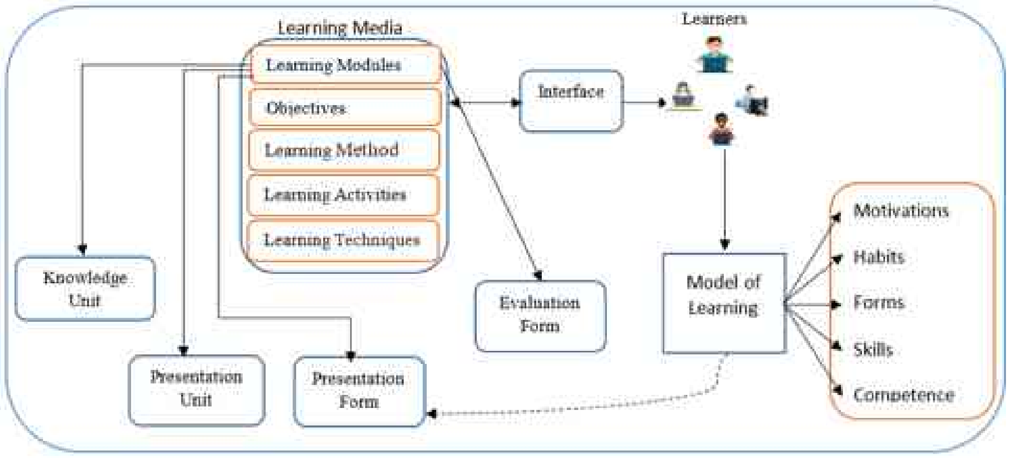


Figure 3.10 Top layered architecture [88]

The conceptual model layer is to define the framework of the actual process of LPs to the learners displayed in Figure 3.11. This framework contains four components knowledge unit, presentation unit, presentation form, and evaluation form.

- the knowledge unit, which contains the presentation of learning unit in the form of definitions, methods, applications, activities, rules, and techniques of the LPs.
- the presentation unit, which contains the presentation of the lesson module in the form of examples, explanations, exercises, assignments, projects, and discussions of the LPs,
- the presentation form which contains the presentation of the learning materials in the form of text, video, audio, visual-silent, and audio-visual to present the LPs.

• the evaluation form which contains a quiz, calculation, multi-choice question, task project, and participation of learning module and the relationship between the different units.

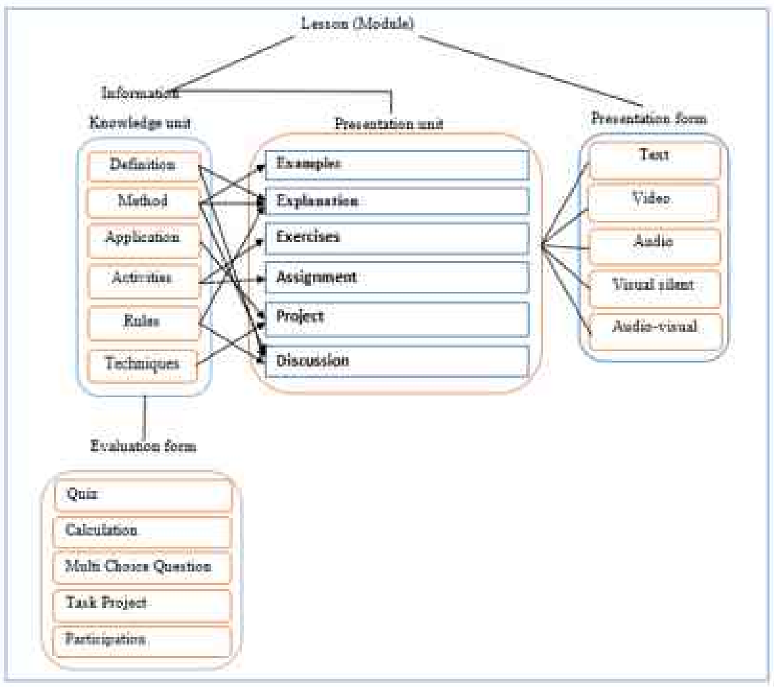


Figure 3.11 Conceptual model layers [88]

The course ontology model refers to a course in the education domain. An education course is a unit of teaching offered by an educational institute or organization to a group of learners and directed by one or more instructors [93]. For the variety of domains in which different courses are involved, the development method of ontology is also not the same. So, there is no standard method to build ontology models. Regarding reusability and maintainability, an ontology engineering methodology developing phases is followed to develop course ontology. In addition, considering the particular domain of E-learning courses, with different users and different education levels focused. The objective of this model is to define the domain ontology model of the course ontology. The model consists of the students, the teacher, and the primary components of a course such as learning objectives, teaching methods, learning contents, learning media, and assessment, and then the other components of the course structure displayed in Figure 3.12.

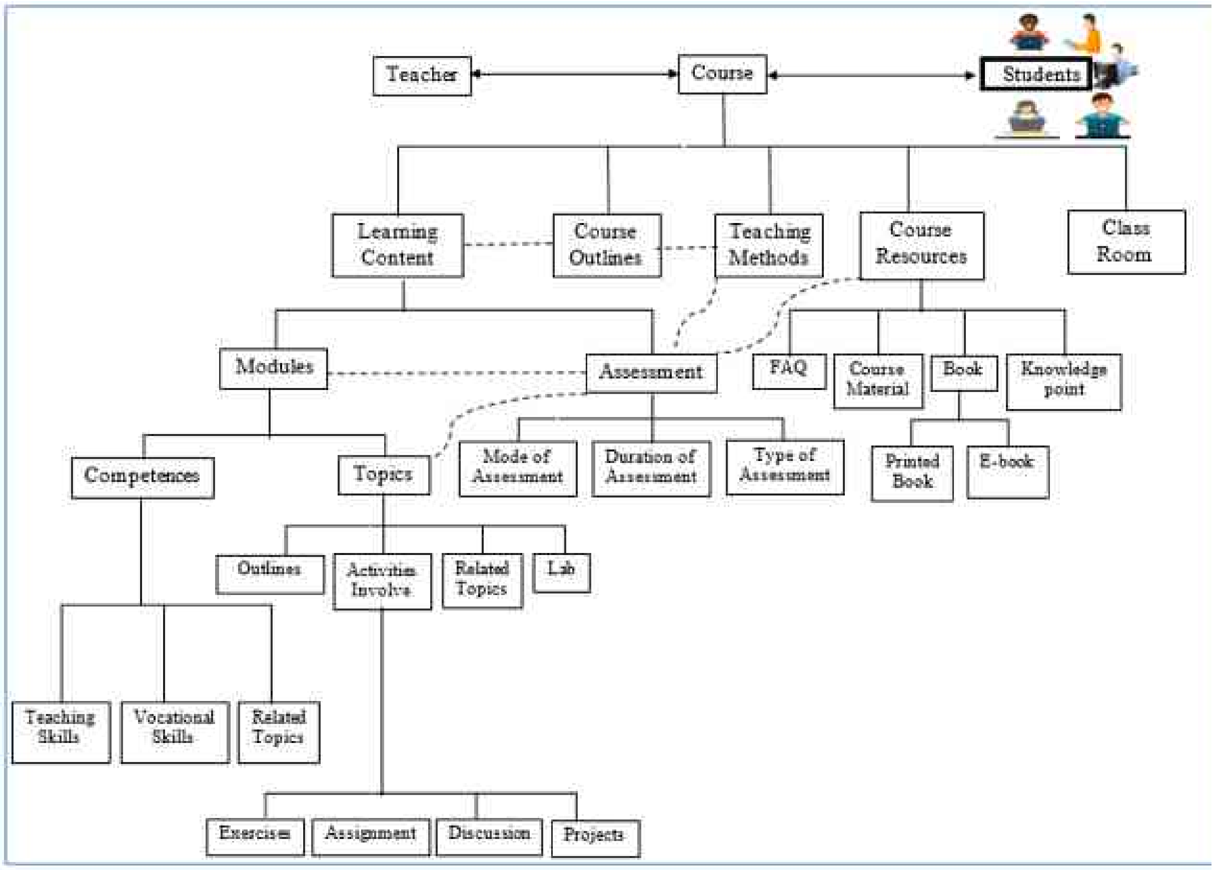


Figure 3.12 Course ontology model [88]

3.6.2 Ontology-based Domain Knowledge Model for IT Domain in E-tutor Systems

In this research, I have suggested an ontology-based Knowledge domain model for the IT domain in E-tutoring systems. Depending on the properties of the learning materials, two types of ontologies are implemented as a form of general concepts: domain knowledge ontology and specific domain knowledge ontology. These modules represent the knowledge to be learned, deliver input to the expert model, and eventually provide detailed feedback, select problems, generate guidance, and support the student model. The introduced domain knowledge model is constructed based on the current research directions, as shown in Figure 3.13. The presented model suggests the topics, concepts, attributes, tasks, competencies, assessments, and relations. To facilitate the sharing and reusing of the domain knowledge model features in E-tutoring systems, ontologies are utilized to organize and represent the domain knowledge model. The benefit of this model is to personalize the materials for learners.

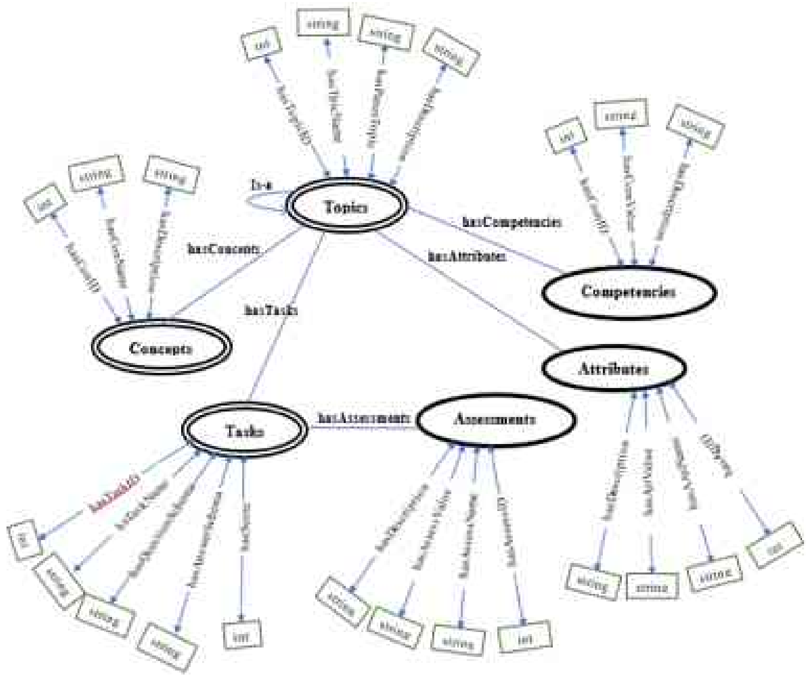


Figure 3.13 The ontology-based domain knowledge model schema

In the ontology model double line denotes core concepts and single line means core properties. According to the key concepts of the domain knowledge ontology shown in Figure 3.13. The core components of this model are topics, concepts, attributes, tasks, competencies, and assessment terms refer to the following definitions:

- Topics refer to present domain knowledge or a comprehensive overview of a subject or course.
- The concept identifies the sub-domain or unit of a subject or course.
- Competency refers to a priori competencies required for understanding the topics and demonstrating the features and skills that allow and enhance the efficiency of student performance to gain new knowledge. In other words, the competencies gained by completing the topic (can be measure after completing a knowledge unit)
- The task refers to representing how a student can complete a task within a given period of time.
- The attribute represents a topic or domain attribute within a domain model.
- The assessment refers to presenting how the system can evaluate or assess the student activities required within a given period of time.

Figure 3.14 displays the design of a specific domain knowledge case study for the IT domain (computer programming) in an E-tutoring system. I use different types of relationships in the case study, such as specialization or generalization, association, and containment. Containment means that a specific topic within a domain contains different concepts (has-a for example a topic can have a topic part). The specialization or generalization means that certain topics or domains have specific concepts (is-a for example a topic can contain another topic). The association means that a specific topic or concept is associated with another topic.

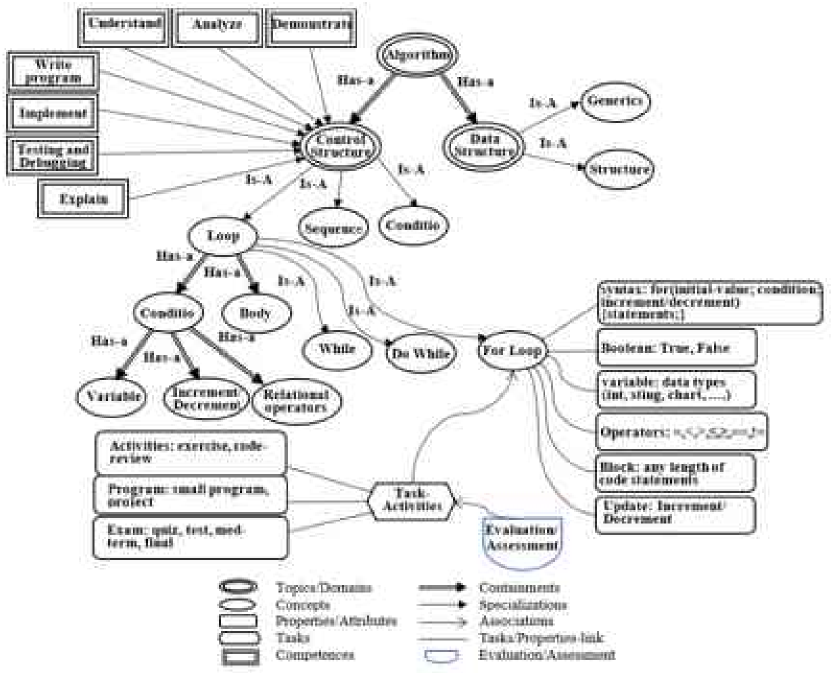


Figure 3.14 The domain knowledge module for E-tutoring frameworks

Based on Figure 3.13 and Figure 3.14, the following list shows a brief description of a control structure subject:

- *Topic*: Control Structure.
- *Concept*: Loop, Sequence, and Condition.
- *Competency*: understand, analyze, implement.
- *Task*: program, code review, project
- *Attribute*: syntax, operators
- *Assessment*: activities such as quizzes, tests

3.6.3 Ontology Supported Domain Knowledge Module For E-Tutor Systems

In this research, I suggested an ontology-supported domain knowledge module for an E-tutoring system based on the properties of the learning materials. Two styles of ontologies were introduced: a) general concepts for domain knowledge module ontology and b) specific domain knowledge module ontology. These modules describe the topic to be studied, provide input to the domain module, provide specific feedback, select problems, generate suggestions, and support the learner module. The underlying construction of the suggested domain knowledge module is demonstrated in Figure 3.15. The model is based on topics, attributes, task assessments, material forms, learning levels, learning rules, and relations. In addition, these model components are designed to support the features of shareability, standardability, flexibility, and reusability for the knowledge domain module. These features can be integrated with the E-tutoring platforms, and I utilized ontology to manage and represent the domain knowledge module. The benefit of this model is to personalize the material forms, make suggestions, and automatic assessments for students.

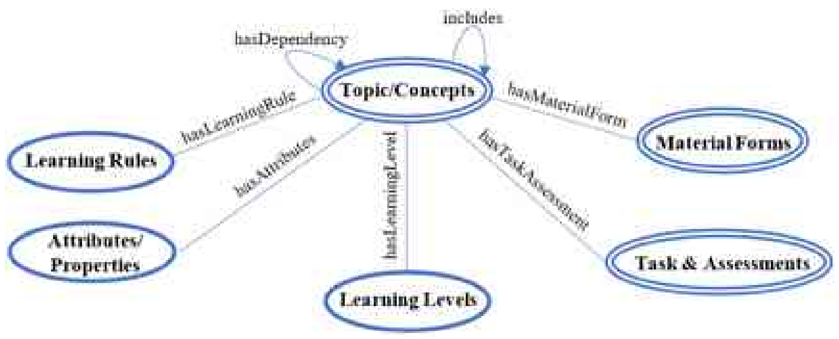


Figure 3.15 The suggested domain knowledge module

Based on the general concepts of the presented domain knowledge module ontology displayed in Figure 3.15, the suggested components are: topics, attributes, task assessments, learning levels, learning rules, and material-forms terms refer to the following:

- The topic refers to knowledge modules as a unit of instruction representing domains, key concepts, or education units of the learning materials.
- The attributes refer to slots as an atomic property of the topics; every slot has a value domain and concept type.
- The learning rules refer to rules or constraints defined on the topics and the attributes. In addition, a learning rule is a group of explicit or implicit ontology constraints or principles managing behavior or procedure in a particular activity area.
- The task & assessments refer to Task assessment is the activity when the system evaluates the student's solution for a given task. Also, task assessment describes an activity to be performed by learners.
- The material forms refer to teaching material for the topic. Moreover, material forms are teaching materials used to learn the topic. In addition, material forms include any parts of the academic institution or education material.
- The learning level is defined by the developmental differences of students and how the learning environments are structured.

Regarding the primary relationships, the ontology model contains the following elements: Topics taxonomy relationship: it defines the specialization among the topics. Topics component relationship: one topic consists of other topics. Topics-competency relationship across the topics. Figure 3.16 indicates the case study structure of specific domain knowledge module ontology for the world history domain in the E-tutoring system. Various types of relationships are used in the case study, such as specialization or generalization, association, and containment. Containment denotes that a specific topic within a domain includes different concepts (has-a). The specialization or generalization indicates that the topic has specific topics (is-a). Finally, association means a specific topic associated with attributes/properties, material forms, and task assessments.

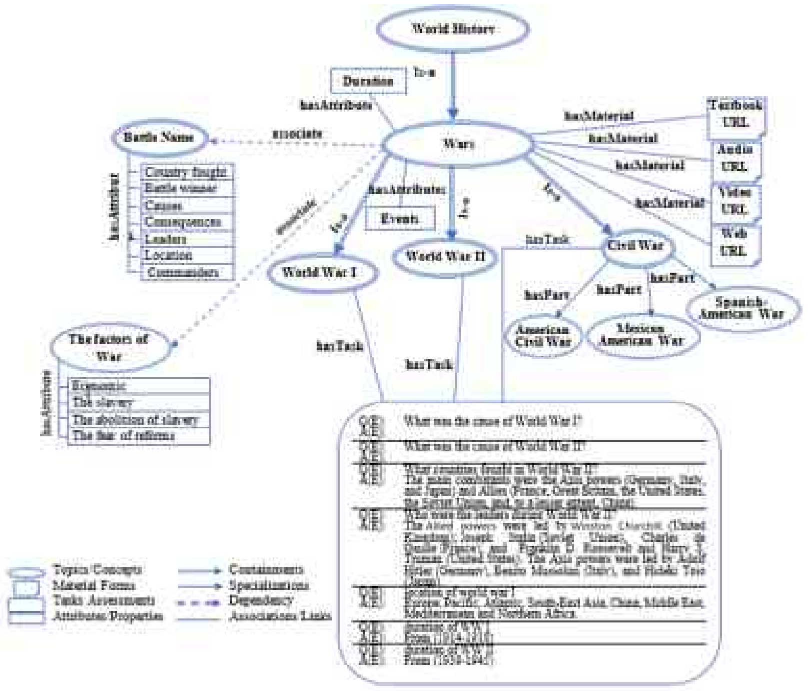

World History
Is-a
Duration
hasAttribute
Battle Name
associate
Wars
Country fought
Battle winner
Causes
Consequences
Leaders
Location
Commanders
hasAttribute
hasAttributes
Events
Is-a
associate
World War I
World War II
hasTask
Civil War
hasMaterial
Textbook URL
Audio URL
Video URL
Web URL
hasPart
Spanish-American War
American Civil War
Mexican American War
The factors of War
Economic
The slavery
The abolition of slavery
The fear of reforms
hasAttribute
hasTask
Q:
A:
What was the cause of World War I?
What was the cause of World War II?
What countries fought in World War II?
The main combatants were the Axis powers (Germany, Italy, and Japan) and Allies (France, Great Britain, the United States, the Soviet Union, and, to a lesser extent, China).
Who were the leaders during World War II?
The Allied powers were led by Winston Churchill (United Kingdom), Joseph Stalin (Soviet Union), Charles de Gaulle (France), and Franklin D. Roosevelt and Harry S. Truman (United States). The Axis powers were led by Adolf Hitler (Germany), Benito Mussolini (Italy), and Hideki Tojo (Japan).
location of world war I
Europe, Pacific, Atlantic, South-East Asia, China, Middle East, Mediterranean and Northern Africa.
duration of W.W I
From (1914-1918)
duration of W.W II
From (1939-1945)
Topics/Concepts
Material Forms
Tasks Assessments
Attributes/Properties
Containments
Specializations
Dependency
Associations Links

Figure 3.16 Domain knowledge module sample

Based on Figure 3.15 and Figure 3.16, the following list displays a brief description of a World History domain:

- *Topic*: World History, WW I, WW II, Civil War.
- *Dependency*: Battel Name, Factor of War.
- *Task Assessment*: Question/Answer.
- *Attributes*: Events, Durations, locations.
- *Material Form*: Web, Text, Media.

3.7 Formal Description of the Knowledge Model

In this section, I introduced the formal definition of the proposed knowledge domain model, which presents the conceptual components of teaching materials frameworks. The domain knowledge model is presented as a coloured graph (or knowledge unit taxonomy) with different node elements and relationships among these nodes. The core concept elements of this model are knowledge units, knowledge slots, task units, rule units, material units, and educational units covering the specific teaching and competency dependency units. I also introduce many types of relationships among the knowledge units in the domain ontology. These relationships are specialization or generalization relationships, component relationships, association relationships, and dependency graph relationships. I will explain this model in the remaining section.

The domain model Δ is given with a tuple $\Delta\ (\Sigma, T, \Pi, \Phi, M, \Lambda, \preccurlyeq, \sqsubseteq, \Rightarrow, \epsilon)$, where:

- Σ : the set of knowledge units (domains, knowledge topics, teaching unit, and key knowledge concepts)
- T : task units (activity, competency related to the knowledge units)
- Π : knowledge slots (property of the knowledge units)
- Φ : rule units (constraints on property values and knowledge units)

- M : material units or teaching materials (textbook, media (video, audio), web)
- L: educational units (learner level(
- Λ : entity type-set (Name, Attributes, on the knowledge units)
- $\preccurlyeq\ \subseteq\ \Sigma \times \Sigma$: generalization, specialization (among the knowledge units)
- $\sqsubseteq \subseteq\ \Sigma \times (\Sigma \cup \Pi)$: components, the containment relationship on the knowledge units and slots; which means the knowledge unit includes some other knowledge units or slots.
- $\Rightarrow \subseteq\ \Sigma \times \Sigma$: dependency graph, the competency level dependency on the knowledge units
- $\sim\ \subseteq\ \Sigma \times \Sigma$: association (among the knowledge units)
- $\epsilon\ \subseteq\ \mathrm{T} \times \Sigma$: task – Knowledge-units assignment (activity, assessment on the knowledge unit)

3.7.1 Knowledge Units Description

Knowledge units (K-units) refer to a unit of education and it denoted as domains, knowledge topics, teaching units, key knowledge, and key concepts. K-units are fundamental building blocks of the teaching materials. The K-units are a vital component of domain knowledge, which includes different attributes such as name, complexity levels, presentation style, detail, or explanation level. Every k-unit is organized into a k-units group. The K-units group includes topics that are closely related to k-units of a specific field. Also, K-units are information that applies to illustrate Knowledge content, ideas, overviews, and workflows. The K-units are more objective, including definitions, rules, and guidelines.

The set of knowledge units is denoted by $\Sigma = \{u\}$ where u is a knowledge unit(k-units). K-units refer to a unit of education that defines a course, knowledge topics, domains, key concepts, or teaching units of the teaching materials. in other terms, K-units are the general basic building block of the domain, indicating knowledge topics, teaching units, and key knowledge concepts related to a domain for learning or studying. It is defined as general concept knowledge. K-units are parts of a knowledge hierarchy, and one k-unit element can contain many other k-unit elements.

In the graph representation format, the k-unit elements are denoted by circles with thick double borderline. Figure 3.17 shows an example of a loop structure k-unit; in this example, I connected different k-units concepts related to the loop structures and the relationships between these k-units concepts.

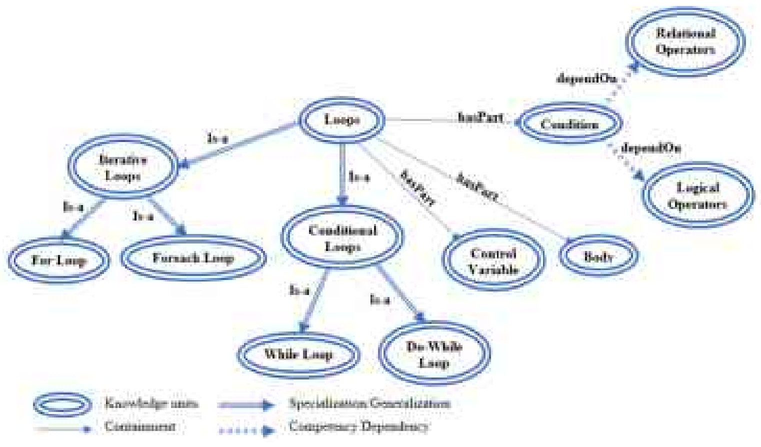


Figure 3.17 Knowledge units example

3.7.2 Knowledge Slots Description

knowledge slots (K-slots) refer to an atomic property of the k-units. Every slot has a value domain and a value range. K-slots can have various facets defining the value type, allowed values, the number of the values (cardinality), and other characteristics of the values the k-slot can consider. Slot cardinality specifies how many values a slot can hold. Slots, also named attributes of properties, represent the concept features. The set of all slots is defined as $\Pi = \{p\}$. The standard facets involve:

- Slot cardinality determines the number of slots values. Some applications offer single and multiple cardinalities.
- Slot value type defines which kinds of values can fill in the slot, for example string, Boolean, float, etc..

Every slot p can be assigned to many different knowledge units. The symbol

$$p \sqsubseteq u$$

Denotes that p is valid for unit u. The set of all slots for unit u is given by Π_u . The graphical symbol for a slot is a rectangle with a single thin borderline. Figure 3.18 indicates an example for loop structures k-units and k-slots; in this example, I presented different k-units and k-slots related to the loop structures and the relationships between these k-units and k-slots.

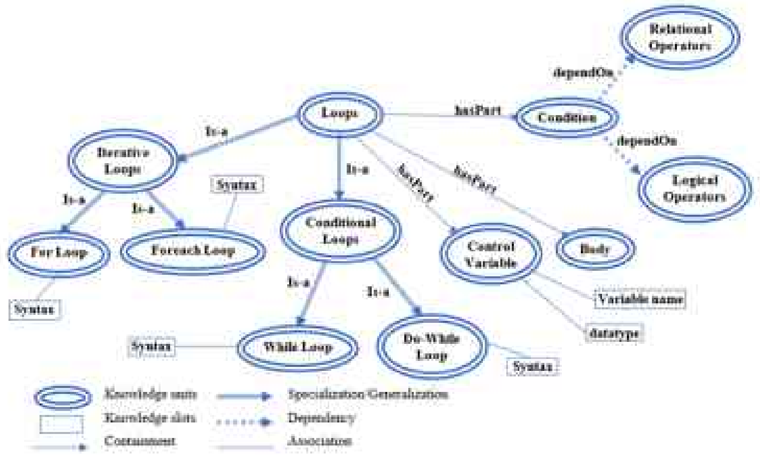


Figure 3.18 Knowledge slots example

3.7.3 Task Units Description

Task units (T-units) refer to activities related to the k-units and k-slots. T-units can be represented in the form of activities performed by the learners. It contains a task type followed by a list of arguments. It also can be given as a question-answer pair. Syntactically, a T-unit contains a task type followed by an argument list. The T-units may be either primitive or compound. A primitive T-unit was considered to be performed by beginner learner, for example print “hello five times”. A compound task requires separation into smaller tasks using a method; any method whose title unifies the task type, and its arguments may probably be suitable for satisfying the task unit, for example write a program to count the student result for six subject. A Task is aimed at a procedure that defines how to accomplish it. A Task is a combination of steps users follows to produce an expected outcome.

The task can be represented as a pair of questions Q and answer A. The set of all T-units is denoted by $\mathrm{T} = \{t\}$ where t is a T-unit, the T-unit is given as a question-answer applying as a function form as shown below:

- *Q(S, Student): list the S of ST <S is a field, ST is a table>*
- *A(S, Student): select S from Student*

Example: T-unit: task or activity related to the knowledge units, activity units, or assessment. T-unit can be given by a question-and-answer pair applying as a function form in different knowledge domains IT refers to information technology domain, M refers to mathematic domain, and H refers to history domain as below:

IT: Q(S, ST): list the S of ST <S is a field, ST is a table>
A(S, ST): **select** S **from** ST
Q: list the students name and department <student and dept are tables>
A: **select** student.st_name, dept.st_dept **from** student, dept.
Or A: **select** s.st_name, d.s_dept **from** student **as** s, dept **as** d.
Q: list the students names by the academic year.
A **select** s_name **from** student **where** a_year **between** 2020 **and** 2021.
Q list the employee tuples whose salary is greater than $3000.
<tuples are the fields; employee is a table>
A: **select** tuples **from** employee **where** salary>$3000.

