A·L·I·C·E
Adaptive Learning via Intuitive/Interactive Collaborative and Emotional systems

Project Number:	**257639**
Project Title:	ALICE: ADAPTIVE LEARNING VIA INTUITIVE/INTERACTIVE, COLLABORATIVE AND EMOTIONAL SYSTEMS
Instrument:	Specific Targeted Research Projects
Thematic Priority:	ICT-2009.4.2:Technology-Enhanced Learning
Project Start Date:	June 1st, 2010
Duration of Project:	24 Months
Deliverable:	**D7.1.2: Models and Methodologies for Knowledge Model Contextualization v2**
Revision:	2.0
Workpackage:	WP7: Adaptive Technologies for e-Learning Systems
Dissemination Level:	Public
Due date:	November 30th, 2011
Submission Date:	November 30th, 2011
Responsible	CRMPA
Contributors:	N/A

PROJECT CO-FUNDED BY THE EUROPEAN COMMISSION WITHIN THE SEVENTH FRAMEWORK PROGRAMME (2007-2013)

SEVENTH FRAMEWORK PROGRAMME

Version History			
Version	**Date**	**Changes**	**Contributors**
0.1	30/09/2010	The deliverable structure has been completed, the introduction and the background chapters have been written. Sections 1 and 2 have been completed.	CRMPA
0.2	31/10/2010	A study about the state of the art on context-aware e-learning has been performed. A learning context model has been defined. Section 5 has been completed. Section 3 is under development.	CRMPA
0.3	30/11/2010	The abstract domain model, the domain contextualization algorithm and the updated learning life-cycle have been defined. Section 3 has been completed.	CRMPA
1.0	31/12/2010	The extensions needed to IWT in order to implement defined models and methodologies have been studied. Section 4 has been completed.	CRMPA
1.1	21/04/2011	Minor spelling corrections.	CRMPA
1.2	30/09/2011	Revised section 3.1 about the learning content. Added the extended contextualization algorithm (3.2.2) to propagate relations during domain model contextualization.	CRMPA
1.3	31/10/2011	Related work section improved with additional research on course sequencing and a comparison with similar systems.	CRMPA
1.4	15/11/2011	The notions of derivation between context profiles have been introduced in 3.1.2. Revision of section 4 about technological perspective. Conclusions and future work section has been revised.	CRMPA
2.0	30/11/2011	Final version after peer review.	CRMPA

Table of Contents

1 Introduction

The purpose of this document is to provide the theoretical foundation to introduce knowledge model contextualisation in the ALICE learning system with respect to requirements described in [7] (section 5.2). This will allow to improve and extend existing models, methodologies and components of ALICE reference platform IWT in order to prepare it for a smooth integration of methodological and technological components coming from other ALICE research lines.

This document is structured in the following sections.

- **Section 2** provides an introduction about models and algorithms that are currently applied by IWT to obtain learner modelling and learning experience individualisation features. This includes the domain model, the learner model, the learning resource model. Moreover the learning life-cycle applied by IWT is also described as well as the connected algorithms. This is a needed background to understand new models and algorithms defined in section 3.
- **Section 3** defines improvements and extensions needed to IWT, from a theoretical perspective, to support contexts and contextualisation features. The learning context model, the abstract domain model (an evolution of the existing IWT domain model) and two different algorithms for domain model contextualization will be defined as well as an evolution of the learning life-cycle applied to IWT.
- **Section 4** shows the proposed improvements from a technological perspective by defining new software components to be developed and how they must be integrated in the existing IWT architecture. Such components include an improved version of LIA (the IWT component providing functions discussed in section 2), a service for abstract domain models storing and retrieving, a contextualized course manager, an editor for abstract domain models, a context manager and an improved component for learner models management.
- **Section 5** contextualizes performed research with respect to the relevant literature about Intelligent Tutoring Systems (ITS), personalized e-learning, course sequencing, context modelling and context-based e-learning.
- **Section 6** concludes the report and introduces next steps.

The document updates and extends [31]. It updates the defined learning context model, it introduces the notions of derivation between contexts, it includes an extended algorithm for domain model contextualization that relaxes a constraint required by the standard one. The technological perspective section adds two new components (a context editor and a learner model manager) and revises the abstract domain model editor as well as the contextualized course manager. It also provides an extended version of the section on related work.

2 Background

The Learning Intelligent Advisor (LIA) is a component of IWT (Intelligent Web Teacher) that provides advanced features like learner modelling and learning experience individualisation. LIA is based on a set of models able to represent the main entities involved in the process of teaching/learning and on a set of methodologies, leveraging on such models, for the generation of individualised learning experiences with respect to learning objectives, pre-existing knowledge and learning preferences.

This chapter describes applied models (section 2.1) and related methodologies and how they are used during the whole learning life-cycle (section 2.2) by the current version of LIA. The subsequent chapter deals instead with improvements proposed by ALICE with respect to the contextualisation of the knowledge model. This chapter is strongly based on [1] that can be read to obtain further details about architectural aspects related to the integration in IWT and results of past experimentations activities.

2.1 LIA Models

The next sub-paragraphs describe the four modes adopted by LIA to formally represent the main actors involved in the teaching/learning process. The next sub-paragraph shows how they are used in such process.

2.1.1 The Domain Model

The domain model describes the knowledge that is object of teaching through a set of concepts (representing the topics to be taught) and a set of relations between concepts (representing connections among topics). Such structure can be formally represented with a concepts graph G (C, R_1,… R_n) where C is the set of nodes representing domain concepts and each R_i is a set of arcs corresponding to the i-th kind of relation. Two categories of relations are supported: hierarchical relations are used to factorise high-level concepts in low-level concepts while ordering relations are used to impose partial orderings in the concept set. IWT uses a concept graph G (C, *BT*, *IRB*, *SO*) with three relations *BT*, *IRB* and *SO* whose meaning is explained below (where a and b are two concepts of C):

- *BT* (a, b) means that the concept a belongs to the concept b i.e. b is understood iif every a so that a belongs to b is understood (hierarchical relation);
- *IRB* (a, b) means that the concept a is required by the concept b i.e. a necessary condition to study b is to have understood a (ordering relation);
- *SO* (a, b) means that the suggested order between a and b is that a precedes b i.e. to favour learning, it is desirable to study a before b (ordering relation).

Any number of additional relations may be introduced provided that they belong to one of the two categories described above. The figure 1 shows a sample domain model in the didactics

of artificial intelligence exploiting the relations defined above and stating that to understand "logics" means to understand "formal systems", "propositional logic" and "first order logic" but, before approaching any of these topics it is necessary to have an "outline of set theory" first. Moreover, "formal systems" must be taught before both "propositional logics" and "first order logic" while it is desirable (but not compulsory) to teach "propositional logics" before "first order logic".

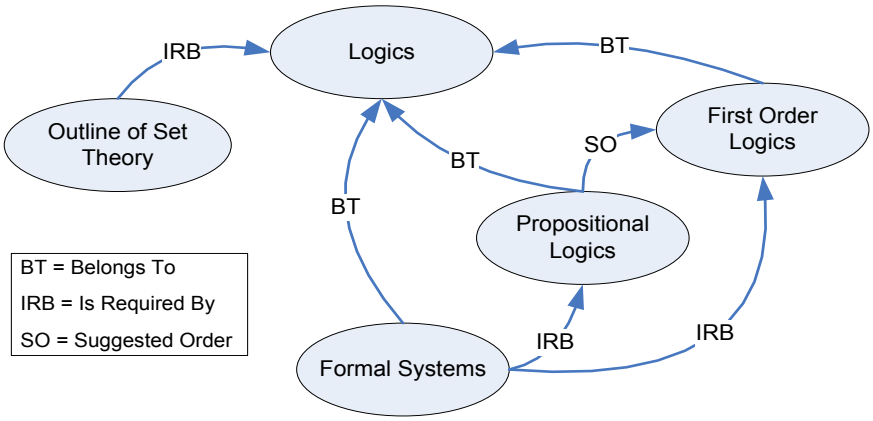

Figure 1. A sample concepts graph.

A set of **teaching preferences** may be added to the domain model to define feasible teaching strategies that may be applied for each available concept. Such preferences are represented as an application $TP\ (C \times Props \times PropVals) \rightarrow [0, 10]$ where *Props* is the set of didactical properties and *PropVals* is the set of feasible values for such properties. The table 1 provides some (non exhaustive) example of didactical property and associated feasible values. It is worth noting that *TP* is defined only for couples of *Props* and *PropVals* elements belonging to the same row in table 1.

Properties	**Feasible values**
Didactic method	Deductive, inductive, etc.
Resource type	Text reading, video clip, simulation, virtual experiment, etc.
Interactivity level	High, medium, low

Table 1. Example of didactical properties and feasible values.

As an example, the following definition for TP states that in order to teach formal systems a deductive didactic method is more suitable with respect to an inductive one while, to give an outline of set theory, both methods are equally good.

- *TP* (“formal systems”, “didactic method”, “deductive”) = 10;
- *TP* (“formal systems”, “didactic method”, “inductive”) = 4; (1)
- *TP* (“outline of set theory”, “didactic method”, “deductive”) = 8;
- *TP* (“outline of set theory”, “didactic method”, “inductive”) = 8.

2.1.2 The Learner Model

The learner is the main actor of the whole learning process and it is represented with a cognitive state and a set of learning preferences. The **cognitive state** represents the knowledge reached by a learner at a given time and it is represented as an application *CS* (*C*) → [0, 10] where *C* is the set of concepts of a given domain model. Given a concept *c*, *CS* (*c*) indicates the degree of knowledge (or grade) reached by a given learner for *c*. If such grade is greater then a given “passing” threshold θ then *c* is considered as known, otherwise it is considered as unknown. For example, assuming that $\theta = 6$, the following definition states that a given learner masters the outline of set theory but has a very poor knowledge of propositional logics.

- *CS* (“outline of set theory”) = 8; (2)
- *CS* (“formal systems”) = 4.

The **learning preferences** provide an evaluation of learning strategies that may be adopted for a given learner. They are represented as an application *LP* (*Props* × *PropVals*) → [0, 10] where *Props* and *PropoVals* are the same sets defined in 2.1.1. Differently from teaching preferences, learning preferences are not linked to a domain concept but refer to a specific learner. As an example, the following definition states that a given learner prefers an inductive didactic method on a deductive one.

- *LP* (“didactic method”, “deductive”) = 5; (3)
- *LP* (“didactic method”, “inductive”) = 8.

The cognitive state of any learner is initially void (i.e. *CS* (*c*) = 0 for any *c* included in a given domain model) and may be initialized on a teaching domain with a pre-test. Learning preferences may be initialized by the teacher or directly by learners through a questionnaire capable of evaluating learners styles and transform them in suitable values for learning preferences. Both parts of the learner model are automatically updated during learning resources by a “learner model updating algorithm” (see 2.2.4).

2.1.3 The Learning Resource Model

A learning resource represents a learning content that must be played by a learner to acquire one or more domain concepts. The “learning presentation generation algorithm” (see section 2.2) uses learning resources as building blocks to generate learning experiences so the structure of a single learning resource is out of the scope of this document.

In order to be effectively used as a building block, a learning resource *A* is described through the following elements:

- a set of **concepts** C_A part of a given domain model, that are covered by in the learning resource (only leaf concepts with respect to the *BT* relation can be included in C_A i.e. learning resources are associated with leaf concepts);
- a set of **didactical properties** expressed as an application DP_A (*property*) = *value* representing learning strategies applied by the learning resource;
- a set of **cost properties** expressed as an application CP_A (*property*) = *value* that must be taken into account in the optimisation process connected with the "learning presentation generation algorithm".

Didactical properties components have the same meaning with respect to teaching and learning preferences i.e. *property* and *value* may assume values from a closed vocabulary (see table 1). Differently from learning and teaching preferences, they are neither linked to a domain concept nor to a specific student but to a learning resource. As an example, the following assertions state that the resource *A* is a simulation that uses an inductive didactic method and have a high interactivity level.

- DP_A ("didactic method") = "inductive";
- DP_A ("resource type") = "simulation"; (4)
- DP_A ("interactivity level") = "high".

Cost properties are couples that may be optionally associated to learning resources, whose properties may assume values from the closed vocabulary {*price*, *duration*} and whose values are positive real numbers representing, respectively the eventual price of a single learning resource and its average duration in minutes. As an example, the following assertions state that the price of a resource *A* is 1.50 € while its average duration is 5 minutes.

- CP_A ("price") = 1.5; (5)
- CP_A ("duration") = 5.

