3

The INTUITEL Approach: Foundations and Design

This chapter explains the modeling in the INTUITEL approach. It picks up the idea of using ontologies and reasoning to model didactic expertise from the previous chapter. The concept of the ontology of pedagogies, the idea of learning pathways and the learner model are described. Didactic factors are introduced. The model how they are used to deduce recommendations and feedback in real-time is developed. In the fourth part we describe the software architecture and in the last part the data model and communication standard is explained. The decision for semantic technologies and the OWL2 specification is justified.

3.1 Pedagogical Ontology and Reasoning

Christian Swertz, Alexander Schmoelz, Alessandro Barberi and Alexandra Forstner

An ontology needs to be consistent from a technical perspective [35]. In contrast, teaching and learning is inconsistent due to the artistic nature of educational actions. Thus, the challenge is to build an inconsistent consistency, which is an ontology that opens up a consistent room which is necessary to meet the logical structure of computer technology and that allows for the creative design of teaching and learning processes. The gap that is indicated by this contradiction can be filled by teachers and students when playing with the system.

We suggest providing a meta-data system, a learner model and a reasoning engine as tools to create learning environments. The meta-data system allows teachers to describe different possibilities to learn certain content. It can be formulated logically in an ontology in the Web Ontology Language.

The flexible elements are circled around learning pathways. The learning pathways, defined as relations between concept containers (CCs), between knowledge types, and between media types can be altered by teachers and by learners. If a teacher, for example, prefers other steps than suggested by a didactic model, he can mix those steps with steps from other pathways or create steps. While doing so, he plays with the teaching and learning models that were applied while creating the meta-data system. Some basic teaching and learning models are suggested (inquiry-based learning and multistage learning), but the teacher neither has to follow these models nor to apply these models at all. He is always free to create his own learning pathways and offer them to the learner.

Thus, the meta-data system allows teachers to play with various teaching models. Still, he has to describe his learning material with this meta-data. In his game he still uses the meta-data system, but as a toy. Since the teacher uses the meta-data system an automatic reasoning engine is still able to react on the results from teachers play. Since the learning material and the meta-data developed by the teacher are offered to learners they can use these to play too. If for example a teacher creates a learning sequence, the learner can learn the material backwards or in any creatively created order. This order can automatically be identified, converted in a personal learning strategy and applied to further material. Since the different learning pathways and the descriptions are offered to the learner, a flexible room is created where learners can play with learning models.

Understanding teaching and learning (at least partly) as play and computer technology as a toy used to create a playground sheds some light on the position that is taken when creating a pedagogical ontology for machine support in didactic practice: we are creating a game for people who play a "create a game" game. With computer technology, the playground can be best modeled by an ontology [69]. This form of a semantic network specifies the rules of the game. In order to do so, it is necessary to open up different possibilities for expressing ideas of teaching and learning creatively. Still, some rules have to be set when creating games. In order to keep the possibilities open, these rules can be developed from an analysis of computer technology as a medium, since the properties of a medium applied in teaching and learning always limit the possible actions.

The consistent part of the ontology we propose consists of a three level meta-data system for learning objects [66]. Learning Objects include instructional scaffolding such as learning objectives and outcomes, assessments, and other instructional components, as well as information objects [67]. We accommodate the levels of learning objects by using three types of Learning Objects: (i) Knowledge Domain (Course Level), (ii) Concept Container (Lesson Level), and (iii) Knowledge Objects (KOs; Content Level). The term Knowledge Domain refers to a certain amount of knowledge, which is defined by a specific curriculum, syllabus and/or course requirements.

One CC contains one instructionally framed concept within a Knowledge Domain. A CC is a container for one or more KOs. A KO is an item of knowledge, which typically corresponds to about one screen page of content and to an estimated learning time of 3–10 min for the average learner. A KO might contain learning content as well as learning activities such as a discussion in a forum, an assignment where a video has to be handed in or reading an explanation. KOs are described by a pedagogical knowledge type and a media type. CCs and KOs can be connected by relations.

In order to support different learning pathways, a vocabulary has been developed. The vocabulary for the CCs is intended to express the structure of the knowledge domain. It considers the hierarchical relations has child, has parent, and has sibling as well as the chronological relations is before, is after and is beside. The vocabulary for the knowledge types is intended to express pedagogical concepts. The vocabulary for the media types is also intended to express pedagogical concepts.

3.1.1 Learning Objects

The INTUITEL ontology is based on the concept of learning objects. Learning Objects include instructional scaffolding such as learning objectives and outcomes, assessments, and other instructional components, as well as information objects [67]. INTUITEL will accommodate Metros dimensions of learning objects by using three types of learning objects:

1. Knowledge Domain (Course Level)
2. Concept Container (Lesson Level)
3. Knowledge Objects (Content Level)

Thus, learning objects contain learning objects of different object types (see Figure 3.1).

The term knowledge domain in general refers to the part of the world investigated by a specific discipline. In INTUITEL, the term knowledge domain refers to a certain amount of knowledge, which is defined by a specific curriculum, syllabus and/or course requirements. In INTUITEL four partners

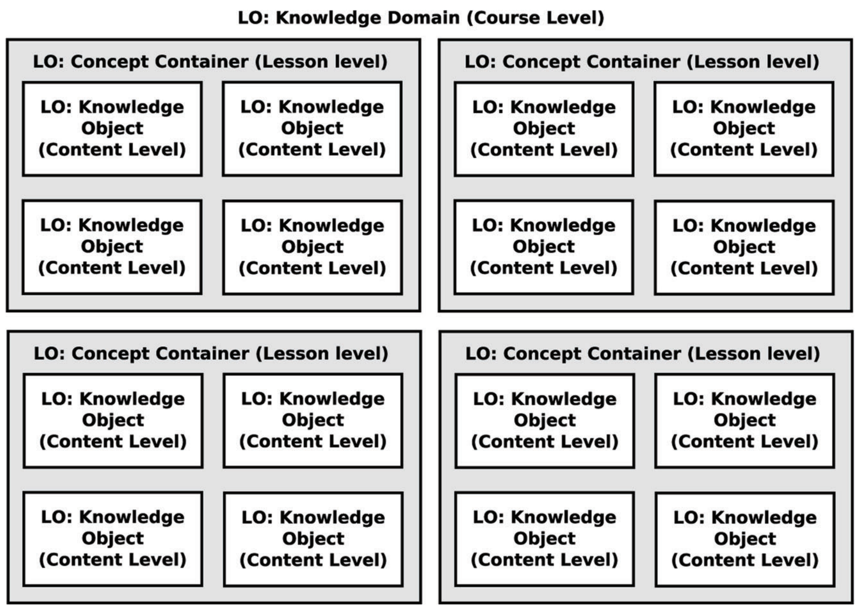
LO: Knowledge Domain (Course Level)
LO: Concept Container (Lesson level)
LO: Knowledge Object (Content Level)

Figure 3.1 Learning object hierarchy in the pedagogical ontology of INTUITEL.

(IOSB[1], URE[2], UVA[3], and UVIE[4]) will provide four cognitive models of four different knowledge domains, which correspond to the different example courses of INTUITEL. Knowledge Domains consist of several concept containers. Course is a synonym for knowledge domain. Knowledge Domains have a title and consist of knowledge containers.

One CC contains one instructional scaffolding concept within a knowledge domain. CCs are part of a knowledge domain. CCs are linked by typed relations within the knowledge domain. CCs are assembled and structured corresponding to the logic of different pedagogical CC models that are derived from learning pathways and expressed by the typed relations. Concept containers have a title, typed relations to other CCs, and are part of a knowledge domain.

Knowledge objects contain about one screen page of content and correspond to a learning time of 3–10 min. A KO covers mainly one knowledge type and one media type. The content of a KO can be anything like:

[1]Fraunhofer Institute of Optronics, System Technologies and Image Exploitation.
[2]Universiy of Reading.
[3]University of Valladolid.
[4]University of Vienna.

- a discussion in a forum (knowledge type: discussion and media type: text),
- an assignment where a video has to be handed in (knowledge type: hand in assignment and media type: video),
- reading an explanation (knowledge type: explanation and media type: text).

Knowledge objects are assembled and structured corresponding to the logic of different pedagogical knowledge type models and media type models that are derived from learning pathways. KOs have a learning time, a knowledge type, and a media type, are part of a CC and consist of content.