M: Q(Equ): calculate the solution of Equ <Equ is quadratic equation>
A(Equ):

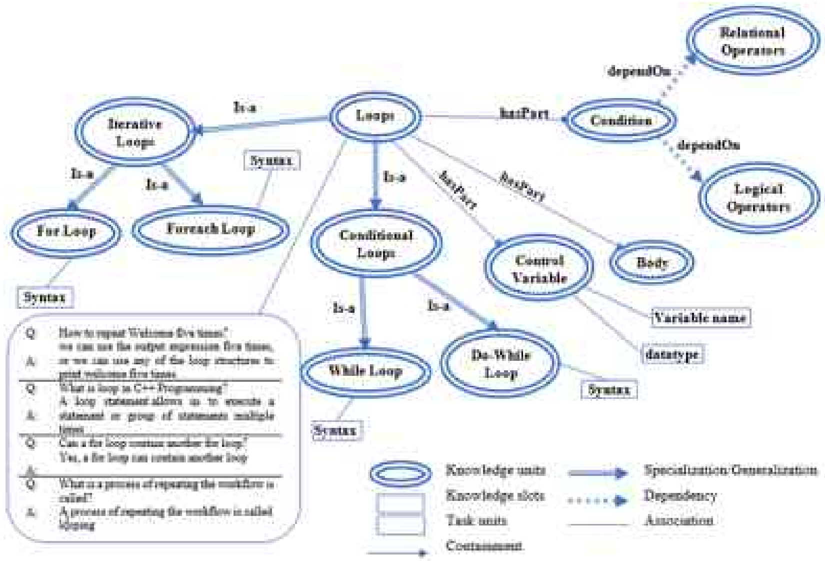
```
function(Equ) {
        discriminant = b*b - 4*a*c;
        if (discriminant > 0)
                x1 = (-b + sqrt(discriminant)) / (2*a);
                x2 = (-b - sqrt(discriminant)) / (2*a);
                return (x1,x2)
        if (discriminant == 0)
                x1 = -b/(2*a);
                return(x1)
        if (discriminant < 0) {
                return (none); }
```

H: Q(E): explain the cause of the event E.name, where E stand for Event
A(E): E.cause
Q(E): mention the leaders' names during WW II.
A(E): E.leader_name
Q(E): mention the battle location of WW I.
A(E): E.battle_location

Every task unit t can be assigned to many different knowledge units and knowledge slots. The symbol

$$t \sqsubseteq u, p$$

Denotes that t is valid for unit u and slots p. The set of all T-units for unit u and task t are given by $\mathrm{T}_{u,p}$. The graphical symbol for a task is a rectangle with rounded corners and a single thin borderline. Figure 3.19 displays an example. In this example, I considered the loops from the loop structures unit and the tasks related to this unit.



Q How to repeat Welcome five times?
A we can use the output expression five times, or we can use any of the loop structures to print welcome five times.
Q What is loop in C++ Programming?
A A loop statement allows us to execute a statement or group of statements multiple times.
Q Can a for loop contain another for loop?
A Yes, a for loop can contain another loop
Q What is a process of repeating the workflow is called?
A A process of repeating the workflow is called looping

Figure 3.19 Task units example

3.7.4 Material Units Description

Material units (M-units) refers to teaching materials for the k-units. M-units are teaching materials used to learn the k-units when using the E-tutoring systems. Moreover, M-units contain any parts of academic materials or teaching materials. Teaching material is a set of training resources, including textbooks, media (video, audio), tools, technologies, web pages, and other learning materials related to the knowledge units. M-units in E-tutoring systems are the primary concrete features that allow E-tutoring to teach the knowledge units effectively. If the teaching material delivered by an academic community or educational institution is not of high quality in the E-tutoring systems in that case, there is no motivation for building an e-learning system. M-units utilized in the learning process constructed for the knowledge unit cannot be effectively used to conduct an E-tutoring system's goals without proper planning, systematizing, coordinating, and management.

The set of all teaching resources is defined by $M = \{m\}$ where m is a teaching material related to the knowledge units. Every material m can be assigned to many different knowledge units. The symbol

$$m \sqsubseteq u$$

denotes that m is valid for unit u. The set of all materials for unit u is given by M_u .The graphical symbol for a material is a rectangle with folder corners and a single thin borderline. Figure 3.20 shows an example of the teaching material related to the knowledge unit. In this example, I considered loops in C++ Programming.

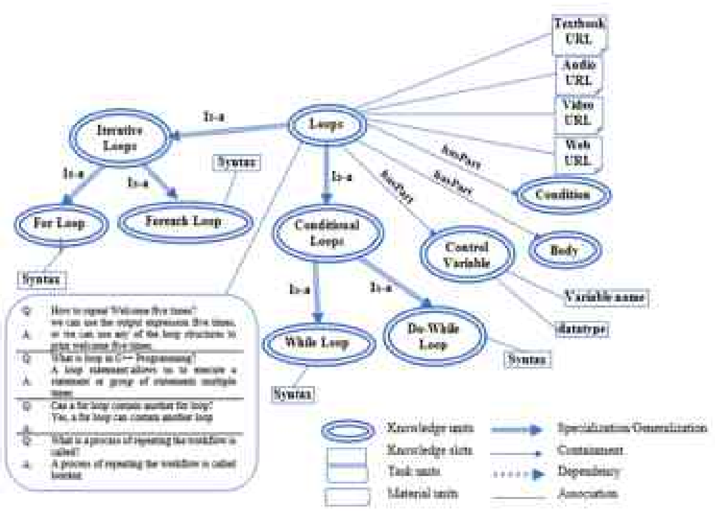

Textbook URL
Audio URL
Video URL
Web URL
Iterative Loops
Is-a
Loops
Syntax
Is-a
Is-a
Is-a
hasPart
hasPart
hasPart
Condition
For Loop
Foreach Loop
Conditional Loops
Control Variable
Body
Syntax
Variable name
datatype
Is-a
Is-a
While Loop
Do-While Loop
Syntax
Syntax
Q: How to repeat 'Welcome five times'?
A: we can use the output expression five times, or we can use any of the loop structures to print welcome five times.
Q: What is loop in C++ Programming?
A: A loop statement allows us to execute a statement or group of statements multiple times.
Q: Can a for loop contain another for loop?
Yes, a for loop can contain another loop
Q: What is a process of repeating the workflow is called?
A: A process of repeating the workflow is called looping
Knowledge units
Knowledge slots
Task units
Material units
Specialization/Generalization
Containment
Dependency
Association

Figure 3.20 Material units example

3.7.5 Rule Units Description

Rule units (R-units) refer to rule units as rules or constraints defined on the k-units and the k-slots. R-units are explicit or understood regulations or principles governing behavior or procedure in a particular activity area. R-units also refer to statements representing how actions or processes usually occur in a specific condition. R-units are used to define the constraints put on the K-units and the K-slots. The goal is to control the participation of the K-units in a given knowledge domain. A rule represents the condition, restriction, or assertion related to K-units and K-slots. The constraint is usually restricted by a Boolean declaration, which can consider true or false. The condition must be satisfied (i.e., considered true) by properly designing the E-tutoring system. Generally, rules are applied for different K-units and K-slots in the knowledge domain. The R-units are designed as constraints put on the k-units and k-slots. It can be in the form of DL rules. A rule axiom consists of a body (also called antecedent which means if-clause part of the rule) and ahead (also called consequent which means then-clause part of the rule), consisting of some atoms. A URI reference can also provide a rule axiom, which could help identify the rule. The rule can be handled using the Semantic Web Rule Language (SWRL) or Manchester Syntax (MS). SWRL rules are DATALOG rules, including unary predicates representing classes and data types, binary predicates for properties, and special built-in n-ary predicates [2]. According to [3] SWRL intends to be the rule language of the Semantic Web. It covers a higher-level abstract syntax for the rules. All rules are represented in terms of OWL concepts in the forms of Entities, classes, properties, and individuals. MS is a compressed syntax for OWL 2 ontologies. MS is frame-based, based on the Manchester ontology [4].

The R-units are given in the form of DL rules or SWRL rules. The set of all rule units is defined as $\Phi = \{r\}$ where r is a rule unit related to the k-units and k-slots.

Examples:

IT: every loop has syntax, and every loop has a control variable.

M: <3 QuadraticEquation.root

H: EventName in Duration from 1939-1945, the war has one winner, the war happens in a specific palace.

Every rule r can be assigned to many different k-units and k-slots. The symbol

$$r \sqsubseteq u, p$$

denotes that p is valid for unit u. The set of all slots for unit u is given by $\Phi_{u,p}$. The rule syntax is given in SWRL language for the K-units and K-slots with their constraints.

- *KnowledgeUnit(?k), hasSlot(?k, ?s), hasCompetencyDependency(?k, ?d), taskScore(?k, 2), taskOf(?k, task1)-> hasEduLevel(?k, ?bl)*
- *KnowledgeUnits(?k), hasSlot(?k, ?s), hasCompetencyDependency(?k, ?d) -> taskOf(?k, ?t)*
- *KnowledgeUnits(?k), hasSlot(?k, ?s), hasCompetencyDependency(?k, ?d), hasTasks (?k, ?t) -> taskScore(?k, 2)*

3.7.6 Educational Units Description

Educational units (E-units) refer to the level of education that define the learners level when learning different knowledge units, usually at a school, college, or university. E-units are a set of classifications intended for a group of educational programs concerning the level of learning backgrounds, knowledge skills, and competencies each program is designed to teach. Therefore, E-units are created based on the idea that education programs can be grouped into classes. These classes describe general steps of educational advancement regarding the difficulty of educational material. E-units are employed to define the level of the learners when using the E-tutoring system. The level can be beginner, intermediate, upper-intermediate, and advanced.

The set of all educational level is defined by $L = \{l\}$ where l is a learner level related to the knowledge units. Every level l can be assigned to many different knowledge units. The symbol

$$l \sqsubseteq u$$

denotes that l is valid for unit u. The set of all levels for unit u is given by L_u .The graphical symbol for a level is a circle with a single thin borderline. Figure 3.21 shows an example of the educational level related to the knowledge unit. In this example, I considered loops in C++ Programming.

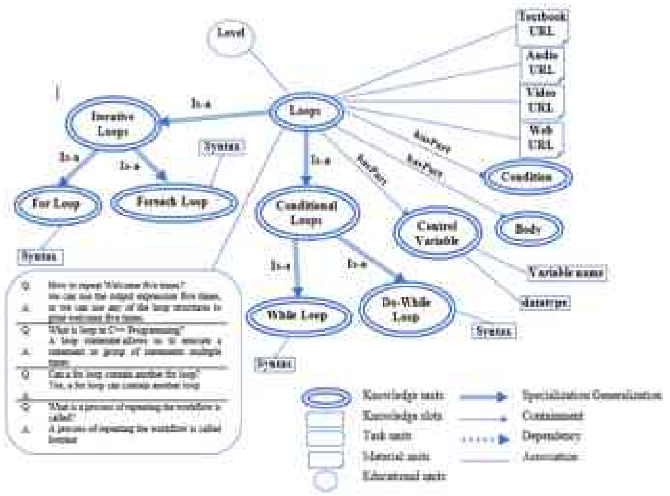


Figure 3.21 Educational units example

3.7.7 Specialization/Generalization Relationship

Generalization is a kind of relationship that indicates one component called a subclass is the basis of another component known as a superclass. Generalization relations are utilized in the k-units to demonstrate that the subclass holds all the properties, attributes, operations, and relationships represented in the superclass. The generalization relationship requires to be a similar type. For example, the generalization relationship can apply to k-units and their components. In other words, generalization is a taxonomy relationship, which means different specialization levels and competencies in the k-units. In addition, generalization also can be a binary taxonomic used as a directed relationship with a more general component known as superclass and a more distinct component called subclass on the k-units and their components. Every individual or instance of a typical component is an indirect individual or instance of a general component, so the generalization relationship is also defined in the form of "Is-A" relation. The generalization and specialization relations are denoted by $\preccurlyeq$ where the relation $\preccurlyeq$ is related to the k-units.
Examples:

IT: relational selection, relational projection, and rename $\longrightarrow$ relational algebra operation.
relational union, relational intersection, and relational join $\longrightarrow$ relational algebra operation
tuple relational calculus $\longrightarrow$ relational calculus
M: integration and derivation $\longrightarrow$ calculus
derivation of polynomials $\longrightarrow$ derivation of functions
H: I WW and II WW $\longrightarrow$ world wars
American W, Mexico W, and Spanish W $\longrightarrow$ civil wars

Every relation $\preccurlyeq$ can be assigned to many different knowledge units. The symbol

$$\preccurlyeq \subseteq \Sigma \times \Sigma$$

denotes that $\preccurlyeq$ it is valid for unit u. The set of all relations for unit u is given by $\preccurlyeq_u$ .The graphical symbol for the relation is an arrow with a double-thin borderline. Figure 3.22 is an example demonstrates the specialization/generalization relationship applied on loops related to the control structures k-unit.

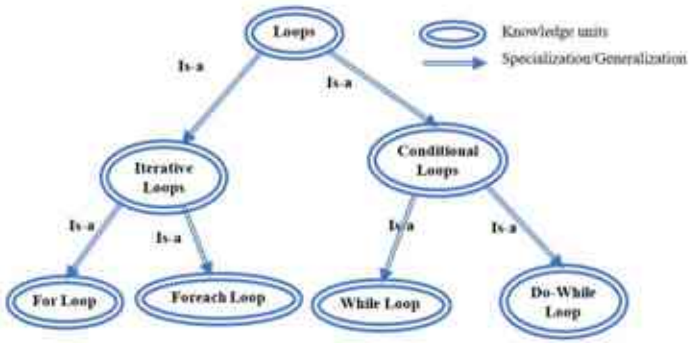


Figure 3.22 Specialization/Generalization relation example

3.7.8 Components Relationship

A component is a relationship, which means one unit consists of other units on the K-units. The component represents the containment relationship, and it is a one-to-many relationship from a superclass entity to a subclass entity. The Containment relationship represents the containment of the knowledge unit on the knowledge graph. For example, I will define a containment relationship between the knowledge units in a graphical representation. A Component represents all types of knowledge units that construct a knowledge model of an E-tutoring system. A Component can always be considered an independent unit within an E-tutoring system or module in a subsystem. The component relations are denoted by $\sqsubseteq$ where the component $\sqsubseteq$ relation is related to the k-units and k-slots.

Examples:

IT: relational models consists of relational algebra and relational calculus.

relational algebra operation consists of syntax and parameters.

selection operation consists of conditions and attributes.

M: function analysis contains derivation and integration

H: duration consists of events, persons, building

Every relation $\sqsubseteq$ can be assigned to many different k-units and k-slots. The symbol

$$\sqsubseteq \subseteq \Sigma \times (\Sigma \cup \Pi)$$

Denotes that $\sqsubseteq$ it is valid for unit u and slot p. The set of all relations for unit u and slot p is given by $\sqsubseteq_{u,p}$. A graphical symbol for the relationship is an arrow with a single thin borderline. Figure 3.23 displays an example of containment relationships on the k-units. In this example, loops in the C++ programming unit are considered.

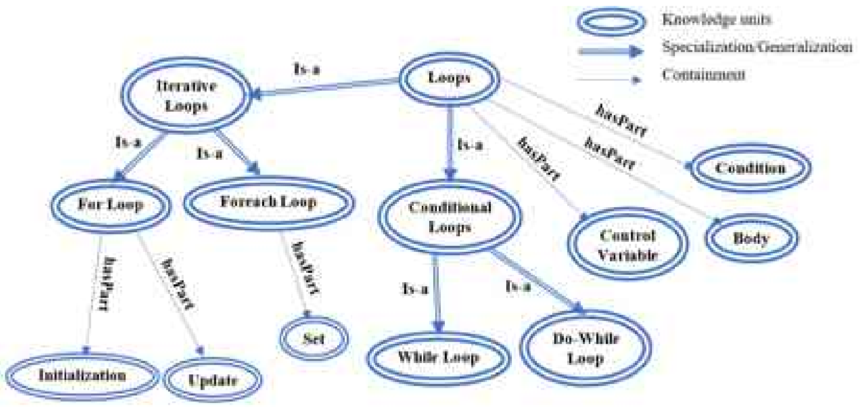


Figure 3.23 Components relation example

3.7.9 Associations Relationship

An association describes the relationship between entities or k-units based on shared attributes. In addition, the association enables entities or k-units to access the data of other entities or k-units through a persistent reference. The association defines a relationship that can happen between individuals or instances of the k-units. The relationship is a tuple with a single value for each unit so that each value can be an individual or instance of k-units. The association relations are denoted by $\sim$, where the association $\sim$ relation is related to the k-units. Every relation $\sim$ can be assigned to many different k-units. The symbol

$$\sim \subseteq \Sigma \times \Sigma$$

denotes that $\sim$ is valid for unit u and slot p. The set of all relations for unit u is given by $\sim_u$. The graphical symbol for a relation is a line with a single thin borderline. Figure 3.24 is an example illustrates the association relation on the k-units. In this example, I considered loops in the C++ programming unit.

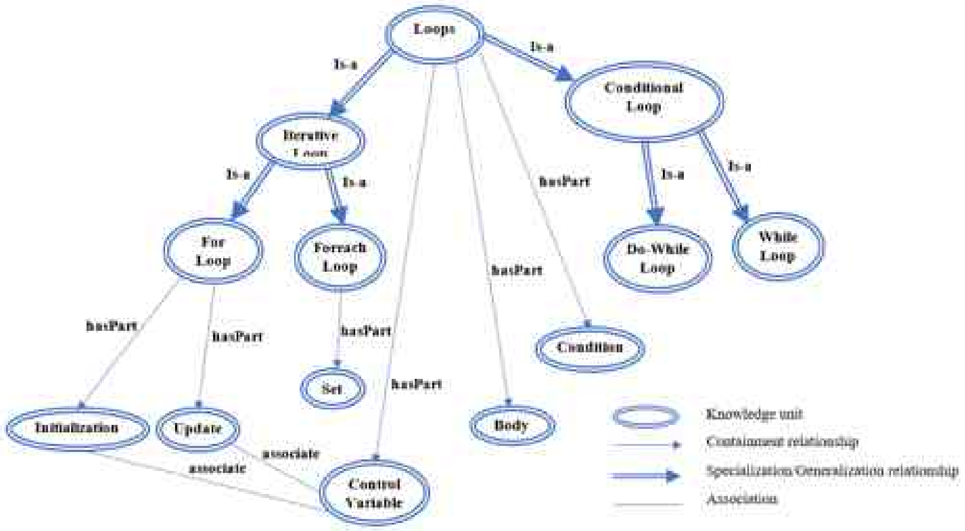


Figure 3.24 Association relation example

3.7.10 Dependency graph Relationship

Dependency is a binary relationship that shows knowledge unit components require or depend on other knowledge unit components for specification or implementation. Dependency is a category of a relationship among the named knowledge unit components, which involves several components, e.g., k-slots, to solve specific tasks related to the k-units. Dependency is a relation representing the competency-level dependency between the knowledge units. The dependency graph is denoted $\Rightarrow$, where the dependency $\Rightarrow$ relation is related to the knowledge units and knowledge slots. Every relation $\Rightarrow$ can be assigned to many different k-units and k-slots. The symbol

$$\Rightarrow \subseteq \Sigma \times \Sigma$$

denotes that $\Rightarrow$ it is valid for unit u and slot p. The set of all relations for unit u is given by $\Rightarrow_u$ .The graphical symbol for a relation is an arrow with single dashes thin borderline:

IT: relational algebra depends on (relational schema) Closed.

M: derivation of polynomials depends on calculus, polynoms

H: II WW depends on (events, geopolitics, ….) Open

Where closed refers to close dependency graph and open refers to open dependency graph. Figure 3.25 shows an example of the dependency graph. I considered a loop statement in the control structures unit in this example.

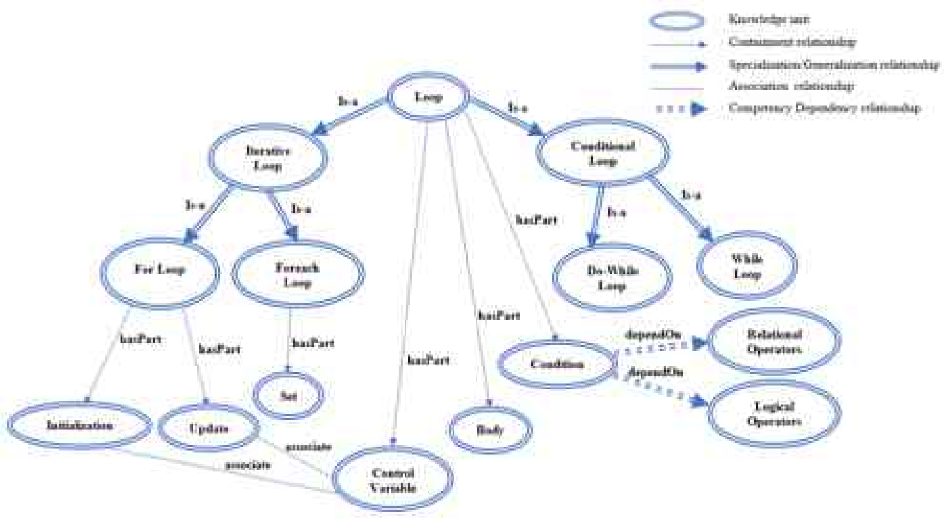


Figure 3.25 Dependency relation example

3.8 The Proposed Ontology Domain Knowledge Model Concepts

The key structure of our proposed domain knowledge model is shown in Figure 3.26 The model is based on selected components such as knowledge units, knowledge slots, task units, material units, educational units, rule units and relations. Moreover, to share and reuse the domain knowledge model and integrate it with the E-tutoring systems, ontology is utilized to manage and represent the domain knowledge model. The benefit of this model is to provide personalized learning and teaching, generate material units, make suggestions, and make automatic assessments for students.

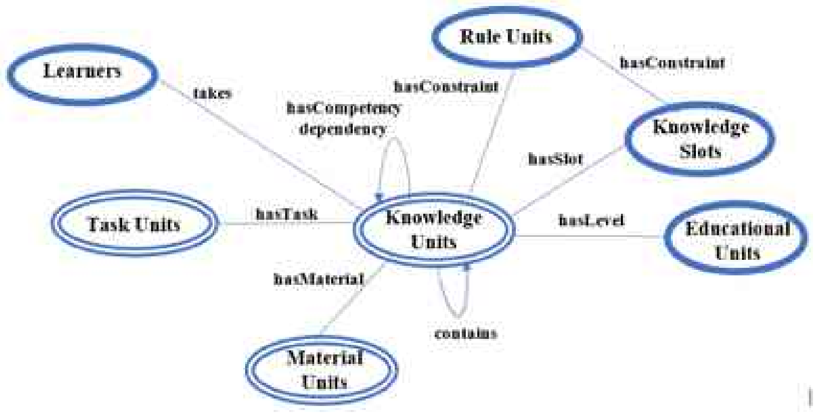


Figure 3.26 Ontology model schema of teaching materials

3.9 Conceptual knowledge representation of the proposed ontology model

The conceptual knowledge model (CKM) is a structured relations system of the knowledge needed to promote business processes, describe business activities, and track corresponding performance measures. CKM concentrates on specifying the knowledge employed in business applications. CKM also represents the overall knowledge structure required to support the business requirements independent of any application or knowledge repository structure. Figure 3.27 displays the conceptual knowledge model of the teaching materials.

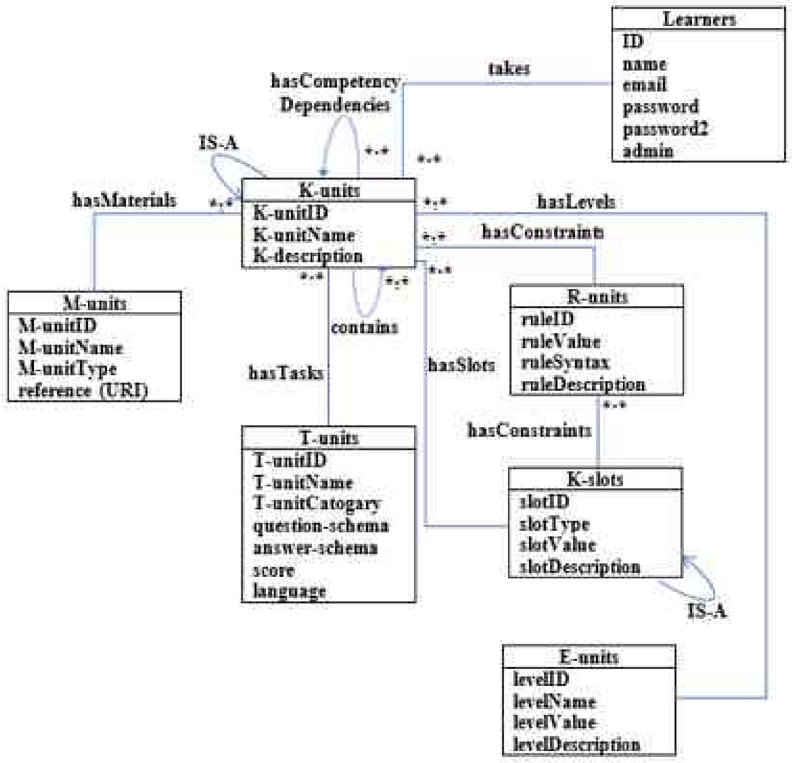


Figure 3.27 Conceptual knowledge model of teaching materials

Ontology provides more coherent and easy navigation for moving from one concept to another in the ontology structure. Another valuable feature is that ontology is easy to extend as relationships and concept matching is easy to add to existing ontology. Ontology also provides the means to represent any type of knowledge domain format, including unstructured, semi-structured, or structured knowledge, enabling smoother knowledge integration, easier concept and text mining, and data-driven analytics.

The conceptual knowledge model is derived from the proposed ontology domain knowledge model displayed in Figure 3.26. The aim is to define and communicate high-level relationships between domain knowledge model components.

3.10 Ontological Model of the Domain Knowledge Model

The ontological model of the domain knowledge model (OMDKM) is one of the most used forms to model the knowledge for Intelligence software applications. This model is converted from the proposed ontology domain knowledge model displayed in Figure 3.25. It constructs upon the requirements delivered by the knowledge analysts. It has a different level of detail, helping both the software system and related knowledge requirements. OMDKM also provides the specifications for knowledge describing the concepts, relationships, and interpretation of knowledge values. OMDKM is an abstraction of the knowledge specifications. Figure 3.28 shows the ontological knowledge model of the domain knowledge model for teaching materials. This Figure also displays the ontology schema of the domain knowledge model using the object and data properties, for example hasSlotRole as a data property means the role that slot can play in the system (optional, obligatory). With this schema the ontology database for that model can be generated.

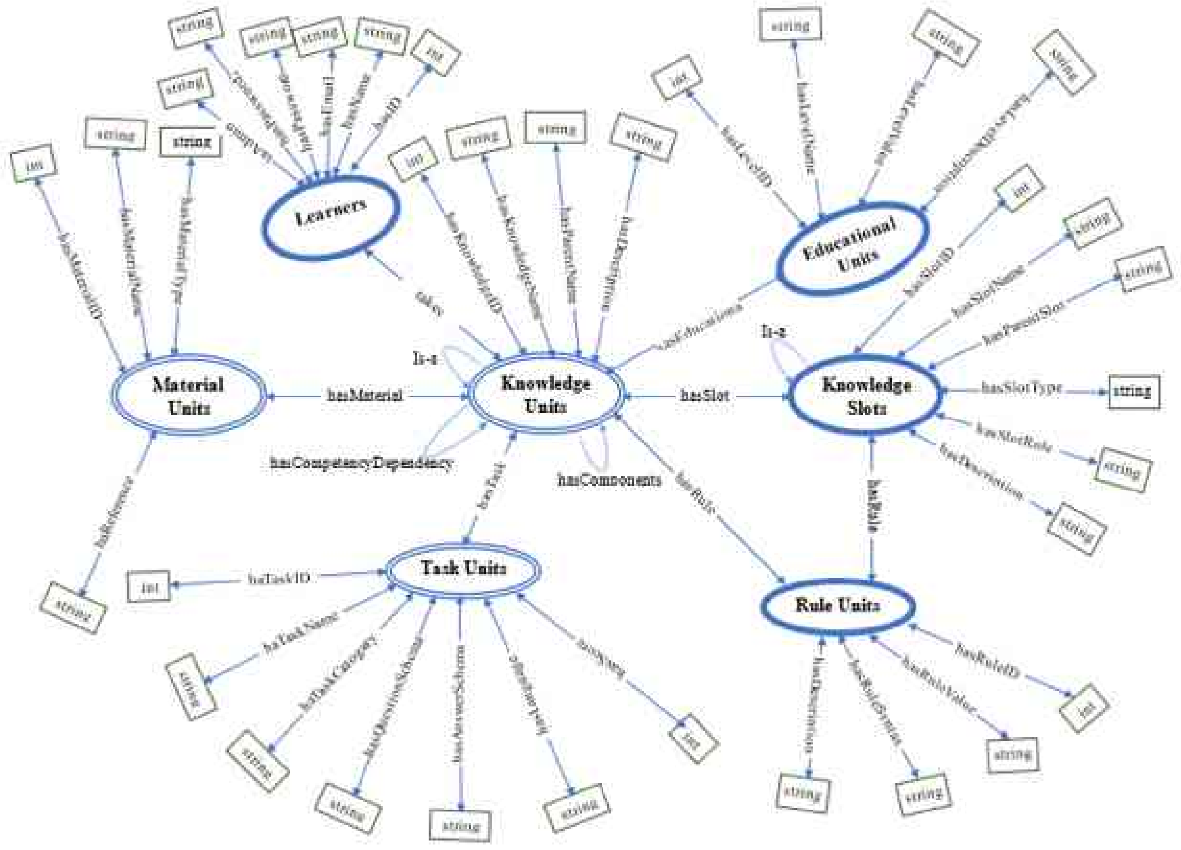


Figure 3.28 Ontology knowledge model schema

Table 3.1 shows a set of defined classes in the domain knowledge model ontology related to the OMDKM schema which is displayed in Figure 3.28. These classes can be modeled in the ontology database. The top abstract class for the OMDKM is Knowledge units, of which subclasses are defined. In this table the first column describes the OMDKM classes, the second column explain the OMDKM relations, and the third and fourth columns denote the domain and range class for the relationships.