Testing resources are learning resources used to verify the knowledge acquired by the learner. As learning resources, a testing resource *T* is connected to a set C_T of covered concepts indicating, in this case, the list of concepts verified by the resource. Once executed by a specific learner, they return an evaluation E_T belonging to the range [0, 10] indicating the degree of fulfilment of the test by that learner. Differently from other resources here DP_T and CP_T are not significant.

2.1.4 The Unit of Learning

A unit of learning represents a sequence of learning resources needed to a learner in order to understand a set of target concepts in a given domain with respect to a set of defined cost constraints. It is composed by the following elements:

- a set of **target concepts** *TC* part of a given domain model, that have to be mastered by a given learner in order to successfully accomplish the unit of learning;
- a set of **cost constraints** *CC* (*property*) = *value* that must be taken into account in the optimisation process connected with the "learning presentation generation algorithm" (see section 2.2);
- a **learning path** *LPath* (c_1, …, c_n) i.e. an ordered sequence of concepts that must be taught to a specific learner in order to let him master target concepts;
- a **learning presentation** *LPres* (a_1, …, a_m) i.e. an ordered sequence of learning resources that must be presented to a specific learner in order to let him/her master the target concepts.

While target concepts and cost constraints are defined by the course teacher (in case of supervised learning) or by the learner himself (in case of self-learning), the learning path and the learning presentation are calculated and updated after each testing resource by specific generation algorithms described in section 2.2 basing on the domain model, on the learner model associated to the target learner and on available learning resources. Concerning cost constraints, the property may assume values from the closed vocabulary {*price*, *duration*}. Feasible values are positive real numbers representing, respectively the maximum total price and the maximum total duration of the unit of learning.

As an example the following assertions state that the system have to build an unit of learning explaining the concept of "logics" that have a maximum total price of 100€ and during a maximum time of 6 hours (= 360 minutes).

- *TC* = {"logics"};
- *CC* ("price") = 100; (6)
- *CC* ("duration") = 360.

2.2 The Learning Life-Cycle

After having seen how the main involved actors and objects are represented by means of appropriate models, it is necessary to see how LIA uses such models to automate some of the phases of the teaching/learning process. LIA sees the learning lifecycle as composed by five phases as shown in figure 2. Each phase includes one or more activities that have to be performed by the actors involved in the process (namely the teacher and the learner) or by LIA itself.

- In the ***preparation phase*** the teacher defines or selects a feasible domain model and prepares or selects a set of learning resources that may be used in the learning experience while learners may define their learning preferences.
- In the ***starting phase*** the teacher initializes a unit of learning by setting target concepts and cost constraints and associates one or more learners to it (in self-directed learning, learners settle own target concepts and constraints by themselves).

Then LIA generates a personalised learning path for each learner through a “learning path generation algorithm” described in 2.2.1 and then introduces placeholders for testing resources through a “milestone setting algorithm” described in 2.2.2.

- In the ***execution phase***, LIA selects a fragment of the learning path and generates the best learning presentation for each enrolled learner by applying the “learning presentation generation algorithm” described in 2.2.3. The learner then undertakes the learning and testing resources of the learning presentation until its end.
- In the ***evaluation phase***, when the learner ends a learning presentation fragment, his/her learner model is updated on the basis of the results of testing resources included in the fragment according to the “learner model updating algorithm” described in 2.2.4 and a new execution phase starts by generating a new learning presentation fragment that will possibly include recovery resources for concepts that the student did not understand.
- In the ***closure phase***, once all concepts of the unit of learning are mastered by the learner, the system collects statistical information on the process and the teacher may use this information as a basis for possible improvements of the domain model and/or of learning and testing resources.

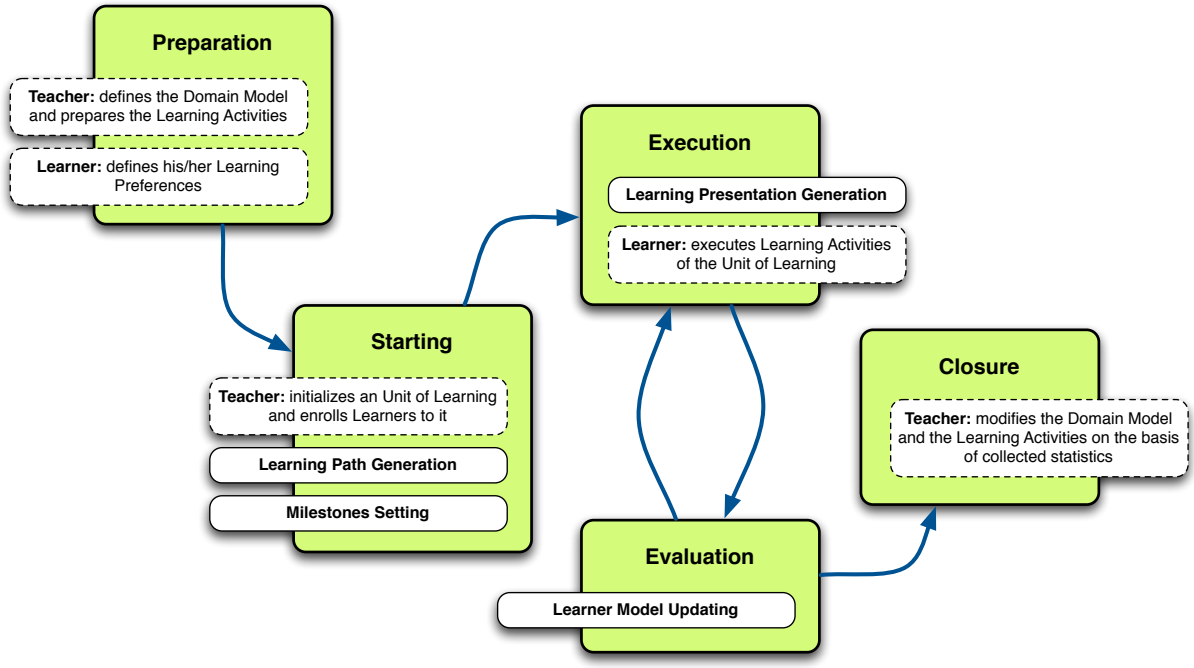

Figure 2. The Learning Lifecycle.

The following paragraphs describe in more details the algorithms exploited by LIA in the several phases of the learning lifecycle (un-dotted boxes in figure 2).

2.2.1 Learning Path Generation

The generation of the learning path is the first step to completely generate a unit of learning. Starting from a set of target concepts *TC* and from a domain model, a feasible learning path must be generated taking into account the concepts graph *G*(*C*, *BT*, *IRB*, *SO*) part of the domain model (with $TC \subseteq C$). The four steps of the learning path generation algorithm are summarizes below.

- The **first step** is purposed to build the graph *G*'(*C*, *BT*, *IRB*', *SO*') by propagating ordering relations downward the hierarchical relation. *IRB*' and *SO*' are initially set to *IRB* and *SO* respectively and, by applying the theory of partial ordered sets, they are modified by applying the following rule: for each arc $ab \in IRB' \cup SO'$ we have to substitute it with arcs *ac* for all $c \in$ C such that there exist a path from *c* to *b* on the arcs from *BT*.
- The **second step** is aimed to build the graph *G*"(*C*', *R*) where *C*' is the subset of *C* including all concept that must be taught according to *TC* i.e. *C*' is composed by all nodes of *G*' from which there is a ordered path in $BT \cup IRB'$ to concepts in *TC* (including target concepts themselves). *R* is initially set to $BT \cup IRB' \cup SO'$ but all arcs referring to concepts external to *C*' are removed.
- The **third step** finds a linear ordering of nodes of *G*" by using depth-first search so by visiting the graph nodes along a path as deep as possible. The obtained list *L* will constitute a first approximation of the learning path.
- The **fourth step** generates the final learning path *LPath* by deleting from *L* all non-atomic concepts with respect to the graph *G* i.e. *LPath* will include any concept of *L* apart concepts *b* so that $ab \in BT$ for some *a*. This ensures that only leaf concepts (i.e. concepts having some associated learning resource) will be part of *LPath*.

As a first example we may consider the concept graph in figure 1 as *G* and the set {"Logics"} as *TC*. The figure 3 (first case) shows the intermediate and final results of the learning path generation algorithm on these inputs carrying out to the following result stating that, to understand logics, the learner has to learn the outline of set theory, then formal systems, then propositional logics and, finally, first order logics:

LPath = ("Outline of Set Theory", "Formal Systems", "Propositional Logics", "First Order Logics"). (7)

As a second example we may consider the same concept graph *G* while setting *TC* = {"First Order Logics"}. Algorithm results in this case are also shown in figure 3 (second case) and reported below:

LPath = ("Outline of Set Theory", "Formal Systems", "First Order Logics"). (8)

It is important to note that the algorithm converges iif *G* is acyclic. If this condition cannot be guaranteed then an additional step zero must be added in order to remove cycles in *G*.

Several algorithms may be applied to do that. The condition that must be respected is that, for each detected cycle, arcs are discarded starting from the less significant (arcs belonging to *SO* are less significant than arcs belonging to *IRB* that are less significant than arcs belonging to *BT*) until the cycle disappears.

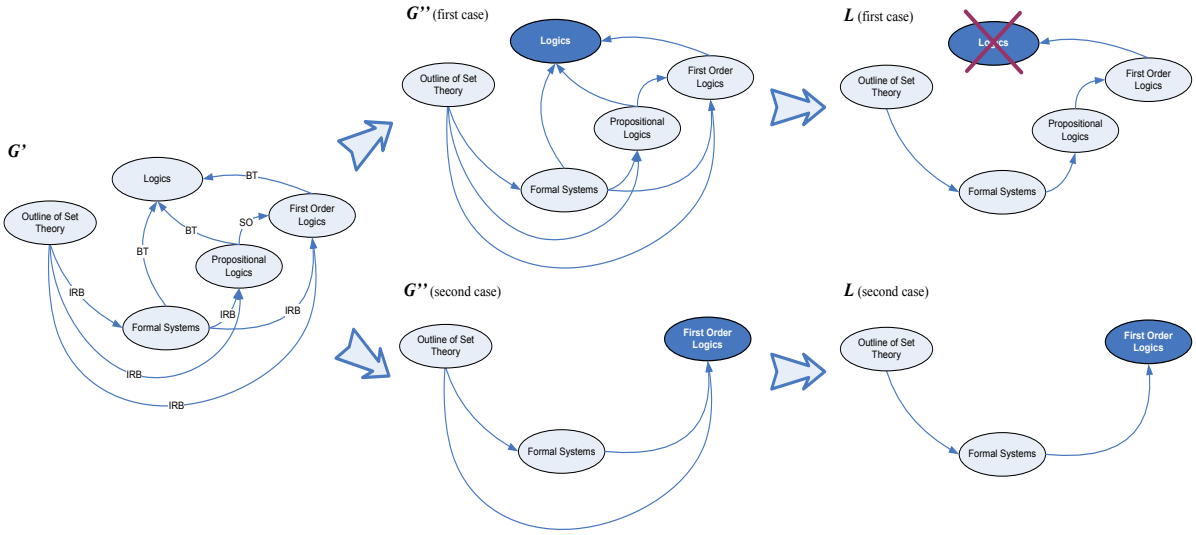

Figure 3. Application example of the learning path generation algorithm.

2.2.2 Milestones Setting

Once the learning path is generated, it is necessary to insert in it placeholders for testing resources named milestones. While concepts of the learning path will be converted in learning resources different from tests by the presentation generation algorithm, milestones will be converted in testing resources by the same algorithm. Several different possibilities exist to place milestones on the learning path:

- they can be placed directly by teachers;
- they can be placed basing on a list of percentages given by the teacher (e.g. the input list [0.2, 0.5, 0.7, 1] means that four milestones should be placed in the learning path approximately at 20%, 50%, 70% and at the end);
- they can be placed dynamically after each sub-list of concepts belonging to the same higher-level concept following the hierarchical relation (i.e. after each sub-list of concepts $c_1, \ldots, c_n$ so that $c_i a \in BT$ for some a).