3.1.2 Vocabulary of Knowledge Types

Knowledge Types are due to didactical requirements. However, this structure of knowledge must be always seen as preliminary, because it can only be structured according to the goals of the knowledge type structure. Knowledge types are structured by means of the function within the learning process. This is the didactical goal of the organization of knowledge for the learning process. Functions within the learning process are presentation (receptive knowledge), trial (interactive knowledge) and communication (cooperative knowledge).

3.1.2.1 Receptive knowledge types

Receptive Knowledge Types (e.g., Orientation, Explanation) contain media for presentation. Within the media, the knowledge is displayed but without changing the presentation because of the media. The presentation is static. The learner is receiving the knowledge but is not active beyond that. Receptive knowledge may be orientation, explanation or source knowledge.

Orientation Knowledge gives orientation in one field. Knowledge is orientation knowledge, if it is naming and relating the field with other knowledge and if it can be connected to previous knowledge of the learner. This knowledge is represented in terms of: facts, history, news, log, overview, knowledge map, abstract, and scenario.

Knowledge is an explanation, if it gives reasons for representations or claims. An explanatory statement is necessary, because representations can always be different. An explanatory statement for a representation names the method, which is used by the representation. Explanatory Statements are arguments, examples, descriptions, interviews, comments, definitions, exemplifications or ideas/tips.

Sources answer the question as to find information. If a person is in possession of sources, he/she can answer the question “where to find knowledge”.

Therefore, the sources must be published and known. References on sources are made through indications of sources. Sources differ in types. The important types of sources are link lists, lists of literature, and download (which can be addresses or archives).

3.1.2.2 Interactive knowledge types

Learning items with interactive knowledge contain knowledge, whose presentation is influenced by the activity of the learner. The activity of the learner within the learning process is very useful if knowledge can be learned in an explorative way and if this knowledge can be proved in action or the knowledge is tested within an assignment.

3.1.2.3 Cooperative knowledge types

A didactical cooperation is a communication between humans, in which they work together on a certain topic in order to understand each other above expertise. Cooperative knowledge items are essential in order to react on unscheduled required knowledge. Cooperative Knowledge can be procured planned or spontaneous.

3.1.3 Media Type Vocabulary

3.1.3.1 Communication

Communication Media Types are described as tools for people to communicate directly with each other. In this list are only media types which are used online within networked computer technology. This may – for example – comprise chats, audio-conferences, video-conferences, and shared applications.

3.1.3.2 Interaction

An example for interactive media types are forms, where structured documents with blank spaces have to be filled out for further processing through a LMS. These blank spaces, which have to be filled out by the learner can be check boxes, radio buttons, lists, etc. Another example is interactive videos, where the user can at least interact through stop-and-go-functions with the computer. It would get better if the learner could also influence the plot of the interactive video.

3.1.4 Learning Pathways

Theoretically, learning pathways can be deduced from the logical structure of a knowledge domain that is expressed in the typed relations. In practice this would require a very well-written hypertext with precise typed and set relations. Unfortunately, authors tend to make little mistakes considerably in larger courses. Additionally, authors would need to know a lot about the logic of the adaptive assistant system in order to predict the outcomes that will be created based on their input. Finally, the automatic deduction of learning pathways would restrict authors to the pathways that are predefined in the system. Since there are hundreds of models for teaching and learning available and new ones created very often, this restriction does not make much sense. It would just create a tendency to undermine the theory-practice transformation competence of teachers. And that should be avoided.

That's why in INTUITEL the simple possibility to set different learning pathways among the same learning objects is considered. The learning pathways have to be set as directed acyclic graphs. No further restrictions apply. In addition to setting the learning pathways the teachers have to create a description that supports the learner in the pathway selection. Since Concept Containers and KOs are distinguished, Macro-Learning Pathways among CCs and Micro-Learning Pathways among KOs are possible. The Macro-Learning Pathways are on the level of the Content Container within one Knowledge Domain. The Macro-Learning Pathways describe how the learner might proceed within one Knowledge Domain. Within one Knowledge Domain, there can be more than one CC. These CCs are assembled and structured by learning pathways. The pathways are expressed by typed relations. In an example Knowledge Domain four Macro Learning Pathways have been used by the teachers:

- Chronologically from old to new
- Chronologically from new to old
- Hierarchically top down
- Hierarchically bottom up

Concept containers have a title; typed relations to other concept containers, and are part of a knowledge domain. The Micro-Learning Pathways are on the level of the KOs in one CC. The Micro Learning Pathways describe how the learner might proceed within one CC.

Within one CC, there can be more than one KO. If there are many KOs, they are assembled and structured by learning pathways. In INTUITEL there were three Micro Learning Pathways created by teachers for testing purposes:

- the MultiStage Approach
- the Inquiry-Based Learning Approach
- the Programed Instruction Approach.

An example metadata set for one KO is listed in Table 3.1.

Table 3.1 Example meta-data for a KO about Comenius

Meta Tag	Value
ID	KO_ComeniusOrientierungVideo LMS
Project	INTUITEL
Licence	Creative Commons Attribution Non-Commercial Share-Alike 3.0 Unported License
Author	Christian Swertz
Date	4.9.2013
Knowledge Domain	General Didactics
ContentContainer	Comenius
KnowledgeType	OrientationReceptive
Media Type	VideoReceptive
MicroLearningPathways	KO ComeniusOrientierungVideo isMoreConcreteThan KO ComeniusOrientierungText
Level	All
European Qualification Framework	LearnerEqfLevel6
EstimatedLearningTime	00:07:00
Suitable For Blind	Learner Is Not Blind
Suitable For Deaf	Learner Is Not Deaf
UnableToSpeak	All
Age	LearnerIsChild
Gender	All
Lang	De-de
Screen Minimum	320X240
Screen Recommended	640X480
Subtitle	None

This meta-data describe a seven minute video about Comenius. For an improved readability, only one Micro-Learning Pathway is reproduced here. These metadata are used as an input for the learning analytics integrated in INTUITEL. The results of learning analytics are used for adaptations, recommendations, and feedback.

3.2 Learning Analytics by Didactic Factors

Christian Swertz, Alexander Schmoelz, Alessandro Barberi, Alexandra Forstner, Alexander Streicher and Florian Heberle

As inputs for the learning analytics component of INTUITEL, three sources are available:

1. Observations of teachers and learners
2. Input of teachers
3. Input of learners

Observation data exist in the form of log files where it is recorded which learning objects were when accessed by teachers and learners. The main input from teachers is the meta-data described previously. Input from learners is either derived from profile data that is available from learning manage ment systems or from answers learners gave to requests for input from the INTUITEL system which is called TUG messages.

Since it is pretty difficult for learners to analyze raw data while learning takes place, it seems appropriate to offer some results from learning analytics to the learner. Unfortunately, we do not know beforehand which results will be relevant to the learner, but have to prepare analytics before the learning takes place. Thus, the results should only be turned into recommendations to the learner.

If for instance observation data show that the last login was a fortnight ago, it might make sense to recommend a repetition of the last topic instead of continuing with the next one. Another example is the recommendation for a learning pathway based on the age and gender of the current user:

"This course can be learned by multi-stage learning, inquiry based learning or programmed instruction." Other learners of your age and gender preferred programmed instruction. Which model do you prefer? Unfortunately, it is not known yet which Feedbacks are useful. Since this is an empirical question, the system needs to be designed in a way that allows for subsequent adaptations. That's why the rules to create feedback will be written in OWL and not as software.

A Didactic Factor is a compound of a number of data items from INTUITEL in a way so that the combination of them describes a fact that is relevant for the recommendation creation. They are the fundamental building blocks of the Rating Factors, which are used to evaluate the suitability of KOs. For this purpose, everything that is available in the whole collection of INTUITEL data, meaning the SLOM meta-data, the Learning Pathways and especially the learner-specific information (e.g., the learning history as contained in the INTUITEL logs) that are stored or collected just-in-time from the LMSs, can be used.

From a technical perspective, a Didactic Factor is an OWL class which contains its own textual description. It furthermore also links to a Java class, containing its Transformation Rule. These are the instructions that specify in which combination of input data the respective Didactic Factor is valid. This combination of OWL and Java allows a very high flexibility regarding their specification, because all features of a high-level-programming language can be used. This especially also includes functionalities that would not be available in an OWL-only solution, as, for instance, mathematical methods to calculate the ratio between two values.