Table 3.2 Relationships of the domain knowledge model Class

Classes	Relationships	Domains	Ranges
Learners	takes	Learners	Knowledge Units
Knowledge Units	hasParent	Knowledge Units	Knowledge Units
Knowledge Slots	hasPart	Knowledge Units	Knowledge Units
Task Units	hasCompetencyDependency	Knowledge Units	Knowledge Units
Rule Units	associate	Knowledge Units	Knowledge Units
Educational Units	hasSlot	Knowledge Units	Knowledge Slots
Material Units	hasTask	Knowledge Units	Task Units
	hasURule	Knowledge Units	Rule Units
	hasSRule	Knowledge Slots	Rule Units
	hasEduLevel	Knowledge Units	Educational Units
	hasMaterial	Knowledge Units	Material Units

3.11 General Form of the Ontology Domain Knowledge Model Components

Ontology is a framework for representing shareable and reusable knowledge within a specific domain. The ability to describe relationships and their high interconnectedness make them the basis for modeling a high-quality, linked, coherent knowledge base. In addition, an ontology formally explains the knowledge domain as a collection of concept elements within a specific domain and their relationships. To enable this explanation, we must formally determine knowledge domain components. These components are classes, attributes, individuals (instances/objects), relationships, restrictions, rules, and axioms. As an outcome, ontology not only presents a reusable and sharable knowledge representation but can also generate new knowledge related to the knowledge domain. The ontology knowledge domain can be applied to a collection of individual facts to create a knowledge graph using a collection of entities where nodes and edges between these nodes express the types and the relationships between them.

Different methods utilize formal specification formats for knowledge representation. These formal specifications are taxonomies, topic maps, vocabularies, thesauri, and logical ontology models. One of the key features of logical ontology is that it enables automated reasoning about the knowledge domain by having the essential relationships between concepts built into them. Moreover, such a reasoning mechanism is easy to implement in semantic graph knowledge bases that use ontology as their semantic schemata.

In Figure 3.25 I displayed the components integration of the proposed ontology domain Knowledge model, which I described in section 3.7 in detail. The purpose of this section is to show these components and their relationships in a general form. The structure of this model has fundamental concept elements which denote knowledge units, knowledge slots, task units, rule units, material units, and educational units covering the specific teaching and competency dependency units. I also introduce many types of relationships among the knowledge units in the ontology domain. The selected relationships are specialization or generalization relationships, component relationships, association relationships, and dependency graph relationships. The core components of the suggested model are the knowledge unit, knowledge slot, task unit, material unit, rule unit, and educational unit, which can help the learners to automatically get a personalize teaching material for their learning process. The following is a description of domain knowledge model elements mapping to ontology classes, properties in the ontology domain.

Class categories

- Learners (Class):
- Knowledge Units (Class)
- Knowledge Slots (Class)
- Task Units (Class)
- Rule Units (Class)
- Educational Units (Class)
- Material Units (class); subclasses:
  - Text source (Class)
  - Video source (Class)
  - Audio source (Class)

  - Web URI (Class)

Property categories

- Relationship (Object Property); subclasses:
  - Containment (Object Property)
  - Specialization (Object Property)
  - Competency Dependency (Object Property)
  - Association (Object Property)
  - Takes (Object Property)
- Attribute (Data Property):
  - Task Assessments (Data Property)
  - Complexity levels (Data Property)
  - Names (Data Property)
  - Rules (Data Property)
  - Syntax (Data Property)
  - Materials (Data Property)
  - Slots (Data Property)
  - Descriptions (Data Property)
  - Values (Data Property)
  - Types (Data Property)
  - Questions (Data Property)
  - Answers (Data Property)
  - Scores (Data Property)
  - Language (Data Property)
  - Categories (Data Property)

The following are OWL examples of the domain knowledge components mapping to ontology classes, properties, individuals (instances), data values, and the restrictions on classes. For example, we can define a class KnowledgeUnits as follows:

```
<owl:Class rdf:ID=" KnowledgeUnits ">
  <rdfs:subClassOf rdf:resource="#DomainKnowledgeModel"/>
</owl:Class>
```

Regarding the properties for the domain knowledge model components, the following sub code is an example description of the object property.

```
<owl:ObjectProperty rdf:ID="takes">
  <rdfs:range rdf:resource="#KnowledgeUnits "/>
  <rdfs:domain rdf:resource="#DomainKnowledgeModel"/>
  <owl:inverseOf rdf:resource="#isTaughtBy"/>
</owl:ObjectProperty>
```

The sub code below is an example description of a datatype property for the domain knowledge model components.

```
<owl:DatatypeProperty rdf:ID="hasKnowledgeID">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XLMSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

The following is an example of the property restriction applying on owl:onProperty, owl:maxCardinality, and owl:minCardinality properties for the domain knowledge model components.

```
<owl:Class rdf:about="#KnowledgeUnits">
```

```
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#isTaughtBy"/>
      <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger"> 1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

An example of the syntactic form of the special properties offered by OWL.

```
<owl:ObjectProperty rdf:ID="hasSameScoreAs">
    <rdf:type rdf:resource="&owl;TransitiveProperty" />
    <rdf:type rdf:resource="&owl;SymmetricProperty" />
    <rdfs:domain rdf:resource="#Learner" />
    <rdfs:range rdf:resource="# Learner " />
</owl:ObjectProperty>
```

The following examples are instances of the For_Loop knowledge unit as modeled in Figure 3.25.

```
<owl:NamedIndividual rdf:about="#For_Loop">
    <rdf:type rdf:resource="#KnowledgeUnit"/>
    <hasCompetencyDependency rdf:resource="#Logical_Operator"/>
    <hasCompetencyDependency rdf:resource="#Relational_Operator"/>
    <hasEduLevel rdf:resource="#intermediate_level"/>
    <hasMaterial rdf:resource="#m_unit2"/>
    <hasParent rdf:resource="#Iterative_Loops"/>
    <hasPart rdf:resource="#Body"/>
    <hasPart rdf:resource="#Condition"/>
    <hasPart rdf:resource="#Control_Variable"/>
    <hasPart rdf:resource="#Initialization"/>
    <hasPart rdf:resource="#Update"/>
    <hasSlot rdf:resource="#syntax"/>
    <knowledgeName rdf:datatype="string">For Loop</knowledgeName>
</owl:NamedIndividual>
```

3.12 Summary

This chapter introduces a survey of the most common knowledge representation schemes and related limitations. Moreover, I argue that knowledge representation schemes, especially ontologies, can offer solutions to current drawbacks with a well-designed form. An ontology as a means of knowledge sharing and interoperability will significantly increase the reusability of the domain knowledge model.

In this research, I have developed a model for the purpose of enhancing the learning and teaching processes in 3 ways.. These models are the ontology-based model to support the learning process in LMS, the ontology-based Knowledge domain model for the IT domain in E-tutoring systems, and the ontology-supported domain knowledge module for an E-tutoring system. Moreover, I presented a novel and general module called the domain knowledge model for teaching materials using ontology as a knowledge representation technique.

To facilitate the sharing and reusing of the domain knowledge model features in E-tutoring systems, ontologies are utilized to organize and represent the domain knowledge model. The expected benefits of this model are to personalize the learning materials for the learners, enhance the learning and teaching process, support recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of new material.

Thesis 1 ***[2] [3] [4] [8]***

I have developed an adaptive E-tutoring system architecture. Beyond the standard components, this architecture also includes a common shared database and knowledge-based background. The main benefits of this architecture are that it supports shared databases providing more features. The features are efficient knowledge management, flexibility, better performance, scalability, increased accessibility and availability, better security, and automatic recovery. They also provide the required technical and methodological background for developing smart tutoring systems. Besides this adaptive architecture, I have developed and tested three model versions of the domain knowledge module based on ontology methods in order to create a general and flexible model and enhance the learning processes. The proposed model can improve the semantic support of the related decision-making processes. The main role is to support the automatic control of teaching processes and automatically selecting assessment material from a predefined pool according to the student's capabilities. The proposed model is presented as a model structure containing two types of ontology models, general concepts of domain knowledge ontology and specific domain knowledge ontology. The expected benefits are to enhance the learning and teaching process, provide support for recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of teaching materials.

Thesis 2 ***[4]***

I have introduced a novel formal logical ontology domain knowledge model for an E-tutoring system, which presents the conceptual components of teaching material frameworks. The core components of this model are knowledge units and related units covering the specific teaching and competency dependency units, including the specialization, containment, association, and dependency graph relationships among the knowledge units in the ontology domain. For storing this model, I compared three model storage systems the OWL ontology file, Jena Fuseki, and the relational model. The key aspect is determining the suitable storage model to store the ontology model in a standard, efficient, scalable, and consistent format. The main role of this comparison is to improve application performance, enable semantic knowledge representation, and achieve efficient information retrieval. The investigation shows that Jena Fuseki and the relational model are the best ontology storage tools. They provide support facilities for storing, inferring new knowledge, and managing the ontology domain knowledge.

Chapter 4 Assessment Module

As learning is a complicated process, we cannot accurately predict whether the student's knowledge of a domain knowledge concept is known or unknown. For example, a new domain knowledge concept may be completely unknown to the student, but in other cases, it may be known partly due to previous related knowledge of the learner. Because of the above, representing the student's knowledge is a dynamic goal, and the learning process strategy is an uncertainty system. A solution to this problem is using the Intuitionistic Fuzzy Sets (IFSs) technique. In literature, IFSs have been used to manage uncertainty systems [4]. Atanassova introduced the IFSs as a technique for measuring uncertainty factors, which cannot be implemented effectively well with other uncertainty approaches [94].

The IFSs technique is valuable in several application areas, including robotics, control systems, computer science, online education, and other engineering fields. The knowledge and semantic representation of the IFS become more meaningful, innovative, and applicable because it includes the membership degree, the non-membership degree, and the hesitation margin [95]. In view of the flexibility of the IFS technique in managing uncertainty, it is a mechanism for more consistent human reasoning under insufficiently defined facts and uncertain knowledge [96]. Different applications of the innovative IFS approach have been conducted through a distance measures method. Many scholars have investigated multiple applications of an IFS, including medical and diagnosis applications, career determination, real-life situations, and the teaching and learning domain [97 – 99].

The aims of my research is to develop an assessment module for evaluating the student knowledge level with uncertain conditions using Intuitionistic fuzzy logic values and implement this module to automatically detect the students' knowledge level, update their status, and generate the following question for the student.

This research work has been organized in the following structure. The introduction is presented in the first section. The second section displays the related works, including the assessment module, domain model, student knowledge model, and uncertainty of the knowledge. The Student Assessment Using Atanassov's Intuitionistic Fuzzy Technique is illustrated in the fourth section. The fifth section presents updating the student's knowledge using the Intuitionistic Fuzzy logic value. The sixth section demonstrates the results, and the conclusion is in the seventh section.

4.1 Assessment Module

Assessment plays a significant role in the process of learning and teaching. Assessment is a method for acquiring information to understand better the strengths and weaknesses of learners' knowledge levels [100]. The assessment also can be expressed as a systematic gathering and studying of learner knowledge to enhance learning outcomes [100]. As such, the assessment provides meaningful feedback to both tutors and learners. In other words, assessment seems to be a mechanism to give the tutors information for enhancing education and learning strategies as well as guiding and motivating learners to be actively involved in their learning processes [100]. Well-designed assessment techniques provide helpful information about learner knowledge. Assessing learner knowledge systematically collects and analyzes the information to improve student learning outcomes. The assessment makes learning more effective and consistent by systematically connecting tasks, knowledge unit structure, and learning materials practices to intended learning goals. It also helps tutors become better educators by offering specific feedback on what is working or not working in their tutoring and provides frequent feedback to learners about their progress.

The assessment module offers benefits for the learners and tutors. For learners, it facilitates and improves student learning by clarifying their tutors' expectations, focusing on education by connecting learning and knowledge unit materials and providing learners with an understanding of their strengths and limitations. For the tutor, it offers personalized learning, uses different assessment techniques, adjusts the teaching process to accommodate gaps in the learning process connected to teaching techniques, and facilitates learning for acquiring knowledge.

In any E-tutoring platform, both tutor and learner must keep track of the learner's progress and review whether the learners have understood the knowledge unit. In the traditional learning system, the instructor and the learner are involved in face-to-face communication at a particular place and time. Regular knowledge assessment for the learners can be carried out naturally so that instructors can efficiently review the understanding of learner progress. At the same time, instructors must evaluate the assessment for all learners. This technique may be inflexible and a time-consuming process. In the form of the e-learning system, it is very important to assess student learning automatically. One of the primary uses of E-tutoring is facilitating personalized learning such that tutors can dynamically modify and deliver instructional materials related to learners' status. In general, personalized teaching and learning refer to using knowledge about learners, a priori or via interactions, to modify how a learning background extends and improves learners' success and satisfaction. In E-tutoring, a regular knowledge assessment must be carried out using different styles of tests for many reasons. Unfortunately, assessing students' learning by questionings, quizzes, or assignments only assesses the learners at the basic levels. Learners require better understanding and need to give better elaboration when evaluated at higher levels.

The assessment module is used for evaluating the learner's knowledge in the knowledge domain model, which is implemented in the E-tutoring system as a task unit assessment model. This assessment module can be applied as a pair of question-and-answer models. Every task belongs to a knowledge unit or knowledge slot. The following elements define the components of the assessment module, which is suggested to be used for the learner's knowledge assessment to detect the performance of learning. The knowledge unit related to the knowledge domain model integrated into the E-tutoring system.

(1) T_i is a set of all task activities $(i = 1 \dots n)$, $T_i = (t_1, t_2, \dots, t_n)$.

(2) Each task is a pair of (Q, A), and every task belongs to a knowledge unit, knowledge slot, concept, or sub-concept,

(3) A Q/A is a triple tuple (Q_T, A_T, P_i), Where:
Q_T is a set of all questions, $Q_T = (Q_1, Q_2, \dots, Q_n)$,
A_T is a set of all answers, $A_T = (A_1, A_2, \dots, A_n)$,
P_i is a set of parameters entities, $(i = 1 \dots n)$, $P_i = (P_1, P_2, \dots, P_n), P_i$ may have attributes (P_i, A), concepts (P_i, C), or domains (P_i, D).

(4) The question and answer for a task unit is presented as a pair function in the form of $[Q_T(P_i), A_T(P_i)]$

4.2 Domain Model

The domain model represents a finite set of domain knowledge concepts that describe the conceptual components of the teaching materials. The entire collection of this domain knowledge model is organized into knowledge elements or knowledge concepts represented as a graph (or knowledge unit taxonomy) with different node elements and relationships among these nodes. The key-concept components of this model are knowledge units, knowledge slots, task units, rule units, material units, and educational units covering the specific teaching and competency dependency units. This domain model introduces many types of relationships among the knowledge unit concepts. Therefore, the selected relationships are specialization or generalization relationships, component relationships, association relationships, and dependency graph relationships. The selected domain knowledge model elements are the loops in the computer programming domain, especially in the C++ programming sub-domain, as shown in Figure 4.1. “For example, a knowledge unit 'for loop' is a type of loop structure with part initialization, condition, update, control variable, and body”, which has competency dependency logical and relational operators and has slots syntax, variable name, and datatype, which can be described as $Ku = \{ku_1, ku_2, \dots, ku_n\}$ where n is the total number of domain knowledge concepts. Learning dependencies (specialization or generalization, component, association, and competency dependency) among domain knowledge concepts are represented by the ordered prerequisite relation $R = \{R_s, R_c, R_a, R_d\}$ as provided in the following:

$$R \subseteq Ku \times Ku$$

$$R = \{(ku_i, ku_j): ku_i \prec ku_j; ku_i, ku_j \in Ku\}$$

Here, the prerequisite knowledge unit ku_i must be known to the student to understand the second knowledge unit ku_j. Therefore, the student can only start learning the next knowledge unit ku_j after learning the previous related knowledge unit ku_i. For example, after learning the domain knowledge unit of the loops, a task activity is conducted to evaluate the student's knowledge level. The domain knowledge model of the teaching and learning material is represented in the domain model because it is one of the most important and core components of an E-tutoring system.

4.3 Student Knowledge Model

The student knowledge model (SKM) represents the student's knowledge level. As an essential part of the E-tutoring system, when a student interacts with learning and teaching materials, it will follow the progress made by the students. The SKM will allow the system to adapt to the student. The main goal of the SKM in an E-tutoring platform is to create a storage model for providing all information about the student progress in learning via following their task activities with the system to store model information of a profile that represents the student. In addition, the importance of building a SKM is to ensure that the E-tutoring system has adequate information about each student so that it can reply effectively, engage students' interest, and promote the learning process. Personalized learning and feedback are key features to generating learning material tailored to students' preferences, increasing their interest in learning, and enhancing the learning process. The E-tutoring system can support weak students' knowledge, develop their skills, and adapt the material according to their cognitive and motivational characteristics. An E-tutoring system utilizes the information stored in the SKM to adopt this method of interacting with the student. This module is used to evaluate a student's possible answer, provide personalized feedback, and recommend learning material that fits the student's current level of knowledge. The SKM contain knowledge unit similar to the knowledge unit, which is in the domain model, but here knowledge units are extended by competency level, test results, and task attributes as displayed in Figure 4.1.

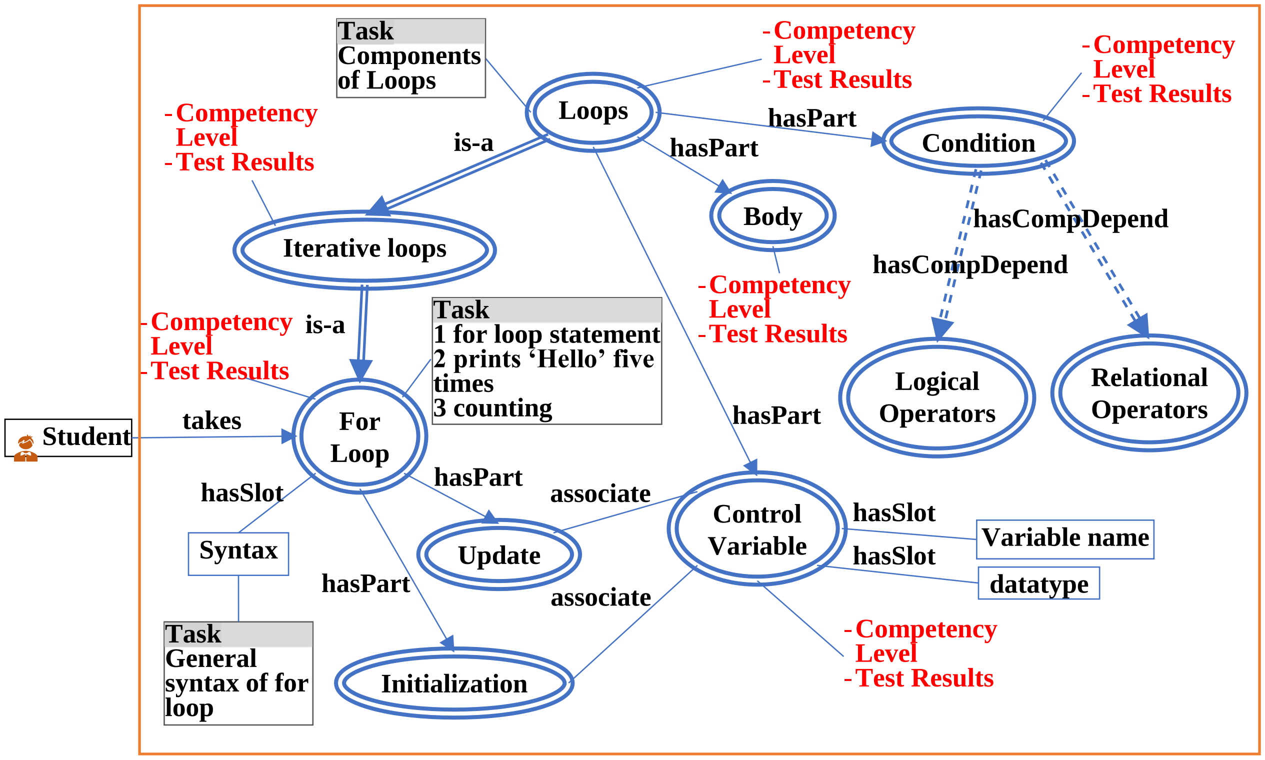


Figure 4.1 knowledge model of the student

4.4 Uncertainty of the Knowledge

In most systems, an evaluation is performed based on the crisp response of the test taken by the student during the learning process. If the learner chooses an answer by guessing or chooses the most probable correct answer in multiple-choice questions, this leads to uncertainty about the learner's knowledge. Many techniques have been used in the literature to evaluate the uncertainty of the student's knowledge. The techniques include certainty factor rules, fuzzy logic, Bayes probability network, and Intuitionistic logic value. The researchers in [101] are working on the uncertainty of the learner's knowledge using the theory of intuitionistic fuzzy set, an approach that was introduced by Atanassov in 1986 [102], in which the student has the opportunity of identifying the percentage that they think each answer to be correct in multiple-choice questions.

For managing the uncertainty of students' knowledge and identifying their status in learning the domain knowledge model in an E-tutoring framework, we used Intuitionistic fuzzy sets (IFSs) to represent the student knowledge model from the domain knowledge model concepts. The main goal is to evaluate learner knowledge based on an accumulative task assessment using question-and-answer pairs and updating a student model. In the following, we introduce some basic concepts related to IFSs.

In this work, Atanassov's intuitionistic fuzzy logic technique is used to evaluate the student based on multiple-choice questions. The information is updated based on the student's knowledge level, which is collected in the student model. In addition, we used multiple-choice questions to measure students' understanding, requiring students to express their more in-depth understanding of the knowledge unit concepts. This assessment is being taken to evaluate the student's knowledge and to update the student's status.

4.5 Student Assessment Using Atanassov's Intuitionistic Fuzzy Technique

According to Atanassov, IFSs have defined a pair of membership $t_A(x)$ and non-membership $f_A(x)$ values to define the logic value. The following is a mathematical definition of IFSs theory.

Definition 1. Let $X = \{x_1, x_2, \dots, x_n\}$ is a non-empty and finite set, then IFS A in X is an object that is defined as follows in function (1):

$$A = \{x. t_A(x), f_A(x) | \ x \in X\} \qquad (1)$$

Where $t_A: X \rightarrow [0, 1]$ and $f_A: X \rightarrow [0, 1]$ defines the membership degree and non-membership degree of the element $x \in X$ belonging to IFS A, with the condition in function (2):

$$\forall \mathrm{x} \in \mathrm{X} \quad 0 \leq t_A(x) + f_A(x) \leq 1 \qquad (2)$$

All of the fuzzy sets can be represented in the form of an IFS as shown in function (3) and (4):

$$A = \{x, t_A(x), 1 - t_A(x) \mid x \in X\} \qquad (3)$$

$$\pi_A(x) = 1 - t_A(x) - f_A(x) \qquad (4)$$

for all x ∈ X, then $\pi_A(x)$ is called the indeterminacy degree of x in A. It is the hesitancy degree of x in A as displayed in (5):

$$0 \leq \pi_A(x) \leq 1, \forall \mathrm{x} \in \mathrm{X} \qquad (5)$$

Definition 2. Let $a = (t_A, f_A)$ be an Intuitionistic fuzzy value (IFV). The score function S of a can be evaluated as follows in function (6):

$$S(a) = t_A - f_A \qquad S(a) \in [-1, 1] \qquad (6)$$

Atanassov's Intuitionistic fuzzy logic is used for managing inaccurate and incomplete information and has been proven helpful in different science and technology domains.

The focus of this article is to provide a methodology for measuring the learner's knowledge level with uncertain conditions and while incomplete information is available on the learner's answers to the question templates. To this purpose, we applied the IFS theory to evaluate the student's knowledge level.

We generated multiple-choice questions (MCQs) for each knowledge unit concept in the domain knowledge model. In the presented technique, the learner can choose a correct answer for every MCQ, which has a score value and level. This value demonstrates the correct answer that the learner can answer for each choice is the multiple-choice question. If the learner has no idea about the correctness of the choice option, they can leave its correctness blank. The outline of the correct answer can be lower than or equal to one. The strength of the IFS theory lies behind this fact, i.e., it enables integrating the lack of the learner's knowledge about their answer.

4.6 Knowledge Level Representation

In order to provide personalized training, the E-tutoring system should contain a separate knowledge model for each student. This model covers the training history and the assessment results for every knowledge unit. The knowledge map of the students is in synchrony with the general domain model. Thus, the student's knowledge maps can be considered as an extended version of the domain model. The student knowledge map consists of the following core elements.

1) $S = (s_{a1}, s_{a2}, s_{a3}, \dots, s_{am})$: feature vector of the student. It contains, among other the personal data identifiers.
2) $KM = (KU, R)$: the knowledge map. Graph of knowledge units (k-units). Every k-unit in KU has a corresponding item in the domain model. The component R denotes the set of relationships among the k-units, including the specialization relationship or the containment relationship or the competency dependency relationship.
3) A_L: the set of knowledge-level attributes of the KU units. The main attributes are as follows.
    a) Measured knowledge level value (A_{LM}) at the k-unit.
    b) Expected knowledge level value (A_{LE}) at the k-unit.
    c) History of learning activities (A_H) related to the k-unit.
    d) History of related assessments (A_A)

Regarding the knowledge level value attribute, we use an intuitionistic logic value to denote the knowledge and the competency level. The primary motivation for selecting this representation mode is that this formalism is able to present the uncertainty level. Thus, we are able to use three aspects in the description:

- Perfect competency
- No competency
- We have no information about the competency.

According to the formalism of intuitionistic logic, $A_L = (l_T, l_F)$, where:
l_T: the level of perfect competency $0 \leq l_T \leq 1$
l_F: the level of no competency $0 \leq l_F \leq 1$ and $l_T + l_F \leq 1$

Initially the A_L values are set to $(0, 0)$ as we have no information about the knowledge level of the student.

The main difference between the expected and measured knowledge levels is that the measured value is based on the concrete assessment results of the student. As the assessments usually cover only a fraction of the related knowledge units, the ratio of unknown values is relatively high. The other component, the expected competency level, is based on the processing of the measured competency levels of other students, using a machine learning approach for value prediction.

4.6.1 Calculation of the competency levels

The context of the calculation of the expected competency value can be given with a competency matrix CM, where the notation of the columns:

$A_1, \dots, A_m$: attributes. Every attribute corresponds to a value range of measured competency (or feature value component in the student description).
$C_1, \dots, C_v$: value categories of the expectation value at the target knowledge unit. The values in the cells are True (1) or False (0) values, depending on whether the attribute is valid at the given object (row) or not. The following table (Table 4.1) displays the expected competency level value related to the knowledge unit.

Table 4.1 The expected competency level

A_1	A_2	A_3	A_4	C_1	C_2
1	0	0	1	1	0
1	1	0	0	0	1

For the prediction of the expected value for a given (student, knowledge unit (k-unit)) pair, we use the Bayes classifier. The Bayes classifier output is the category with the highest score which is calculated from the corresponding conditional probabilities.

$$C_w = argmax_c \left\{ P(c) \prod_a P(a/c) \right\} \qquad (7)$$

The expected value shows that the k-unit with higher score value can be suggested to the student as a new k-unit for learning.

The measured competency values are changed after every assessment unit. In the selection of the k-unit for the assessment, we should take those k-units, which help to discover the unknown areas in the competency map.

The algorithm to select a knowledge unit:

1. Suppose there are k-units with uncertainty or with low-level value. In that case, we should choose one of these k-units where this value is equal to the product of the uncertainty level of the measured level and the inverse of the adjusted expected competency level. We order a selection probability for each task unit (the higher the relevance value, the higher value has the chance of selection).
2. The student performs the assigned task, and we get the result.
3. We evaluate the response and get an accuracy level for each k-units related to this task.
4. We update the knowledge map of the student.
5. We give more suggestions to the student on which parts and study aids should he/she learn and practice.

If the current value is (l_T, l_F) for the k-unit, and the score of the solution is g (g is normalized in [0,1]) and the relevance of the k-unit in the task was r (r in [0,1]), then.

$$l_T = \frac{(n * l_T + g * r)}{(n+1)} \qquad (8)$$

$$l_F = \frac{(n * l_F + (1-g) * r)}{(n+1)} \qquad (9)$$

Where n is the number of previous updates.

The assessment process in the prototype system is an interaction cycle between a student and the system. This process contains a student login to his knowledge model, a random selection of a knowledge unit, a task, a related question, a student answer, an evaluation, and a student knowledge model update. Figure 4.2 shows an interaction process between a student and the prototype framework.