Each milestone covers all preceding concepts in the learning path until the beginning of the course apart concepts already known by the learner according to his/her cognitive state (i.e. any concept a so that $CS(a) \geq$ the "passing" threshold θ as defined in 2.1.2). As an example, starting from the learning path LPath given in (7) and the cognitive state CS defined in (2), a

list of percentages [0.5, 1] carries out to the following updated learning path where the first milestone M_1 covers the concept "Formal Systems" (given that "Outline of Set Theory" is considered already known by the learner) while the second milestone M_2 covers the concepts "Formal Systems", "Propositional Logics" and "First Order Logics":

LPath' = ("Outline of Set Theory", "Formal Systems", M1, "Propositional Logics", "First Order Logics", M2) (9)

A milestone at 0% of the learning path is a special milestone meaning that a ***pre-test*** is necessary before entering in the unit of learning. Differently for other milestones, pre-test milestones may be of three different kinds:

- M_R indicates a ***pre-test on requirements*** testing concepts of the learning path considered as known by the learner and it is purposed to verify if such knowledge is still retained by the learner before entering in the unit of learning;
- M_C indicates a ***pre-test on content*** testing concepts of the learning path considered as unknown by the learner and is purposed to discover any possible further known concepts not stored in the cognitive state yet and to adapt the unit of learning accordingly in order to avoid to re-explain such known concepts;
- M_I indicates an **integrated pre-test** including both the pre-tests above so testing all concepts of the learning path.

As other milestones, pre-test milestones may be placed by teachers or by an algorithm based on a list of percentages. As other milestones they are converted in testing resources by the presentation generation algorithm.

2.2.3 Learning Presentation Generation

The presentation generation algorithm is purposed to build a fragment of presentation (part of an unit of learning) suitable for a specific learner basing on a learning path *LPath*' that have to be covered, on a set of teaching preferences *TP* belonging to a domain model, on a cognitive state *CS* and a set of learning preferences *LP* (both part of the learner model associated to the target learner), on a set of optional cost constraints *CC* and on a set of available learning resources (including tests). The three steps of the presentation generation algorithm are summarizes below.

- The ***first step*** is to select the sub-list *L* of *LPath*' that have to be converted in a presentation. *L* is the sequence of all the concepts of *LPath*' not already known by the learner (i.e. any concept a so that $CS(a) < \theta$) from the beginning to the first milestone preceded by at least one concept not already known by the learner. If *L* is empty then the algorithm ends because the learner knows all concepts of the learning path.
- The ***second step*** is to define the best sequence of learning resources *P*, selected from available learning resources (not including tests), covering *L* on the basis of *TP*, *LP* and *CC*. This requires the resolution of a optimisation problem that is not discussed here but in [1].

- The ***third step*** is to add testing resources at the end of *P* so obtaining the final learning presentation *Pres*. Testing resources are selected to cover all concepts of *L* without taking into account didactical and cost properties potentially linked to testing resources (as said in 2.1.3 they are not significant for tests).

It is important to note that, in the case that *LPath* starts with a pre-test milestone (M_R, M_C or M_I), then *L* will be defined in order to include all concepts that should be tested according to the kind of pre-test milestone (as defined in 2.1.3), *P* is then settled to be void and only the third step is executed. Then the pre-test milestone is removed from *LPath*.

2.2.4 Learner Model Updating

For each testing resource *T* executed by the learner in the last learning presentation fragment, the test returns (as explained in 2.1.3) an evaluation E_T between 1 and 10 representing the degree of fulfilment of the test by the involved learner. For each concepts *c* belonging to C_T, the cognitive state of the learner is modified in this way: if $CS(c)$ is not defined then $CS(c) = E_T$, otherwise $CS(c) = (CS(c) + E_T) / 2$. This is repeated for any executed test *T* in *LPres*.

An optional procedure consists in propagating the evaluation of each concept over required concepts in the concepts graph following backward the ordering relation *IRB*. Some times in-fact iterated failures to understand a set of concepts indicates a possible misconception in a common requirement. Such procedure helps to find these implicit misconceptions and forces the learning presentation generation algorithm to introduce resources covering such misconceptions in subsequent fragments by decreasing their grade in the learner cognitive state.

To apply this procedure, the grade of each concept *c'* so that there is an ordered path in *IRB'* (see 2.2.1) from *c'* to *c* should be modified in the learner cognitive state as follows: $CS(c') = (a \cdot E_T + (n - a) \cdot CS(c')) / n$ where *a* is a constant in (0,½] representing the intensity of the propagation while *n* represents the number of the arcs of the path between *c'* and *c*.

After a testing phase, learning preferences may be also modified in the following way: if the same learning resource *A* has been proposed *n* times to the learner (for a given *n*) without success (so bringing to a negative score in testing resources on related concepts), then the system decreases, for each didactical property $DPA(property) = value$ associated with *A*, learner preferences $LP(property, value)$ of a given constant δ. In this way the learning resource *A* as well as learning resources with similar didactical properties will be less often selected in future learning presentations.

Conversely learning resources that demonstrate to be successful (so bringing to positive scores in testing resources) will increase learner preferences related to resources' didactical properties. In this way, similar resources (i.e. resources with similar didactical properties) will be more often selected in future learning presentations.

3 Methodological Perspective

The domain model of IWT is currently unable to deal with learning contexts i.e. (virtual or real) places where the learning process happens. This means that, for example, we need different models to teach mathematical analysis in different university faculties (topics to be taught may vary as well as connected learning methods) or at different school levels.

In order to overcome this limitation one of the improvements that ALICE foresees is to give teachers the possibility to specify learning contexts and to provide algorithms and integrated software tools able to automatically adapt domain models defined by teachers with respect to several different learning contexts.

Functions that must be included in ALICE with respect to knowledge model contextualization have been described in [7] (section 5.2). They can be summarized as follows:

- the teacher will be able to define available contexts;
- the teacher will be able to include, in domain models, contextualization information that define how models change according to contexts;
- the system will be able, given a student, a context, a set of target concepts and a set of constraints, to build a customized unit of learning.

The following sub-section introduces the notion of learning context. The subsequent focuses on theoretical improvements needed to models and algorithms described in section 2 to implement such functions.

3.1 The Learning Context

The IMS Learning Resource Meta-Data Information Model Specification [2] defines a context as “the typical learning environment where use of learning object is intended to take place”. Other authors [6] define the context as the learner environment and talk about three main environments: the external environment (classroom, working space, in-person coaches, etc.), the internal environment (previous beliefs, thoughts, hopes, etc.) and the digital environment.

According to [5] the learning context (in e-learning settings) describes a class of learners within a technological infrastructure with a set of parameters related to the learner category, the educative modality and the educational objective. In [3] a static context model for context-aware e-learning is defined as composed by the personal context, the abstraction context and the situation context. Other researches, in [4], defines a context model for e-learning as composed by seven levels i.e. technological, pedagogical, methodological, organisational, psychological, related to the subject domain and related to the course.

3.1.1 The Learning Context Model

Starting from these researches we have defined our model of learning context by trying to take into account all aspects of the environment surrounding the learner without considering aspects related to the learner itself like profile, previous knowledge, learning preferences, etc. that, as explained in the previous section, are already considered by IWT in the learner model. We then tried to organize these aspects in five levels, each connected with an extensible lists of possible values as reported in table 2.

Level	Description	Values
Educational Context	Identifies the environment where the learning takes place	…
Country		UK, Italy, Spain, Austria, etc.
Educational level		Primary, Secondary, Higher Education, University 1st or 2nd Cycle, Post-Grade, Technical School, Professional Formation, Continuous Formation, Vocational Training, etc.
Course Subject Context	Identifies the subject domain	Literature, Mathematics, Physics, Computer Science, etc.
Methodological Context	Identifies the e-learning form and the level of formality to apply in the context	Self-learning, Asynchronous, Synchronous, Blended/Asynchronous, Blended/Synchronous, Informal learning
Instructional Context	Identifies the instructional strategy to apply in the context	Active learning, Collaborative learning, Direct instruction, Drill and practice, Experiential learning, Game based learning, Inquiry learning, Problem based learning, etc.
Technological Context	Identifies the main technological constraints linked to the context	…
Device Constraints		Screen size, computational power, etc.
Network Constraints		Bandwidth, availability, etc.

Table 2. The learning context model.

The *Educational Context* identifies the environment where the learning takes place and is composed by the country and by an educational level according to the instruction system of the selected country. The *Course Subject Context* identifies the learning subject domain. The *Methodological Context* identifies the e-learning form and the level of formality to apply in the context i.e.:

- self-learning (without presence and without e-communication),
- asynchronous (without presence and with e-communication),
- synchronous (with virtual presence and with e-communication),
- blended/asynchronous (with occasional presence and with e-communication),
- blended/synchronous (with presence and with e-communication),
- informal learning (learning that occurs during daily life or working activities).

The *Instructional Context* identifies the instructional strategy to apply in the context e.g.:

- active learning (approach engaging learners by matching instruction to the learner's interests, understanding, and developmental level);
- collaborative learning (approach that heavily relies on cooperation and sharing among participants, possibly divided in groups);
- direct instruction (an highly teacher-directed approach, usually effective for providing information or developing step-by-step skills);
- drill and practice (approach based on practice by repetition, often used to reinforce grammar and basic mathematical skills);
- experiential learning (inductive, learner centred, activity oriented approach to learning that emphasise the process of learning rather than the product);
- game based learning (approach based on competition, social interaction and some form of prize to award best performing learners);
- inquiry learning (approach in which students solve problems by providing hypotheses and collecting data to provide evidence for or against their hypotheses);
- problem based learning (inductive teaching approach where a teacher poses a real problem and students work cooperatively to solve it).

Eventually the *Technological Context* identifies the main technological constraints linked to the context, divided in device and networking constraints.

3.1.2 The Context Profile

A learning context (also called *context* in the next pages) can be so defined as a feasible configuration of the preceding parameters. Starting from such configuration it is possible for teachers to define a context profile stating feasible teaching strategies that may be applied for each available context.

A **context profile** can be defined as an application *CXP* (*CX* x *Props* × *PropVals*) → [0, 10] where *CX* is a set of available contexts, *Props* is the set of didactical properties and *PropVals* is the set of feasible values for such properties. The table 3 provides didactical properties and associated feasible values as an extension of those defined in table 1. Additional

properties, with respect to table 1, correspond to a selection of fields of the "educational" group of the IEEE 1484.12.1-2002 Standard for Metadata (see [1] for more details).

Properties	Feasible values
didactic method	deductive, inductive, etc.
resource type	text reading, video clip, simulation, virtual experiment, etc.
interactivity level	low, medium, high
context	school, higher education, training
age range	0-5, 6-10, 11-13, 14-18, 19-24, 25+
language	English, Italian, Spanish, German, etc.
semantic density	low, medium, high
interactivity type	active, expositive, mixed
difficulty	easy, medium, difficult

Table 3. Extended didactical properties and feasible values.

For example, the following definition for *CXP* states that, in the context *cx*, learning resources for school should be selected, preferably thought for an age range of 6-10 or 11-13, and an inductive didactic method is more suitable with respect to a deductive one.

- *CXP* (*cx*, "context", "school") = 10;
- *CXP* (*cx*, "age range", "6-10") = 10;
- *CXP* (*cx*, "age range", "11-13") = 9;
- *CXP* (*cx*, "didactic method", "inductive") = 10;
- *CXP* (*cx*, "didactic method", "deductive") = 4.

The definition of a given context can be *derived* by a broader one. In this case, the narrower context inherits all values provided to properties by the broader context but can provide more restrictive values for some of them. If the context *cx'* is derived from the context *cx* we write that $cx' \sqsubseteq cx$. We can then say that if $cx' \sqsubseteq cx$ then $CXP(cx', p, v) \leq CXP(cx, p, v)$ for each $p \in Props$ and $v \in PropVals$ so that $CXP(cx, p, v)$ is defined.

By exploiting the derivation, context profiles can be organized in ***context taxonomies*** i.e. hierarchies of contexts where each context inherits properties and property values from all ancestors.

3.2 Extension of LIA

From the methodological point of view, in order to support the contextualization of the domain model, LIA models and algorithms need the following improvements:

- the definition of an **abstract domain model** able to support the notion of context;
- the definition of a **contextualisation algorithm** able to generate a LIA standard domain model starting from an abstract one;
- a revised version of the **learning lifecycle** including the contextualization algorithm.

The following sub-sections define these elements. In particular two different contextualization algorithms are provided, a standard one (working with a more constrained version of the abstract domain model) and an extended one (working relaxing some of such constraints). The last sub-section focuses on how new components interact with existing ones.

3.2.1 Abstract Domain Model

The abstract domain model describes the knowledge that is object of teaching at a higher level with respect to the standard domain model (see 2.1.1). It supports the notion of context, it allows associating a context to each concept and relation, it allows associating teaching preferences to couples (concept, context) rather then simply to concepts.