As seen from a reasoning point of view, the basic task of a Didactic Factor is to combine information in a way that allows its usage in context of an OWL-reasoner. These complex software modules have foundationally different intentions than programming frameworks like, for instance, Java or .Net. Instead of iteratively executing program code to produce various results, OWL-reasoners are specialized on testing the consistency of statements and the identification of relations between entities. By drawing conclusions on a data set (i.e. an ontology), a reasoner can deduce statements that, for instance, allow to determine whether a CC is fitting for a certain learner's Learning Pathway (LP). The Didactic Factors are especially relevant in this process because natural or real numbers are problematic in that context. This entails that INTUITEL needs to reformat the input in a way that is compatible with such a system. One aspect of the Didactic Factors is consequently to transform the non-nominal values into a nominal form (e.g., by transforming the continuous value 5 into the categorized statement "medium"). There are four fundamental forms of Didactic Factors:

1. Trivial statement: The most basic realization of a Didactic Factor is the *n*: 1 relaying of input data. This means that certain data items are combined and translated into a format that is compatible with the Engine. (example: gender as male or female)
2. Trivial input combination with grading: Different nominal data items can be connected to create a combined statement that entails some kind of grading. (example: connection type as slow, medium, and fast)
3. Complex statement: A more complex use case for a Didactic Factor is the discretization of numerical values into a nominal one (example: noise level in DB is expressed as quiet, tolerable, and loud)
4. Complex input combination with grading: The combination of different (kinds of) input values through, e.g., mathematical functions, can also result in graded Didactic Factors.

In the table below, we provide the list of the Didactic Factors that have been developed in INTUITEL. This list does not claim to be complete or that the respective items are final, since there is no evidence for useful factors available yet. This list will nevertheless give a detailed overview about aspects that might be relevant for the selection of suitable Learning Objects.

#	Didactic Factor	Description
01	Knowledge actuality	Ranking of time between now and the last learning session.
02	Course-focused KO learning speed	Ranking of learning time the learner on average differs from the estimated learning time in contrast to the same measure of the other course participants
03	Learner-focused KO learning speed	Ranking of learning time the learner on average differs from the estimated learning time of completed KOs of this session in contrast to same measure over all KOs over all sessions.
04	Course-focused filtered KO learning speed	Ranking of learning time the learner on average differs from the estimated learning time in contrast to the same measure of the other course participants when only having a look at KOs that have the same KT and MT.
05	Learner-focused filtered KO learning speed	Ranking of learning time the learner on average differs from the estimated learning time of completed KOs of this session in contrast to same measure over all KOs over all sessions when only having a look at KOs that have the same KT and MT.
06	Course-focused session length	Statement about the average session length as compared to the average session length of other course participants.
07	Learner-focused session length	Statement about the current session length as compared to the average session length of the learner.
08	Time exposure	Comparison between the amount of time the learner and the other course participants spent on the course.
09	Learning Pathway permanence	Ranking of the amount of KOs the learner completed on the current LP combination in contrast to the same measure for the other course participants.
10	Recent learning pace	Comparison of the actual learning time the learner needed for the last 10 KOs in contrast to the estimated learning time.
11	Session learning pace	Comparison of the actual learning time the learner needed for the KOs in this session in contrast to their estimated learning time.
12	Course-focused LP usage type	Statement about the LP usage as measured on the learners pathway switches and the switches of the other course participants.
13	Learner-focused learner type	Statement about the LP usage as measured on the learners pathway switches.
14	Course-focused learning success	Success of the learner regarding scores in contrast to the other course participants.

(*Contnued*)

Table Continued

#	Didactic Factor	Description
15	Learner-focused learning success	Success of the learner regarding scores in contrast of the own score history.
16	Course-focused KO repetition quantity	Comparison of the number of repeated KOs with the number of repetitions of the other course participants.
17	Learner-focused KO repetition quantity	Comparison of the number of repeated KOs in the recent KO history and the average of repeated KOs.
18	Course-focused CC repetition quantity	Comparison of the number of repeated CCs with the number of repetitions of the other course participants.
19	Learner-focused CC repetition quantity	Comparison of the number of repeated CCs in the recent KO history and the average of repeated CCs.
20	Course KO completion	Statement about the coverage of the course regarding the completion states of KOs.
21	Course CC completion	Statement about the coverage of the course regarding the completion states of CCs.
22	CC KO completion	Statement about the coverage of the current CC regarding the completion states of the connected KOs.
23	Course-focused KO completion tendency	Comparison of the learners and the other course participants ratio of completed KOs in contrast to the uncompleted ones of the session.
24	Learner-focused KO completion tendency	Comparison of the earners and the other course participants ratio of completed KOs in contrast to the uncompleted ones of the session.
25	Course-focused MT preference	Statement about the MT preference as measured on all course participant selections.
26	Learner-focused MT preference	Statement about the MT preference as measured by the learners learning history.
27	Course-focused MT dislike	Statement about the MT dislike as measured on all course participant selections.
28	Learner-focused MT dislike	Statement about the MT dislike as measured by the learners learning history.
29	Course-focused KT preference	Statement about the KT preference as measured on all course participant selections.
30	Learner-focused KT preference	Statement about the KT preference as measured by the learners learning history.
31	Course-focused KT dislike	Statement about the KT dislike as measured on all course participant selections.
32	Learner-focused KT dislike	Statement about the KT dislike as measured by the learners learning history.
33	LP leaving position	Statement at which point (in the sense of completed LOs) the learner leaves a LP.
34	Course-focused learning efficiency	Ranking of how much time the learner needs to complete a KO in contrast to the time the other course participants needed for it.

Table Continued

#	Didactic Factor	Description
35	Learning attention	Statement about how much attention the learner pays to the content as measured by an eye-tracking device connected to the LMS.
36	Blindness	Statement if the learner is blind.
37	Deafness	Statement if the learner is deaf.
38	Gender	Statement about the learners gender.
39	Age	Statement about the learners age.
40	EQF Level	Statement about the learners European Qualification Framework (EQF) level.
41	Learner Level	Statement about the course specific level of knowledge the learner possesses.
42	Device resolution	Ranking of the relative resolution of the device the learner uses to access the LMS.
43	Connectivity level	Ranking of the connectivity between the learners access device and the LMS.
44	Noise level	Ranking of the environmental noise level of the learner.
45	Learning Environment	Statement about the type of environment the learner is currently located at.
46	Learning Velocity	Ranking of the time the learner needs to successfully complete Learning Objects.

As stated above, these factors need to be transformed into statements that can be computed by a reasoner. To give an example, let us assume that the estimated learning time is 3 min and the actual learning time was 2 min, and 30 s. Let's further assume that the transformation rule differentiates five cases:

1. Estimated time actual time > 2 min $\Rightarrow$ No rating
2. Estimated time actual time < 2 min AND > 1 min $\Rightarrow$ fast learner
3. Estimated time actual time < 1 min AND > -1 min $\Rightarrow$ normal learner
4. Estimated time actual time < -1 min AND > -2 min $\Rightarrow$ slow learner
5. Estimated time actual time < -2 min $\Rightarrow$ No rating

In the present example, this would result in the statement that the learner is a normal learner. Please note that this is only an example. There can be arbitrarily many combinations of input values, but only a subset of them is pedagogically meaningful and exact enough. If, for example, the estimated learning time is quite high (e.g., hour which is non-compliant to the INTUITEL guidelines), completing the KO more than one minute earlier or later is certainly common. Thus, specifying well-engineered rules is an important factor regarding the accuracy of INTUITEL.

Concluding this section, the following examples explain the transformation rules for three of the above mentioned Didactic Factors. Please note that due to a better readability, standard deviation is denoted as *s*. For a full definition of all transformation rules of the 46 Didactic factors, refer to the according Deliverable 3.2 [25] of the INTUITEL project.

Transformation rule for DF "Course-focused KO learning speed"

Input:
lAvgLT = learners average learning time of recent KOs
oAvgLO = others average learning time

Output:
KoSpeedFast, KoSpeedSlow, KoSpeedNormal

Transformation rule:

```
if (lAvgLT > oAvgLT + s)
    output = KoSpeedSlow
else if (lAvgLT < oAvgLT - s)
    output = KoSpeedFast
else
    output = KoSpeedNormal
```

Transformation rule for DF "Course-focused filtered KO learning speed"

Input:
lCouples[] = Learners average difference between actual and estimated learning time of KOs, which is differentiated into KT and MT couples (only the three topmost types each),
oCouples[] = Others average difference between actual and estimated learning time of KOs, which is differentiated into KT and MT couples (only the three topmost types each).