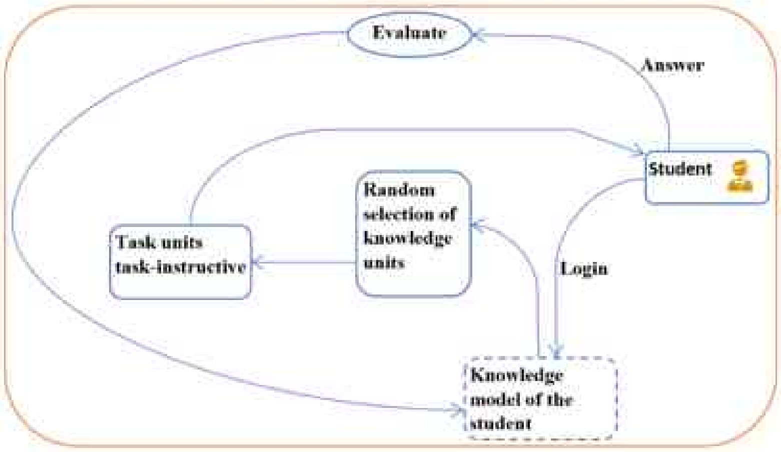

Evaluate
Answer
Student
Random selection of knowledge units
Task units task-instructive
Login
Knowledge model of the student

Figure 4.2 Student task model in an E-tutoring framework

4.6.2 Example Using Intuitionistic Fuzzy Logic Values

In the beginning, the student's knowledge level of each knowledge unit concept in the prototype system that a student wants to learn from is presumed to be completely unknown. MCQ tasks related to the knowledge unit concepts are used to check a student's knowledge level after every attempt of a student to the knowledge unit concept. After the student answers the MCQ tasks on one domain knowledge model concept, it is guessed that this knowledge unit has been learned. If the result is not satisfactory, the level values of the knowledge unit are not updated. A new IFV of membership functions l_T and l_F for a set of learned knowledge unit concepts (learned) is based on the student's answers to the MCQ tasks. The Intuitionistic fuzzy logic method describes the knowledge level of a particular student for a specific knowledge unit concept. The student's knowledge level of each knowledge unit concept is described as a qualitative variable concept knowledge, which has five values: weak, average, good, very good, and excellent. For example, let the units are Loops, For Loops, and While Loops according to (8) and (9):

initially, if $\boldsymbol{l_T = l_F = 0, n = 3, r = 0.7, 0.8, 0.75, g = \frac{1}{4} = 0.25}$

$\boldsymbol{Loops}$: $l_T = \frac{(3*0+0.25*0.7)}{4} = \frac{0.175}{4} = 0.04$ $\quad l_F = \frac{(3*0+(1-0.25)*0.7)}{4} = \frac{0.525}{4} = 0.13$

$\boldsymbol{For\ Loops}$: $l_T = \frac{(3*0+0.25*0.8)}{4} = \frac{0.2}{4} = 0.05$ $\quad l_F = \frac{(3*0+(1-0.25)*0.8)}{4} = \frac{0.6}{4} = 0.15$

$\boldsymbol{While\ Loops}$: $l_T = \frac{(3*0+0.25*0.75)}{4} = \frac{0.188}{4} = 0.047$ $\quad l_F = \frac{(3*0+(1-0.25)*0.75)}{4} = \frac{0.563}{4} = 0.14$

Updating the knowledge progress:

$\boldsymbol{Loops: l_T = 0.04, l_F = 0.13, n = 3,\ r = 0.7, g = \frac{2}{4} = 0.5}$

$l_T = \frac{(3*0.04+0.5*0.7)}{4} = \frac{0.47}{4} = 0.12$ $\quad l_F = \frac{(3*0.13+(1-0.5)*0.7)}{4} = \frac{0.74}{4} = 0.19$

$\boldsymbol{For\ Loop: l_T = 0.05, l_F = 0.15, n = 3, r = 0.8, g = 0.5}$

$l_T = \frac{(3*0.05+0.5*0.8)}{4} = \frac{0.55}{4} = 0.14$ $\quad l_F = \frac{(3*0.15+(1-0.5)*0.8)}{4} = \frac{0.85}{4} = 0.21$

$\boldsymbol{While\ Loop: l_T = 0.047, l_F = 0.14, n = 3,\ r = 0.75, g = 0.5}$

$l_T = \frac{(3*0.047+0.5*0.75)}{4} = \frac{0.516}{4} = 0.13$ $\quad l_F = \frac{(3*0.14+(1-0.5)*0.75)}{4} = \frac{0.795}{4} = 0.20$

Updating the knowledge progress:

$\boldsymbol{Loops: l_T = 0.12, l_F = 0.19, n = 3,\ r = 0.7, g = \frac{3}{4} = 0.75}$

$l_T = \frac{(3*0.12+0.75*0.7)}{4} = \frac{0.885}{4} = 0.22$ $\quad l_F = \frac{(3*0.19+(1-0.75)*0.7)}{4} = \frac{0.745}{4} = 0.186$

$\boldsymbol{For\ Loop: l_T = 0.14, l_F = 0.21, n = 3, r = 0.8, g = 0.75}$

$l_T = \frac{(3*0.14+0.75*0.8)}{4} = \frac{1.02}{4} = 0.255$ $\quad l_F = \frac{(3*0.21+(1-0.75)*0.8)}{4} = \frac{0.83}{4} = 0.207$

$\boldsymbol{While\ Loop: l_T = 0.13, l_F = 0.20, n = 3, r = 0.75, g = 0.75}$

$l_T = \frac{(3*0.13+0.75*0.75)}{4} = \frac{0.953}{4} = 0.238$ $\quad l_F = \frac{(3*0.20+(1-0.75)*0.75)}{4} = \frac{0.788}{4} = 0.197$

Updating the knowledge progress:

$\boldsymbol{Loops: l_T = 0.22, l_F = 0.186, n = 3, r = 0.7, g = \frac{4}{4} = 1.0}$

$l_T = \frac{(3*0.22+1.0*0.7)}{4} = \frac{1.36}{4} = 0.34$ $\quad l_F = \frac{(3*0.186+(1-1)*0.7)}{4} = \frac{0.558}{4} = 0.14$

$\boldsymbol{For\ Loop: l_T = 0.255, l_F = 0.207, n = 3, r = 0.8, g = 1.0}$

$l_T = \frac{(3*0.255+1.0*0.8)}{4} = \frac{1.565}{4} = 0.39$ $\quad l_F = \frac{(3*0.207+(1-1)*0.8)}{4} = \frac{0.621}{4} = 0.16$

$\boldsymbol{While\ Loop: l_T = 0.238, l_F = 0.197, n = 3, r = 0.75, g = 1.0}$

$l_T = \frac{(3*0.238+1.0*0.75)}{4} = \frac{1.464}{4} = 0.366$ $\quad l_F = \frac{(3*0.197+(1-1)*0.75)}{4} = \frac{0.591}{4} = 0.15$

Figure 4.3 shows the representation of the student knowledge level according to the true and false answer of the student.

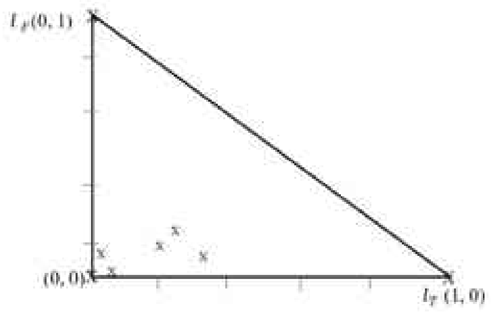


Figure 4.3 Student knowledge level status for a given knowledge unit

- In the case of IFVs (0.0, 0.0), we have no information about the student knowledge level.
- In the case of IFVs (0.0, 1.0), no competency detected about the student knowledge level for a given knowledge unit. This means the student did not learn the given knowledge unit.
- In the case of IFVs (1.0, 0.0), perfect competency about the student knowledge level for a given knowledge unit. This means the student learns the given knowledge unit.

Figures 4.4.a and 4.4.b explain the student knowledge level progress and their results related to the above example according to the true and false answer of the student. In Figure 4.4.a, the x axis shows the given knowledge units and y axis displays the level progress for the given knowledge units, while in Figure 4.4.b the x axis shows the task results (R1 to R6) and y axis displays the level progress for the given tasks (task1 to task4) which the student is tested related to the given knowledge units.

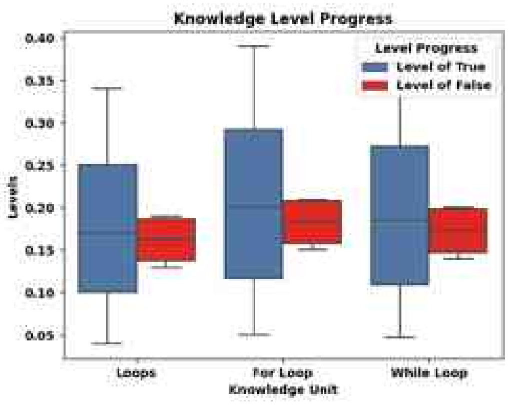


Figure 4.4.a Student knowledge level progress

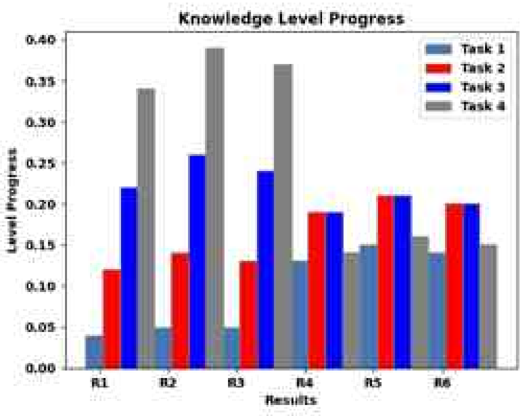


Figure 4.4.b Student results

4.7 Test Result

For evaluating the student knowledge level, we developed a prototype system which is integrated with the ontology domain knowledge model. A functional test was conducted with a MCQ task assessment performed. The result of this test will give feedback according to the student answer showing if the answer is correct or not correct then the student status will update and offer suggestions. The suggestion related to task question, student correct and wrong answer, the student result, knowledge level progress, and suggest reading materials.

Task assessments for different practices are conducted, which allows the student to choose a selected task questions related to the knowledge unit, and then the student select the correct answer and confirm it. After that, the prototype system check the answer whether it is correct or not correct according to that the system will calculate the knowledge level status applying equations (8) and (9) and provides feedback according to the student answer. The test result is described in Table 4.2 which shows the knowledge level status for the student in different task assessments related to the given knowledge units. This table displays the Level of true values and Level of false values, and the result is compared to the previous result to show that if there is a change in the determined status of the student. This status will be updated according to the Level of true values and Level of false values and then the system will detect the competency level of the student.

Table 4.2 knowledge level progress (status)

Results	Unit Name	Level of true	Level of false	Competency Level
r1	For Loop	0.000	0.000	no-competency detected
	While Loop	0.000	0.000	no-competency detected
r2	For Loop	0.178	0.089	very weak
	While Loop	0.167	0.083	very weak
r3	For Loop	0.222	0.044	weak
	While Loop	0.208	0.042	weak
r4	For Loop	0.326	0.118	weak
	While Loop	0.305	0.111	weak
r5	For Loop	0.440	0.123	average
	While Loop	0.412	0.116	average
r6	For Loop	0.516	0.126	good
	While Loop	0.483	0.119	good

In Table 4.2 initially the values of $l_T,\ l_F$ are set to $(0, 0)$ as we have no information about the knowledge level of the student. In other task assessment for the "For Loop" unit task assessment $l_T = 0.326\ (r4), l_F = 0.118(r4)$, which means the result of this unit is weak. In other case the knowledge level progress for the "For Loop" unit task assessment $l_T = 0.440\ (r5),\ l_F = 0.123(r5)$, which means the result of this unit is average. In other case the knowledge level progress for the "For Loop" unit task assessment $l_T = 0.516\ (r6),\ l_F = 0.126(r6)$, which means the result of this unit is good. Figures 4.5.a and 4.5.b show the knowledge level progress and the distribution of the knowledge level progress.

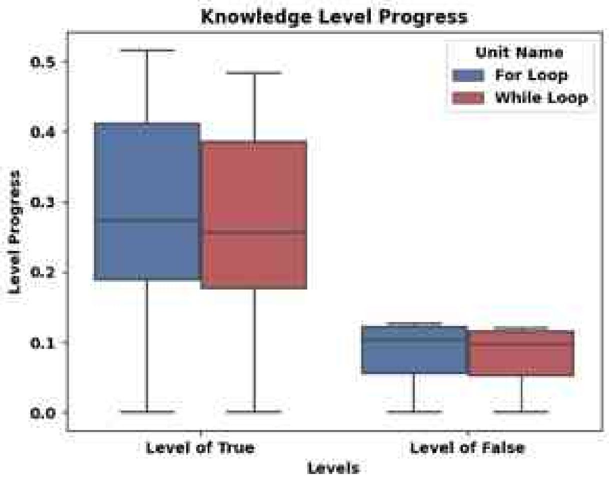


Figure 4.5.a knowledge level progress distribution

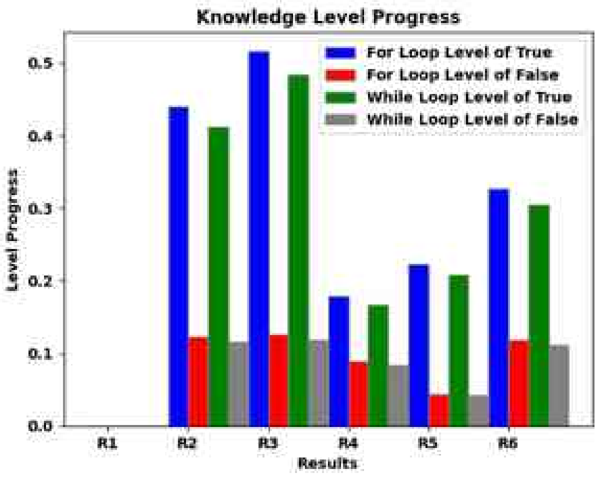


Figure 4.5.b The Results of knowledge units

4.8 Summary

In this chapter, Atanassov's Intuitionistic fuzzy logic technique is used to manage the uncertain or inaccurate information from the student who takes the MCQ task and deals with the uncertainty of the student's knowledge level.

I have used MCQ tasks related to the knowledge unit to check the student's knowledge level after every knowledge unit concept. After the student correctly answers the MCQ tasks related to the knowledge unit concept, it means that the student learned this knowledge unit so that the system will update the student's knowledge level accordingly. If the student's answer is not correct or not satisfactory, the level values of the knowledge unit are not updated. According to the update on student knowledge level, the E-tutoring system can generate a new task and provide personalized teaching and learning materials through a student model based on task scores on domain knowledge unit concepts. The E-tutoring system also offers suggestions. The suggestion related to task question, the student result, knowledge level progress, and suggest reading materials.

Thesis 3 **[5]**

I have presented a knowledge model to manage the students' competency levels and to support automatic decision-making in the smart tutoring framework. The proposed knowledge model supports the representation of uncertainty in students' knowledge and can be used to identify their status in the learning processes. In the assessment module, I used intuitionistic fuzzy logic values to represent the uncertainty in student knowledge status. The intuitionistic fuzzy logic

values can be used to automatically identify the status of the students' knowledge level. I have developed an algorithm for the evaluation of the student's performance using an accumulative task assessment approach with multiple-choice questions. The model automatically updates the student's knowledge level in an adaptive way. The proposed model provides interactive and adaptive assistance techniques by personalizing the teaching and learning materials according to the current knowledge levels.

Chapter 5 Storage model for the Ontology Model of an E-tutoring System

Ontology is a formal and explicit specification of a shared conceptualization. The selection of a specific storage technique always depends on the particular purpose of an application and the core features available in this storage technique to be utilized in particular software applications. Familiarizing with different ontology storage models with their respective features allows the user to choose an appropriate storage structure for high-performance applications. In my research, I investigate and compare three main different ontology storage methods from the viewpoint of adaptivity to the E-tutoring framework. The three models are ontology OWL, Jena Fuseki, and the relational model. I tested and evaluated the selected model storage systems by conducting an experiment applying ten test queries. The test result shows that Jena Fuseki and the relational model are the best model storage systems compared with the ontology OWL file. The suggested module can be improved by adding some functionalities and automatically supporting the generation of problems and their solutions.

5.1 Model Storage Systems

Many prominent investigators and software application agencies have suggested and created various ontology storage techniques and tools with different features. Different model storage systems are currently represented in the literature to integrate an ontology model with the E-tutoring frameworks [103], [104].

Three model storage systems are compared to integrate the ontology domain knowledge model with an E-tutoring framework. The models are OWL file, Jena Fuseki, and relational data model. Implementing an effective and efficient model storage system for the ontology model can improve application performance and enable semantic knowledge retrieval. The use of a particular storage model always depends on the specific purpose of the application and the core features available in the storage model to be utilized in particular applications [105]. Familiarizing different ontology storage models with distinct features allows us to select a proper storage structure for high-performance applications.

Ontology enables many features, such as supporting application integration, interoperability, semantic information retrieval, etc. Therefore, to achieve these features, there is a need for the ontology models to be stored in a standard form that is efficient, correct, scalable, and consistent in the knowledge domain. Hence, well-organized storage tools or models are necessary for storing and managing ontology models. The ontology storage tools and models are classified into native and database stores, as indicated in Figure 5.1 The native stores are intended to be created on the file system, while relational-based storage systems use relational or object-relational databases as backend stores [105]. Storing ontology models using native storage techniques is easier than using databases, as relational storage systems for storing ontologies do not directly support hierarchical relations. However, relational database management systems provide different key benefits. These features are performance, reliability, availability, and robustness.

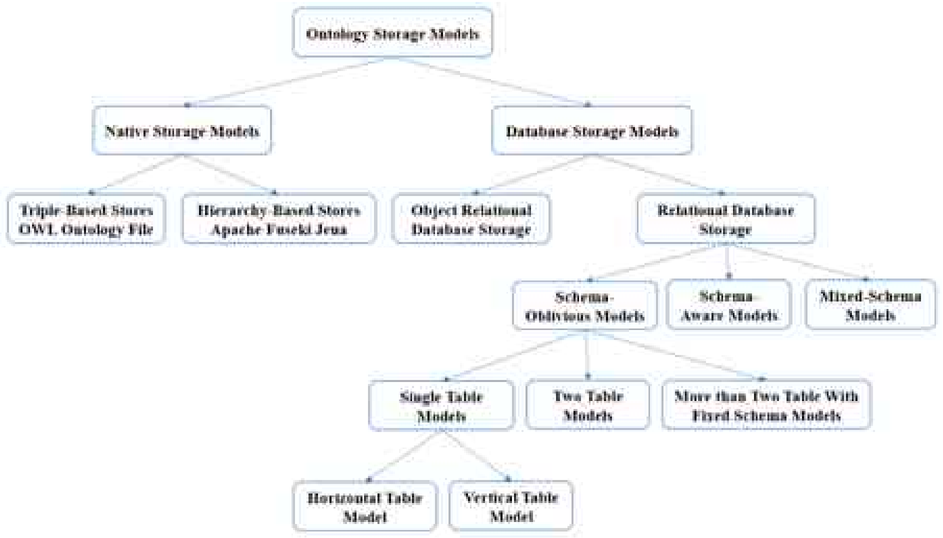


Figure 5.1 Taxonomy of ontology storage approaches [105]

5.1.1 Domain knowledge model using ontology OWL file

Web ontology language (OWL) ontology is a file structure for storing the domain knowledge module. Since OWL is a standard form for describing ontological knowledge, concepts, facts, and related relationships, it is possible to query that knowledge, concepts, and facts to create visual representations and make inferences using any OWL editors and tools like Protégé. In general, knowledge represented in OWL structure is human-readable and computer-readable and can express rich knowledge about the components of the domain knowledge module and the relationships between them.

The following list summarizes the key benefits of the ontology OWL file structure.

- Human-readable: human users can read and understand the data represented in ontology OWL structure. In other terms, the representation of the knowledge or information described in ontology OWL format can easily be read by humans.
- Computer-readable: computer can read and execute the data represented in an ontology OWL structure. In other words, a computer machine can automatically read and process the structure of the data represented in an ontology OWL format.
- Rich knowledge: the data represented in ontology OWL structure can contain massive knowledge.
- Consistency: the ontology OWL form has a consistent structure, meaning the representation of knowledge has the same formats.
- Flexibility: ontologies enable the separation of design and implementation concerns, so they are flexible to changes in specific implementation technologies.
- Efficient: OWL allows you to use your data model to support many different kinds of reasoning tasks. By "reasoning," we mean machine reasoning that makes implicit data explicit.
- Expressive: the feature of OWL is that we can use it to express highly complicated and subtle ideas about the data. OWL specifies concepts, relationships, and characteristics of these concepts and relationships in a human and machine understandable model.
- Reusability: ontologies enable the reuse of foundational concepts in (upper) ontologies that are domain-independent, and we can use them across different disciplines.

- Extensibility: ontologies allow further growth of the ontology for specific applications.
- Maintainability: ontologies facilitate identifying and correcting defects, accommodate new requirements, and cope with changes in an ontology.

The drawbacks of an ontology OWL file model is summarized in the following list.

- Low performance
- Time-consuming in the query execution which means it takes time to get the query result.
- Several issues we cannot do with user-created properties and classes that are possible with classes and properties that are part of the OWL language.
- There is no way to represent relationships between more than two things directly.
- It is challenging to build performant inference engines when we combine the ontology with too much enterprise data.
- Lack of explicit declarations of ontology entities, this problem actually makes the OWL specification incomplete and difficult to implement in practical systems.

5.1.2 Domain knowledge model using Jena Fuseki model

Apache Jena Fuseki is one of the widespread open-source RDF graph triple-stores. It is an essential component of a reliable, low-cost Knowledge Graph infrastructure. In addition, Fuseki supports SPARQL queries and updates, as well as SPARQL Protocol and Graph Store Protocol. Moreover, Apache Jena Fuseki [106] developed to create Linked Data and Semantic Web applications using RDF, Triple store, and OWL as data storage models. RDF uses RDF API and ARQ (SPARQL) to store data. RDF API interacts with the Key API elements to read and create RDF graphs. It is used to serialize the triples using Common forms like RDF/XML or Turtle.

ARQ (SPARQL) is Query RDF data using ARQ, a SPARQL 1.1 compliant engine. Therefore, ARQ provides integrated remote queries and text searches. Fuseki has a low resource usage regardless of great scalability, which makes it appropriate as a backend for any RDF datasets, from private data sources to medium-sized enterprise Knowledge Graphs [107]. In addition, Fuseki exposes triples via a SPARQL endpoint that can be accessible via HTTP. Fuseki also supports REST-style interactivity among RDF data. OWL employs ontology API and inference API as data storage models. Ontology API works through models, OWL, and RDFS to add more semantics to RDF data. While the inference API is used to reason over data to extend and check the content of the triple store. We can configure inference rules or use the built-in OWL and RDFS reasoners.

Triple-store repository is implemented with TDB database and Fuseki server architecture data store models. For persistent data, TDB supports a native high-performance triple-store. TDB also provides a full range of Jena APIs. Jena TDB [108] is a java-based robust built-in ontology storage and query layer for Jena Fuseki. It supports Jena API to implement it using the form of an RDF of triplets storage on a specific machine. Therefore, we can access and manage the TDB store with command line scripts through the Jena API. It uses N-Triple files to store ontologies. In addition, it supports SPARQL to query RDF data. In addition, TDB uses B+ tree indexing for RDF triples to enhance retrieval performance. Figure 5.2 illustrates the Apache Jena Fuseki framework architecture.

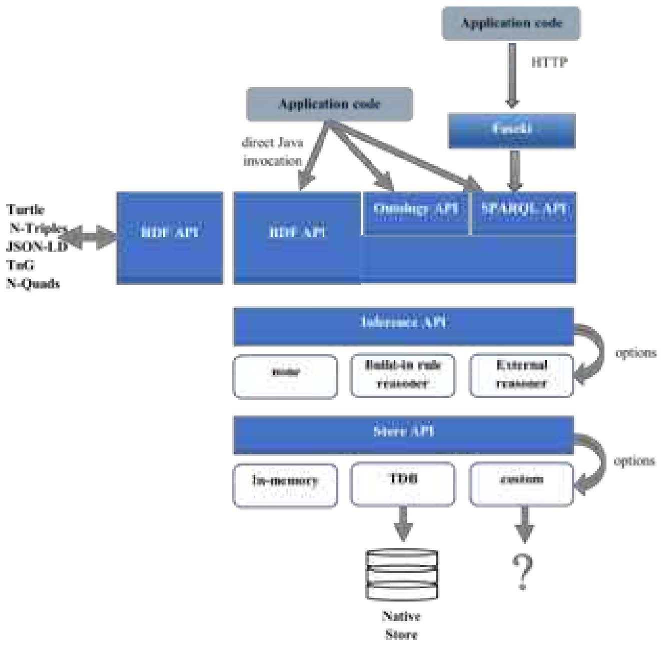


Figure 5.2 Apache Jena Fuseki architecture [106].

The ontology can be represented by a set of triples, including a collection of knowledge assertions. Each knowledge assertion can describe a triple with subject, predicate, and object. The subject of a triple is the thing about which knowledge classes can communicate with each other. The triple predicate indicates the sort of thing being communicated to different things; it corresponds to a way that two things can be related, or something is related to a literal. The triple object is the individual or a literal linked to the subject via the predicate. When the object is literal, it is often thought of as a value. Figure 5.3 and Table 5.1 show an example of a triple store structure.

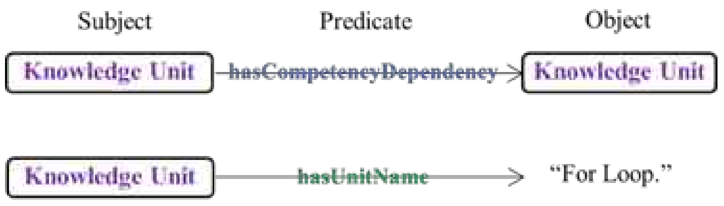


Figure 5.3 Triple stores structure

Table 5.1 Sample triple stores structure

Subject	Predicate	Object
Control Structures	hasTask	control structures in C++ Programming
Loop Structures	hasSlot	Syntax
For Loop	hasPart	Control_Variable
While Loop	hasMaterial	Chapter 5: Control Structures II Repetition https://slideplayer.com › slide

The Benefits of Jena Fuseki model is described in the following list.

- Interoperability
- Scalability
- Consistency
- Automatic reasoning
- Reliability

The drawback of Jena Fuseki model is expressed in the following list.

- Poor performance when parsing huge literal in query.

5.1.3 Domain knowledge module using a relational data model

Although native file systems use lower-level database techniques, they do not provide full database functionality. Several investigators have presented solutions for storing and organizing ontological data in database management systems to support full database functionality. There are primarily two possibilities for storing ontologies in database models, such as relational and object-relational models [105]. Relational database models directly support the storage of hierarchical relations in ontologies. Object-relational database models support through inheritance. Therefore, storing ontologies in object-relational databases is more straightforward than storing ontologies in relational database models. On the other hand, relational databases provide key features over object-relational databases. These features are performance, robustness, reliability, availability, etc. The benefits of database stores are optimized querying, recovery mechanism, and access privileges. But this technique has some limitations. The limitations are that ontology must have a fixed structure, and ontologies must be parsed and converted into a relational schema.

Investigators have proposed several techniques for storing and managing ontological data in relational database models. Relational ontology storage techniques are classified into schema-oblivious, schema-aware, and mixed-schema [105]. Schema-oblivious employs a fixed relational format for all of the ontological content. Schema-aware adopts a variable number of relational tables to represent ontological data closely. Finally, mixed-schema acquires both schema-oblivious and schema-aware techniques.

The following example explains the relational model storage systems with two schema tables and their relationship.

Knowledge Units			
<u>unit-id</u>	unit_name	parent_unit	unit_description
1	Loops	Control Structures	Loop in Programming Language

Task Units							
<u>task-id</u>	<u>unit-id</u>	task_name	task_category	question-schema	answer-schema	score	language
1	1	Repeat Good luck five times	Activity	How to repeat Good luck five times?	We can use print command five time, or we can use the loops to control the process.	2	English

Benefits of the relational data model is summarized in the following list.

- Performance
- Robustness
- Reliability
- Availability
- Optimized querying
- Recovery mechanism
- Access privileges

The following list describes the drawbacks of the relational data model. Namely, it does not support:

- application integration.

- the interoperability.
- the scalability.
- the consistency.
- automatic reasoning.

In the summary comparison table (Table 5.2), I applied the key features necessary for efficient ontology management. These criteria are query language, remote interface, access control, inference support, storage technique, as well as insert operation, update operation, delete operation, query cost, performance, robustness, reliability, availability, support application integration, Interoperability, Scalability, Consistency, and automatic reasoning.

Table 5.2 Comparison of the model storage system

Criteria	Jena Fuseki	OWL File	Relational Model
Query language	SPARQL	SPARQL	SQL
Remote interface	HTTP	HTTP	HTTP/DBMS net
Access control	HTTP based	HTTP based	HTTP based/ DBMS Security
Inference support	Yes	Yes	Yes
Storage technique	Triple file	Triple file	RDBMS
Insert operation	Yes (using API)	No	Yes
Update operation	Yes (using API)	No	Yes
Delete operation	Yes (using API)	No	Yes
Query cost	Fast	Slow	Fast
Performance	High	Low	High
Robustness	Yes	Yes	Yes
Reliability	Yes	Yes	Yes
Availability	Yes	Yes	Yes
Interoperability	Yes	Yes	No
Scalability	Yes	No	Yes
Consistency	High	Low	High
Automatic reasoning	Yes	Yes	No

5.2 Implementation and Test of the selected model storage systems

The domain knowledge model is implemented in three ways: ontology OWL file structure, relational data model, and Jena Fuseki model storage systems. The goal is to determine whether each of these models is better when evaluating them by selecting the execution time criteria.

Regarding the domain knowledge module implementation converted from the general concepts of formal logical ontology, which is explained in Chapter 3, this model is stored as a specific domain knowledge ontology case study for the loops in computer programming shown in Figure 5.4. Many relationships are used in the selected case study, such as specialization or generalization, association, competency dependency, and containment. Containment means that a specific knowledge unit within a domain contains different knowledge units (hasPart). Specialization or generalization means that certain knowledge units have specific knowledge units (is-a). The association means that a specific knowledge unit is associated with another knowledge unit. Competency dependency means specific knowledge units can have a competency dependency with other knowledge units.