Such structure can be formally represented with:

- a **concepts graph** G (C, BT, IRB, SO) representing concepts object of teaching and relations between them (the same defined in 2.1.1);
- a set of **contexts** $CX = \{cx_1, \ldots, cx_n\}$ that is a vocabulary of supported contexts;
- a **concepts labelling relation** $CL \subseteq (C \times CX)$ purposed to label each domain concept with one or more contexts of CX.

The figure 4 shows a graphical representation of a sample abstract domain model in the didactics of artificial intelligence obtained as an extension of the one depicted in figure 1. It defines six concepts and two contexts. Formally speaking we can say that:

- C = {“Logics”, “Formal Systems”, “Propositional Logics”, “First Order Logics”, “Description Logics”, “Outline of Set Theory”};
- CX = {“Computer Science Course at University”, “Mathematics Course at University”}.

It also defines a set of relations between concepts that can be represented as follows:

- BT (“Formal Systems”, “Logics”);
- BT (“Propositional Logics”, “Logics”);
- BT (“First Order Logics”, “Logics”);
- BT (“Description Logics”, “Logics”);
- IRB (“Outline of Set Theory”, “Logics”);

- *IRB* (“Formal Systems”, “Propositional Logics”);
- *IRB* (“Formal Systems”, “First Order Logics”);
- *SO* (“Propositional Logics”, “First Order Logics”);
- *SO* (“First Order Logics”, “Description Logics”).

Finally it labels domain concepts as follows:

- *CL* (“Logics”, “Computer Science Course at University”);
- *CL* (“Logics”, “Mathematics Course at University”);
- *CL* (“Formal Systems”, “Mathematics Course at University”);
- *CL* (“Propositional Logics”, “Computer Science Course at University”);
- *CL* (“Propositional Logics”, “Mathematics Course at University”);
- *CL* (“First Order Logics”, “Computer Science Course at University”);
- *CL* (“First Order Logics”, “Mathematics Course at University”);
- *CL* (“Description Logics”, “Computer Science Course at University”);
- *CL* (“Outline of Set Theory”, “Computer Science Course at University”).

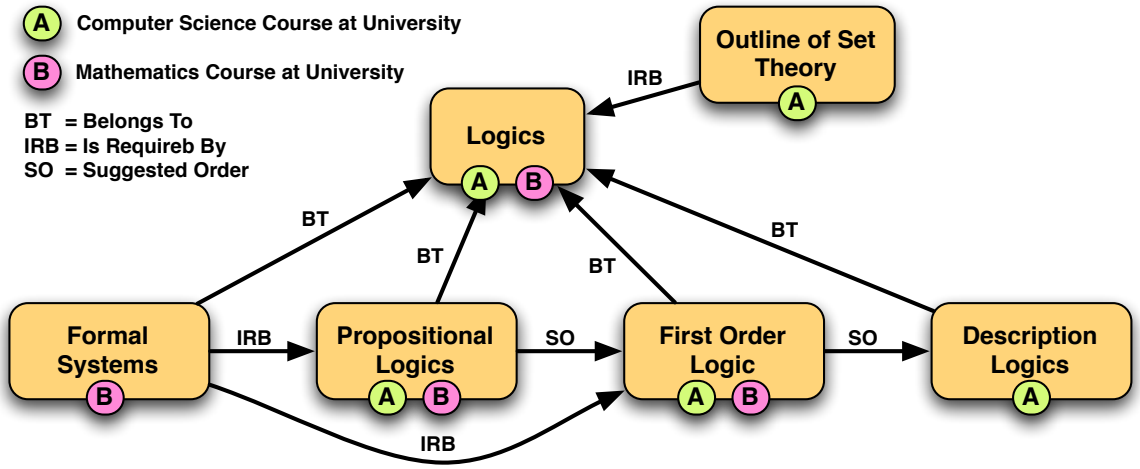

Figure 4. Graphical representation of a sample abstract domain model.

Informally speaking this domain model specifies that to understand “logics” means to understand “formal systems”, “propositional logic”, “first order logic” and “description logics” where “formal systems” is only required in university courses about mathematics while the concept of “description logics” is only required in university courses about computer science. Moreover in this latter context (computer science), before approaching any of these topics it is necessary to have an “outline of set theory” first.

Moreover, in any case, “formal systems” must be taught before both “propositional logics” and “first order logic” while it is desirable (but not compulsory) to teach “propositional logics” before “first order logic” and “first order logic” before “description logics”.

To be used by LIA algorithms, the concept graph *G* must be acyclic and, to be coherent, the following property must be verified by editing software: if concept *a* is related to concept *b* with any relation (*BT*, *IRB* or *SO*) then *CL* (*a*, *cx*) → CL (*b*, *cx*) for any *cx* ∈ *CX* i.e. the fact that *a* applies to a given context implies that *b* applies to the same context too (this constraint can be partially relaxed if the extended contextualization algorithm is used in place of the standard one, as described in 3.2.3).

The abstract domain model also includes a **context profile** *CXP* as defined in section 3.1, stating feasible teaching strategies that may be applied for any available context in *CX*. A set of additional **teaching preferences** can be also added to define exceptions to the context profile i.e. to specify feasible teaching strategies that may be applied for a given concept in a specific context (so excepting general rules included in the context profile).

Teaching preferences are an application *TP* (*C* × *CX* × *Props* × *PropVals*) → [0, 10] where *Props* is the set of didactical properties and *PropVals* is the set of feasible values for such properties. The table 3 provides examples of didactical property and associated feasible values. It is worth noting that *TP* is defined only for couples of *Props* and *PropVals* elements belonging to the same row in table 3.

As an example, the following definition for *CXP* and *TP* states that the preferred didactic method to be used to explain concepts of the logics domain in the context of a computer science course at university is the deductive one. Despite that, to teach the concept of first order logics, an inductive method should be preferred.

- *CXP* (“Computer Science Course at University”, “didactic method”, “deductive”) = 7;
- *CXP* (“Computer Science Course at University”, “didactic method”, “inductive”) = 4; (10)
- *TP* (“First Order Logics”, “Computer Science Course at University”, “didactic method”, “inductive”) = 10;

3.2.2 Contextualization Algorithm

The contextualisation algorithm is purposed to the generation of a standard domain model (as defined in 2.1.1) composed by a conceptual graph *G’* = (*C’*, *BT’*, *IRB’*, *SO’*) and a set of teaching preferences *TP’* starting from:

- an abstract domain model composed by a conceptual graph *G* = (*C*, *BT*, *IRB*, *SO*), a set of contexts *CX*, a context profile *CXP*, a concepts labelling relation *CL*, a set of teaching preferences *TP*;
- a context *cx* ∈ *CX*.

The algorithm build *G'* components in this way:

- *C'* = {*c* ∈ *C* | (*c*, *cx*) ∈ *CL*} i.e. it is obtained by considering only concepts of *C* that apply in the context *cx*;
- *BT'* = {*(a, b)* ∈ *BT* | *CL* (*a*, *cx*) ∧ *CL* (*b*, *cx*)} i.e. it is obtained by considering only arcs of the *BT* type connecting two concepts that apply in the context *cx*;
- *IRB'* = {*(a, b)* ∈ *IRB* | *CL* (*a*, *cx*) ∧ *CL* (*b*, *cx*)} i.e. it is obtained by considering only arcs of the *IRB* type connecting two concepts that apply in the context *cx*;
- *SO'* = {*(a, b)* ∈ *SO* | *CL* (*a*, *cx*) ∧ *CL* (*b*, *cx*)} i.e. it is obtained by considering only arcs of the *SO* type connecting two concepts that apply in the context *cx*;

Then *TP'* is composed in this way: *TP'* (*c*, *p*, *pv*) = *CXP* (*cx*, *p*, *pv*) for any *c* ∈ *C*, *p* ∈ *Props* and *pv* ∈ *PropVals* i.e. the teaching preferences for each concept are those defined by the context profile. Then it is updated in this way: *TP'* (*c*, *p*, *pv*) = *TP* (*c*, *cx*, *p*, *pv*) for any defined *TP* i.e. if teaching preferences are explicitly defined for a given concept in the context *cx*, then they override default teaching preferences stated in the context model.

As an example, the figure 5 shows the two domain models that can be obtained through the contextualization algorithm, starting from the conceptual graph in figure 4, by selecting the context “Mathematics Course at University” (left) or “Computer Science Course at University” (right).

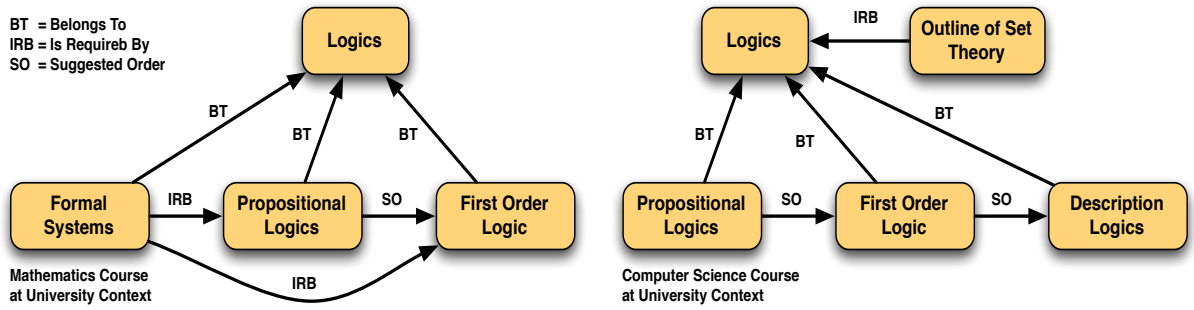

Figure 5. Two samples of contextualized domain models.

Formally speaking the first graph (mathematics) is defined as follows:

- *C* = {“Logics”, “Formal Systems”, “Propositional Logics”, “First Order Logics”};
- *BT* (“Formal Systems”, “Logics”);
- *BT* (“Propositional Logics”, “Logics”);
- *BT* (“First Order Logics”, “Logics”);
- *IRB* (“Formal Systems”, “Propositional Logics”);
- *IRB* (“Formal Systems”, “First Order Logics”);

- *SO* (“Propositional Logics”, “First Order Logics”);

While the second graph (computer science) is defined as follows:

- *C* = {“Logics”, “Propositional Logics”, “First Order Logics”, “Description Logics”, “Outline of Set Theory”};
- *BT* (“Propositional Logics”, “Logics”);
- *BT* (“First Order Logics”, “Logics”);
- *BT* (“Description Logics”, “Logics”);
- *IRB* (“Outline of Set Theory”, “Logics”);
- *SO* (“Propositional Logics”, “First Order Logics”);
- *SO* (“First Order Logics”, “Description Logics”).

Moreover, starting from context profile and teaching preferences defined in (10), the following preferences will be automatically generated by the contextualization algorithm for the computer science context:

- *TP* (“Logics”, “didactic method”, “deductive”) = 7;
- *TP* (“Logics”, “didactic method”, “inductive”) = 4;
- *TP* (“Propositional Logics”, “didactic method”, “deductive”) = 7;
- *TP* (“Propositional Logics”, “didactic method”, “inductive”) = 4;
- *TP* (“First Order Logics”, “didactic method”, “deductive”) = 7;
- *TP* (“First Order Logics”, “didactic method”, “inductive”) = 10;
- *TP* (“Description Logics”, “didactic method”, “deductive”) = 7;
- *TP* (“Description Logics”, “didactic method”, “inductive”) = 4;
- *TP* (“Outline of Set Theory”, “didactic method”, “deductive”) = 7;
- *TP* (“Outline of Set Theory”, “didactic method”, “inductive”) = 4;

3.2.3 Extended Contextualization Algorithm

As reported in the section 3.2.1, the contextualization algorithm works well if the following property is verified by the editing software: if the concept *a* is related to concept *b* with any relation (*BT*, *IRB* or *SO*) then *CL* (*a*, *cx*) → CL (*b*, *cx*) for any *cx* ∈ *CX* i.e. the fact that *a* applies to a given context, implies that *b* applies to the same context too. This means, as an example, that the relations depicted in the next image are not allowable.

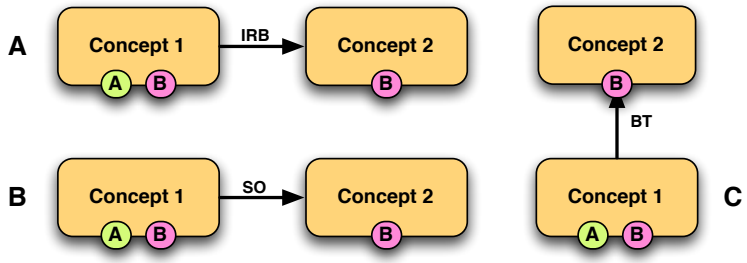

Figure 6. Samples of not allowable relations.