Output:
FilteredKoSpeedFast, FilteredKoSpeedSlow, FilteredKoSpeedNormal

Transformation rule:

```
For each couple {
    if (couple not null for learner) {
        lAvg += learner s value for couple
              oAvg += others value for couple
           }
}
lAvg /= count of not null couples
oAvg /= count of not null couples
if (lAvg > 110\% of oAvg)
   output = FilteredKoSpeedSlow
else if (lAvg < 90\% of oAvg)
   output = FilteredKoSpeedFast
else
   output = FilteredKoSpeedNormal
```

Transformation rule for DF "Learner-focused learning success"

Input:
scoRec = Recent average learner score
scoGen = General average learner score

Output:
SuccessBetter, SuccessStable, SuccessWorse

Transformation rule:

```
if (scoRec > scoGen + s)
   output = SuccessBetter
else if (scoRec < scoGen - s)
   output = SuccessWorse
else
   output = SuccessStable
```

3.3 Learning Progress and Learning Pathways

Alexander Streicher and Florian Heberle

The central information mediating component of INTUITEL is the Learning Progress Model (LPM) module, which connects all other components. It acts as a preliminary stage for the INTUITEL Engine. By providing functions to perform transformations of learner scores, history, and pedagogical as well as domain knowledge into a position within a cognitive space, the LPM prepares the data to be usable by the semantic reasoner in the INTUITEL Engine.

In order to achieve this, it relies on multiple data sources which carry data about the users. First, there is the data about the learner as represented in the LMS. Secondly, the pedagogical and domain knowledge in form of the Pedagogical Ontology and the respective Cognitive Models, provide information about the actual learning content and how to guide a learner through the learning material. And in the third place, a user model of internal INTUITEL relevant user data like session data or the navigation history. The specific tasks of the LPM regarding the recommendation creation process can be summarized with three core themes:

1. Data aggregation
2. Data storage
3. Data transformation

To complete its range of functions, the LPM includes components to trigger direct user feedback before the reasoning process starts, determine the most suitable Learning Pathway (LP) and create the optimized data basis for the Reasoning Engine. Explicitly not task of the LPM is finding suitable Learning Objects (LO) for a learner. This is solely task of the INTUITEL Engine. The LPM is only indirectly involved in this process since it provides and edits the relevant information to be later used by the INTUITEL Engine.

3.3.1 INTUITEL Recommendation Process

There are multiple components that are part of the recommendation creation process and the individual tasks need to be distributed between them. The LPM is one of these components and the following descriptions show which tasks are conducted by the LPM and which tasks the other modules take over. This also depicts how the LPM is positioned in the INTUITEL overall system design.

The functional process has three stages, as depicted in Figure 3.2. There firstly is the stage of data preparation where the LPM collects and reformats the data for the INTUITEL Engine. Secondly, the reasoning process is conducted to determine the elements in the different result sets. Thirdly, the results are sorted and reformatted to be in a format that is compatible with the rest of the INTUITEL system. Also, learning recommendation messages are created to guide the LMS user.

3.3.2 Comparison to Real-Life Tutoring

Speaking in terms of a real-life learning scenario, INTUITEL is a workflow engine to process well established learning methods, in particular by deducing relevant and individual recommendations for learners. Without loss of generality, Table 3.2 outlines the analogy between the decision process of a real teacher and INTUITEL.

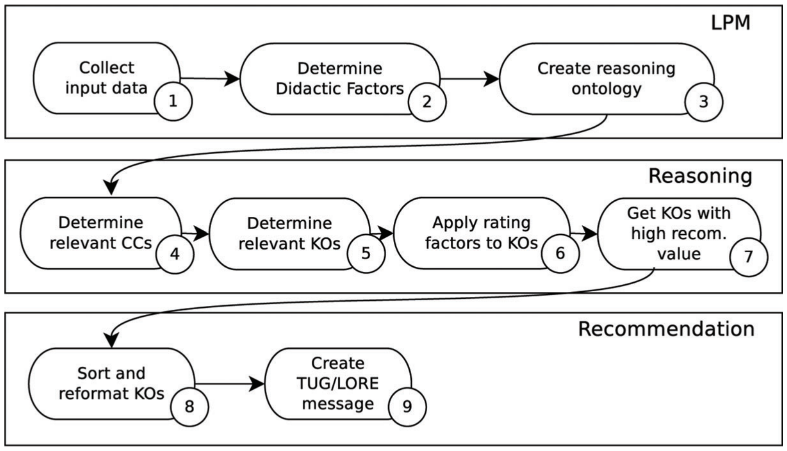
LPM
Collect input data 1
Determine Didactic Factors 2
Create reasoning ontology 3
Reasoning
Determine relevant CCs 4
Determine relevant KOs 5
Apply rating factors to KOs 6
Get KOs with high recom. value 7
Recommendation
Sort and reformat KOs 8
Create TUG/LORE message 9

Figure 3.2 Simplified functional procedures for the recommendation creation in the back end.

Table 3.2 Comparison between learner and INTUITEL behavior in learner counseling

Pedagogical Aspect	INTUITEL Analogy
A human teacher senses when a learner needs assistance. The sensing process may rely on subconscious experience based processes as well as involve cognitive reasoning	Via the LMS-interfaces, INTUITEL registers and protocols the current and precious states of the learner. The LPM serves as the sensing instance, either by simple rules based inference or by more complicated ontology based reasoning.
In looking at test results and carrying out a spoken dialog, a teacher tries to assess the learner's current needs.	Usage of personalized information INTUITEL has collected for each learner and request of up-to-date information via USE or TUG dialog.
By applying personal didactic/domain knowledge or by intuitive reaction, the teacher forms an opinion about the learner's situations.	Application of transformation rules to create the different Didactic Factors and their integration in the learner-specific Learner-State-Ontology.
The teacher evaluates the situation on a topical basis and aligns it to his or her personal knowledge about the topic.	With the Cognitive Model stored in the SLOM file, INTUITEL draws conclusions about necessary alignment of the learning process.
The teacher ponders which advice fits best.	Reasoning process in the INTUITEL Engine to select suitable KOs and/or the decision which natural language feedback should be given.
Teacher gives the learner an advice which fits his/her current situation best.	Recommendation of specific content via the LORE interface and/or natural language messages via the TUG interface.

3.3.3 Reflex Reactions

Like a human teacher the INTUITEL system should always be able to react to the learner. At best, the recommendation process is always based on sound considerations of all input information. But sometimes even a real teacher is not able to find the best recommendation to help the learner. Possible reasons for that could be the lack of some specific knowledge or the deduction problem is too complex to be solved in an appropriate time. In such cases teachers must rely on their intuition. Their "reflexes" kick in and the teachers give nearly-as-good answers as recommendations.

Ordinarily, the more experience a teacher has, the better the recommendations fit the learners' situation and needs. INTUITEL aims to do the same in order to provide for the best possible guidance. It thus needs a mechanism which decides when to bypass the reasoning process and when to trigger a direct reaction. Therefore, the LPM contains a "reflex module". Like in the real-life scenario, the reflex module acts when it is foreseeable that a response is not going to be created in time or when it is obvious that a certain action has to take place.

Because a recommendation cannot necessarily be created in all cases, the reflex module triggers a message that either informs the learner that his recommendation is pending (e.g., "I am still thinking about it, just one moment please.") or enforces the creation of a specific question that the learner should answer (e.g., "Which Learning Pathway do you want to choose? Select one of the following"). Another example would be to create a welcome message, if the learner just started a new session. For such a message, no computationally expensive reasoning has to take place.

3.3.4 LPM Input

This section defines and explains the different input types for the LPM. It firstly describes the learner dependent set of data. Further, the concept of Learning Pathways and their realization in the Pedagogical Ontology is outlined. Afterwards, the concept of the Cognitive Models is described.

3.3.5 Learner Input

Learner input basically describes all personalized learner information INTUITEL collects from and via the LMS and which can be used as a source for the recommendation creation process. This not only includes learner scores in terms of grades or results of tests, but also additional data which can be collected by INTUITEL. There is a multitude of different types or kinds of learner scores that provide diverse information about the learner from a learning habit, a learning progress, and a situational perspective. Depending on the possibilities of the LMS and the available information, some of it might also be collected by directly asking the learner. Examples of the available learner data are, apart from the rather obvious question how good the learner is in terms of grades, amongst others:

- What and when did the learner access content?
- In which order did the learner access learning content?
- How long was the learner working on certain a Learning Object?