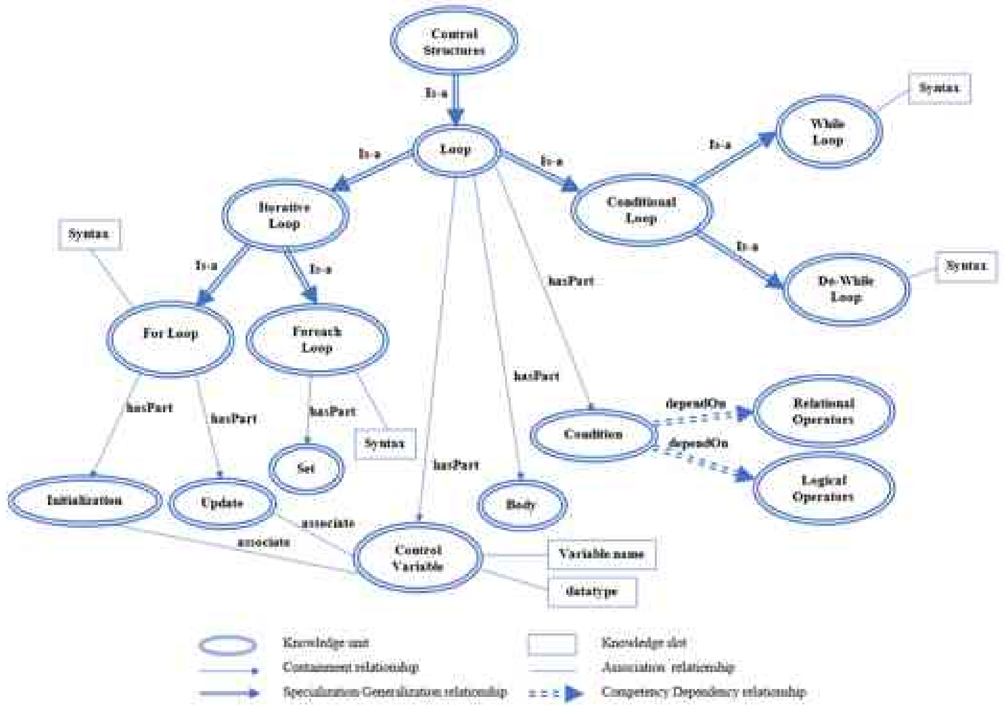


Figure 5.4 The specific domain knowledge module of teaching materials

Different styles of test queries are executed for testing and evaluating the performance of the selected model storage systems ranging from a very simple query, normal query, advanced simple query, intermediate query, upper intermediate query, and advanced intermediate query to a very complex query. The simple query refers to the problem that only needs to query atomic triple data with a relationship to get the answer. The query example for the very simple is "List all the knowledge units?". The intermediate query refers to the problem that needs to query for more than atomic triples data with one or two relationships to get the results. For example, the intermediate query is "What are the task units of each knowledge unit?". The upper intermediate query in the example is "Find all task units, related questions, and material units of the knowledge units?". The complex query refers to the problem that requires querying for more than atomic triples data with many relationships to get the results. The complex query in an example is "Find all task units, related questions, material units, and educational units of the knowledge units?". The levels of test queries cover the computer programming course.

5.2.1 Test of the domain knowledge module using an ontology OWL file

The ontology OWL file of the domain knowledge module is generated using Python programming language and the OWLready2 module. Python can be integrated with an OWL ontology using the Owlready2 module. Owlready2 was used to get transparent access to ontologies, manipulating the classes, object and data properties, individuals, property domains, ranges, annotations, constrained datatypes, disjoints, and class expressions.

The following code is an example of a complex query test using a SPARQL query language and RDFLib module in Python.

The expression g.parse("D:/Research Works/Dissertation/dkm_onto_module.owl") is used to connect Python and ontology OWL file.

```
g.parse("D:/Research Works/Dissertation/dkm_onto_module.owl")
PREFIX unit: <http://test.org/dkm_onto_module.owl#>
SELECT DISTINCT ?k_unit ?k_slot ?task ?question ?score ?level ?material
WHERE { ?k unit:unitName ?k_unit; unit:hasSlot ?s; unit:hasTask ?t; unit:hasMaterial ?m; unit:hasEduLevel ?e.
    ?s unit:slotName ?k_slot.
    ?t unit:taskName ?task; unit:questionSchema ?question; unit:taskScore ?score.
    ?m unit:materialName ?material.
    ?e unit:educationalName ?level.
}
```

5.2.2 Test the domain knowledge module using Apache Jena Fuseki

Apache Jena [106] stores knowledge as RDF triple-stores in directed graphs and supports the addition, removal, manipulation, storage, and publishing of that knowledge. Jena has several primary subsystems with clearly defined interfaces between them. Fuseki is an information publishing server that can display and update RDF models through the web utilizing SPARQL and HTTP.

The query test for Apache Jena Fuseki model is implemented using the Python language, SPARQLWrapper2 module and SPARQL query, which is indicated in the following code applying for a complex query test. The following statement is used for connecting Python with Jena Fuseki server.

sparql = SPARQLWrapper2("http://127.0.0.1:3030/Domain_Knowledge_Model/sparql")

```
sparql = SPARQLWrapper2("http://127.0.0.1:3030/Domain_Knowledge_Model/sparql")
sparql.setQuery("""PREFIX unit: <http://test.org/dkm_onto_module.owl#>
SELECT DISTINCT ?k_unit ?k_slot ?task ?question ?score ?level ?material
WHERE { ?k unit:unitName ?k_unit; unit:hasSlot ?s; unit:hasTask ?t; unit:hasMaterial ?m; unit:hasEduLevel ?e.
    ?s unit:slotName ?k_slot.
    ?t unit:taskName ?task; unit:questionSchema ?question; unit:taskScore ?score.
    ?m unit:materialName ?material.
    ?e unit:educationalName ?level.
}
```

5.2.3 Test the domain knowledge module using a relational data model

The logical knowledge model (LKM) is one of the most commonly used to model the data for intelligence applications. LKM includes further detail, supporting business system-related and data requirements [109]. LKM also provides the specifications for data describing the concepts, relationships, and interpretation of data values. LKM is a business abstraction of the data specifications. Figure 5.5 displays the logical knowledge model of the domain knowledge module for the teaching materials.

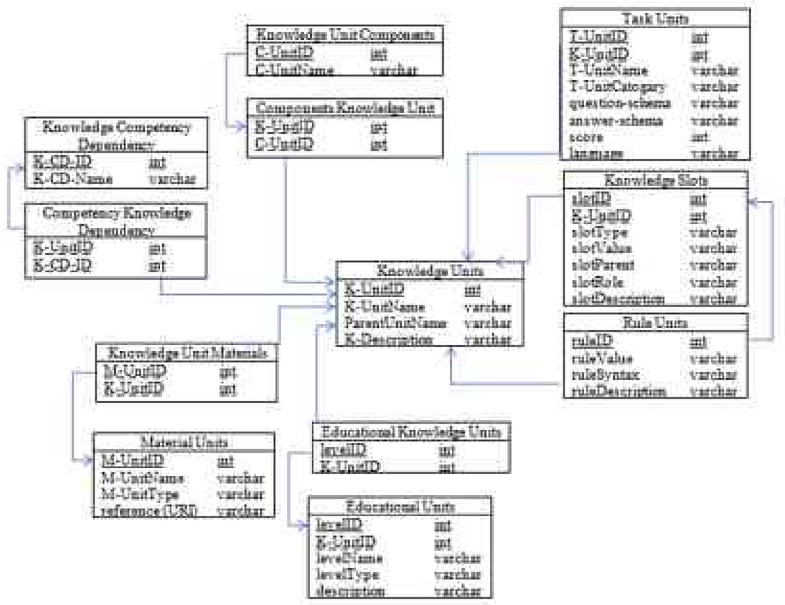


Figure 5.5 Logical knowledge model of teaching materials

A physical knowledge model (PKM) is a specific model that represents relational data objects (such as tables, columns, primary keys, and foreign keys) and their relationships [109]. PKM visually describes data structure as implemented via a relational database schema. In addition, it provides a visual abstraction of the database structure. An essential benefit of determining the PKM is that we can automatically derive the database schema from the model. It is possible due to the richness of metadata captured by the PKM and its close mapping to aspects of the database schema, such as database Tables, columns, Primary Keys and Foreign Keys. The Relational Database's main benefit is to organize the data into different levels and store it efficiently. It also supports data accuracy, integrity, and flexibility, reduces data redundancy and scalability and promotes the easy implementation of security methods, ease of use, and procedure.

Python and MySQL connector modules are used to create a relational model database and the table elements for the domain knowledge module. The test query for the relational data model was implemented using SQL Query language, which is displayed in the following code applying for a complex query test.

```
SELECT unit_name, task_name, slot_value, question_schema, task_score, material_name, level_value
FROM KnowledgeUnit k
    JOIN KnowledgeSlot s ON k.unit_id=s.slot_id
    JOIN TaskUnit t ON k.unit_id=t.unit_id
    JOIN MaterialUnit m ON k.unit_id=m.material_id
    JOIN EducationalUnit e ON k.unit_id=e.level_id
```

5.3 Test Result

The selected model storage systems are tested by conducting an experiment applied to a small and large database concerning ten test queries. Each test query is executed ten times, measuring the execution time for every test query. The result of this experiment was described in Tables 5.3 and 5.4, which show the execution time average for each test query for the three models applying for the small and large amounts of data.

Table 5.3 Test query performance for small database

Query Execution Time	Jena Fuseki	OWL file	Relational Model
Query Test 1	0.020	0.205	0.021
Query Test 2	0.024	0.220	0.028
Query Test 3	0.029	0.253	0.028
Query Test 4	0.031	0.218	0.030
Query Test 5	0.027	0.256	0.030
Query Test 6	0.029	0.210	0.030
Query Test 7	0.033	0.211	0.031
Query Test 8	0.032	0.250	0.031
Query Test 9	0.026	0.253	0.029
Query Test 10	0.037	0.257	0.028

Table 5.4 Test query performance for large database

Query Execution Time	Jena Fuseki	OWL file	Relational Model
Query Test 1	0.024	0.361	0.022
Query Test 2	0.030	0.406	0.031
Query Test 3	0.033	0.357	0.031
Query Test 4	0.031	0.392	0.030
Query Test 5	0.043	0.354	0.034
Query Test 6	0.039	0.408	0.033
Query Test 7	0.039	0.374	0.030
Query Test 8	0.034	0.379	0.030
Query Test 9	0.032	0.373	0.031
Query Test 10	0.035	0.443	0.031

Comparing the execution time between Table 5.4 and 5.4 linking the value of some test query related to the small and large database are given in the following explanations.

In Table 5.3, the query test 1 for the ontology OWL file model takes 0.205 seconds, while Jena Fuseki model takes 0.020 seconds, and the relational data model takes 0.021 seconds. For query test 7, ontology OWL file model takes 0.211 seconds, while Jena Fuseki model takes 0.033 seconds and relational data model takes 0.031 seconds. For other select query test 8, ontology OWL file model takes 0.250 seconds, Jena Fuseki model takes 0.032 seconds, and relational data model takes 0.031 seconds.

In Table 5.4, a query test 2 for the ontology OWL file takes 0.406 seconds, while Jena Fuseki takes 0.030 seconds, and the relational data model takes 0.031 seconds. For query test 7, the ontology OWL file model takes 0.374 seconds, while Jena Fuseki model takes 0.039 seconds and relational data model takes 0.030 seconds. For other select query test 9, ontology OWL file model takes 0.373 seconds, Jena Fuseki model takes 0.032 seconds, and relational data model takes 0.031 seconds.

As a result, the investigation showed that Jena Fuseki and relational data model have a minimum execution time which means that Jena Fuseki and relational data model are the best model storage systems compared with ontology OWL file, as shown in Figures 5.6-5.12. Figures 5.6-5.9 explain the query test result according to the execution time of the three models in a line graph and bar chart. Figures 5.10-5.12 show the distribution of the three models with the execution time.

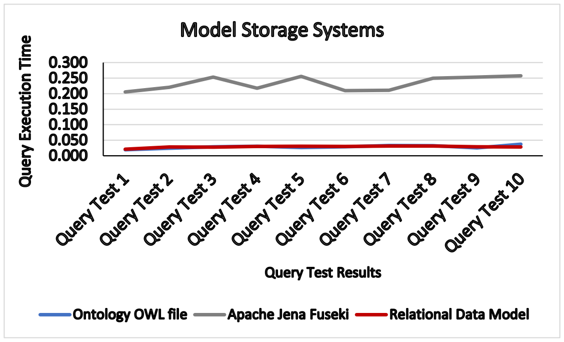


Figure 5.6 Test query performance for small database

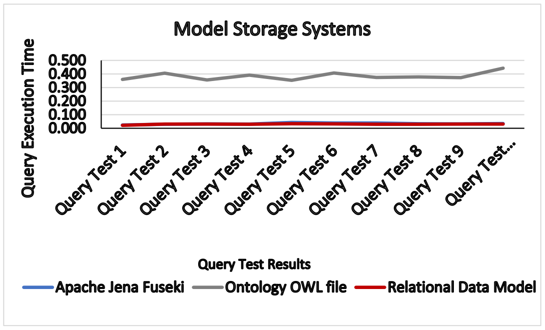


Figure 5.7 Test query performance for large database

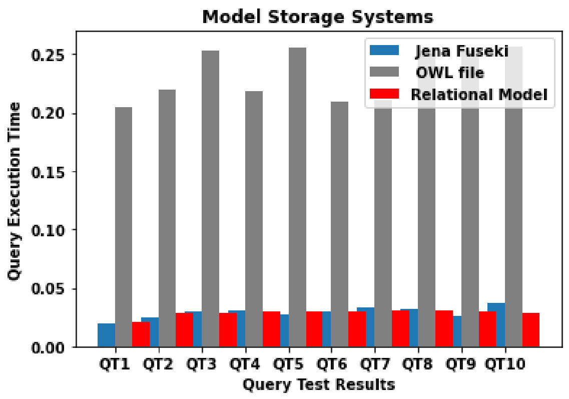


Figure 5.8 Query execution time for small database

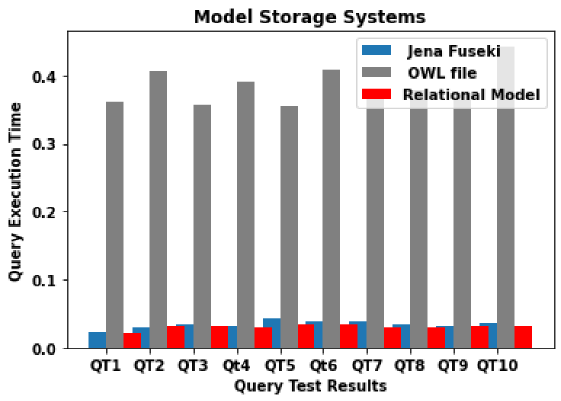


Figure 5.9 Query execution time for large database

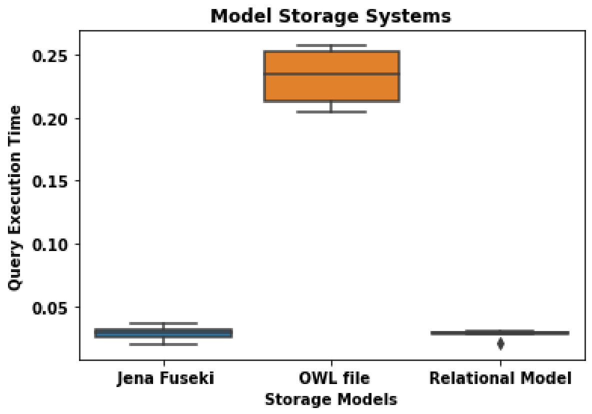


Figure 5.10 Distribution of the model storage systems for small database

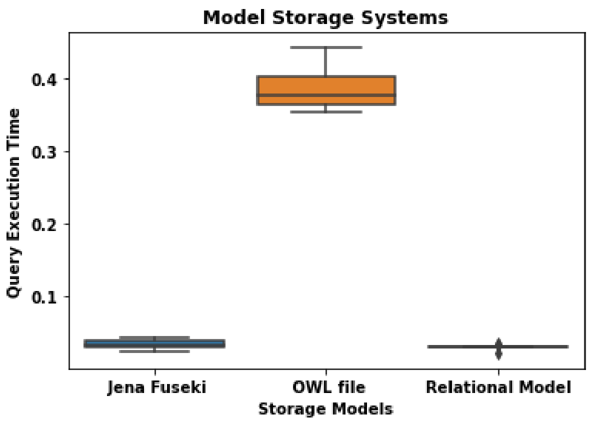


Figure 5.11 Distribution of the model storage systems for large database

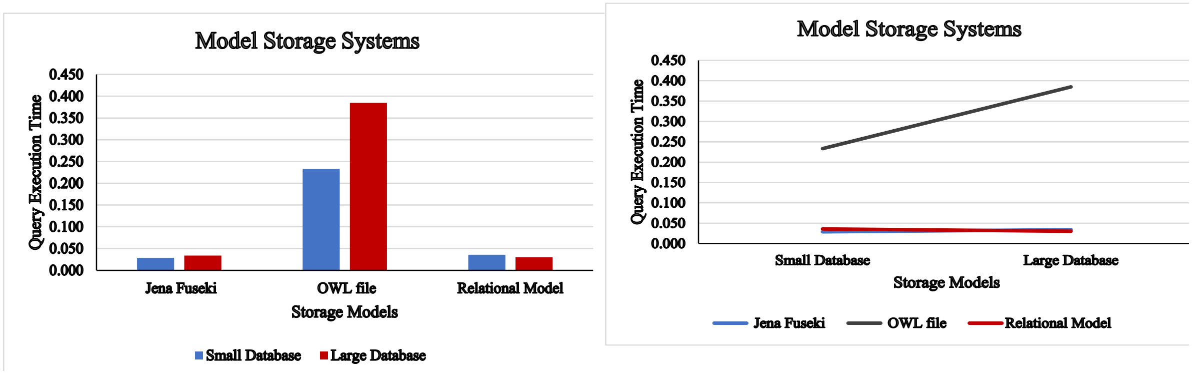


Figure 5.12 Distribution of the storage models for small and large database

Table 5.5 and Figure 5.13 below illustrates the storage size on the disk of the three model storage systems which are investigated in this work.

Table 5.5 Size on disk of the model storage systems for small and large database

Size on the Disk	Small Database	Large Database
Jena Fuseki	72.35 KB	118.48 KB
Ontology OWL File	72.30 KB	118.00 KB
Relational Model	1950 KB	2430 KB

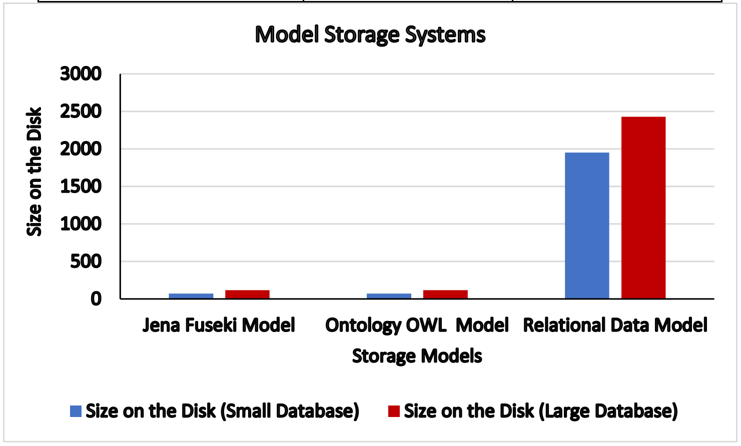


Figure 5.13 Size on disk of the model storage systems

5.4 Summary

In this chapter the author introduces the integration of the formal logical ontology model into an E-tutoring system. Three model storage systems are compared with different criteria.

Ontology provides more coherent and easy navigation for moving from one concept to another in the ontology structure. Another valuable feature is that ontology is easy to extend as relationships and concept matching is easy to add to existing ontology. Ontology also provides the means to represent any type of knowledge domain format, including unstructured, semi-structured, or structured knowledge, enabling smoother knowledge integration, easier concept and text mining, and data-driven analytics.

Implementing an effective and efficient ontology storage system improves application performance and enables semantic knowledge retrieval. We selected three model storage systems for integrating the ontology model with an E-tutoring framework. The models are Ontology OWL Model, Jena Fuseki model, and Relational data model. Familiarizing different ontology storage models with distinct features allows us to select a proper storage structure for high-performance applications. As a result, the investigation showed that Jena Fuseki and relational data model have a minimum execution time which means that Jena Fuseki and relational data model are the best model storage systems compared with ontology OWL file.

By combining the domain knowledge module with E-tutoring systems can enhance the quality of intelligent problem-solving. Also, it will be possible to reuse the knowledge domains. Finally, the domain knowledge module for E-tutoring systems can improve the teaching and learning process, support recommendations, generate hints. In the future the module can be enhanced by adding some functionalities and automatically support the generation of problems and solutions.

Thesis 2 ***[4]***

I have introduced a novel formal logical ontology domain knowledge model for an E-tutoring system, which presents the conceptual components of teaching material frameworks. The core components of this model are knowledge units and related units covering the specific teaching and competency dependency units, including the specialization, containment, association, and dependency graph relationships among the knowledge units in the ontology domain. For storing this model, I compared three model storage systems the OWL ontology file, Jena Fuseki, and the relational model. The key aspect is determining the suitable storage model to store the ontology model in a standard, efficient, scalable, and consistent format. The main role of this comparison is to improve application performance, enable semantic knowledge representation, and achieve efficient information retrieval. The investigation shows that Jena Fuseki and the relational model are the best ontology storage tools. They provide support facilities for storing, inferring new knowledge, and managing the ontology domain knowledge But the relational model storage needs more space and does not provide automatic reasoning. So, Jena Fuseki is the best candidate for model storage and query.

Chapter 6 Functionality of the Prototype System

The rapid growth and development in technologies, especially in the information science and technology domain, provide many languages, modules, and packages for creating and managing ontology models and developing an E-tutoring framework. Many scholars considered Python as one of the common, widely utilized, and adopted languages when implementing an ontology domain for the domain knowledge model. It is an object-oriented, interpreted, and extensible programming language that provides a powerful and excellent combination of simplicity and adaptability in different disciplines [110] and [111]. In addition, Python offers several modules for constructing and manipulating the ontology, developing web application frameworks, and executing the SPARQL query to integrate the prototype system with the ontology domain knowledge model (ODKM). For manipulating the ODKM, OWLready2 module is used for creating the components ontology model and their relationships. The goal is to get transparent access to ontologies, manipulating the classes, object and data properties, individuals, property domains, ranges, annotations, constrained datatypes, disjoints, and class expressions [111]. We used the SPARQLWrapper module for integrating the ODKM with the prototype system. SPARQLWrapper is a SPARQL endpoint Python module to execute the SPARQL query language in Python. For designing the prototype system, I used the Flask web framework to combine the ODKM with the prototype framework interface. Flask is one of Python's most commonly used web development frameworks, based on Werkzeug and Jinja2. Flask uses Jinja2 templates for designing the interface to extend the functionality of the prototype system. Jinja2 is a web application template engine that allows data to be shared and processed before being converted into content and sent back to the user [112]. In addition, it is a web template engine with a simple and organized file structure used to develop and design a user interface for Python. The interface is an interactivity point between the end user and the prototype system. The success of development and the failure of a prototype system depends on Interface Design. Interface design focuses on considering the user needs and ensures that the interface contains components that provide the features of easy to use, easy to access, and easy to understand to simplify user actions of the prototype system.

6.1 Case Study in the Domain of Teaching Programming

The ontology domain refers to an area in teaching programming at introduction level. The selected domain relates to a specific programming language, namely C++. This domain is an essential topic for most students studying in many disciplines, such as education, engineering, and mathematics. The domain knowledge model covered in this example case study is the loop structures in a programming language. The main goal is to help the students to understand and practice the loop structures in computer programming. This example case study is based on the selected ontology model of the domain knowledge model that presents the conceptual components of teaching materials and their relationships, which are proposed in this research. In this section, we construct the ontology model on the programming domain of loop structures. This model includes knowledge units, knowledge slots, task units, rule units, material units, and educational units, with many types of relationships among the knowledge units. The relationships are specialization or generalization, containment, components, associations, and dependency graph relationships. In section 5.2, the example case study is mapped with the proposed domain knowledge model.

In order to measure what the students understand and learn, it is not sufficient to evaluate their knowledge level and skills at the final step of the study program. It is also important to discover the students' present knowledge level of a domain, so that we can determine more specifically the knowledge level and skills they have acquired during the study program. In this case, we need a technique to describe the operations for testing and evaluating the student's knowledge level.

In the prototype system, a knowledge model is separated for every student. This knowledge model is constructed based on the domain knowledge model. It stores a knowledge level for each knowledge unit. The level value should consider that, in most cases, we have no information about the competency level for all knowledge units. Thus, we should manage the case of missing information. In order to denote the uncertainty information, an Intuitionistic logic value is used with the level value of true and the level value of false, which described in more details in Chapter 4. The following steps define the operations of evaluating the student knowledge level:

- Initially, for every knowledge level, I set $(0, 0)$ for the level value.
- First, the system will check if there are knowledge units (k-units) with uncertainty or low value. In that case, the system should choose one of these k-units where this value is equal to the product of the uncertainty level of the measured level and the inverse of the adjusted expected competency level. We order a selection probability for each task unit (the higher the relevance value, the higher value has the chance of selection).
- The student performs the assigned task, and we get the result.
- We evaluate the response and get an accuracy level for each k-units related to this task.
- We update the knowledge map of the student.
- We give more suggestions to the student on which parts and study aids should he/she learn and practice.

6.2 Loop Structures Knowledge Units Case Study

Loops in programming are the knowledge units case study covered in this research. Loops in programming language allow us to repeat one or more expressions many times as required. Loops are instructions that repeat until a particular condition is satisfied. Typically, in a particular operation, such as obtaining and updating data, conditions are verified, such as whether a counter has reached a specific counter. Loops are part of control structures which include For, While, and Do-While loops. In Figure 5.4 I consider the loop structures as a knowledge unit use case.

6.2.1 Knowledge Units (K-Units) Concepts

In this domain knowledge model, we can identify the following knowledge units, Control Structures, Loops, Iterative Loops, For Loops, Foreach Loops, Conditional Loops, While Loop, Do-While-Loop, Condition, Body, Control Variable, Relational Operator, Logical Operator, Initialization, Update, Set

Knowledge unit specialization in loop structures

In the domain knowledge model, we can define different specialization relationships between the selected knowledge units given in the following list.

- Loop Structures is-a Control Structures
- Conditional Loops and Iterative Loops is-a Loop Structures
- for loop, nested for loop, and foreach loop is-a Iterative Loops
- While Loop and Do-While Loop is-a Conditional Loops
- logical operators and relational operators is-a Operators

Knowledge unit Components in the loop structures

In the selected domain knowledge model, each knowledge unit can include other knowledge units. The following list described the components relationship in the domain knowledge model.

- Condition is a part of a loop structure.
- Body is a part of a loop structure.
- Control Variable is a part of a loop structure.
- Update is a part of a loop structure.
- Set is a part of a foreach loop.
- Initialization and Update are parts of for loop.

Knowledge unit Competency Dependency in loop structures

We can identify different competency dependencies between the knowledge units in the domain knowledge model. Regarding for-loop syntax, we can define the following **competency dependencies**: relational operator, logical operator (condition), and update (incrementation or decrementation), which are indicated in the code below.

```
for (counter=0; counter<5; counter++) {
cout<<counter;
}
```

Where (counter =0) is the initial value, (counter <5) is the condition to stop repeating, and (counter++) is the incremental value, in this case, the incremental value is one.

To understand the Loops construct, the learner must first understand the concepts of Variable Assignment (variable=intial-value), Relational Operators (varible<=end-value), and Increment/Decrement Operators (variable++ or ++variable). The syntax of While-loop and Do-While-loop **has the competency dependencies** counter, relational operator, logical operator, condition, and Update (incrementation or decrementation). In computer languages, conditional loops tell the computer to perform an action until something happens. In our case, that ‘something is ‘until counter<5 (While-loop) or counter>5 (Do-While-loop).’ When that happens, the loop can stop printing.

Competency Dependency

We can find many competency dependencies between the knowledge units, such as Condition **depend-on** Logical Operators and Relational Operators.

6.2.2 Knowledge Slots (K-Slots) in loop structures

The knowledge units can have some slots. For example, the explanation of the loop structure has the slots syntax, datatype, and variable-name and includes components (containments), body, condition, and control variables. And has competency dependency logical operators and relational operators, the Knowledge Slots (K-Slots) Concepts are given in the following list.

- Loop structures has-syntax.
- Loop structures has-variable-name.
- Loop structures has-datatype.

6.2.3 Material Units (M-Units) for loop structures

In the domain knowledge model, we can relate each knowledge unit to different material units, given in the list below.

- C++ Programming: From Problem Analysis to Program Design (book: http://www.icivil-hu.com/Civil-team/2nd/c++/)
- Control Structures II Repetition (online material: https://slideplayer.com/slide/5025236/)

6.2.4 Educational Units (E-Units) Concepts

In this domain knowledge model E-units refer to educational level, we can identify several educational levels for students to practice their knowledge units: beginner, intermediate, upper-Intermediate, and advanced.

6.2.5 Rule Units (R-Units) in loop structures

If the learner wants to use the loops without knowledge about the Variable Assignment, Relational Operations, and Increment/Decrement Operators", then the possibility of getting the loop concepts K-unit is not satisfied.

R-Units Concepts

The following list is an example of selected rules for the rule units used to put constraints on students in learning the Loop structures knowledge unit.

- Every loop has a condition.
- Every loop has a body (a block of statement/s)
- Every loop has logical operators.
- Every loop has relational operators (Variable Assignment)
- Every loop has syntax.
- Every loop has an update value.
- Every loop has an Initialization value.
- Every loop has a control variable.

SWRL Rules

Semantic Web Rule Language (SWRL) is a suggested language for the Semantic Web that can be used to extract rules as well as logic, combining OWL DL or OWL Lite with a subset of the Rule Markup Language (itself a subset of Datalog). SWRL rules are used to integrate 'if… then…' rules in ontology model of the domain knowledge model. The SWRL rules are also used to infer new knowledge from the current OWL domain knowledge model. For example:

Students(?x) ^ KnowledgeUnit(?k) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ hasTask(?k, ?t) ^ taskScore(?t, 2) -> hasEduLevel(?k, beginner_level)

The rule above means that students who take a knowledge unit that contains other knowledge units and have the following relationships with the knowledge slot, components and competency dependencies with another knowledge units and related task activities have a score equal to 2 puts them at the beginner level.

We can assign many rules for the selected domain knowledge model in the case study. The following list is several SWRL rules on the knowledge units, knowledge slots, task units, educational units, and their relationships.