This limitation is introduced to avoid situations in which the selection of a specific context can break important concept-to-concept relations. As an example, when the *context A* is selected in the incorrect abstract domain model depicted in figure 7, the standard contextualization algorithm would break *BT* relations between the *concept 1* and the sub-concepts of *concept 2* (because the *concept 2* belongs to another context) as well as *IRB* relations between *concept 7* and *concept 9* and *SO* relations between *concept 4* and *concept 6*.

So, by applying the standard contextualization algorithm, a unit of learning targeting *concept 1* in the *context A*, for the shown abstract domain model, will lead to a meaningless course covering only *concept 7* and *concept 9* without any ordering.

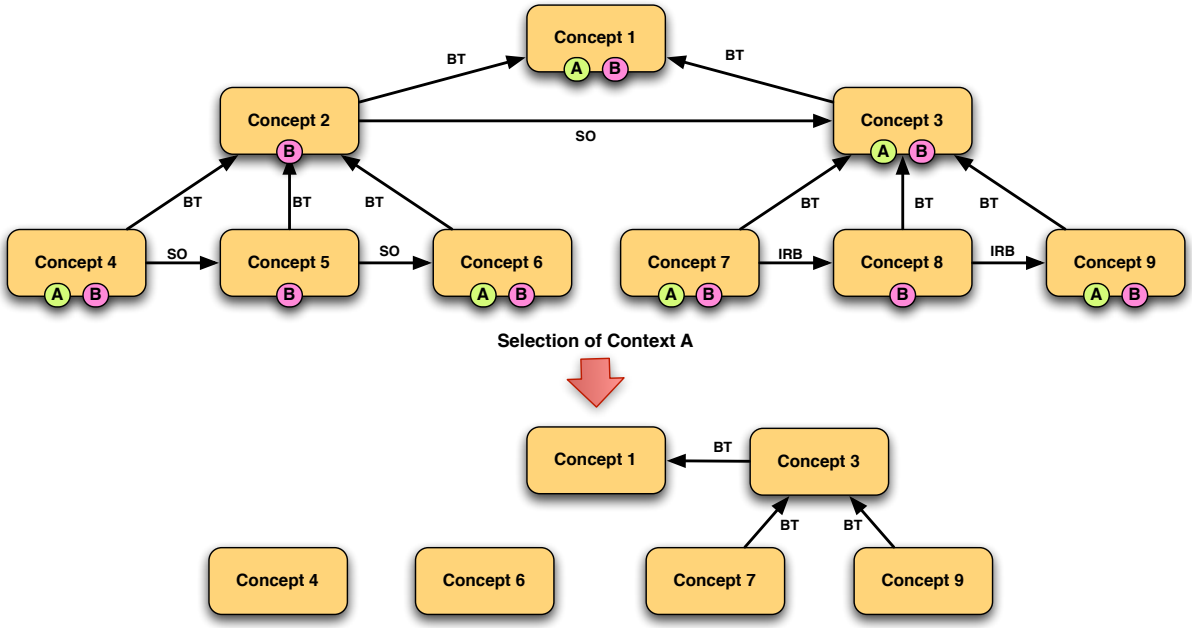

Figure 7. Results of the standard contextualization in presence of incorrect relations.

We want now to relax the reported constraint for *IRB* and *SO* and to subsequently extend the contextualization algorithm in order to let it build meaningful contextualized domain models also in presence of relations of the kind reported in figure 6 (only letters A and B). To do that,

we introduce a set of *relation propagation rules* for *IRB* and *SO* and an extended version of the contextualization algorithm able to apply such rules.

Formally speaking the *extended contextualization algorithm* works with an abstract domain model provided that the following property is verified by the editing software: if $(a, b) \in BT$ then $CL(a, cx) \rightarrow CL(b, cx)$ for any $cx \in CX$.

The *extended contextualization algorithm* is purposed to generate a standard domain model composed of a conceptual graph $G' = (C', BT', IRB', SO')$ and a set of teaching preferences TP' starting from:

- an abstract domain model composed by a conceptual graph $G = (C, BT, IRB, SO)$, a set of contexts CX, a context profile CXP, a concepts labelling relation CL, a set of teaching preferences TP;
- a context $cx \in CX$.

The building process of TP' is already defined in 3.2.2 while the components of G' are calculated with the following algorithm. First of all G' components are initialised as equal to those of G (step 1). Then, for each concept c that does not apply to the context cx (steps 2-16) *IRB* and *SO* relations are propagated over the concept c (steps 3-9), then the concept c is removed together with any incoming and outgoing relation (steps 10-16).

1. $C' = C$; $BT' = BT$; $IRB' = IRB$; $SO' = SO$
2. for each $c \in C'$ so that $(c, cx) \notin CL$ { // for each concept to be removed
3. for each $a \in C'$ so that $(a, c) \in IRB'$ { // propagate *IRB* relations
4. for each $b \in C'$ so that $(c, b) \in IRB'$
5. if $(a, b) \notin IRB'$ then add (a, b) to IRB'
6. for each $b \in C'$ so that $(c, b) \in SO'$ // as well as SO relations
a. if $(a, b) \notin SO'$ then add (a, b) to SO' }
7. for each $a \in C'$ so that $(a, c) \in SO'$
8. for each $b \in C'$ so that $((c, b) \in IRB' \vee (c, b) \in SO')$
9. if $(a, b) \notin SO'$ then add (a, b) to SO'
10. for each $(a, b) \in IRB'$ so that $a = c \vee b = c$ // then remove any relation on the concept
11. remove (a, b) from IRB'
12. for each $(a, b) \in SO'$ so that $a = c \vee b = c$
13. remove (a, b) from SO'
14. for each $(a, b) \in BT'$ so that $a = c \vee b = c$
15. remove (a, b) from BT'
16. remove c from C' } // and the concept itself

For relations' propagation on the concept c, the following *rules* are applied:

- for each incoming *IRB* relation from any concept a and for each outgoing *IRB* relation to any concept b, a *IRB* relation from a to b is added (figure 8, rule A);

- for each incoming *IRB* relation from any concept *a* and for each outgoing *SO* relation to any concept *b*, a *SO* relation from *a* to *b* is added (figure 8, rule B);
- for each incoming *SO* relation from any concept *a* and for each outgoing *IRB* relation to any concept *b*, a *SO* relation from *a* to *b* is added (figure 8, rule C);
- for each incoming *SO* relation from any concept *a* and for each outgoing *SO* relation to any concept *b*, a *SO* relation from *a* to *b* is added (figure 8, rule D).

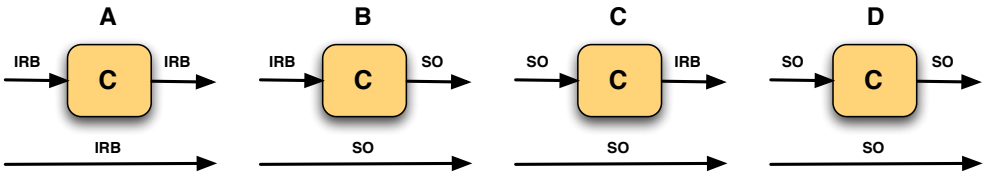

Figure 8. The relation propagation rules.

The number relations introduced by the propagation rules is a combination of the number of existing relations i.e. for a concept *c* having *n* incoming relations and *m* outgoing relations (of *IRB* and *SO* types), the number of introduced relation is $n \times m$ while the number of removed relations is $n + m$.

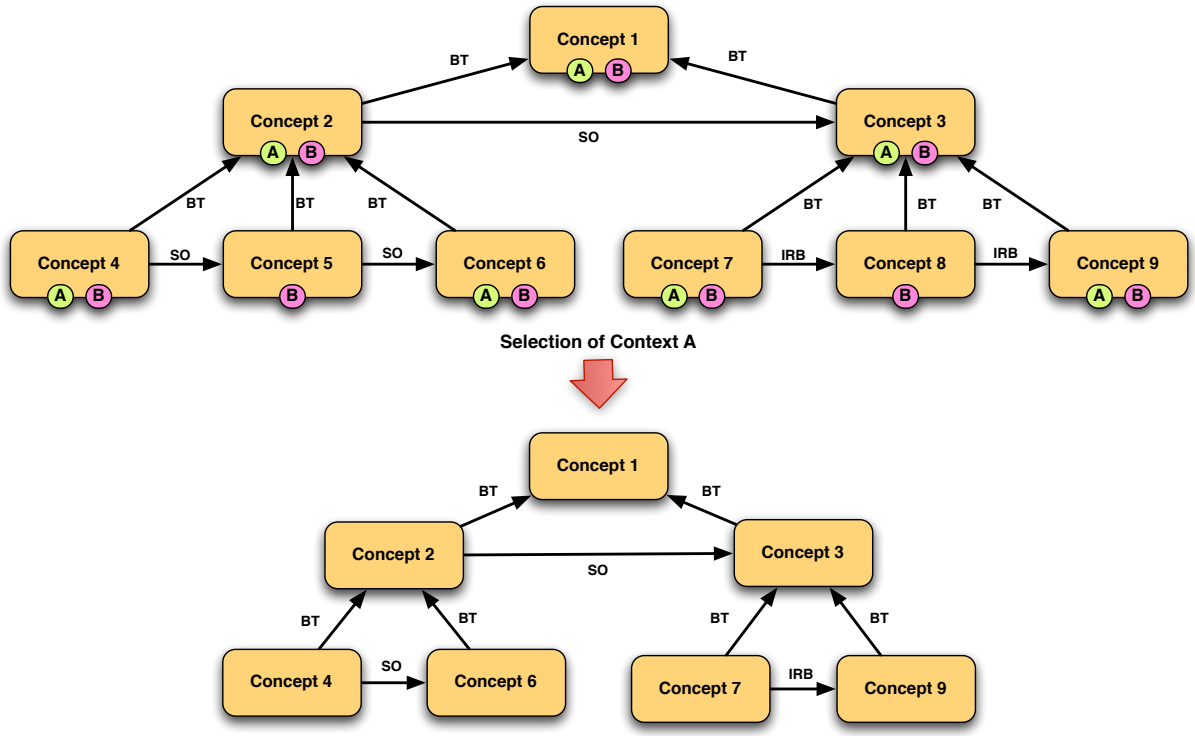

Figure 9. Results of the extended contextualization algorithm.

The figure 9 shows the application of the extended contextualization algorithm on an abstract domain model (obtained from that figure 7) that does not respect the constraint required by the standard contextualization algorithm while respecting that of the extended one. As it can be seen, the contextualized model obtained by selecting *context A* is still correct and capable of generating meaningful units of learning.

3.2.4 Updated Learning Life-Cycle

The LIA learning life cycle is described in section 2.2 (figure 2). Figure 10 shows how it should be modified in order to support domain model contextualization.

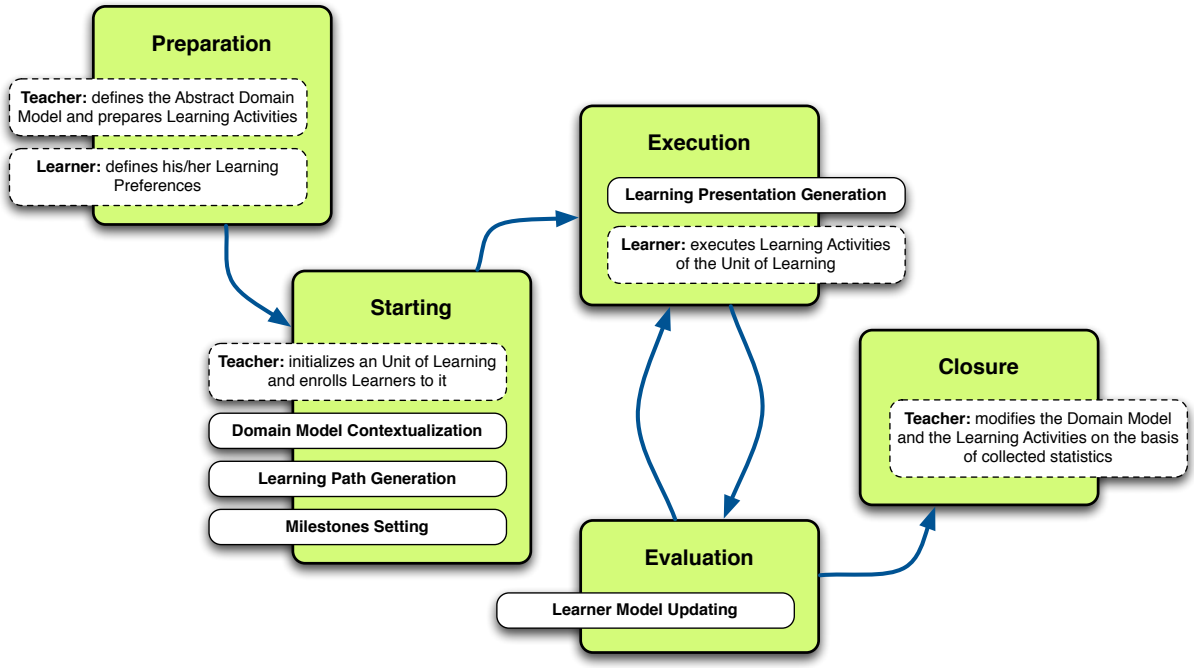

Figure 10. The updated Learning Life Cycle.