- Which kind of device is the learner currently using?
- How good is the learners Internet connection?
- Is the learner currently in a noisy environment?
- Other possible raw data items may indicate emotional or stress measurements involving bodily interfaces to the learner.

In short, learner input describes all information on all aspects regarding how and what a learner currently and also previously has learned in the eLearning system across all courses. For a detailed list and description which data can be accessed through the respective INTUITEL interfaces, see the INTUITEL data model in Section 3.5. Examples could be age, name, gender etc.

3.3.6 Pedagogical Input

The pedagogical input for the LPM consists of two parts. On the one hand, there is the terminology specified in the Pedagogical Ontology. It provides a comprehensive set of entities for the modeling of learning material in INTUITEL. The structuring of learning material, independent of its actual LMS realization, allows a semantically rich description of e-learning courses. The central elements, namely CCs for abstract topic-based structuring of courses and KOs for the description of the actual content, allow the LPM to understand the meaning of the Learning Objects in the course. However, what is more important for the pedagogical input is the second aspect, the availability of Learning Pathways. In order to guide learners through an e-learning course, INTUITEL needs a "map" describing how to find a suitable route. This basically is what Learning Pathways (LP) provide for the back end. Modeling Learning Pathways with them, the LPM and the INTUITEL Engine are able to deduce a didactically reasonable route through the learning material.

3.3.7 Domain Input

The previously described pedagogical input alone is as itself insufficient for the back end, since it only provides the technical and didactic foundations. The description of the actual learning content is missing so far. In order to include this, the LPM needs the description of the knowledge domain from the lecturer. This is provided through the Cognitive Model (CM) which is created by a domain specialist. The CM outlines the curriculum of a certain domain of knowledge. This OWL-based description specifies which CCs are available in a specific Knowledge Domain (KD). Based on this a course in a LMS is connected by completing the course specification with the semantically rich description of the KOs. The thereby defined meta-data contains detailed information about the learning material itself (e.g., the contained media or what type of knowledge they represent).

To summarize, the domain input subsumes all information which the LPM and INTUITEL Engine need to understand the internal coherences in an eLearning course.

This does not mean that INTUITEL understands the content itself, i.e., what the learner is trying to learn, but is able to react on the semantic data as mentioned before.

A figurative example would be the simplified e-learning course "Math for absolute beginners" (see Figure 3.3). Consisting of a number of pages introducing to the basics of what numbers are and the basic arithmetic operations addition and subtraction, it depicts a very brief example of domain input in INTUITEL. On the first level, there is the start point of the course itself, the Knowledge Domain (KD). Branching from there, the individual CC are connected via macro LPs and allow navigating between them in a didactically reasonable way. The KOs is attached to the CCs and the micro LPs specify how a learner should best range between them on basis of the KO meta-data. With this information, the back end can deduce an optimal route for each individual learner, based on how many different LPs are available for that specific course.

Although this is not categorized as domain input, it should be noted that the individual Learning Pathways between the elements are firstly possible because the domain input provides a description of the content between which a route is possible. INTUITEL can thereby rely on the different macro LPs that the tutor has added into the Cognitive Model and also the micro LPs that are available due to their definition in the Pedagogical Ontology.

3.3.8 Set-based Rating of Learning Objects

Due to the usage of OWL-reasoning in INTUITEL, the approach of how recommendations are created is different to how other software solution approach that task. One of the main features of OWL-reasoners is the identification of elements in a set. This is a very useful functionality for INTUITEL, because creating a KO recommendation can be interpreted as the task of finding the elements of the set that contains the optimal KOs for the learner. Thus, the fundamental question of the back end is: What are the defining criteria of this set and how to evaluate them? Building on the previous introductions on what input is available for the LPM, it can be registered that there basically are three influential factors concerning the suitability of a KO:

- the learners macro Learning Pathway
- the learners micro Learning Pathway
- the situational dependent information of the learner

The first two points are relatively unproblematic, because this basically only requires data that is already available in the present system. This is at first the data which specifies the Learning Objects that are part of the course. Secondly, the history of the learner is needed to filter those LOs that are not available for the recommendation (e.g., because they are already finished). Thirdly, the learners personal Learning Pathway is required, to find the LOs that are next in the sequence. Given that this information is handed over to the Engine in a suitable way, the LOs that are relevant in context of the learners LP can be identified. This is possible, because the present problem

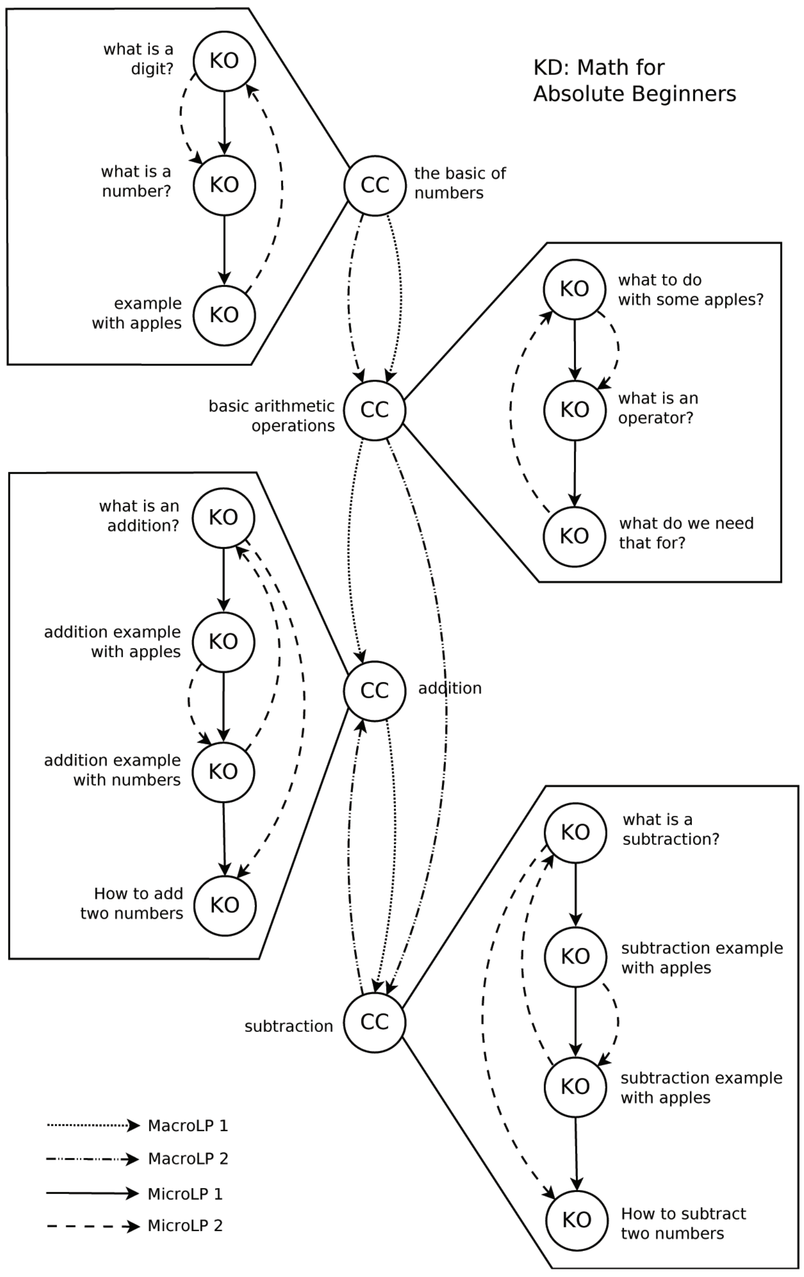
KD: Math for Absolute Beginners
what is a digit?
KO
what is a number?
KO
example with apples
KO
CC
the basic of numbers
CC
basic arithmetic operations
KO
what to do with some apples?
KO
what is an operator?
KO
what do we need that for?
what is an addition?
KO
addition example with apples
KO
addition example with numbers
KO
How to add two numbers
KO
CC
addition
CC
subtraction
KO
what is a subtraction?
KO
subtraction example with apples
KO
subtraction example with apples
KO
How to subtract two numbers
KO
MacroLP 1
MacroLP 2
MicroLP 1
MicroLP 2

Figure 3.3 Exemplary different LPs diagram showing the coherences in the domain input, including different LPs.

can be described on basis of sets, allowing the Engine to iteratively reduce the number of LOs. In order to understand this procedure, it is advisable to illustrate this step-by-step.