- KnowledgeUnit(?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ associate(?k, ?k) ^ hasCompetencyDependency(?k, ?k) -> knowledgeName(?k, Loops)
- KnowledgeUnit(?k)^hasTask(?k, ?t) ^ taskScore(?t, 2) -> hasEduLevel(?k, beginner_level)
- KnowledgeUnit(?k) ^ hasTask(?k, ?t) ^ taskScore(?t, 3) -> hasEduLevel(?k, intermediate_level)
- KnowledgeUnit(?k) ^ hasTask(?k, ?t) ^ taskScore(?t, 4) -> hasEduLevel(?k, upper_intermediate_level)
- KnowledgeUnit(?k) ^ hasTask(?k, ?t) ^ taskScore(?t, 5) -> hasEduLevel(?k, advanced_level)
- KnowledgeUnit(?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) -> associate(?k, ?k)
- KnowledgeUnit(?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) -> taskScore(?k, 2)
- KnowledgeUnit(?k) ^ taskScore(?k, 2) -> hasEduLevel(?k, beginner_level)
- KnowledgeUnit(?k) ^ hasPart(?k, ?k) -> hasCompetencyDependency(?k, ?k)
- KnowledgeUnit(?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ taskScore(?k, 5) ^ hasTask (?k, ?t) -> hasEduLevel(?k, advanced_level)

Also, we can assign some rules for the students. The following list is some SWRL rules for the students to practice the knowledge units, knowledge slots, task units, and their relationships.

- Student(?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ hasTask(?k, ?t) ^ taskScore(?t, 2) -> hasEduLevel(?k, beginner_level)
- Student(?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ hasTask(?k, ?t) ^ taskScore(?k, 3) -> hasEduLevel(?k, intermediate_level)
- Student (?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ hasTask(?k, ?t)) ^ taskScore(?k, 4) -> hasEduLevel(?k, upper_intermediate_level
- Student (?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) ^ hasTask(?k, ?t) ^ taskScore(?k, 5) -> hasEduLevel(?k, advanced_level)
- Student (?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ associate(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) -> hasTask(?k, ?t)
- Student (?x) ^ takes(?x, ?k) ^ hasPart(?k, ?k) ^ associate(?k, ?k) ^ hasSlot(?k, ?s) ^ hasCompetencyDependency(?k, ?k) -> knowledgeName(?k, Conditional_Loops)

Figure 6.1 displays the design of a specific domain knowledge ontology model as a case study for the IT domain in an E-tutoring system for computer programming. We use different types of relationships in the case study, such as specialization or generalization, association, and containment. Containment defines that every knowledge unit can contain sub- knowledge unit components (has-a) within the selected domain. Specialization or generalization means certain knowledge units or domains have specific knowledge unit components (is-a). The association means that a specific knowledge unit or concepts are associated with another knowledge unit.

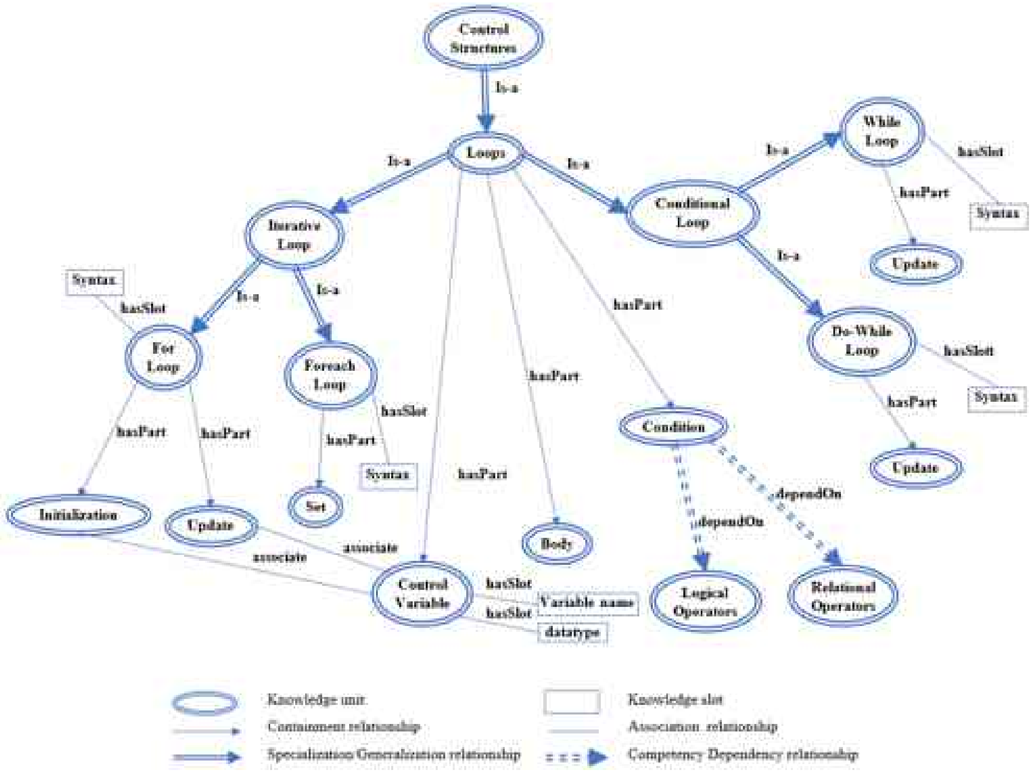


Figure 6.1 Loop structures knowledge units

6.2.6 Task Units (T-Units) in loop structures

Task Units (T-units) refer to a set of activities related to different knowledge units or knowledge slots. In other terms, T-units are activities generated from the knowledge units and knowledge slots to be performed by students when they need to practice their knowledge. The T-units contain task types followed by a list of arguments: taskID, taskName, taskQuestion, taskAnswer, and taskScore. It is also given as a form of question-answer pair.

We generated different task activities from the knowledge units (k-units) and knowledge slots (k-slots) in the selected domain knowledge model in the programming domain. Each k-units and k-slots can include one or many task activities. The following are some task activities selected from the generated task units.

Task activity 1: Print "Good luck" five times. (Task activity of control structure)

The traditional solution is to write each statement action in a single line using print command related to specific programming language, which is given here using "cout" print command in C++ programming.

```
cout<<" Good luck";
cout<<" Good luck";
cout<<" Good luck";
cout<<" Good luck";
cout<<" Good luck";
```

Task activity 2: Print “Good luck” five times applying for-loop. (Task activity of for loop)
The solution considering for-loop: for (counter =0; counter <5; counter ++)
First, what action do we need to repeat? “Good luck”. Second, add “Good luck” to the for-loop statement:

```
 for (counter =0; counter <5; counter++) {
       cout<<” Good luck”; }
```

Task activity 3: Print good luck five times using a while loop. (Task activity of while loop)

The solution considering While-loop. We need to keep performing these instructions over and over. Also, We need to know when to stop. We will stop when we meet the condition.

```
 counter=0;
 while(counter<5) {
        cout<<” Good luck”;
        counter+=1; }
```

Task activity 4: Print good luck five times using do-while loop. (Task activity of Do while loop)
The solution considering Do-While-loop:

```
      counter=0;
       Do {
          cout<<” Good luck”;
          counter+=1;}
       while(counter<5);
```

The following are OWL examples of the task activity mapping to ontology individuals (instances). We can define an instance of the task units as follows:

```
<owl:NamedIndividual rdf:about="#task10">
        <rdf:type rdf:resource="#TaskUnit"/>
        <questionSchema rdf:datatype="#string">What will be the output after the following statements?
                        x = 1;
                        while (x < 10){
                            cout<<x<<" ";
                            x++; }
        </questionSchema>
        <A rdf:datatype="#string">1 2 3 4 5 6 7 8 9</A>
        <B rdf:datatype="#string">0123456789</B>
        <C rdf:datatype="#string">10</C>
        <D rdf:datatype="#string">1 2 3 4 5 6 7 8 9 10</D>
        <answerSchema rdf:datatype="#string">A</answerSchema>
        <taskScore rdf:datatype="#integer">3</taskScore>
    </owl:NamedIndividual>
```

6.3 Integration of an ontology model with the E-tutoring framework

The Flask framework is used to develop the prototype system and integrate the constructed ontology for the selected domain knowledge model with the E-tutoring system. Flask framework is a Python web development framework module that allows the creation of web applications quickly. It has simple, easy-to-extend vital features, including URL routing and a template engine.

Furthermore, a web framework describes a set of modules and libraries that allow web application designers to develop applications without concerning low-level details such as protocol, thread control, Etc.

6.4 Implementation of The Ontology Domain Knowledge Model

Software system design defines the components of an E-tutoring framework like modules, architecture, interfaces, and domain knowledge models based on the selected specifications. The methodology of building a software application is to specify and develop a framework that satisfies the specific requirements of an E-tutoring system. System design applies different techniques and principles to create an E-tutoring system, which allows its teaching material to the learners.

Information science and technology provide many modules and packages for developing E-tutoring systems and constructing and managing ontology databases. For the last decades, formal ontologies have become widely used in computer science to structure data and knowledge. Python programming language has become more widespread in teaching, industry, and research.

Ontology-oriented is a method of programming provided by Owlready2 that looks like object-oriented programming in which objects and classes are the ontology entities [110]. Owlready2 offers many benefits as follows:

- The expressiveness of standard ontologies offers the ability to describe detailed knowledge in depth, link them together, and reason about the knowledge.
- The speed of access to a relational database has quick storage and search capabilities.
- The ability of object-oriented features of the programming languages.
- The capability to execute imperative lines of code offering orders to the computer program which is impossible with a knowledge base or an isolated ontology.

Owlready incorporates a graph-based knowledge base with an OWL level of semantics. This knowledge base is known as quadstore as it keeps quadruplets in an RDF structure called RDF triples structure (subject, property, object), which can be added to an ontology identifier [110].

Quadstore stores all knowledge from the imported ontologies in a compressed form. It may be set in RAM or on a drive in the form of an SQLite3 knowledge base file. In addition, Owlready imports the entities of an ontology on request into Python when used and releases them from RAM automatically, when they are no longer required [110]. Furthermore, when these entities are updated in Python, Owlready automatically modifies the quadstore. Figure 6.2 below displays the general architecture of Owlready.

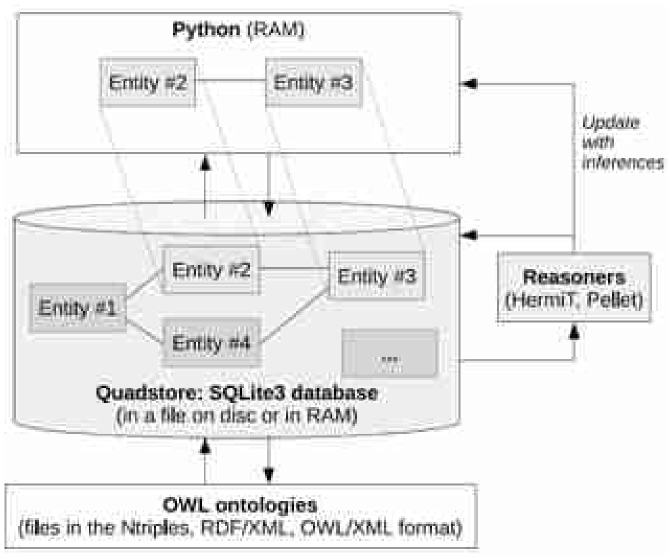

Python (RAM)
Entity #2
Entity #3
Update with inferences
Entity #2
Entity #3
Entity #1
Entity #4
...
Reasoners (HermiT, Pellet)
Quadstore: SQLite3 database (in a file on disc or in RAM)
OWL ontologies (files in the Ntriples, RDF/XML, OWL/XML format)

Figure 6.2 Owlready general architecture for Python [110]

Figure 6.2 is an architecture that allows the possibility to process large ontologies while fast locating specific entities, such as a textual search and reasoning. It also allows a level of semantics equivalent to OWL ontologies. However, Owlready can also be employed as a simple object database, a graph-based, or an Object-Relational Mapper (ORM) without benefiting from the advantages that ontologies' expressiveness can provide [113]. Table 6.1 provides a relationship between the terminologies of the world of object-oriented programming and that of formal ontologies:

Table 6.1 Relationship between terms in object-oriented programming and the formal ontologies [113]

Object-oriented programming	Formal ontology
Object	Entity
Module	Ontology
Class	Class
Class inheritance	Class Property, role, or predicate, also called “is-a” relation
Not allowed	Property inheritance
Instance	Individual
Attribute or property	Property, role, or predicate
Value of an attribute for an instance	Relation
Class name	IRI
Datatype	Datatype
Method	Not allowed
Not allowed	Logical constructor
Not allowed	Restriction
Not allowed	Disjoint

An ontology can contain a group of entities in which the entities can be classes, properties, or individuals. Regarding to the object oriented model of Python, there are three main differences:

- Properties are defined independently and outside the classes.
- Individuals can belong to one class and several classes (this is multiple instantiations, similar to multiple inheritances, but for instance).
- The ontology is based on the Open-World assumption, which means that anything not expressly prohibited is considered possible. For example, suppose that the knowledge unit

"Loops in Programming" has as task "Print Welcome five times." In that issue, the concept of an Open-World assumption leaves the possibility that other additional tasks exist for this knowledge unit. Since "Print Welcome five times" is the only task, it must also indicate that "Loops in Programming" has no other tasks than "Print Welcome five times" (typically using an OWL restriction).

6.4.1 Ontology Construction Using Python and Owlready2 module

Regarding the ontology domain knowledge model implementation as a back end of an E-tutoring framework, Python and Owlready2 modules are used. Owlready2 is used to obtain transparent access to ontologies and manipulate the classes and the individuals, object properties, data properties, annotations, property domains, ranges, constrained datatypes, disjoints, and class expressions (such as intersections, unions, property value restrictions, and others). Python offers some functions and modules for managing ontology to implement, create, and modify ontologies. The get_ontology() function allows building an empty ontology from its IRI using the Owlready2 module. Owlready2 uses the syntax "with ontology: ..." to indicate the ontology that will receive the new RDF triples. For creating an ontology, the following code is used:

```
from owlready2 import *
ontology = get_ontology()
with ontology: <Python code>
```

Concerning the implementation of the domain knowledge model and the construction of its components: the learners, knowledge units, knowledge slots, task units, material units, rule units, and educational units. The following code deals with the design of the core classes of the presented model.

```
from owlready2 import *
ontology = get_ontology("http://test.org/dkm_ontology.owl#")
# Construction of Domain Knowledge Model Components
with ontology:
    class Learner(Thing): pass
    class KnowledgeUnit(Thing): pass
    class KnowledgeSlot(Thing): pass
    class TaskUnit(Thing): pass
    class RuleUnit(Thing): pass
    class MaterialUnit(Thing): pass
    class EducationalUnit(Thing): pass
```

For adding some restrictions to the designed model using the Owlready2 restriction function, such as is_a, and equivalent_to. The remaining code deals with the implementation of this function.

```
#Some of the Restrictions Added to the Domain Knowledge Model
    class KnowledgeUnit(Thing):
        is_a = [hasParent.some(KnowledgeUnit)]
        equivalent_to = [KnowledgeUnit & hasPart.some(KnowledgeUnit)
                    & hasCompetencyDependency.some(KnowledgeUnit)
                    & hasTask.some(TaskUnit)
                    & hasSlot.some(KnowledgeSlot)
                    & hasURule.some(RuleUnit)
                    & hasMaterial.some(MaterialUnit)
                    & hasEduLevel.some(EducationalUnit) ]
```

The example below corresponds with some of the object property relationships defined for the constructed components of the selected model.

```
#Object Property Relationships added to the Knowledge Domain Model
class hasPart(KnowledgeUnit >> KnowledgeUnit):  pass
class hasCompetencyDependency(KnowledgeUnit >> KnowledgeUnit): pass
class hasParent(KnowledgeUnit >> KnowledgeUnit): pass
class hasSlot(KnowledgeUnit >> KnowledgeSlot): pass
class hasTask(KnowledgeUnit >> TaskUnit): pass
class hasMaterial(KnowledgeUnit >> MaterialUnit): pass
class hasURule(KnowledgeUnit >> RuleUnit): pass
class hasEduLevel(KnowledgeUnit >> EducationalUnit): pass
```

The following example deals with some of the data property relationships defined for the constructed components of the determined model.

```
# Knowledge Unit Data Property Relationships
   class knowledgeID(KnowledgeUnit >> int, FunctionalProperty): pass
   class knowledgeName(KnowledgeUnit >> str, FunctionalProperty): pass
   class parentKnowledge(KnowledgeUnit >> str, FunctionalProperty): pass
   class knowledgeDescription(KnowledgeUnit >> str, FunctionalProperty): pass
```

The following code deals with the issue of when there is a need for inserting instances to the domain knowledge model components.

```
#Creating instances for the knowledge unit component
Control_Structures = KnowledgeUnit('Control_Structures', knowledgeID = 1, knowledgeName = 'Control Structures',
parentKnowledge = 'C++ Programming', knowledgeDescription = '''Control Structures is a statement used to control the flow
execution of the program.''')
```

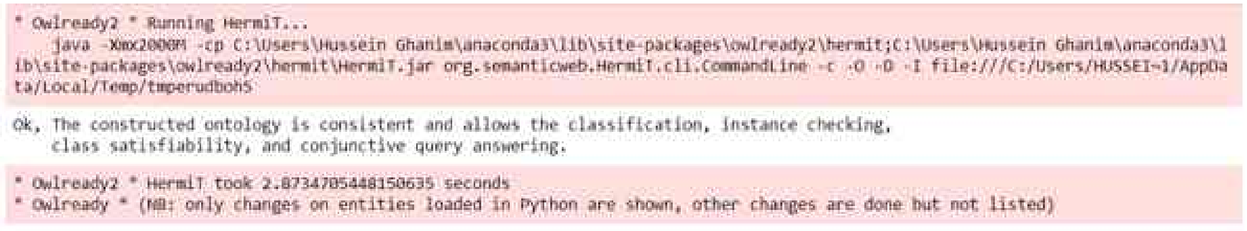
The following code deals with adding different relationships to the created domain knowledge model of the selected components.

```
#Adding slot relationships to the knowledge unit
For_Loop.hasSlot = [syntax]
#Adding Competency Dependency relationships to the knowledge unit
For_Loop.hasCompetencyDependency = [Relational_Operator, Logical_Operator]
#Adding Containment relationships to the knowledge unit
For_Loop.hasPart = [Condition, Body, Control_Variable, Initialization, Update]
```

Owlready2 offers many reasoners for manipulating the domain ontology, such as Pellet, ELK, and HermiT. To check the consistency of the developed domain ontology for the selected model, the HermiT reasoner are used, it is the most commonly used in ontology engineering. Reasoning is a process of inferring implicit facts from collected explicit facts. These facts can be expressed in OWL 2 ontologies and stored in RDF triple stores. However, OWL 2 ontology addressed reasoning capabilities: consistency, classification, instance checking, class satisfiability, and conjunctive query answering [114]. Ontology Consistency is a process of ensuring that an ontology is free of contradictions. In other terms, OWL reasoners are employed for checking the consistency of the constructed domain ontology and infer new facts in the domain ontology, typically reclassing Classes into new Superclasses and Individuals into new Classes, depending on their relationships [115]. For example, examine the following two facts: 1) all knowledge units can have parts, and 2) control variables are knowledge units that cannot have a task [114]. Using this approach would cause an inconsistency because if a control variable is a knowledge unit, logically, it can have a task, but another fact is stating otherwise. Class Satisfiability is the procedure of checking whether the class can have instances without causing inconsistency [114]. For example, if the learners can either be good or bad, then the class GoodAndBadStudent would be unsatisfiable (i.e., it cannot have instances for the ontology to be consistent). Typically, to model classes with at least one instance; therefore, having unsatisfiable classes usually suggests a modeling error. Classification is a process of determining the subclass relationships between classes in ontology to complete the class hierarchy [114]. Instance Checking is the process of verifying whether the individual is an instance of a concept [114]. For example, given the facts “a Learner is a Human” and “Ghanim is a Learner,” and if issue were to check if the fact “Ghanim is a Human” holds, the answer is true. Conjunctive Query Answering: this is the task of answering a (SPARQL) query regarding an ontology [114]. For example, if the query is “return all individuals who take a selected Knowledge unit” and the facts are “a Learner is a Human,” “Ghanim is a Learner,” and “Ghanim takes a selected Knowledge unit,” the answer to the query should be “Ghanim.”

```
try:
    sync_reasoner()
    print("Ok, the constructed ontology is consistent and allows the classification, instance checking, class
satisfiability, and  conjunctive query answering.")
except OwlReadyInconsistentOntologyError:
    print("The constructed ontology is inconsistent! and didn't allow the classification, instance checking, class
satisfiability, and conjunctive query answering. ")
```

Figure 6.3 displays the result when implementing the above code by using HermiT reasoner for checking the consistency of the knowledge domain model ontology.



```
* Owlready2 * Running HermiT...
    java -Xmx2000M -cp C:\Users\Hussein Ghanim\anaconda3\lib\site-packages\owlready2\hermit;C:\Users\Hussein Ghanim\anaconda3\l
ib\site-packages\owlready2\hermit\HermiT.jar org.semanticweb.HermiT.cli.CommandLine -c -O -D -I file:///C:/Users/HUSSEI~1/AppDa
ta/Local/Temp/tmperudboh5

Ok, The constructed ontology is consistent and allows the classification, instance checking,
    class satisfiability, and conjunctive query answering.

* Owlready2 * HermiT took 2.8734705448150635 seconds
* Owlready * (NB: only changes on entities loaded in Python are shown, other changes are done but not listed)
```

Figure 6.3 the consistency of the knowledge domain model ontology

There is an increasing demand for using ontologies in the education field and obtaining knowledge from ontologies by reasoning. Developing reasoning frameworks for Semantic Web applications require dealing with both rules and ontologies to infer new knowledge from the selected ontology, which is currently supported in managing the knowledge base in well-defined forms [116]. The author have considered several rule-based formalisms to work on top of or with ontology bases using SWRL forms. The following code deals with writing rules for inferring new knowledge, which can add to the created domain knowledge model of the selected ontology components.

```
with ontology:
    imp = Imp()
    imp.set_as_rule("""Learner(?x), KnowledgeUnit(?k), takes(?x, ?k), hasPart(?k, ?k), hasSlot(?k, ?s),
hasCompetencyDependency(?k, ?k), hasTask(?k, ?t), taskScore(?t, 2) -> hasEduLevel(?k, beginner_level)""")
```

The rule above means that the leaner x how takes the knowledge unit k. This knowledge unit which contains another knowledge unit, and it has slot, competency dependency with another knowledge unit and it has task with score 2 his knowledge level is beginner level.

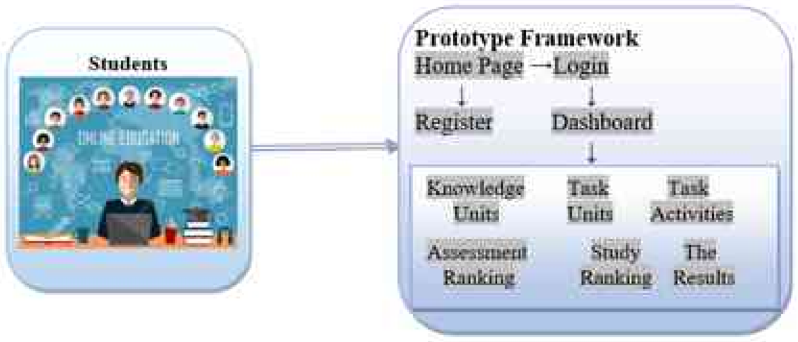
6.4.2 Application Framework Using Python Flask Framework

Flask is a higher-level Python Web development framework that enables the rapid development of secure and maintainable web applications [113].

In the interface design, the designer should focus on expecting the learners, which might require accomplishing and confirming that the interface design has components that are easy to use, easy to access, and easy to understand in order to facilitate using the actions. In addition, interface design combines concepts from different fields, such as interaction design, visual design, and information architecture.

Task Session in the Prototype Framework

The students can use the prototype framework to practice their knowledge and its related task activities. The prototype system personalizes the knowledge units, task units, and related questions according to the learner's knowledge level. First, the students' need to login if he/she already has an account, or he/she needs to register on the system. After the registration student is required to login, when he/she logged in to the framework, the dashboard will be displayed, and then the student can view the knowledge units or the task units or directly go to the task activities. Figures 6.4, 6.5, 6.6, and 6.7 indicate the interaction of the student with the Prototype Framework, home page, login, and register interfaces for the student to interact with the prototype system.



6.4 The interaction of the student with the prototype framework

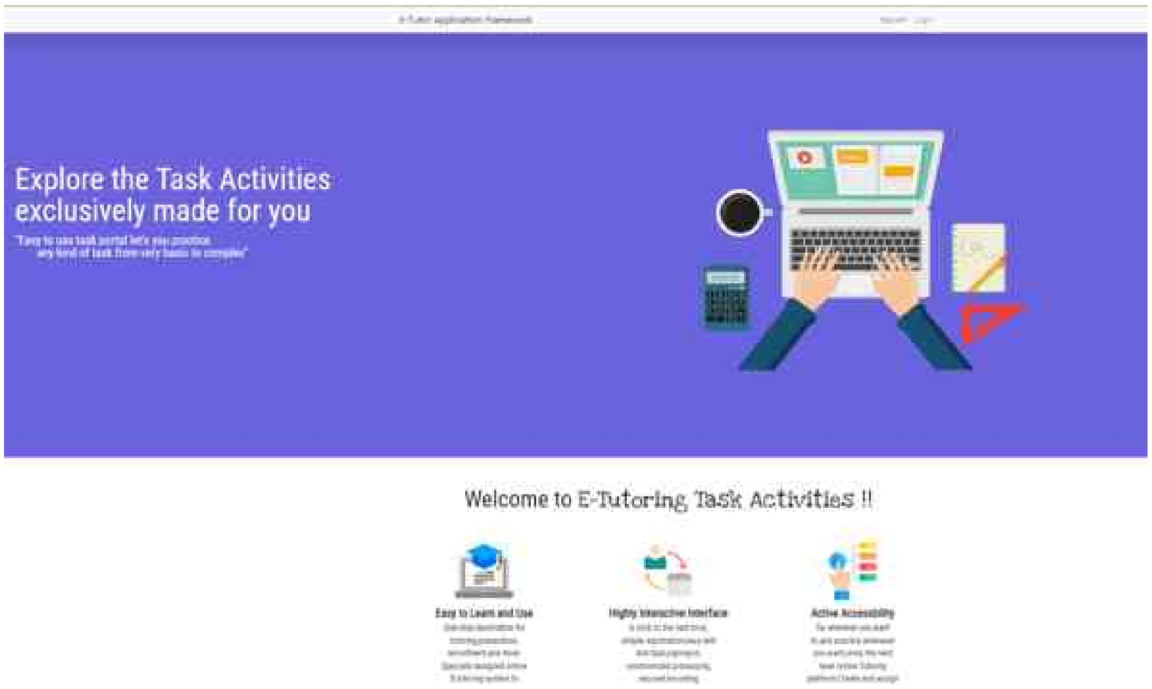
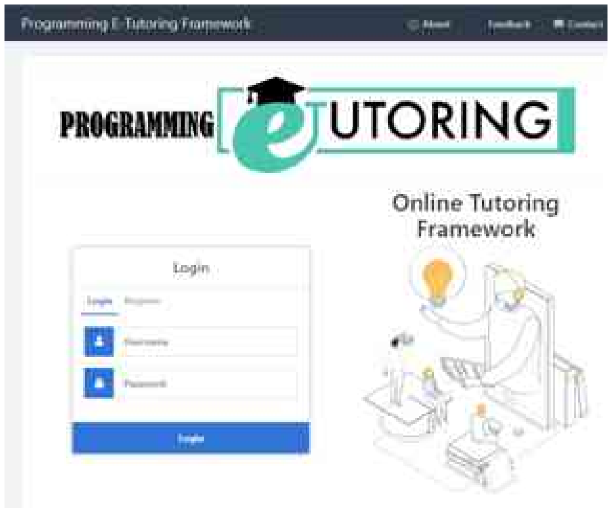

Explore the Task Activities exclusively made for you
"Easy to use task portal let's you practice any kind of task from very basic to complex"
Welcome to E-Tutoring Task Activities !!
Easy to Learn and Use
Highly Interactive Interface
Active Accessibility

Figure 6.5 The main E-tutoring framework interface



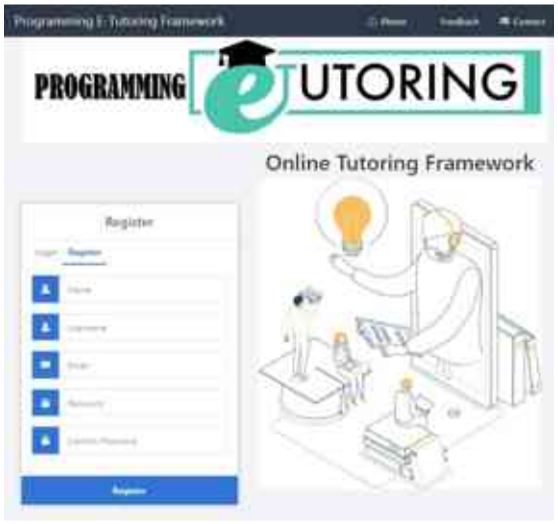


Figure 6.6 The E-tutoring framework login interface Figure 6.7 The E-tutoring framework registration interface

The dashboard interface allows the student to take a task activity, show the knowledge units or the task units or the study aid ranking or the assessment ranking or their results, which is shown in Figure 6.8.