Differently from the one shown in figure 2, the new process foresees:

- the definition of an abstract domain model rather then a domain model by the teacher during the preparation phase;
- a different initialization of the unit of learning by the teacher in the starting phase i.e. the teacher selects not only target concepts and cost constraints but also the target context among those available;
- the domain contextualization is inserted as the second step in the starting phase i.e. the abstract domain model defined in the preparation phase is contextualized (so transformed in a standard domain model) before using it by subsequent algorithms to generate the learning path.

4 Technological Perspective

The IWT logical architecture is divided in four main layers as shown in figure 11. The first layer at the bottom of the stack is the **IWT Framework** used by developers to design and implement core services, application services and learning applications. The second layer is composed by **Core Services** providing basic IWT features like resources management, ontology storing, user authentication, content storing, metadata, role and membership management, learning customisation (i.e. LIA), logging, profiling etc. Core services are used by application services and learning applications.

Application Services are used as building blocks to compose e-learning applications for specific domains. They include document management, conferencing, authoring, learning management, learning content management, ontology management, communication and collaboration, process management and information search services. On the top of the stack, **Learning Applications** covering specific learning scenarios obtained as integration of application services are built.

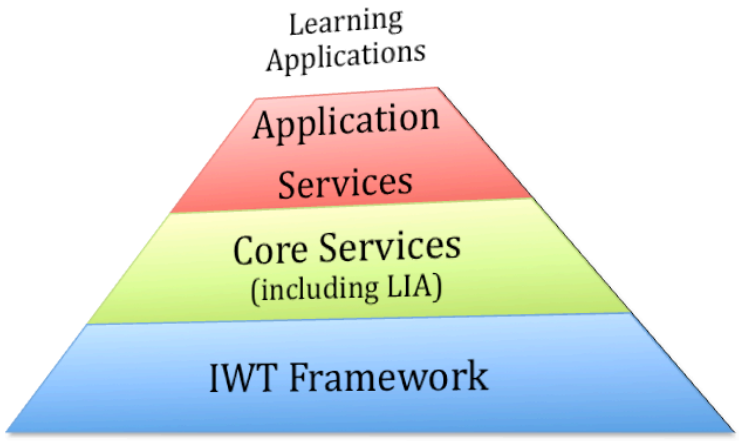

Figure 11. The four layers composing the IWT Logical Architecture.

Such architecture is modular enough to allow the deployment of solutions capable to cover application scenarios of different complexity and for different domains by composing service building blocks. ALICE in this context is a Learning Application based on existing and new IWT components to be integrated at different levels of the architecture. Next paragraphs focuses on extensions needed to such architecture in order to obtain context management and knowledge model contextualization functions.

4.1 Extensions Needed to IWT

From the technological point of view, the integration in the reference platform IWT of context management and knowledge model contextualization functions, as specified in [7] (section 5.2), requires a set of additional/improved components as depicted in figure 12 (additional components are in grey while improved ones are in black). The figure also contextualizes such components with respect to the IWT architecture.

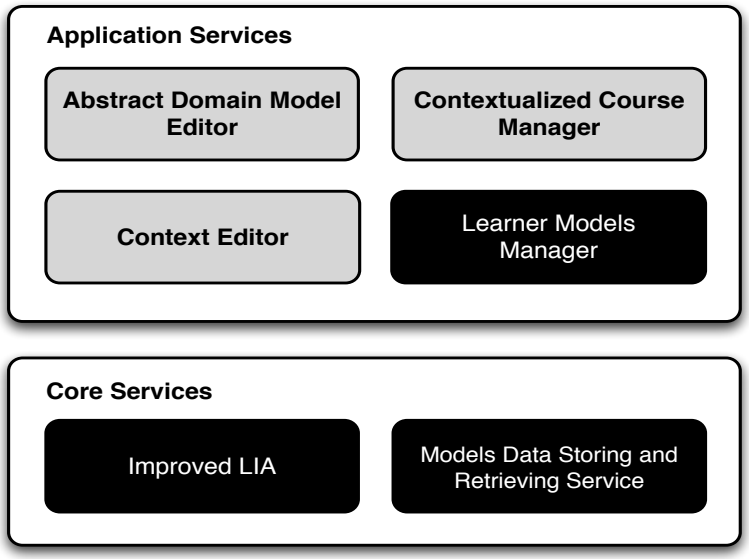

Figure 12. Additional IWT components foreseen.

In the following we briefly describe needed components and their impact on the architecture.

- **Improved LIA**. The new version of LIA will be able to apply the updated learning life cycle defined in 3.2.4 as well as the standard one (defined in 2.2) according to input parameters. In this sense it can also deal with abstract domain models as defined in 3.2.1 and will implement the extended contextualisation algorithm defined in 3.2.3.
- **Models Data Storing and Retrieving Service**. We foresee the extension of existing data structures to include the set of contexts *CX* and to maintain the additional information required by the abstract domain model like the concepts labelling relation *CL,* the context profile *CXP* and a revised version of teaching preferences *TP* as defined in 3.2.1. Related storing and retrieval services will be modified accordingly.
- **Contextualized Course Manager**. This is a component for managing contextualised courses that will be developed by extending the one already existing for personalized courses. The component will allow teachers to select target concepts on an abstract domain model rather than on a standard one and to select the correct context when the course is assigned to a given student. In this way the updated learning life cycle will be used in place of the standard one.

- **Abstract Domain Model Editor**. The editor will consent to visually build an abstract domain model including concepts (to select a dictionary, add and remove a concept), relations between concepts (to add and remove a relation), contexts (to select, add and remove a context), context profiles (to modify teaching preference for each selected context), context labels linked to concepts (to link and unlink a context for each concept), and teaching preferences linked to context labels (to modify teaching preference for each context connected to a given concept).
- **Context Editor**. The editor will allow knowledge managers to create and modify the list *CX* of contexts managed by the system. Contexts can be added, modified and removed from the list. Only unambiguous context names are provided through this editor while each teacher can associate different profiles to defined contexts through the abstract domain model editor.
- **Learner Models Manager**. This is an already existing component allowing teachers to modify learner models. It will be improved in order to allow teachers to associate one or more contexts to a learner among those available.

The next paragraphs will give further details on the Abstract Domain Model Editor and on the Contextualized Course Manager.

4.2 The Abstract Domain Model Editor

The figure 13 shows a mock-up of the abstract domain model editor. Available contexts are listed in the left side of the window. The user can add or remove contexts by exploiting menu items over the list. The list of available contexts is defined outside the tool through the context editor. When the *add* button is pressed, the list of available contexts is presented and the user can select one of them. A colour is associated to each context.

Domain concepts are listed under contexts. The user can add or remove concepts by exploiting menu items over the list. The list of available concepts is defined outside the tool through the already existing IWT dictionary editor. When the *add* button is pressed, the list of available dictionaries is presented and the user can select one of them. Once a dictionary is selected the user can chose to add all dictionary concepts or just a subset. Available relations are listed under the list of concepts.

The user can drag a concept from the list and drop it in the workspace on the right. Dropped concepts are represented as rounded boxes with the name of the concept and a sequence of coloured circles, one for each available context. Concepts can be moved and connected with available relations (a relation must be selected and a line must be traced between the two concepts to connect).

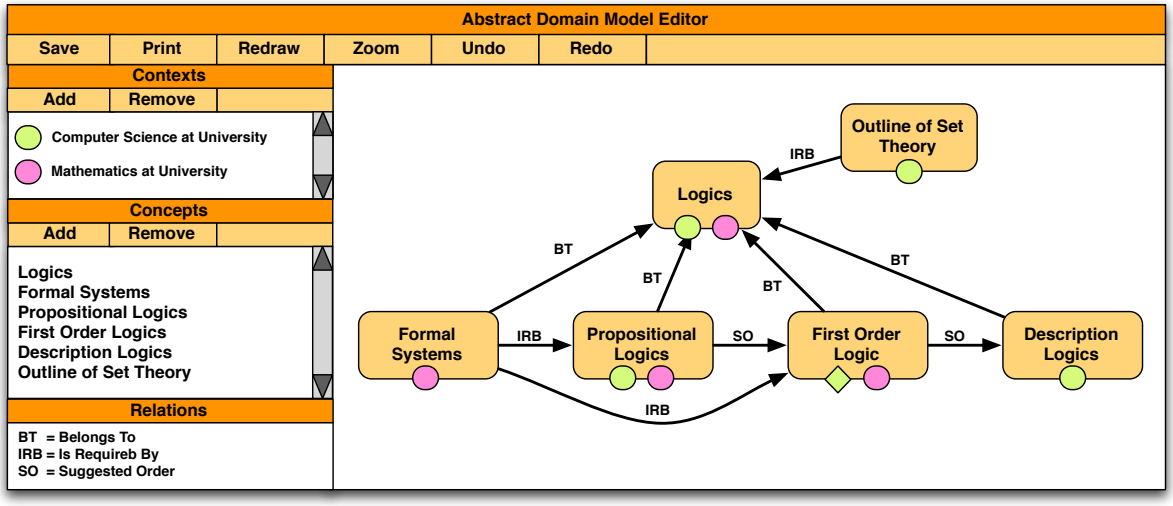

Figure 13. Mock-up of the Abstract Domain Model Editor.

By double clicking on a context, the context profile is shown and connected teaching preferences can be settled. In this way it is also possible to rename a context i.e. to select a different context from the list of available contexts. By clicking on a coloured circle inside a concept, teaching preferences for the concept (in the corresponding context) can be modified. Such preferences are initially settled according to the context profile but may be changed. Once preferences for a concept in a context are changed the corresponding context label become a diamond to emphasise the modification.

Context labels can be removed from concepts by right clicking on them and selecting the “remove” menu item from the contextual menu. Removing a context label from a concept means that the concept is not active in that context. Once removed a context may be added again by dragging it from the list and dropping it on the interested concept. By right clicking on a diamond shaped context label it is also possible to select the “remove teaching preferences” button. By doing that teaching preferences connected to the concept (in the corresponding context) are modified and the diamond becomes a circle.

A designed model can be saved, printed, redrawn (to improve readability) and zoomed (zoom percentage may be settled). Undo and redo buttons will be also provided to remove and redo the last performed action. During the design, the editor monitors that the graph is acyclic and that the constraint reported in 3.2.3 is respected.

4.3 The Contextualized Course Manager

This component allows the creation and the delivery of personalized and customized units of leaning. The creation of a course works through the following steps:

1. The teacher selects a standard or an abstract domain model among those available.
2. If he selects an abstract domain model then he can chose:

a. to pre-select a context – in this case the list of available contexts is presented and he selects the one he prefers;

b. to not pre-select a context – in this case the course will be contextualized on the fly, basing on the context connected to the enrolled student.

3. The domain model is displayed:

   a. in case of selection (in step 1) of a standard domain model, the system displays the model itself;

   b. in case of selection (in step 1) of an abstract domain model and of pre-selection (in step 2) of a context, the system displays the abstract model contextualized for the selected context;

   c. in case of selection (in step 1) of an abstract domain model without pre-selection (in step 2) of a context, the system displays the abstract model itself.

4. The teacher selects one or more target concepts.

5. The teacher selects all IWT course parameters.

6. In case of selection (in step 1) of an abstract domain model without pre-selection (in step 2) of a context, the system asks the teacher to specify:

   a. a default context among those supported by the abstract domain model to be used when the list of contexts associated to a learner do not overlap the list of contexts associated to the domain model;

   b. an ordered list of contexts among those supported by the abstract domain model to be used when the domain model supports more then one contexts among those associated to a given learner – in this case the context with the higher ranking in the list is selected.