As the first step, it should be clear that a LO is an umbrella term that combines CCs and KOs (for the sake of simplicity, Knowledge Domains (KDs), which are by definition also LOs, are excluded here). Each LO is thus either a CC or a KO. So, when starting with a set that contains all LOs of a certain course, it is possible to segment it into the set of CCs of that course and into the set of KOs of that course. These two sets can further be segmented into sets that differentiate LOs depending on whether they are rated as unseen, unfinished or already completed (Figure 3.4).

Independent of these sets is it possible to subdivide the set of available LOs of a course regarding their distance to the current Learning Pathway positions. For this, firstly the set of currently active KOs is identified which is represented by the small dark circles in Figure 3.4. This set of currently active KOs is either empty or contains exactly one item. It is empty if the learner has never worked on the course and has not yet accessed a KO in the current session. If the learner has already accessed a KO in the LMS, the set contains exactly this KO (i.e., the last one), which is rated as either unfinished or already complete.

Based on that, it is possible to identify the set of current CCs, which contains all CCs that has reference on the currently active KO. The set thus contains either one CC or multiple ones, because a KO can be part of more than one CC. The set of CCs that are next when following the macro LP is a logical consequence of the previous

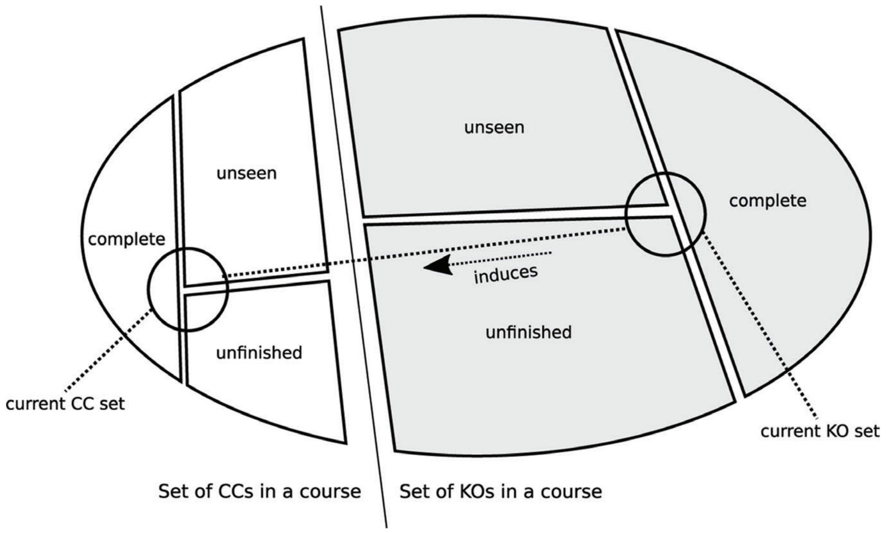

Figure 3.4 Sets of current CCs and KOs in the set of available LOs.

set. When it is known which CC(s) is/are currently active, the next CCs are those that are subsequent as described in the respective LP definition.

With these sets, the back end is able to identify the CCs that have recommendation relevance for the respective learner. This firstly is the intersection of the set of current CCs with the union of unfinished and unseen Ccs. If the set of CCs with recommendation relevance is empty, the course is either complete or there are no more CCs available in that particular macro LP. In the first case, a recommendation cannot be created. In the second case, a new macro LP needs to be selected or the process is finished too.

If at least one recommendation relevant CC has been identified, it is subsequently possible to create a list of KOs that have a CC-based relevance. This is the set of KOs that are attached to this/these particular CC(s). This is already a big reduction of the available LOs, but it can be reduced even further, when also including the micro LP information (Figure 3.5). This is the set that contains all KOs that are "next" as seen from the currently active KO and those KOs that are "next" regarding the "next" CCs. As previously, the term "next" is a matter of definition, which is based on the question what is reasonable to include here.

As the last step in the procedure, to identify those KOs that are recommendation relevant, the back end has to select those KOs that fulfill a list of requirements:

- KO is next or currently active
- KO is attached to a CC that is next or currently active
- KO must be unfinished or unseen (i.e., not already completed)

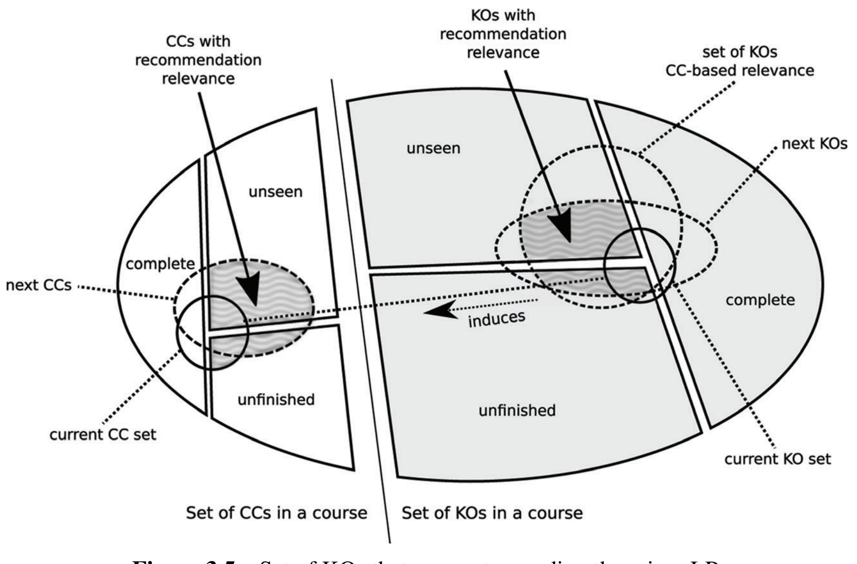

Figure 3.5 Set of KOs that are next regarding the micro LP.

Expressed on a set-basis, this is the intersection of the set of currently active KOs, with the set of KOs with CC-based relevance and the union of unfinished and unseen KOs. For an easier understanding, the diagram in Figure 3.5 clarifies these coherences graphically. If this respective set is empty, the same method as already used for an empty set of CCs with recommendation relevance is applied. Either has the micro and/or macro LP to be changed to include other LOs, or the process is finished, since there are no more KOs available to choose from.

When now coming back to the initial statement that there are three influential factors for the recommendation (macro LP, micro LP and situational dependent information), this so far only includes the first two aspects. In order to further personalize the recommendations the information that INTUITEL collects about the learner must be included in this procedure. For this, the back end specifies a method to include this information in a suitable way, which is implemented via the so called Didactic Factors that are explained in Section 3.2.

On the next level, these Didactic Factors are combined with properties of KOs to state their individual suitability for the learner. The procedure of connecting this information with the learning content is also expressible via the set-based approach as it has been applied to find the (LP-) relevant KOs. Therefore, for each Rating Factor, a set is specified that fulfills a particular rating rule. The task of the back end is then to find the elements that meet the respective conditions and to combine all this knowledge (Figure 3.6). This is done by calculating the intersection of the set of KOs with a general recommendation value (i.e., those elements that have been selected previously on basis of LPs) and those sets that express the optimum regarding the different Rating Factors.

3.3.9 Learning Progress and Learner Position

In INTUITEL the learner is being located in a multidimensional space. This section explains the intentions behind the Multidimensional Cognitive Space (MCS) and the associated Cognitive Content Space (CCS). Both are newly introduced concepts that the INTUITEL project uses to formalize the task of finding the position of the learner in the e-learning content. It has been designed to translate the basic educational conditions to a mathematical level, which is much better applicable on the domain of computer science. The presented geometrical representation of Learning Pathways and learning content in form of a multidimensional hypercube facilitates a better understanding of the technical realization of this task.