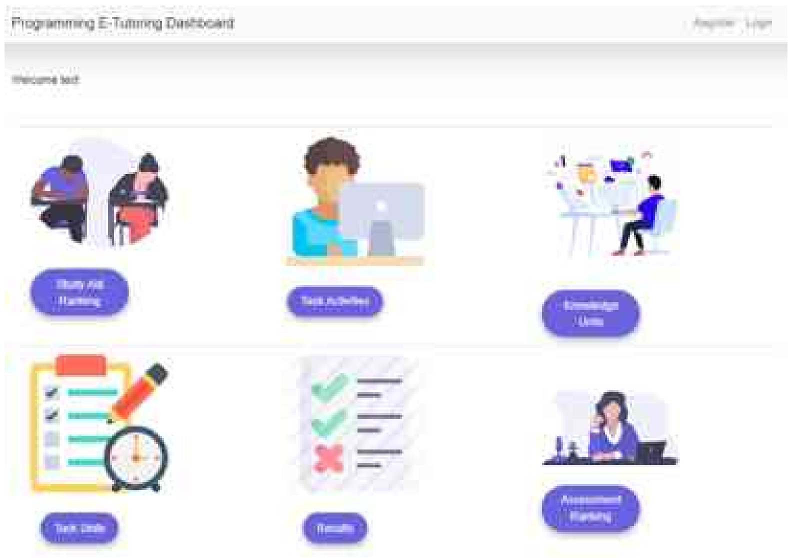


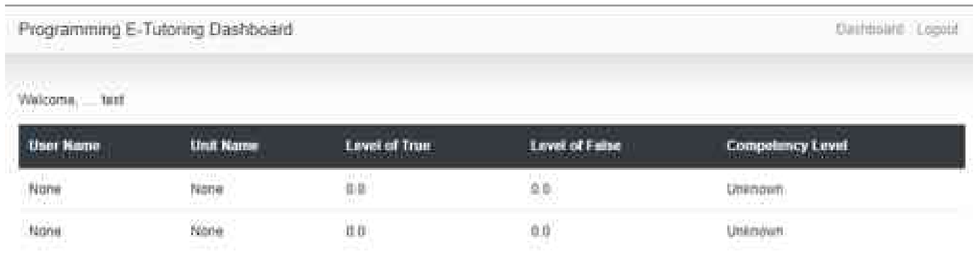
Figure 6.8 The E-tutoring framework dashboard interface

6.5 Testing and evaluation

Testing is a process of checking the design and functionality of the prototype system or components that meet the requirements and specifications via applying testing techniques in the software engineering domain [117]. The developer or the learners can evaluate the outcomes to assess the improvement of the design, performance, supportability, etc. The development of testing is an engineering mechanism used to reduce the risks during the software design process. Operational testing is the actual or simulated application, by specific users, of a system under practical, functional conditions [117].

A functional test was done to evaluate the student knowledge level in the prototype system, which is integrated with the ontology domain knowledge model. Several task assessments with MCQ were performed. According to that, the result gives feedback according to the student's answer showing if the answer is correct or incorrect. The system will update the student status according to his/her task assessment solution and the offer suggestions. The suggestion is related to task questions, student correct and wrong answers, the student result, knowledge level progress, and suggested reading materials.

Initially, the level of progress is set as unknown, meaning the level of true and false is equal to zero, as shown in Figure 6.9.

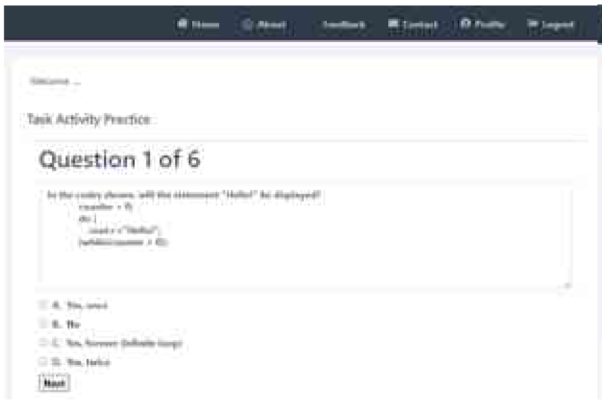


Welcome, ... test

User Name	Unit Name	Level of True	Level of False	Competency Level
None	None	0.0	0.0	Unknown
None	None	0.0	0.0	Unknown

Figure 6.9 Snapshot of the initial progress

Figures 6.10 and 6.11 display a test evaluation of the task assessments for different practices that allow the student to answer the task questions related to the knowledge unit, and then the student chooses the correct answer and moves to the next task question. After that, the prototype system checks whether the answer is correct or not and provides feedback according to the student's answer.



Welcome ...

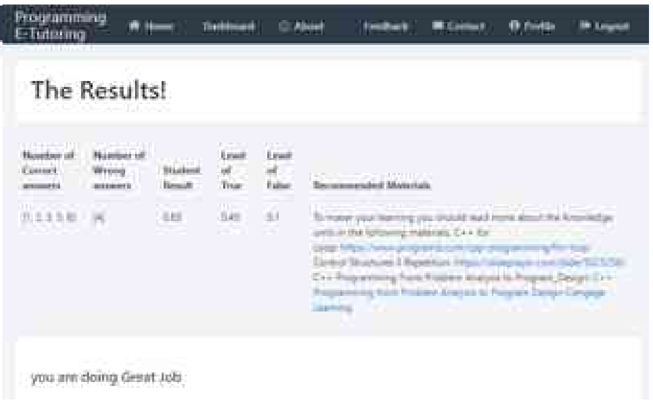
Task Activity Practice

Question 1 of 6

In the codey shown, will the statement "Hello!" be displayed?

```
        counter = 0;
        do {
            cout << "Hello!";
        }while(counter > 0);
```

- A. Yes, once
- B. No
- C. Yes, forever (infinite loop)
- D. Yes, twice



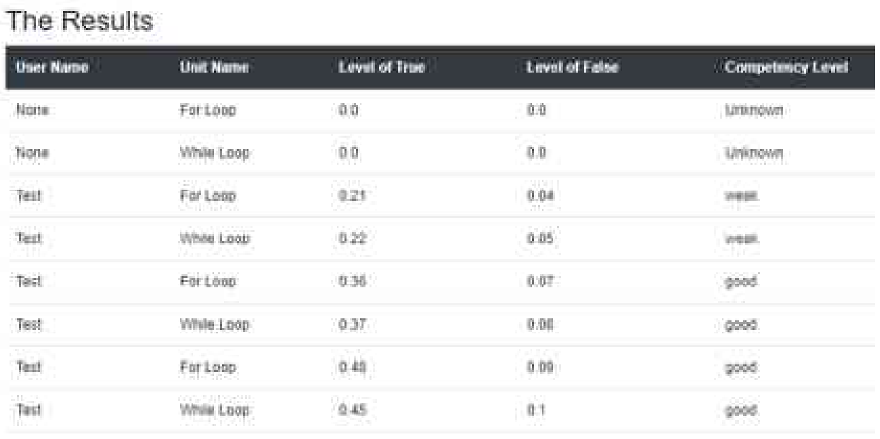
The Results!

Number of Correct answers	Number of Wrong answers	Student Result	Level of True	Level of False	Recommended Materials
[1, 2, 3, 5, 6]	[4]	0.83	0.45	0.1	To master your learning you should read more about the Knowledge units in the following materials: C++ for Loop: https://www.programiz.com/cpp-programming/for-loop Control Structures 2 Repetition: https://slideplayer.com/slide/[illegible] C++ Programming From Problem Analysis to Program Design: C++ Programming From Problem Analysis to Program Design Cengage Learning

you are doing Great Job

Figure 6.10 Snippets of practice task assessment Figure 6.11 Snippets of the task assessment result

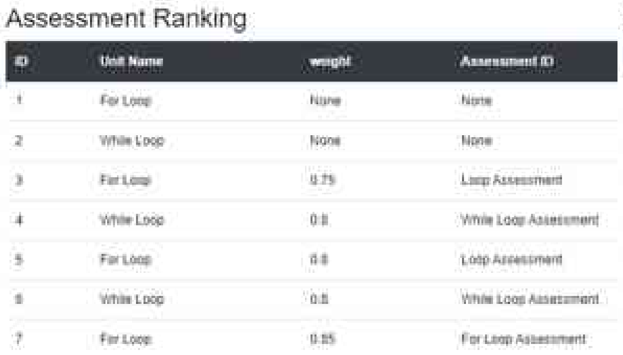
Figures 6.12, 6.13, 6.14 and 6.15 shows the generated result in detail for the task assessments, the selected task question related to the knowledge unit, and the learner's answer, and suggests related material if the learner wants to learn more about the chosen knowledge unit. Figure 6.11 shows the student's name, the tested knowledge unit, the level progress (level of true, level of false), and the competency level which shows the status of the student in each tested knowledge unit (weak, average, good). This gives an indicator of where the student needs to enhance his knowledge level. Figure 6.12 displays the assessment ranking for each knowledge unit and their weight (the task question related to the knowledge unit) and the assessment ID. While figure 6.13 denotes the study aid ranking for the knowledge unit which suggests the learning material for the tested knowledge unit it means that the prototype system can be used as self-learning.



The Results

User Name	Unit Name	Level of True	Level of False	Competency Level
None	For Loop	0.0	0.0	Unknown
None	While Loop	0.0	0.0	Unknown
Test	For Loop	0.21	0.04	weak
Test	While Loop	0.22	0.05	weak
Test	For Loop	0.36	0.07	good
Test	While Loop	0.37	0.08	good
Test	For Loop	0.40	0.09	good
Test	While Loop	0.45	0.1	good

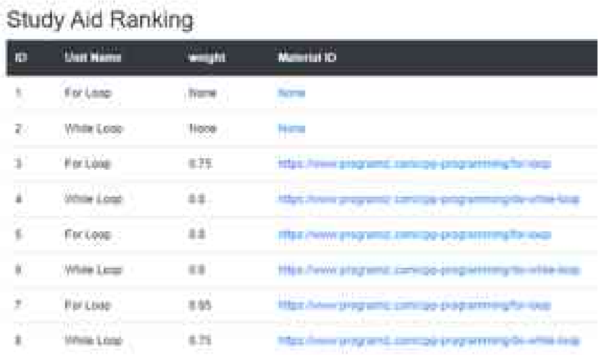
Figure 6.12 Snippets of the result interface



Assessment Ranking

ID	Unit Name	weight	Assessment ID
1	For Loop	None	None
2	While Loop	None	None
3	For Loop	0.75	Loop Assessment
4	While Loop	0.8	While Loop Assessment
5	For Loop	0.8	Loop Assessment
6	While Loop	0.8	While Loop Assessment
7	For Loop	0.85	For Loop Assessment

Figure 6.13 Assessment ranking



Study Aid Ranking

ID	Unit Name	weight	Material ID
1	For Loop	None	None
2	While Loop	None	None
3	For Loop	0.75	https://www.programiz.com/cpp-programming/for-loop
4	While Loop	0.8	https://www.programiz.com/cpp-programming/do-while-loop
5	For Loop	0.8	https://www.programiz.com/cpp-programming/for-loop
6	While Loop	0.8	https://www.programiz.com/cpp-programming/do-while-loop
7	For Loop	0.95	https://www.programiz.com/cpp-programming/for-loop
8	While Loop	0.75	https://www.programiz.com/cpp-programming/do-while-loop

Figure 6.14 Study aid ranking

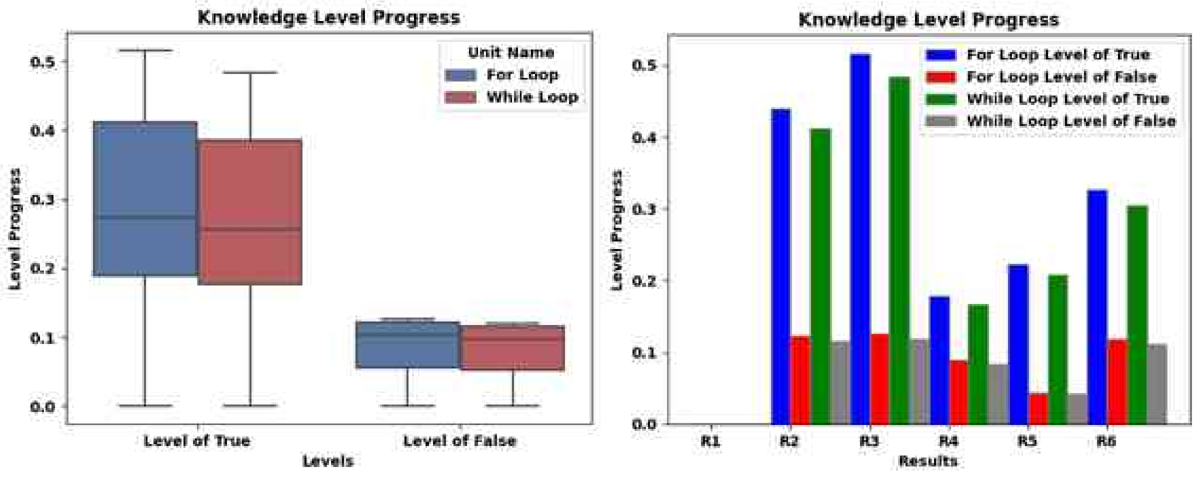


Figure 6.15 knowledge level progress

6.6 Summary

In this chapter, a prototype framework is developed and added functionality by implementing and integrating the knowledge domain model mentioned in Chapters 4 and 5 for an E-tutoring framework. Regarding the domain knowledge model implementation for an E-tutoring system and the ontology as a back end. The prototype system and the ontology model are implemented using the Python Flask framework and OWLReady2. OWLReady2 is used for constructing the ontology domain knowledge model, and the Flask framework for the interface design is connected with the ontology model.

Several task assessments were performed to evaluate the prototype system integrated with the ontology domain knowledge module and the functional test. According to that, the result gives correct feedback according to the student's answer showing if the answer is correct or incorrect and offering suggestions. The suggestion related to unit name, task name, task question, learner answer, the result, suggested reading material, and the task level score.

As a result, the developed ontology domain knowledge model integrated with the prototype system can be used in managing adaptive intelligent E-learning frameworks in the future. Furthermore, the domain ontology knowledge model can meet various pedagogical goals. The goals include understanding specific domain facts and solving problems, obtaining a conceptual and intuitive understanding of the material in the selected domain, and learning problem-solving and metacognitive skills. A significant feature of the selected model is that the knowledge representation techniques have a standard structure. This standard structure provides the proposed model as representational inference tools and control mechanisms and facilitates a pedagogical analysis of a knowledge domain. In addition, combining the proposed ontology domain knowledge model with an E-tutoring system can enhance the quality of intelligent problem-solving. Also, it will be possible to reuse the knowledge domains. Finally, a proposal of the domain knowledge model for the E-tutoring system can enhance the teaching and learning process, support recommendations, generate hints, and automatically support the generation of problems and solutions.

Thesis 4 ***[3] [5]***

I have developed a prototype software system based on the proposed E-tutoring framework integrated with the ontology domain knowledge model. The framework implements the presented decision support algorithms for personalized E-tutoring. The prototype E-tutoring system can enhance the quality and functionality of intelligent e-learning systems. The performed experiments and tests with the prototype framework show that flexible functionality and personalization of the learning materials, automatically generating new task assessments, and providing hints and recommendations to the student, can significantly improve the efficiency of the e-learning systems and can enhance the functionalities of teaching and learning processes.

General Summary

In this research, a survey of the current investigation is presented on E-learning domains and the historical evolution of education technologies from the 1990s to now, existing Intelligent Tutoring Systems (ITS) or E-tutoring Systems models, Ontology model in an E-learning environment, domain knowledge model, knowledge representation schemes and related problems.

The problem of the most current ITS systems are found that they focuses on solving a single domain and creating only a single domain knowledge like introduction to computer and programming tutor and Java object tutor. Most solutions use an isolated knowledge base storage model, and these local knowledge bases can provide only limited knowledge background. The limitations of an isolated knowledge base are lack of standardability, lack of shareability, lack of flexibility, lack of reusability, and a limited knowledge base. Moreover, it also found that knowledge representation schemes, especially ontologies, can offer solutions to current problems with a well-designed form.

An ontology as a knowledge sharing and interoperability will significantly increase the reusability of the domain knowledge model. Ontology has a reasoning engine and supports logic formulations. Therefore, this will make it possible to design both reusable functional components for the E-tutoring systems. Using ontology to represent the domain knowledge models can be integrated into an E-tutoring system to improve user modeling and intelligent problem-solving support.

The investigation found that intelligent E-Learning, especially E-tutoring systems, is a key technology for supporting educational activities and processes. Many E-tutoring architectures, models, and components that significantly impact the evolution of E-tutoring frameworks are found in the literature; in this work, some of them are presented. Concerning the E-tutoring architecture, the first E-tutoring systems architecture was based on a traditional model, including three elementary components: domain, learner, and tutoring module. one of the researchers developed the earlier architecture model by adding a learner interface module.

In this research, I contributed to the development of the extended ITS architecture and based on this experience I later developed the adaptive ITS architecture based on the current research directions and trends presented in the previous work. Besides the standard models, the proposed architecture includes a common shared database and knowledge-based background. The advantages of the shared database are to share a common understanding of the knowledge structure, reuse the knowledge, and mix different knowledge bases.

In this research, I have developed a model for the purpose of enhancing the learning and teaching processes in 3 ways. These models are the ontology-based model to support the learning process in LMS, the ontology-based Knowledge domain model for the IT domain in E-tutoring systems, and the ontology-supported domain knowledge module for an E-tutoring system. The main goals of these models is to enhance the learning and teaching process, support recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of new material.

In the first model, depending on the properties of the learning materials, two types of ontologies are implemented as a form of general concepts: domain knowledge ontology and specific domain knowledge ontology. These modules represent the knowledge to be learned, deliver input to the expert model, and eventually provide detailed feedback, select problems, generate guidance, and support the student model. The introduced domain knowledge model is constructed based on the current research directions. The presented model suggests the topics, concepts, attributes, tasks, competencies, assessments, and relations. To facilitate the sharing and reusing of the domain knowledge model features in E-tutoring systems, ontologies are utilized to organize and represent the domain knowledge model. The benefit of this model is to personalize the learning materials for learners.

This research developed and presented a novel domain knowledge model for teaching materials using ontology as a knowledge representation technique. In addition, the presented model allows the knowledge construction for tailoring the explanation of tutoring material through the information gathered in the learner model. The primary purpose is to construct a general and flexible knowledge model. However, flexibility means allowing for extensions and modifications of the core model. In other words, flexibility also refers to using the knowledge model in different domains and allowing a dynamic and automated framework.

There are many types of knowledge forms to be considered in this work focusing on procedural knowledge and declarative knowledge. Declarative knowledge deals with the selected model's facts, methods, and practices. Procedural knowledge deals with the problem-solving process. Combining the declarative and procedural knowledge to add functionality to the proposed model and provide access to existing domain-specific glossaries, taxonomies, and ontologies is preferred to build the domain knowledge model. A well-designed structure for the knowledge domain requires adopting a proper methodology, especially in ontology representation, because it provides a suitable form for learning material representation. The presented model can be used effectively for the intelligent processing of teaching material and form a base for interaction and collaboration supporting E-tutoring framework components. Using an ontology can reuse the knowledge domain stored on it. The suggested model can support solving the limitations of existing models.

The formal description of the proposed model is based on a fundamental concept element, denoted as a knowledge unit covering the specific teaching and competency dependency units. Many types of relationships are introduced between the knowledge units in the domain ontology. The core components of the suggested model are knowledge unit, knowledge slot, task unit, material unit, rule unit, and educational unit. Loops in the C++ programming language are selected as knowledge units for testing the suggested model. The proposed model is introduced to explain how the ontology domain knowledge model can be combined with an E-tutoring system to improve the quality of intelligent problem-solving. Also, it will make it possible to reuse knowledge components, and it can serve to enhance the teaching and learning process. The problem of isolated knowledge bases can be avoided using ontology as a knowledge representation technique for building the domain knowledge model. The developed ontology can be involved in managing adaptive intelligent e-learning frameworks in the future. The domain ontology knowledge model considered in chapter 3 has various pedagogical goals. These goals include understanding specific domain facts and solving standard problems, obtaining a conceptual and intuitive understanding of the material in the selected domain, and learning general problem solving and metacognitive skills. A significant feature of the selected model is that the knowledge representation techniques have a standard structure. This standard structure allows general representational inference tools and control mechanisms, facilitating the pedagogical analysis of knowledge.

The researcher used some technologies that deal with adding the functionality by implementing and integrating the knowledge domain model mentioned in chapter 4. with an E-tutoring framework. Regarding the domain knowledge model implementation for an E-tutoring system and the ontology as a back end, Python programming language, Python Flask web framework module, and the Owlready2 module are used. Python offers different libraries for the Owlready2 module for querying the knowledge stored in the domain ontology, such as RDFLib, SPARQL-Client, and SPARQLWrapper. In querying the model, RDFLib and SPARQLWrapper are used. The Flask framework is used to develop the prototype system to integrate the constructed ontology for the selected domain knowledge model with the E-tutoring framework.

For evaluating the prototype system which is integrated with the ontology domain knowledge module functional test was done several task assessments were performed. According to that the result gives correct feedback according to the user answer showing if the answer is correct or incorrect and offering suggestions. The suggestion related to unit name, task name, task question, learner answer, the result, suggest reading material and the task score.

As a result, the developed ontology domain knowledge model integrated with the prototype system can be used in managing adaptive intelligent E-learning frameworks in the future. Furthermore, the domain ontology knowledge model can meet various pedagogical goals. These goals include understanding specific domain facts and solving standard problems, obtaining a conceptual and intuitive understanding of the material in the selected domain, and learning general problem solving and metacognitive skills. A significant feature of the selected model is that the knowledge representation techniques have a standard structure. However, the standard form of the proposed model can allow for general representational inference tools, control mechanisms, and facilitating pedagogical analysis of knowledge. In addition, combining the proposed ontology domain knowledge model with an E-tutoring system can enhance the quality of intelligent problem-solving. Also, it will be possible to reuse the knowledge domains. Finally, a proposal of the domain knowledge model for the E-tutoring system can enhance the teaching and learning process, support recommendations, generate hints, and automatically support the generation of problems and solutions.

For managing the uncertainty of students' knowledge and identifying their status in learning the domain knowledge model in an E-tutoring framework, we used Intuitionistic fuzzy logic values (IFVs) to represent the student knowledge model from the domain knowledge model concepts. The main goal is to evaluate learner knowledge based on an accumulative task assessment using question-and-answer pairs and updating a student model. IFVs is used for calculating student knowledge level by means of which the system can automatically recommend further reading materials/tasks.

Chapter 7 Conclusion

7.1 Contribution

In this dissertation, a contribution to the domain knowledge model representation is presented in an E-tutoring system. The proposed model uses ontology as a knowledge representation technique for the domain knowledge model. First, an adaptive E-tutoring system architecture is developed based on the current research directions and trends. Beyond the standard models, the proposed architecture includes a shared database and knowledge-based background. The advantages of the shared database are to share a common understanding of the knowledge structure, reuse the knowledge, and mix different knowledge bases. Second, a model for the purpose of enhancing the learning and teaching processes are developed in 3 ways. These models are the ontology-based model to support the learning process in LMS, the ontology-based Knowledge domain model for the IT domain in E-tutoring systems, and the ontology-supported domain knowledge module for an E-tutoring system. Third, according to these models, a formal and general ontological model for the domain knowledge model is developed, which presents the conceptual components of teaching materials frameworks. The benefit of this model is providing personalized learning and teaching, automatically supporting the generation of new material units, making suggestions, generating hints, and supporting automatic task assessments for students. Fourth, I have developed an assessment module for evaluating student knowledge levels in E-tutoring Systems using the intuitionistic fuzzy logic technique. Finally, a prototype E-tutoring framework is created and integrated with the ontology domain knowledge model. The goal is to enhance the learning and teaching process. The following four thesis can summarize the novel scientific results.

Thesis 1 [2] [3] [4] [8]

I have developed an adaptive E-tutoring system architecture. Beyond the standard components, this architecture also includes a common shared database and knowledge-based background. The main benefits of this architecture are that it supports shared databases providing more features. The features are efficient knowledge management, flexibility, better performance, scalability, increased accessibility and availability, better security, and automatic recovery. They also provide the required technical and methodological background for developing smart tutoring systems. Besides this adaptive architecture, I have developed and tested three model versions of the domain knowledge module based on ontology methods in order to create a general and flexible model and enhance the learning processes. The proposed model can improve the semantic support of the related decision-making processes. The main role is to support the automatic control of teaching processes and generate the assessment material according to the student's capabilities. The proposed model is presented as a model structure containing two types of ontology models, general concepts of domain knowledge ontology and specific domain knowledge ontology. The key benefits are to enhance the learning and teaching process, provide support for recommendations, generate hints, automatically support the generation of problems and solutions, and automatically support the generation of teaching materials.

Thesis 2 [4]

I have introduced a novel formal logical ontology domain knowledge model for an E-tutoring system, which presents the conceptual components of teaching material frameworks. The core components of this model are knowledge units and related units covering the specific teaching and competency dependency units, including the specialization, containment, association, and dependency graph relationships among the knowledge units in the ontology domain. For storing this model, I compared three model storage systems the OWL ontology file, Jena Fuseki, and the relational model. The key aspect is determining the suitable storage model to store the ontology model in a standard, efficient, scalable, and consistent format. The main role of this comparison is to improve application performance, enable semantic knowledge representation, and achieve efficient information retrieval. The investigation shows that Jena Fuseki and the relational model are the best ontology storage tools. They provide support facilities for storing, inferring new knowledge, and managing the ontology domain knowledge. But the relational model storage needs more space and does not provide automatic reasoning. So, Jena Fuseki is the best candidate for model storage and query.

Thesis 3 [5]

I have presented a knowledge model to manage the students' competency levels and to support automatic decision-making in the smart tutoring framework. The proposed knowledge model supports the representation of uncertainty in students' knowledge and can be used to identify their status in the learning processes. In the assessment module, I used intuitionistic fuzzy logic values to represent the uncertainty in student knowledge status. I have developed an algorithm for the evaluation of the student's performance using an accumulative task assessment approach with multiple-choice questions. The model automatically updates the student's knowledge level in an adaptive way. The proposed model provides interactive and adaptive assistance techniques by personalizing the teaching and learning materials according to the current knowledge levels.

Thesis 4 [3] [5]

I have developed a prototype software system based on the proposed E-tutoring framework integrated with the ontology domain knowledge model. The framework implements the presented decision support algorithms for personalized E-tutoring. The prototype E-tutoring system can enhance the quality and functionality of intelligent e-learning systems. The performed experiments and tests with the prototype framework show that flexible functionality and personalization of the learning materials, automatically generating new task assessments, and providing hints and recommendations to the student, can significantly improve the efficiency of the e-learning systems and can enhance the functionalities of teaching and learning processes.

7.2 Future work

I developed an ontology domain knowledge for teaching and learning models, integrated with a prototype E-tutoring framework that uses task assessment as multiple-choice questions. This framework gives a provision for evaluating the student's knowledge level. It can be improved in the following areas.

- The future work direction is to automatically construct the ontology domain knowledge model from the teaching and learning materials. This approach can be used to control adaptive intelligent E-tutoring frameworks.
- The future direction also enables the suggested module to be improved by adding some functionalities and automatically supporting the generation of problems and their solutions. Furthermore, future work recommends dealing with problem-solving processes that work on procedural knowledge to help students understand how to use the domain knowledge model in solving a practical problem.
- Currently, the prototype system task assessment works for MCQ questions. In the future, the system can be modified to improve the work for subjective task assessment questions. This modification requires artificial intelligence or machine learning algorithms for evaluation schemes.

Author’s Publications

Publications Related to the Dissertation

Journal Articles in Q Ranking

[1]. **Ghanim Hussein Ali Ahmed and László Kovács**, Development of Ontology-based Model to Support Learning Process in LMS, INDONESIAN JOURNAL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE 24(1), pp. 507-518, (2021) **Scopus (Q3)**, **Impact Factor (1.51),** Journal Article.

[2]. **Ghanim Hussein Ali Ahmed, Jawad Alshboul and László Kovács**, Development of Ontology-based Domain Knowledge Model for IT Domain in E-tutor Systems, INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS (2158-107X 2156-5570): 13(5), pp. 28-34, (2022), **Web of Science (WoS), Scopus (Q3)**, **Impact Factor(1.16),** Journal Article.

[3]. **Ghanim Hussein Ali Ahmed and László Kovács**, Ontology Supported Domain Knowledge Module For E-Tutoring System (accepted with minor revision), ACTA CYBERNETICA, **Scopus (Q3)**, **Impact Factor(0.64),** Journal Article.

[4]. **Ghanim Hussein Ali Ahmed and László Kovács**, Model Storage Systems for Integrating the Ontology Domain Knowledge Module in E-Tutoring Systems (submitted, in review process), INDONESIAN JOURNAL OF ELECTRICAL ENGINEERING AND INFORMATICS (IJEEI), **Scopus (Q3)**, **Impact Factor(1.5)** Journal Article.

[5]. **Ghanim Hussein Ali Ahmed and László Kovács**, Assessment Module for Evaluating Student Knowledge level in E-tutoring Systems Using Intuitionistic Fuzzy Sets Technique (under progress), expected to be published in **Scopus (Q2) journal.**

Other Publications Journal Articles in non-Q Ranking

Local Journals

[6]. **Walelign Tewabe Sewunetie, Ghanim Hussein Ali Ahmed and László Kovács** The Development and Analysis of Extended Architecture Model for Intelligent Tutoring Systems, GRADUS 6(4), (2019), pp. 128-138., Journal Article.

[7]. **Ghanim Hussein Ali Ahmed and László Kovács**, Evaluation of Domain Ontology Requirements in E-tutor Framework, Multidiszciplináris Tudományok: A Miskolci Egyetem Közleménye 9 : pp.508-516. , (2019), Journal Article.

[8]. **Ghanim Hussein Ali Ahmed and László Kovács**, Ontology Domain Model for E-tutoring System, JOURNAL OF SOFTWARE ENGINEERING & INTELLIGENT SYSTEMS 5(1) pp. 37-44. (2020), **International Journal,** Journal Article.