During the delivery step for a given learner, the course manager will act in this way:

1. If the course starts from an abstract domain model then the model is contextualized with respect to the context(s) associated with the learner.

2. The course is personalized according to learner knowledge state and preferences.

3. The learner goes through course activities.

4. The learner model is updated after each assessment, then the course is personalized with as usual IWT personalized courses.

5 Related Work

This research falls in the field of Intelligent Tutoring Systems (ITS) and deals mainly with personalized and context aware e-learning.

Personalized e-learning is defined in [9] as an educational model that is tailored to the individual learner's needs and interests. Personalized learning can be used for developing the individual learning programs and also engage these learners into the learning process so that learner's learning potentials and success can be optimized. Personalized e-learning is not restricted by time, place and learner's other requirements. It is mostly focusing on learner's preferences and current state of a learner to provide the learning content correctly. Differently from context-aware e-learning, it does not consider a learner's situation.

Context Aware e-Learning, on the other side, provides a learner with highly customized learning content [10]. The customization of content is made by selecting or filtering learning resources in order to make the e-Learning content more relevant and suitable for the learner in his/her situation. The filtering process is done by considering several parameters like the learner's personal information, learning style preferred by him, learner's situation, etc. These parameters constitute for the learner's context.

Several definitions of context are available in the literature. In the sub-section 5.1 we analyse and compare such definitions in order to define best parameters to be applied in our case.

5.1 Related Research about Learning Context

According to [11], context is defined as "that which surrounds, and gives some meaning to, something else". In [12] instead it is defined as "any information that is used to characterize the situation of an entity". Moreover, according to the ubiquitous computing, "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." [10].

The IMS Learning Resource Meta-Data Information Model Specification [2] defines a learning context as "the typical learning environment where use of learning object is intended to take place". It also proposes a list of feasible contexts for a learning object i.e. primary education, secondary education, higher education, university first cycle, university second cycle, university post-grade, technical school first cycle, technical school second cycle, professional formation, continuous formation, vocational training. The last version of the specification (1.3) reduces the list of feasible values to school, higher education, training and provides a free text field to provide further details.

Other authors [6] define the learning context as the learner environment and talk about three main environments: the external environment (classroom, working space, in-person coaches, etc.), the internal environment (previous beliefs, thoughts, hopes, etc.) and the digital

environment. According to [4], instead, a context model for e-learning is composed by seven levels (i.e. technological, pedagogical, methodological, organisational, psychological, related to the subject domain and related to the course), each one characterised by several aspects and variants as reported in table 4.

Level	Aspect	Variants
Technological	Hardware	Desktop computer, handheld, mobile phone, interactive board.
	Networking	Availability, bandwidth, stability, connection/transfer price.
	Software	Virtual learning environment for delivering DSC, standard software (such as MS PowerPoint).
	User interface	Textual, graphical, web-based, 3D.
Pedagogical	Learning theory	Behaviourism (based on training with impact to behaviour), Cognitivism (based on analysis and change of processes of thinking), Constructivism (oriented to “guiding” students rather than “teaching”).
	Instructional strategy	Adult learning (based on principles, defining how adults acquire new knowledge and skills), Active learning (emphasizes active participation of students, instructions must match learner's interests), Blended learning (covers different strategies), Collaborative learning (based on collaborative work in groups), Direct instruction (lecturer-centred instruction which includes lectures, presentations, and receiving rapid feedback), Drill (based on practice by repetition), Problem based learning (based on solving authentic problems).
E-Learning methodology	Delivery models (time aspect)	Synchronous (learning activities take place at specific times), Asynchronous (learners can access study material and activities at any time), Self-study (independent according to time aspect, there is no instructor).
	Delivery models (the main figure)	Self-study (based on students’ workload), Instructor-led (based on instructions, provided by lecturer), Instructor-facilitated (based on student learning, guided and supported by instructor activities).
	Delivery models (dependence on content)	Content + support (content is predefined and structured, support is separated from content), Wrap Around (content is partially predefined, learners have more freedom and responsibility), Integrated model (content is mostly developed or gathered during learning process by learners).

Level	Aspect	Variants
	E-learning forms	With physical presence and without e-communication (face-to-face), Without presence and without e-communication (self-learning), Without presence and with e-communication (asynchronous), With virtual presence and with e-communication (synchronous), With occasional presence and with e-communication (blended/hybrid-asynchronous), With presence and with e-communication (blended/hybrid-synchronous).
	Interactivity level	Passive (not responding to the actions of learner), Reactive (responsive only to the last action of user), Interactive (responsive to the some previous actions).
Organisational	Studies type (formality)	Formal learning (occurs in an organised environment, and leads to accredited certification), Informal learning (occurs during daily life activities).
Psychological	Motivation	Low, Medium, High.
	Preferred senses	Visual learner, Auditory learner, Kinaesthetic learner.
	Learning style	Activist (prefers hands-on case studies and simulations), Reflector (prefers lectures and then brainstorming), Theorist (prefers conceptual readings), Pragmatist (prefers field work in the workplace).
	Myers-Brigg types	Attitudes (extraversion / introversion), Lifestyle (judgment / perception), Functions (sensing / intuition and thinking / feeling).
Subject domain	Level of Structure	Well structured, Ill structured.
	Didactics	Didactical requirements and methodologies for specific subject areas are different.
Course	The main aims of learning	To acquire new knowledge; To acquire practical abilities; To change attitudes, viewpoints, feelings; To acquire transferable abilities (critical thinking, etc.).
	Previous experience	Differences existing in matching previous knowledge and abilities of student and the required knowledge and abilities in particular course.

Table 4. Levels of learning contextualisation according to [4].

In [3] a “static” context model for context-aware e-learning has been defined basing on the analysis of the relevant literature about the topic. The static nature of the context is due to the fact that only parameters that do not change within the entire e-Learning course structure

have been considered. The defined model is composed by several context parameters divided in several sub context parameters. In [10] the same authors have systematized and aggregated such parameters into the following sub-contexts:

- *Profile Context* giving information about learner's personal information, personality type and learner's level of expertise;
- *Preference Context* containing information about learner's approach or preferences and learner's intention and learning style;
- *Infrastructure Context* describing the information about learner's situation, network and device used by the learner;
- *Learning Context* describing the information about the learning pace, learning state and the comprehension level of the learner.

Table 5 summarizes defined sub-contexts, parameters and sub-parameters.

Sub-Context	Parameter	Sub-Parameters
Profile Context	Learner Profile	Name, ID, DOB, Gender, Address, Email-id, Phone Number, Technologies Known, Knowledge Level, OS Experience, Internet Usage.
	Level of Expertise	Beginner, Practitioner, Expert.
Preference Context	Learning Style	Video, Audio, Text, Animation, Slides.
	Learning Preference	Conceptual, Example-Oriented, Case Study, Simulation, Demonstration.
	Learning Intention	Research, Survey/Overview, Quick Reference, Basic Introduction, Project, Assignment, Seminar.
Infrastructure Context	Learner Situation	Private, Public, Driving.
	QoLS	Functional Requirement, Non-Functional Requirement.
	Network	Wired, Wireless.
	Device	Mobile, PC, Laptop, PDA.
Learning Context	Learning Pace	Slow, Medium, Fast.
	Learning State	Studied, To be Studied, To be Revised.
	Comprehension Level	Not Understood, Understood a Little, Understood Well, Understood Completely.

Table 5. Parameters of the static context model defined in [10].

According to [5] the learning context (in e-learning settings) describes a class of learners within a technological infrastructure with a set of parameters related to the learner category, the educative modality end the educational objective.

- The *Learner Category* is composed by the average cognitive state (concepts already acquired by the class) and by the social context (learner cultural background, which is established by the title of study and the country where it has been acquired).
- The *Educative Modality* is composed by the learning experience period (duration of the whole learning experience) and by the interaction modality (typologies of learning experience to choose e.g. distance or blended).
- The *Educative Objective* is composed by the educational context (the target kind of instruction e.g. high school, university, PhD, training, etc.), by the motivation for the education (kind of expertise the learner is interested in e.g. theoretical knowledge, applied knowledge, know-how, etc.) and by the deepening level (the depth level of the study to achieve).

Basing on the context description, the system proposed in [5] is able to automatically select suitable learning objects by matching context parameters and fields of the IMS metadata [2] connected to learning objects.

Starting from these researches, we provided, in section 3.1, a custom definition of a learning context composed by six parts and several parameters and a context profile stating feasible teaching strategies that may be applied for available contexts. Moreover an abstract domain model able to support the notion of context was defined in section 3.2 as well as an algorithm able to generate a contextualised domain model starting from an abstract one.

5.2 Related Research about Course Sequencing

Learning path and presentation generation algorithms available in LIA (described in section 2), are able to generate personalized learning experiences starting from a contextualized learning model. It is worth noting that this latter part of the research also falls in the field of Intelligent Tutoring Systems and deals with course sequencing techniques [13].

Several approaches to course sequencing have been defined and exploited by experimental ITS. The majority of such systems, like [14] and [15], only deals with task sequencing i.e. they are able to generate sequences of problems or questions to improve the effectiveness and the efficiency of the evaluation process. Other systems like [16] and [17], only deal with lessons intended as big learning objects fully explaining and assessing the knowledge about a given topic. Only some system, like [18] and [19], is able to generate sequences composed by different kinds of learning objects like presentations, tests, examples, etc. LIA falls in this latter category given its ability to sequence several kind of learning resources.

Models and methodologies behind LIA integrate and extend results coming from researches on knowledge representation and ITS [20]. A first prototype of a learning intelligent system, based on conceptual graphs and software agents was proposed in [21]. Defined models and

methodologies were improved introducing description logic as well as planning and machine learning techniques leading to a new prototype described in [22]. Further improvements have been made to models and algorithms leading to the last version of the system described in [1] that, currently, constitutes the reference paper for LIA within ALICE.

Other adaptive e-learning systems that deal with course sequencing exist in literature. The **AHA!** (Adaptive Hypermedia for All) system [23] employs adaptive techniques like fragment and link hiding for course delivery. Knowledge domain is modeled using concepts and actual content is kept directly in pages. Individual pages are stored as XML files with information about concepts, HTML fragments and adaptation rules. AHA! is used in some universities in Belgium and the Netherlands. It does not include contextualization features.

The **INSPIRE** (INtelligent System for Personalized Instruction in a Remote Environment) system [24] offers adaptive link annotation, adaptive link sorting, and adaptive curriculum sequencing techniques to guide learner through a learning space using a path respecting his learning style (based on Kolb's theory and determined through an entrance questionnaire). The system supports the definition of learning goals on a concept graph. It does not include contextualization features.

InterBook [25] is a tool for creation and presentation of adaptive electronic textbooks. It offers adaptive link annotation and direct guidance as well as automatically generated glossaries and indexes. Domain structure is modeled using a concept network, where each concept represents an elementary part of knowledge. Content units can have prerequisite concepts and outcome concepts. The system monitors student's progress and keeps track of how much he knows about each concept. It uses this information to recommend pages with all prerequisites known.

ActiveMath [26] is an adaptive e-Learning system specifically thought to teach mathematical topics but usable also in other contexts that has evolved from a prototype to a full-blown platform used by an international community centered in Germany. The course generator uses information about learning resources, the learner and his/her learning goals to generate an adapted sequence of learning objects that supports the learner in achieving his goals. It does not include contextualization features.

The **NetCoach** system [27] simplifies the course authoring process by offering the possibility to insert content and define relations between documents using concept networks. Indexes, glossaries and data for adaptivity are generated automatically. Adaptive annotation of links and curriculum sequencing features are supported. It does not include contextualization features. The system is now available as commercial learning solution from Orbis.

ACGs (Adaptive Course Generation System) [28] is an adaptive course delivery prototype designed and implemented by the Vietnam National University of Hanoi. It is an agent-based system providing adaptive curriculum sequencing features basing of learner profiles. It does not include contextualization features.

INES (INtelligent Educational System) [29] is a prototypal adaptive learning system from the University of Vigo, Spain. It is based on a multi-agent engine and exploits ontologies to model the knowledge about the domain. It also uses natural language processing technology to communicate with students. Basing on student progresses, the system is able to suggest to each learner specific tasks to perform in order to achieve his/her particular learning objectives. It does not include contextualization features.