3.3.10 Determination of the Next KO

In the following it is outlined how to determine initial recommendations for a “next” KO to be processed by the learner, which are then modified by the Didactic Factors. Note, that throughout this section we are not dealing with Micro Learning Pathways

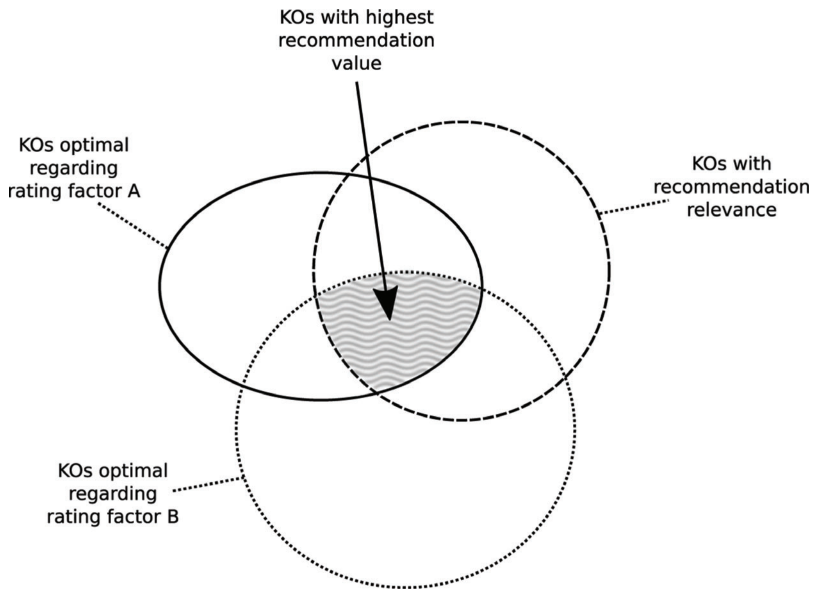
KOs with highest recommendation value
KOs optimal regarding rating factor A
KOs with recommendation relevance
KOs optimal regarding rating factor B

Figure 3.6 Set of KOs with the highest recommendation value.

or LOs that are considered alternatives, but only with Macro Learning Pathways and singular LP nodes. Consequently, for the purpose of this consideration, each KO may be considered as fundamentally different from all the others because it leads to a different knowledge strain. The basic paradigm of outcome-oriented constructivist learning is that it should lead to a certain learning goal. This goal might be quite complex, e.g., might require that

- a certain set of concepts is learned with a required precision
- a certain set of temporal, causal, or logical connections between these concepts is learned
- a certain set of methods and algorithms is learned which enables the learner to process new situations

Mathematically, this complex goal – which in fact is an ontology – may be expressed as a target position in an abstract space spanned by the concepts.

We term this the Multidimensional Cognitive Space MCS. Learning therefore may be considered a movement in this abstract space and learning should bring the learner closer to the target position. In modern human-centered teaching processes, teachers determine the current position of a learner by assessing the learners current knowledge and standing in regard to the topic and on a "meta"-level (e.g., how the learner learns). This happens either subconsciously by sensing that there are problems, delays or disturbances while the lessons are taking place, by evaluating the assignments of the learners (e.g., homework) or by having a direct conversations with them (e.g., questions during the lessons). These observations and feedback enables the teacher to come to conclusions regarding the learners' state of knowledge, habits, and characteristics to ultimately support the learner in the learning process.

INTUITEL provides means to carry out this tutorial guidance process also in technology enhanced learning, where no human teacher in traditional form is present and the learning so far may be seen as a self-directed process. A learning object recommendation, as it is generated by the INTUITEL system, therefore is a computer generated hint towards the self-directed learner on how the learner may come closer to the target position. Note that this recommendation is not entirely a global recommendation ("direct way to the target"), but may be locally deviant from the direct way to the target. Consider, for example, the possibility of a recommendation indicating that the learner should repeat a certain Learning Object: In this case it might correspond to a movement of the learner in the Multidimensional Cognitive Space (MCS) which for some time takes the learner further away from the final learning target position. The INTUITEL system therefore is handling the following tasks:

- Determination of the learner position in MCS
- Determination of a set of Didactic Factors influencing the learning process
- Application of the personal Learning Pathway on the course material
- Taking into account these Didactic Factors and the Learning Pathways, determination of the next step the learner has to take in order to ultimately come closer to the target position

The principles of the MCS will be reconsidered below, for now let us consider the Didactic Factors as they were explained in Section 3.2. Some of them are:

- Cognitive speed, which is the learning speed in the context of environment, personal attitude and current learner position is this a fast, a medium or a slow learner?
- Learning success is this a learner with superior, good, average or inferior results?
- Learning discipline does this learner follow suggestions easily, does the learner adhere to ascribed learning pathways?

For the moment, the INTUITEL consortium relies on a rather simple scale for each of these factors, e.g., "learning speed" is not measured in objective terms but in simple concepts of fast medium slow. This means, that a crucial module of the LPM is responsible to translate raw input data (from the LPM) into these Didactic Factors. Consider, for example the Didactic Factor of "learning speed". For each learner, the LMS delivers the actual learning time for each KO to the LPM. Furthermore, the LPM reads the metadata accompanying the learning content and knows the target learning time entered there by the cognitive engineer (henceforth called "estimated time"). The corresponding transformation rule for the LPM then reads, for example:

> "If the learner needs more time than the estimated time to process a KO in at least 70 percent of all KOs, the learner will be assigned the value "slow" on the dimension of learning speed. If the learner needs less time than the estimated time to process a KO in at least 70 percent of the KOs, the learner will be assigned the value 'fast' on the dimension of learning speed. In any other case, the learner will be assigned the learning speed value "medium" (70-70-rule)."

However, suppose that the learner has already achieved a high level of knowledge, easy questions should then be answered faster. Therefore, one could think to shift from a 70-70-rule for learner speed determination to a 50-85-rule in this case. Or it may turn out that the learner is working in a high noise environment providing lots of distraction in this case one could revert to an 85-50-rule.

3.3.11 Cognitive Content Space

The Cognitive Content Space (CCS) is spanned by the Cognitive Model and the Semantic Learning Object Model (SLOM). SLOM is explained in more detail in Section 3.6. The Cognitive Model contains the pedagogical structure of a course; it is a concretion of the pedagogical ontology for a given domain of knowledge. The SLOM contains meta-data referring to the actual content. Both of these might contribute to the desired learning goal. For later development it is also possible that a more complex learning goal is defined in the Cognitive Model. For example, the cognitive engineer could specify, that in his Cognitive Model the desired learning goal puts more emphasis on practical knowledge than on theoretical knowledge. In such a case, the Cognitive Model would contain a transformation rule to determine the cognitive position from the raw input data.

3.3.12 Multidimensional Cognitive Space

The cognitive position of a learner is estimated by the amount the learner has learned from each of the KOs. Hence, the position for a course consisting of n KOs is determined by an n-dimensional vector within the Multidimensional Cognitive Space (MCS). The value of each vector element can be determined as follows:

- In a simple LMS one only knows that a KO has been processed. In this case we follow the strong optimistic learner assumption: Processing means learning, consequently the grade of progress of the respective KO jumps from 0 to 100 percent when a KO has been accessed.
- A more advanced LMS may tell INTUITEL which part of a KO has been processed by the learner. In this case we will follow the weak optimistic learner assumption: A partial processing of a KO means that the learner has learned the same percentage of this KO.

- A very advanced LMS will tell INTUITEL the result of a measurement, like, e.g., the percentage of points reached in a concluding test of this KO.

Let us now consider how a certain learning goal may be achieved as a sequence of cognitive positions. Let $P = (x_1, x_2, \ldots, xN)$ denote a vector representing the learners position in the Multidimensional Cognitive Space (MCS) with x_i representing the grade of progress of the i^{th} KO.

The default learning goal states, that each KO has to be learned. The target position in the MCS, therefore, is a value of 100% = 1.0 for each component of this vector. The learner starts at position $Ps = (0, 0, \ldots, 0)$, his target position is $Pf = (1, 1, \ldots, 1)$.

Should there be a more complex learning goal (say, target learning profile) instead of all KOs being processed completely, the Cognitive Model defines a transformation, which for simplicity may be seen as a linear transformation (a matrix) reducing the dimensionality of the MCS from N to $M < N$. Therefore, without loss of generality, even in this case the same INTUITEL concepts and algorithms may be used as for the default learning goal.

If we assume a learner who completes each KO before moving on, the movement of this learner in MCS would always be along the edges of a N-dimensional hypercube: The learner has completed KOs 1, 2, 3, 4, . . ., $k - 1$. He is currently working himself through the k^{th} KO and still has to consider himself with KOs $k + 1$, $k + 2$, ..., N. Hence, his cognitive position would be

$$Lk = (1,\ 1,\ 1, ..., x_k, 0,\ 0, ..., 0)$$

Whereby x_k represents the KO at the current position. Here, without loss of generality, we have ordered the KOs in exactly the sequence as it is processed by the Learner, henceforth called User Learning Path (ULP). However, we might also assume a learner who is much less disciplined, and therefore does not complete any KO before moving on. Obviously, the sequence of cognitive positions in the MCS would be a curve inside (and not along the edges, but possibly across bounding surfaces) of the hypercube. In the extreme example, this learner would switch back and forth between the KOs in the model and finish each KO to the same degree. While this learner might nevertheless reach the final learning goal, his actual learning curve is a line close to the hyper-diagonal of the MCS. In order to achieve his final goal however, this learner would have to perform a very large number of switches between KOs, in order not to emphasize a particular KO.