International Journal

[9]. Ghanim Hussein Ali Ahmed, László Kovács and Gfary Hassan Hajhamed. Evaluation Model for the Usability of Web-Based Learning Management Systems with user profile, JOURNAL OF SOFTWARE ENGINEERING & INTELLIGENT SYSTEMS 6(1) pp. 1-10. (2021), Journal Article.

[10]. Jawad Alshboul1, Ghanim Hussein Ali Ahmed and Erika Baksa-Varga, Semantic Modeling for Learning Materials in E-tutor Systems, JOURNAL OF SOFTWARE ENGINEERING & INTELLIGENT SYSTEMS 6(2) pp. 1-5. (2021), Journal Article

International Conference Proceeding

[11]. László Kovács , László Csépányi-Fürjes and Ghanim Hussein Ali Ahmed, Automated Assessment Generation in Intelligent Tutoring Systems, The 15th International Conference Interdisciplinarity in Engineering, **Springer** International Publishing (2022), pp. 808-819. **Scopus**, Conference paper.

[12]. Ontology Supported Domain Knowledge Module For E-Tutoring System, The 13th Conference of PhD Students in Computer Science, Szeged, Hungary, June 29 - July 1, 2022.

Book of Abstract

[13]. Ghanim Hussein Ali Ahmed and László Kovács, Ontology Model of Learning Process in Web Based LMS, Association of Hungarian PHD and DLA Students (2019), pp. 431-431, Abstract.

[14]. Ghanim Hussein Ali Ahmed and László Kovács, Development of Recommendation Systems Based on Collaborative Filtering Using KNN and SVD, Association of Hungarian PHD and DLA Students (2020), pp. 355-355, Abstract.

[15]. Jawad Alshboul1, Ghanim Hussein Ali Ahmed and Erika Baksa-Varga, Development of a Semantic Model for Learning Materials in Intelligent Tutoring Systems, International PhD & DLA Symposium 2021, Pollack Press (2021). pp. 91-91, Abstract.

Local Conference Proceeding

[16]. Ghanim Hussein Ali Ahmed and László Kovács, Ontology Support Domain Model for E-tutoring System, Doktoranduszok Fóruma , (2019) pp. 93-99. Conference paper.

[17]. Ghanim Hussein Ali Ahmed and László Kovács, Development of Knowledge Model for IT Domain on E-tutor System, Doktoranduszok Fóruma , (2020) pp. 53-58. Conference paper.

Reference

[1] N. W. Rahayu, R. Ferdiana, and S. S. Kusumawardani, “A systematic review of ontology use in E-Learning recommender system,” *Comput. Educ. Artif. Intell.*, vol. 3, p. 100047, 2022, doi: 10.1016/j.caeai.2022.100047.

[2] A. Behaz and M. Djoudi, “Adaptation of learning resources based on the MBTI theory of psychological types,” *Int. J. Comput. Sci. Issues*, vol. 9, no. 1, pp. 135–141, 2012.

[3] R. Nkambou, “Modeling the domain: An introduction to the expert module,” *Stud. Comput. Intell.*, vol. 308, pp. 15–32, 2010, doi: 10.1007/978-3-642-14363-2_2.

[4] R. Freedman, S. Ali, and S. McRoy, “Links: what is an intelligent tutoring system?,” *Intelligence*, vol. 11, no. 3, pp. 15–16, 2000, doi: 10.1145/350752.350756.

[5] D. Darai, S. Sing, and S. Diswas, “Knowledge Engineering - an overview,” *Int. J. Comput. Sci. Inf. Technol.*, vol. 1, no. 4, pp. 230–234, 2010.

[6] Sayantini, “What is Knowledge Representation in AI? Techniques You Need To Know,” *Edurekha*, 2020, [Online]. Available: https://www.edureka.co/blog/knowledge-representation-in-ai/#techniques.

[7] I. Panagiotopoulos, L. Seremeti, A. Kameas, and V. Zorkadis, “PROACT: An ontology-based model of privacy policies in ambient intelligence environments,” *Proc. - 14th Panhellenic Conf. Informatics, PCI 2010*, pp. 124–129, 2010, doi: 10.1109/PCI.2010.30.

[8] S. C. Shapiro, “Knowledge Representation and Reasoning Logics for Artificial Intelligence,” *Production*, pp. 2004–2009, 2009.

[9] R. Studer, V. R. Benjamins, and D. Fensel, “Knowledge Engineering: Principles and methods,” *Data Knowl. Eng.*, vol. 25, no. 1–2, pp. 161–197, 1998, doi: 10.1016/S0169-023X(97)00056-6.

[10] A. Caravantes and R. Galán, “Generic educational knowledge representation for adaptive and cognitive systems,” *Educ. Technol. Soc.*, vol. 14, no. 3, pp. 252–266, 2011.

[11] S. Kumar Malik, N. Prakash, and S. Rizvi, “Ontology Creation towards an Intelligent Web: Some Key Issues Revisited,” *Int. J. Eng. Technol.*, vol. 3, no. 1, pp. 44–52, 2011, doi: 10.7763/ijet.2011.v3.199.

[12] S. Somyürek, “The new trends in adaptive educational hypermedia systems,” *Int. Rev. Res. Open Distance Learn.*, vol. 16, no. 1, pp. 221–241, 2015, doi: 10.19173/irrodl.v16i1.1946.

[13] S. A. El-Seoud, H. F. El-Sofany, and O. H. Karam, “The semantic web architecture and its impact on E-learning systems development,” *Int. J. Emerg. Technol. Learn.*, vol. 10, no. 5, pp. 29–34, 2015, doi: 10.3991/ijet.v10i5.4754.

[14] A. Phillips, J. F. Pane, R. Reumann-Moore, and O. Shenbanjo, “Implementing an adaptive intelligent tutoring system as an instructional supplement,” *Educ. Technol. Res. Dev.*, vol. 68, no. 3, pp. 1409–1437, 2020, doi: 10.1007/s11423-020-09745-w.

[15] E. Llc, *e-learning Concepts, Trends, Applications*, 1st ed. San Francisco, California, CA 94104: Epignosis LLC, 2014.

[16] S. Edition, *E-learning methodologies and good practices*. 2021.

[17] M. Srivastava, H. Pandey, S. Shukla, and B. K. Thakur, “A Literature Review of E-Learning Model Based on Semantic Web Technology,” *Int. J. Sci. Eng. Res.*, vol. 5, no. 10, pp. 174–178, 2014, [Online]. Available: http://www.ijser.org.

[18] S. Lonn and S. D. Teasley, “Saving time or innovating practice: Investigating perceptions and uses of Learning Management Systems,” *Comput. Educ.*, vol. 53, no. 3, pp. 686–694, 2009, doi: 10.1016/j.compedu.2009.04.008.

[19] I. A. Almrashdeh, N. Sahari, N. A. M. Zin, and M. Alsmadi, “Distance Learning Management System Reqiurements From Student ’ S Perspective,” *J. Theor. Appl. Inf. Technol.*, pp. 17–27, 2011.

[20] D. Dicheva, “Ontologies and Semantic Web for E-Learning,” *Handb. Inf. Technol. Educ. Train.*, pp. 47–65, 2008, doi: 10.1007/978-3-540-74155-8_3.

[21] K. Maheshwari and D. Boro, “Web Based Educational Systems : An Application of Information & Communication Technology,” *Media*, pp. 230–235.

[22] S. Ninoriya, “CMS, LMS and LCMS For eLearning,” *Int. J. Comput. Sci. Issues*, vol. 8, no. 2, pp. 644–647, 2011.

[23] B. J. J. Muries, “Explaining Electronic Learning Management Systems (ELMS) Continued Usage Intentions among Facilitators in Higher Education Institutions (HEIs) in Tanzania.,” *Int. J. Educ. Dev. Using Inf. Commun. Technol.*, vol. 13, no. 1, pp. 123–141, 2017.

[24] . B. V. W., “Study of Content Management Systems Joomla and Drupal,” *Int. J. Res. Eng. Technol.*, vol. 02, no. 12, pp. 569–573, 2013, doi: 10.15623/ijret.2013.0212096.

[25] V. Kadam Babasaheb Ambedkar, V. J. Kadam, S. SDabhade, P. V Dange, and R. B. Gofankar, “How to Choose a Website Content Management System,” no. January, 2013, [Online]. Available: https://www.researchgate.net/publication/274510334.

[26] L. K. Walelign Tewabe Sewunetie, Ghanim Hussein Ali Ahmed, “The Development and Analysis of Extended Architecture Model for Intelligent Tutoring Systems,” *Gradus*, vol. 6, no. 4, pp. 128–138, 2019.

[27] Z. Bezhovski and S. Poorani, “The Evolution of E-Learning and New Trends,” *Inf. Knowl. Manag.*, vol. 6, no. 3, pp. 50–57, 2016, [Online]. Available: https://www.iiste.org/Journals/index.php/IKM/article/view/29274.

[28] A. Goold, J. Coldwell, and A. Craig, “An examination of the role of the e-tutor,” *Australas. J. Educ. Technol.*, vol. 26, no. 5, pp. 704–716, 2010, doi: 10.14742/ajet.1060.

[29] W. Ma, O. Adesope, J. C. Nesbit, and Q. Liu, “Journal of Educational Psychology Intelligent Tutoring Systems and Learning Outcomes : A Meta-Analysis,” *J. Educ. Psychol.*, vol. 106, no. 4, pp. 901–918, 2014.

[30] A. C. Graesser, X. Hu, and R. Sottilare, “Intelligent tutoring systems,” *Int. Handb. Learn. Sci.*, pp. 246–255, 2018, doi: 10.4324/9781315617572.

[31] S. Gamalel-Din, “The smart tutor: Student-centered case-based adaptive intelligent e-tutoring,” in *the Proceedings of the 1st International Conference on Informatics and Systems, Cairo*, 2002, vol. 17, p. 20.

[32] N. Gordon, *Flexible Pedagogies: technology-enhanced learning*, no. January. 2014.

[33] Y. Chen, “Improving student model for individualized learning Université Pierre et Marie Curie Improving Student Model for Individualized Learning,” Université Pierre et Marie Curie, 2016.

[34] K. Livingston, M. Schweisfurth, G. Brace, and M. Nash, “Why Pedagogy Matters: the role of pedagogy in Education 2030-a policy advice paper.” University of Glasgow. https://www. meshguides. org/news/[Accessed 28 July 2018], 2017.

[35] J. E. Wollman-Bonilla *et al.*, “What makes great pedagogy ? Nine claims from research Autumn 2012 Great pedagogy : nine claims from research,” *J. Early Child. Lit.*, vol. 8, no. 3, pp. 167–192, 2008, [Online]. Available: http://ecl.sagepub.com/cgi/doi/10.1177/1468798408096481%5Cnhttp://ecl.sagepub.com/cgi/content/abstract/1/2/167.

[36] M. Al-yahya, R. George, and A. Alfaries, “Ontologies in E-Learning : Review of the Literature,” *Int. J. Softw. Eng. Its Appl.*, vol. 9, no. 2, pp. 67–84, 2015, doi: 10.14257/ijseia.2015.9.2.07.

[37] B. J. A. Lake, R. G. Bennett, and J. F. Kotek, “Copyright 2001 Scientific American, Inc.,” *Sci. Am.*, vol. 284, no. 5, pp. 34–43, 2001, [Online]. Available: https://www.jstor.org/stable/26059207.

[38] A. A. Altowayan and L. Tao, “Simplified approach for representing part-whole relations in OWL-DL

ontologies," in *Proceedings - 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security and 2015 IEEE 12th International Conference on Embedded Software and Systems, H*, 2015, pp. 1399–1405, doi: 10.1109/HPCC-CSS-ICESS.2015.147.

[39] L. Yu, *A Developer's Guide to the Semantic Web*, no. February 1999. 2014.

[40] A. Polleres, A. Hogan, R. Delbru, and J. Umbrich, *RDFS and OWL reasoning for linked data*, vol. 8067 LNAI. 2013.

[41] M. Husáková and V. Bureš, "Formal ontologies in information systems development: A systematic review," *Inf.*, vol. 11, no. 2, 2020, doi: 10.3390/info11020066.

[42] A. Sun, Y. Sun, and C. Liu, "Restructuring E-learning With Ontologies," in *Proceedings - The 2007 International Conference on Computational Science and its Applications, ICCSA 2007*, 2007, pp. 532–536, doi: 10.1109/ICCSA.2007.76.

[43] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Sci. Am.*, vol. 284, no. 5, pp. 34–43, 2001, doi: 10.1038/scientificamerican0501-34.

[44] S. Staab and C. Brewster, "Knowledge Representation with Ontologies : The Present and Future," *IEEE Intell. Syst.*, vol. 19, no. 1, pp. 72–81, 2004, doi: DOI: 10.1109/MIS.2004.1265889.

[45] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, 1993, doi: 10.1006/knac.1993.1008.

[46] D. Dicheva, "Ontologies and semantic web for e-learning," in *Handbook on information technologies for education and training*, Springer, 2008, pp. 47–65.

[47] F. Tunde, A. Sunday, and O. Perpetual, "ONTOLOGY-BASED MODEL FOR E-LEARNING MANAGEMENT SYSTEM (O-BMEMS)," *Int. J. Comput. Sci. Issues*, vol. 12, no. 3, pp. 118–126, 2015.

[48] Q. Zeng, Z. Zhao, and Y. Liang, "Course ontology-based user's knowledge requirement acquisition from behaviors within e-learning systems," *Comput. Educ.*, vol. 53, no. 3, pp. 809–818, 2009, doi: 10.1016/j.compedu.2009.04.019.

[49] H. S. Chung and J. M. Kim, "Ontology design for creating adaptive learning path in e-learning environment," *Lect. Notes Eng. Comput. Sci.*, vol. 2195, pp. 585–588, 2012.

[50] I. Panagiotopoulos, A. Kalou, C. Pierrakeas, and A. Kameas, "An ontology-based model for student representation in intelligent tutoring systems for distance learning," *IFIP Adv. Inf. Commun. Technol.*, vol. 381 AICT, no. PART 1, pp. 296–305, 2012, doi: 10.1007/978-3-642-33409-2_31.

[51] A. Al-Zebari, S. R. M. Zeebaree, and A. Selamat, "ELECTRONIC LEARNING MANAGEMENT SYSTEM BASED ON SEMANTIC WEB TECHNOLOGY: A REVIEW," *Int. J. Adv. Electron. Comput. Sci.*, vol. 4, no. 3, pp. 2393–2835, 2017, [Online]. Available: http://iraj.

[52] S. Hoberman, "Data Modeling Essentials," *Inf. Manag.*, vol. 15, no. 3, p. 42, 2005.

[53] S. N. Akinwalere and V. Ivanov, "Artificial Intelligence in Higher Education: Challenges and Opportunities," vol. 12, no. 1, pp. 1–15, 2022, doi: DOI:10.33182/bc.v12i1.2015.

[54] K. vanLehn, "The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems," *Educ. Psychol.*, vol. 46, no. 4, pp. 197–221, 2011, doi: 10.1080/00461520.2011.611369.

[55] J. Self, "Theoretical foundations for intelligent tutoring systems," 1990.

[56] C. J. Butz, S. Hua, and R. B. Maguire, "A web-based bayesian intelligent tutoring system for computer programming," *Web Intell. Agent Syst.*, vol. 4, no. 1, pp. 61–81, 2006.

[57] M. L. Morales-Rodríguez, J. A. Ramírez-Saldivar, A. Hernández-Ramírez, J. P. Sánchez-Solís, and J. A. Martínez-Flores, "Architecture for an Intelligent Tutoring System that Considers Learning Styles," *Res.*

Comput. Sci., vol. 47, no. 1, pp. 37–47, 2012, doi: 10.13053/rcs-47-1-4.

[58] A. Mitrovic *et al.*, “ASPIRE: An authoring system and deployment environment for constraint-based tutors,” *Int. J. Artif. Intell. Educ.*, vol. 19, no. 2, pp. 155–188, 2009.

[59] S. B. Gilbert, S. B. Blessing, and E. Guo, “Authoring Effective Embedded Tutors: An Overview of the Extensible Problem Specific Tutor (xPST) System,” *Int. J. Artif. Intell. Educ.*, vol. 25, no. 3, pp. 428–454, 2015, doi: 10.1007/s40593-015-0045-0.

[60] B. Vesin, M. Ivanović, A. Klašnja-Milićević, and Z. Budimac, “Ontology-based architecture with recommendation strategy in Java tutoring system,” *Comput. Sci. Inf. Syst.*, vol. 10, no. 1, pp. 237–261, 2013, doi: 10.2298/CSIS111231001V.

[61] V. Aleven, B. M. McLaren, J. Sewall, and K. R. Koedinger, “A new paradigm for intelligent tutoring systems: Example-tracing tutors,” *Int. J. Artif. Intell. Educ.*, vol. 19, no. 2, pp. 105–154, 2009.

[62] V. Aleven *et al.*, “Example-Tracing Tutors: Intelligent Tutor Development for Non-programmers,” *Int. J. Artif. Intell. Educ.*, vol. 26, no. 1, pp. 224–269, 2016, doi: 10.1007/s40593-015-0088-2.

[63] R. A. Sottilare, K. W. Brawner, A. M. Sinatra, and J. H. Johnston, “An Updated Concept for a Generalized Intelligent Framework for Tutoring (GIFT),” *GIFTtutoring.org*, no. May, pp. 1–19, 2017.

[64] S. S. Abu Naser, “ITSB: An Intelligent Tutoring System Authoring Tool,” *Available online www.jsaer.com J. Sci. Eng. Res.*, vol. 63, no. 5, pp. 63–71, 2016.

[65] N. Singh, A. Kumar, and N. J. Ahuja, “Implementation and evaluation of personalized intelligent tutoring system,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 6, pp. 46–55, 2019.

[66] E. D. A. REZA, M. REZA, and H. SAJAD, “Main Components of Intelligent Tutoring Systems,” *Life Sci. J.*, vol. 10, no. 8s, pp. 342–346, 2013, [Online]. Available: http://www.lifesciencesite.com.

[67] W. Yathongchai, J. Angskun, and C. C. Fung, “An Ontology Model for Developing a SQL Personalized Intelligent Tutoring System,” *Naresuan Univ. J. Sci. Technol.*, vol. 25, no. 4, pp. 88–96, 2017.

[68] A. Keleş, R. Ocak, A. Keleş, and A. Gülcü, “ZOSMAT: Web-based intelligent tutoring system for teaching-learning process,” *Expert Syst. Appl.*, vol. 36, no. 2 PART 1, pp. 1229–1239, 2009, doi: 10.1016/j.eswa.2007.11.064.

[69] S. D’mello and A. Graesser, “AutoTutor and affective autotutor: Learning by talking with cognitively and emotionally intelligent computers that talk back,” *ACM Trans. Interact. Intell. Syst.*, vol. 2, no. 4, 2012, doi: 10.1145/0000000.0000000.

[70] A. Latham, K. Crockett, D. McLean, and B. Edmonds, “Adaptive tutoring in an intelligent conversational agent system,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7430 LNCS, pp. 148–167, 2012, doi: 10.1007/978-3-642-34645-3_7.

[71] M. Ikhwan Burhan, E. Sediyono, and K. Adi, “Intelligent Tutoring System Using Bayesian Network for Vocational High Schools in Indonesia,” *E3S Web Conf.*, vol. 317, p. 05027, 2021, doi: 10.1051/e3sconf/202131705027.

[72] K. Kularbphettong, P. Kedsiributt, and P. Roonrakwit, “Developing an Adaptive Web-based Intelligent Tutoring System Using Mastery Learning Technique,” *Procedia - Soc. Behav. Sci.*, vol. 191, pp. 686–691, 2015, doi: 10.1016/j.sbspro.2015.04.619.

[73] B. Cheung, L. Hui, J. Zhang, and S.-M. Yiu, “SmartTutor: An intelligent tutoring system in web-based adult education,” *J. Syst. Softw.*, vol. 68, no. 1, pp. 11–25, 2003.

[74] J. J. P. C. Rodrigues, P. F. N. João, and B. Vaidya, “EduTutor: An intelligent tutor system for a learning management system,” *Int. J. Distance Educ. Technol.*, vol. 8, no. 4, pp. 66–80, 2010, doi: 10.4018/jdet.2010100105.

[75] N. Singh, V. K. Gunjan, A. K. Mishra, and R. K. Mishra, "SeisTutor : A Custom-Tailored Intelligent Tutoring System and Sustainable Education," *Sustainability*, vol. 14, no. 7, pp. 1–24, 2022, doi: https://doi.org/10.3390/su14074167.

[76] J. Hendler, F. Gandon, and D. Allemang, *Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL*. Morgan \& Claypool, 2020.

[77] K. Saks, H. Ilves, and A. Noppel, "The impact of procedural knowledge on the formation of declarative knowledge: How accomplishing activities designed for developing learning skills impacts teachers' knowledge of learning skills," *Educ. Sci.*, vol. 11, no. 10, 2021, doi: 10.3390/educsci11100598.

[78] F. Nickols, "The Knowledge in Knowledge Management," *Knowl. Manag.*, vol. 300, pp. 575–585, 2000.

[79] B. Rittle-Johnson and M. Schneider, "Developing conceptual and procedural knowledge of mathematics," *Oxford Handbook of Numerical Cognition*, vol. 1. pp. 1118–1134, 2014, doi: DOI:10.1093/OXFORDHB/9780199642342.013.014.

[80] V. S. Lenko, V. V Pasichnyk, and Y. M. Shcherbyna, "Knowledge Representation Models," *Bull. Jap. Soc. sci. Fish*, vol. 22, no. 9, pp. 249–265, 2017, [Online]. Available: http://ena.lp.edu.ua.

[81] N. Noy, Y. Gao, A. Jain, A. Patterson, A. Narayanan, and J. Taylor, "Industry-scale knowledge graphs lessons and challenges," *Queue*, vol. 17, no. 2, pp. 1–28, 2019, doi: 10.1145/3329781.3332266.

[82] A. Bhattacharyya and D. Chakravarty, "(Graph Database: A Survey)," *2020 Int. Conf. Comput. Electr. Commun. Eng. ICCECE 2020*, 2020, doi: 10.1109/ICCECE48148.2020.9223105.

[83] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A Survey on Knowledge Graphs: Representation, Acquisition, and Applications," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 2, pp. 494–514, 2022, doi: 10.1109/TNNLS.2021.3070843.

[84] P. Ceravolo and E. Damiani, "Introduction to ontology engineering," *Semant. Knowl. Manag. An Ontol. Framew.*, pp. 25–50, 2008, doi: 10.4018/978-1-60566-034-9.ch002.

[85] H. H. Owaied, M. M. Abu-A'ra, and M. M. d. Qasem, "A hybrid scheme for knowledge representation," *J. Appl. Sci.*, vol. 11, no. 14, pp. 2525–2535, 2011, doi: 10.3923/jas.2011.2525.2535.

[86] N. M. Hewahi, "Intelligent tutoring system: Hierarchical rule as a knowledge representation and adaptive pedagogical model," *Inf. Technol. J.*, vol. 6, no. 5, pp. 739–744, 2007, doi: 10.3923/itj.2007.739.744.

[87] I. Šarić-Grgić *et al.*, "Bayesian Student Modeling in the AC{\&}NL Tutor," in *Adaptive Instructional Systems*, 2020, pp. 245–257.

[88] G. H. A. Ahmed and L. Kovács, "Development of ontology-based model to support learning process in LMS," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 24, no. 1, pp. 507–518, 2021, doi: 10.11591/ijeecs.v24.i1.pp507-518.

[89] B. Hookway, *Interface*. MIT Press, 2014.

[90] M. A. Ramdhani and H. Muhammadiyah, "Proceeding International Conference of Islamic Education: Reforms, Prospects and Challenges Faculty of Tarbiyah and Teaching Training The Criteria of Learning Media Selection for Character Education in Higher Education," *Proceeding Int. Conf. Islam. Educ. Reforms, Prospect. Challenges Fac. Tarb. Teach. Train. Criteria Learn. Media Sel. Character Educ. High. Educ.*, pp. 174–182, 2015.

[91] J. Dunlosky, K. A. Rawson, E. J. Marsh, M. J. Nathan, and D. T. Willingham, "Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology," *Psychol. Sci. Public Interes. Suppl.*, vol. 14, no. 1, pp. 4–58, 2013, doi: 10.1177/1529100612453266.

[92] Y. Chen, "Improving student model for individualized learning," Université Pierre et Marie Curie-Paris, 2016.

[93] E. Katis, "Semantic modeling of educational curriculum and syllabus.," in *European Semantic Web*

Conference, 2018, pp. 55–59.

[94] J. P. Alvarado-Magaña, A. Rodr'iguez-D'iaz, J. R. Castro, and O. Castillo, "Type-2 fuzzy grammar in language evolution," in *Recent Advances on Hybrid Intelligent Systems*, Springer, 2013, pp. 501–515.

[95] K. T. Atanassov, *Intuitionistic Fuzzy Sets Theory and Applications*, vol. 35. Physica-Verlag, Heidelberg: Springer-Verlag Berlin Heidelberg GmbH, 1999.

[96] L. Zhang, "A Novel Similarity Measure and Its Application in Medical Diagnosis under Intuitionistic Fuzzy Settings," *Proc. 2016 7th Int. Conf. Educ. Manag. Comput. Med. (EMCM 2016)*, vol. 59, no. Emcm 2016, pp. 455–458, 2017, doi: 10.2991/emcm-16.2017.87.

[97] A. M. Kozae, A. Elshenawy, and M. Omran, "A Case Study for Engineering Students: Intuitionistic Fuzzy Set and Its Application in Selecting Specialization," *Curr. Top. Math. Comput. Sci. Vol. 4*, vol. 2, no. 6, pp. 1–7, 2021, doi: 10.9734/bpi/ctmcs/v4/10033d.

[98] G. Hacat, "On The Determination Students' Aptitude Using Intuitionistic Fuzzy Logic," *J. Univers. Math.*, vol. 3, no. 1, pp. 94–102, 2020.

[99] M. N. Iqbal and U. Rizwan, "Some applications of intuitionistic fuzzy sets using new similarity measure," *J. Ambient Intell. Humaniz. Comput.*, no. 0123456789, pp. 0–4, 2019, doi: 10.1007/s12652-019-01516-7.

[100] M. L. A. Stassen, K. Doherty, and M. Poe, *Course-based review and assessment: Methods for understanding student learning*. Office of Academic Planning \& Assessment, University of Massachusetts Amherst, 2001.

[101] R. Hosseini and A. Kardan, "Intuitionistic fuzzy-based method for assessing the learner's knowledge level and personalization of learning path," in *Proceedings of the 6th International Conference on Virtual Learning (ICVL2011)*, 2011, pp. 441–447.

[102] K. T. Atanassov, "Intuitionistic fuzzy sets," *Int. J. Bioautomation*, vol. 20, pp. S1–S6, 2016, doi: 10.1007/978-3-7908-1870-3_1.

[103] M. Horridge and P. F. Patel-Schneider, "OWL 2 Web Ontology Language Manchester Syntax (Second Edition)," *W3C Work. Gr. Note*, no. December 2012, pp. 1–10, 2012, [Online]. Available: http://www.w3.org/TR/owl2-manchester-syntax/.

[104] B. McBride, "The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS," *Handb. Ontol.*, pp. 51–65, 2004, doi: 10.1007/978-3-540-24750-0_3.

[105] S. Abburu and S. B. Golla, "Ontology storage models and tools: An authentic survey," *J. Intell. Syst.*, vol. 2015, no. 4, pp. 539–553, 2015, doi: 10.1515/jisys-2014-0167.

[106] APACHE_JENA_ARCHITECTURE, "Apache Jena - Jena architecture overview," *2015*, 2015. https://jena.apache.org/about_jena/architecture.html (accessed Sep. 17, 2022).

[107] P. R. Panchal and P. R. Swaminarayan, "Execution of Sparql Query Using Apache Jena Fuseki Server in Aishe Domain," *Int. J. Adv. Eng. Res. Dev.*, vol. 4, no. 09, pp. 387–395, 2017, doi: 10.21090/ijaerd.89498.

[108] "Jena documentation overview," *Jena TDB, [Online]. http://jena. apache. org/documentati0n*, 2011. https://jena.apache.org/documentation/ (accessed Sep. 15, 2022).

[109] R. Sherman, *Business intelligence guidebook: From data integration to analytics [Skillsoft version]*. 2015.

[110] L. Jean-Baptiste, *Ontologies with Python: Programming OWL 2.0 Ontologies with Python and Owlready2*. Paris: Apress, Berkeley, CA, 2021.

[111] Y. Demchenko, L. Comminiello, and G. Reali, "Designing customisable data science curriculum using ontology for data science competences and body of knowledge," *ACM Int. Conf. Proceeding Ser.*, pp. 124–128, 2019, doi: 10.1145/3322134.3322143.

[112] D. Ashley, *Foundation Dynamic Web Pages with Python*. Berkeley, CA: Springer Apress, 2020.

[113] M. Grinberg, *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.," 2018.

[114] J. Sequeda, "Introduction to : Reasoners," *Data Management education*, 2013. https://www.dataversity.net/introduction-to-reasoners/.

[115] L. Jean-Baptiste, *Ontologies with Python: Programming OWL 2. 0 Ontologies with Python and Owlready2*. Apress, 2021.

[116] T. Eiter, G. Ianni, T. Krennwallner, and A. Polleres, *Rules and ontologies for the semantic Web*, vol. 5224 LNCS. 2008.

[117] О. С. Лагодинський, О. В. Буяло, and С. В. Хамула, "Application of Achievement Testing Software of Cadets in Higher Military Educational Institutions," *Inf. Technol. Learn. Tools*, vol. 81, no. 1, pp. 222–234, 2021, doi: 10.33407/itlt.v81i1.3675.

[118] G. Hussein, A. L. I. Ahmed, L. Kovács, G. Hussein, and A. Ahmed, "Ontology Domain Model for E-Tutoring System," *J. Softw. Eng. …*, vol. 5, no. 1, pp. 37–44, 2020, [Online]. Available: https://www.academia.edu/download/63422352/V5N1-620200525-19169-wgleae.pdf.