The **LIP** (Learning In Process) system [30] was built in the homonymous EC funded project aimed at providing personalized and contextualized learning experiences addressing the needs of knowledge intensive organizations. LIP is able to capture different dimensions related to the user context, including user profile (demographic characteristics, preferences, etc.), technical constraints (location, bandwidth, supported media, etc.), organizational aspects (roles, tasks, etc.) and process-related aspects (current tasks being carried out) and to generate courses accordingly through the composition of learning resources.

GRUNDEV [10] is a "proof of concept" prototype from the Pondicherry University of India able to dynamically recommend the appropriate learning content basing on the formal description of a learning context in terms of profile (including learner's personal information, personality type and learner's level of expertise), preferences (including information about learner's approach and learner's intention and learning style), infrastructure (including learner's situation, network and device used by the learner) and learning (including the learning pace, learning state and the comprehension level of the learner).

In the following sub-section we will compare our approach with the one proposed by the systems here introduced.

5.3 Comparison with Similar Systems

As seen in the previous sub-section, although the research in adaptive e-learning is very active, very few complete course sequencing systems currently exist. Moreover, the greatest part of them are used inside custom learning applications and only two (i.e. ActiveMath and NetCoach) have reached the maturity of stand-alone products. Moreover, among surveyed systems, context adaptation features are provided only by few research prototypes.

The table 6 compares the various available systems and prototypes together and with respect to the prototype for contextualized e-learning we are developing and integrating in the ALICE system on the basis of models and algorithms defined in this report.

As it can be seen our system, also thanks to the integration of other functions coming from the reference platform IWT and from other ALICE tools, is able to offer the greatest set of adaptive techniques i.e. Adaptive Course Sequencing (based on IWT), Page Link Annotation (based on the Compound Learning Resources Management Tool, see [33]) and Content Recommendation (based on the Learning Goals Recommendation Tool, see [32]).

Our prototype, together with LIP and GRUNDEV, is the only system dealing with learning context. Despite that, the LIP context model takes into accounts only few dimensions with

respect to our model and it is thought specifically to address the needs of knowledge intensive organizations differently from our general purpose model. GRUNDEV presents instead a comprehensive context model that unifies in single view information about the learner, the environment and the technological infrastructure. Despite that, the existing system is just a proof of concept that is still far a complete working prototype.

System	Status	Adaptive Techniques	Adaptation based on		
			Knowledge	Preferences	Context
AHA!	Full System	Text fragment hiding, Page link annotation	Yes	No	No
INSPIRE	Full System	Adaptive course sequencing, Page link annotation	Yes	Yes	No
InterBook	Full System	Content Recommendation, Page link annotation	Yes	No	No
ActiveMath	Full System	Adaptive course sequencing, Content Recommendation	Yes	Yes	No
NetCoach	Full System	Adaptive course sequencing, Page link annotation	Yes	No	No
ACGs	Prototype	Adaptive course sequencing	Yes	Yes	No
INES	Prototype	Content Recommendation	Yes	No	No
LIP	Prototype	Adaptive course sequencing, Content Recommendation	Yes	Yes	Yes
GRUNDEV	Prototype	Content Recommendation	Yes	Yes	Yes
Our Prototype	Prototype	Adaptive Course Sequencing, Page Link Annotation, Content Recommendation	Yes	Yes	Yes

Table 6. Comparison with similar systems.

6 Conclusions

We defined in this document the theoretical foundation for the introduction of knowledge model contextualisation facilities in the ALICE learning system. This document updates and extends [31] (as well as results presented in the related paper [34]) and takes into account results of interim experimentation activities. With respect to [31]:

- we improved the learning context model by merging two context levels, rationalising some of the feasible values and providing a justification for the selected context levels and values basing on the analysed literature;
- we introduced the notion of derivation between contexts that can be used in future evolutions to define context ontologies and taxonomies with inheritance features allowing to derive properties and property values between contexts;
- we defined an extended algorithm for knowledge model contextualization that relaxes a constraint required by the standard one by propagating some kind of relations when concepts are removed during contextualization;
- we added a new software component – the context editor – able to organize and manage centrally contexts supported within the system;
- we revised the learner model manager software component in order to let teachers associate contexts to students among those supported by the system;
- we improved the contextualized course manager software component to allow on-the-fly contextualization of a unit of learning basing on the context(s) associated to an enrolled learner;
- we revised the abstract domain model editor software component in order to use centrally managed contexts and to check consistency of defined models;
- we improved the related work section with a comparison of the prototype resulting from this research with similar systems and research prototypes.

After having developed and integrated with other IWT components the defined models and methodologies, a final experimentation phase will follow. Results coming from that can be used for a further step of models and methodologies improvement before industrialization.

References

[1] N. Capuano, M. Gaeta, A. Marengo, S. Miranda, F. Orciuoli, P. Ritrovato, LIA: an Intelligent Advisor for e-Learning, Interactive Learning Environments, Taylor & Francis, vol. 17, no. 3, pp. 221-239, September 2009.

[2] IMS Global Learning Consortium, IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata, 2006.

[3] M. Das, T. Chithralekha, S. SivaSathya, Static Context Model for Context Aware e-Learning, International Journal of Engineering Science and Technology, Vol. 2(6), 2010.

[4] L. Tankelevičienė, R. Damaševičius, Towards a Conceptual Model of Learning Context in e-Learning, Ninth IEEE International Conference on Advanced Learning Technologies, 2009.

[5] ELeGI (European Learning GRID Infrastructure) project, Deliverable D18 “Pedagogical Model, Update version of Didactical and Knowledge representation models for VSE”, project co-funded by the EC within the 6th Framework Programme (2002-2006), n. 002205, 2005.

[6] S. Downes, What is a Learning Context?, Stephen's Web, http://www.downes.ca/post/18, 2004.

[7] ALICE (Adaptive Learning Via Intuitive/Interactive, Collaborative And Emotional Systems) project, Deliverable D1.1 “Requirements”, project co-funded by the European Commission within the 7th Framework Programme (2007-2013), n. 257639, 2010.

[8] IMS Global Learning Consortium, IMS Learning Design Information Model, 2003.

[9] K.K. Thyagharajan, Ratnamanjari Nayak, Adaptive Content Creation for Personalized E-Learning Using Web Services, Journal of Applied Sciences Research, 3(9) pp. 828-836, 2007.

[10] M. M. Das, M. Bhaskar, T. Chithralekha, S. Sivasathya, Context Aware E-Learning System with Dynamically Composable Learning Objects, International Journal on Computer Science and Engineering, Vol. 02, No. 04, 2010, 1245-1253.

[11] D. Howe, Free online dictionary of computing, Imperial College Department of Computing London, UK, 2006.

[12] Dey, and K. Anind, Understanding and Using Context, Personal Ubiquitous Computing, 5(1), pp. 4-7, 2001.

[13] P. Brusilovsky, J. Vassileva, Course sequencing techniques for large-scale web-based education. International Journal of Continuing Engineering Education and Lifelong Learning, vol. 13, nos. 1/2, pp. 75-94, 2003.

[14] C. Eliot, D. Neiman, M. Lamar, Medtec: a web-based intelligent tutor for basic anatomy. Proceedings of WebNet’97, World Conference of the WWW, Internet and Intranet, Toronto, Canada, AACE, pp. 161-165, 1997.

[15] A. Rios, E. Millán, M. Trella, J. Pérez, R. Conejo. Internet based evaluation system. Artificial Intelligence in Education: Open Learning Environments, IOS Press, Amsterdam, pp. 387-394, 1999.

[16] P. Brusilovsky, ILEARN: an intelligent system for teaching and learning about UNIX. Proceedings of SUUG International Open Systems Conference, Moscow, Russia, ICSTI, pp. 35-41, 1994.

[17] P. Capell, R. Dannenberg, Instructional design and intelligent tutoring: theory and the precision of design. Journal of Artificial Intelligence in Education, vol. 4, no. 1, pp. 95-121, 1993.

[18] P. Brusilovsky, A framework for intelligent knowledge sequencing and task sequencing. Intelligent Tutoring Systems, Springer-Verlag, Berlin, pp. 499-506, 1992.

[19] R. Khuwaja, M. Desmarais, R. Cheng, Intelligent guide: combining user knowledge assessment with pedagogical guidance. Intelligent Tutoring Systems, Lecture Notes in Computer Science, Springer Verlag, Berlin, vol. 1086, pp. 225-233, 1996.

[20] G. Albano, M. Gaeta, S. Salerno, E-learning: a model and process proposal. International Journal of Knowledge and Learning, Inderscience Publisher, vol. 2, no. 1/2, pp. 73-88, 2006.

[21] N. Capuano, M. Marsella, S. Salerno, ABITS: An Agent Based Intelligent Tutoring System for Distance Learning. Proceedings of the International Workshop on Adaptive and Intelligent Web-Based Education Systems, ITS 2000, June 19-23, 2000, Montreal, Canada, pp. 17-28, 2000.

[22] N. Capuano, M. Gaeta, A. Micarelli, S. Sangineto. An Integrated Architecture for Automatic Course Generation. Proceedings of the IEEE International Conference on Advanced Learning Technologies, September 9-12, 2002, Kazan, Russia. V. Petrushin, P. Kommers, Kinshuk, I. Galeev eds. pp. 322-326, 2002.

[23] De Bra, P., Stash, N., Smits, D., Romero, C., Ventura, S., Authoring and Management Tools for Adaptive Educational Hypermedia Systems: The AHA! Case Study, in: Studies in Computational Intelligence (SCI) nr. 62, pp. 285-308, Springer Verlag, 2007.

[24] Papanikolaou K., Mabbott, A. Bull S., Grigoriadou M. Designing Personalised Educational Interactions Based on Learning / Cognitive Style and Learner Behaviour. Interacting with Computers: The Interdisciplinary Journal of Human-Computer Interaction 18(3), 356-384, 2006.

[25] Brusilovsky, P., Eklund, J., and Schwarz, E. (1998) Web-based education for all: A tool for developing adaptive courseware. Computer Networks and ISDN Systems (Proceedings of Seventh International World Wide Web Conference, 14-18 April 1998) 30 (1-7), 291-300.

[26] E. Melis, G. Goguadze, P. Libbrecht and C. Ullrich, ActiveMath - A Learning Platform With Semantic Web Features in Ontologies and Semantic Web for e-Learning, 2009.

[27] Weber, G., Kuhl, H.-C., Weibelzahl, S. Developing adaptive internet based courses with the authoring system NetCoach. In Bra, P.D., Brusilovsky, P. and Kobsa, A. (eds.) Proc. of Third workshop on Adaptive Hypertext and Hypermedia, (Sonthofen, Germany, July 14, 2001), Technical University Eindhoven, 35-48.

[28] Viet, A. N., Si, D. H. ACGs: Adaptive Course Generation System, An Efficient Approach to Build E-learning Course. Proceedings of The Sixth IEEE International Conference on Computer and Information Technology (CIT'06).

[29] Mikic Fonte, F.A.; Rial, J.C.B.; Llamas-Nistal, M.; Hermida, D.F.; Using semantics in INES, an Intelligent Educational System. Frontiers in Education Conference, 2009. FIE '09. 39th IEEE, San Antonio, TX, 18-21 Oct. 2009.

[30] Thierry Nabeth, Albert A. Angehrn, Rathish Balakrishnan; Integrating 'Context' in e-Learning Systems Design; ICALT '04 Proceedings of the IEEE International Conference on Advanced Learning Technologies, IEEE Computer Society Washington, DC, USA 2004.

[31] ALICE (Adaptive Learning Via Intuitive/Interactive, Collaborative And Emotional Systems) project, Deliverable D7.1.1 “Models and Methodologies for Knowledge Model Contextualization v1”, project co-funded by the European Commission within the 7th Framework Programme (2007-2013), n. 257639, 2010.

[32] ALICE (Adaptive Learning Via Intuitive/Interactive, Collaborative And Emotional Systems) project, Deliverable D7.2.2 “Models and Methodologies for Upper Level Learning Goals Support v2”, project co-funded by the European Commission within the 7th Framework Programme (2007-2013), n. 257639, 2010.

[33] ALICE (Adaptive Learning Via Intuitive/Interactive, Collaborative And Emotional Systems) project, Deliverable D7.3.2 “Models and Methodologies for Semantic Connections Support v2”, project co-funded by the European Commission within the 7th Framework Programme (2007-2013), n. 257639, 2010.

[34] N. Capuano, M. Gaeta, S. Salerno “An Ontology-Based Approach for Context-Aware e-Learning”. Proceedings of the 3rd International Conference on Intelligent Networking and Collaborative Systems (IEEE INCoS 2011), Fukuoka, Japan, November 30 - December 2, 2011.