For now let us assume that we have a disciplined learner who always moves along the edges of the knowledge hypercube. This amounts to a self-chosen sequence of KOs, each of them is processed (by assumption, learned) completely before moving on. After completing the k_{th} KO, the cognitive position would be

$$L_k = (1, 1, 1, \ldots, 1, 0, 0, \ldots, 0)$$

Obviously, if repetitions are excluded, this learner then has $N - k$ possible choices for his next KO and the direct cognitive distance to the target position is $\sqrt{N-k}$.

Each of those possible steps would reduce the direct cognitive distance by the same amount. Hence, none of the remaining KOs would be preferable from the viewpoint of reducing direct cognitive distance to the target state.

The Cognitive Model however may define certain Learning Pathways $LP_1, LP_2, \ldots, LP_S$. Since we reserved the standard numbering for the ULP, we have to consider each of these predefined Learning Pathways an ordered permutation of the numbers $1, 2, \ldots, N$. Since the current cognitive position after *k* learning steps may be arbitrarily close to Learning Pathway *l* but after a number of *t* steps we may not use simple path deformation rules to compare the actual current cognitive position to one that could be reached by this Learning Pathway. Therefore the LPM implements the following algorithm:

1. Determine the Learning Pathway LP_l from the Cognitive Model which runs closest (in distance d) to the current cognitive position L, and the closest corner point V (vertex) of P_l
2. Check each of the open choices $N - k$, if it would reduce the distance to V from d to $d' < d$ and assign to it the priority $d - d'$.
3. Check on the Learning Pathway P_l which the next KO would be after V, we assume that this would be KO no. v. If v is among the open choices $N - k$, add to the priority assignment of v the value $+1$ and terminate the loop. If not, choose the successor of v in P_l and check if it is among the open choice, add to its priority the value of 0.5, etc. Continue if either this loop terminates or if the end of P_l is reached.
4. Pass the information to the next decision stage.

The current cognitive position *L* is close to P_l and the next object recommended when P_l is followed by the one of the $N - k$ open choices which has the highest cumulated priority. If there is more than one next choice with the same priority, take a random selection among those. This algorithm ensures that a recommendation to the learner will, if the learner follows the recommendation, propagate through the MCS roughly in parallel to a certain Learning Pathway and at the same time towards this particular Learning Pathway. It is possible to relax this and present to the next higher decision stage a selection of alternative Learning Pathways and the corresponding suggestion for the next KO.

3.3.13 Example for Learner Positions

We now consider a system consisting of four different KOs A, B, C and D. The Multidimensional Cognitive Space then is a four-dimensional hypercube or tesseract. A three-dimensional projection of this hypercube is depicted in Figure 3.7.

The cognitive position of a learner is determined by the amount the learner has learned from each of these four KOs. Hence, the position is determined by a four-dimensional vector within the hypercube. The learner starts at position Ps = (0, 0, 0, 0), which is the lower left of the inner cube of the above projection. His target position is

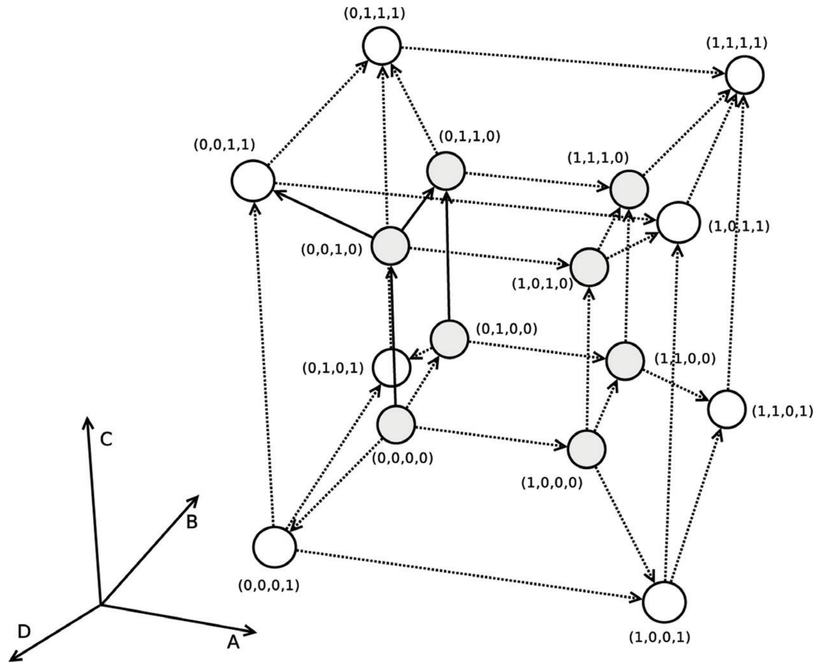

Figure 3.7 Four-dimensional cognitive space (4D hypercube) for four KOs.

Pf = (1, 1, 1, 1), which corresponds to the upper right of the outer cube of the above projection. In principle 4! = 24 distinct Learning Pathways exist and therefore, since the learner is free to choose, may be transgressed by the learner. Not all of these are didactically meaningful; hence let us assume that the KOs are grouped:

- The two pairs A, B, resp., C, D each constitute a thematic group.
- The pairs A, C, resp., B, D each constitute a chronological group. Such a grouping could arise, if
- A describes the contribution of Comenius to educational theory
- B describes the contribution of Habermas to educational theory
- C describes the contribution of Comenius to communicative didactics
- D describes the contribution of Habermas to communicative didactics

Consequently, two Learning pathways are defined in the Cognitive Model:

$$LP_h = (A, B, C, D) = (K_1, K_2, K_3, K_4) \text{ as } \textit{hierarchical} \text{ LP}$$
$$LP_c = (A, C, B, D) = (K_1, K_3, K_2, K_4) \text{ as } \textit{chronological} \text{ LP}$$

These two LPs differ only in one permutation of K_2, K_3. They are depicted by the thick arrows in the projection in Figure 3.8

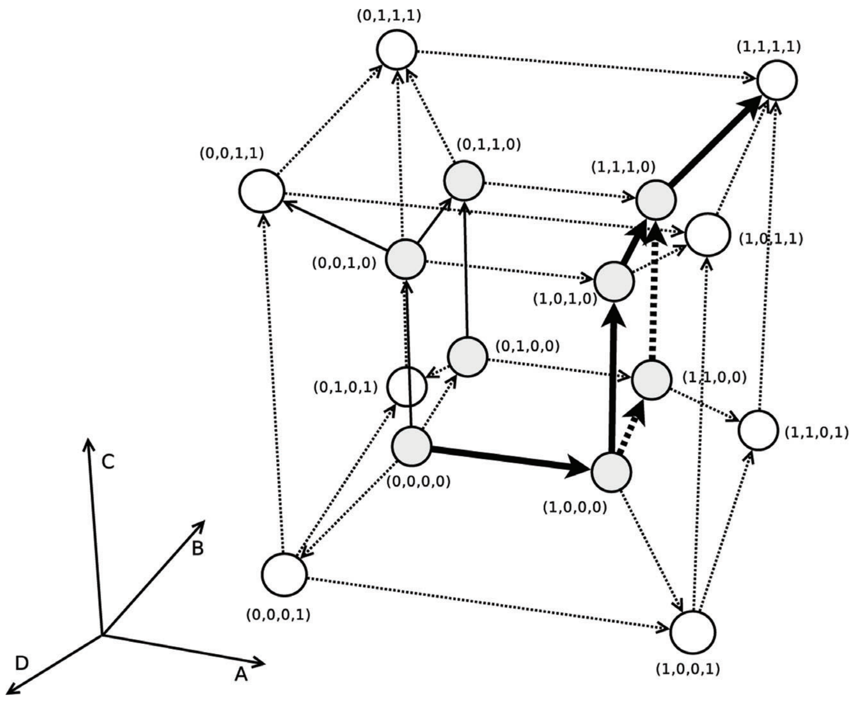

Figure 3.8 The two didactically meaningful learning pathways (hierarchical and chronological), shown as thick arrows.

Let us now follow a learner through this system. First of all let us assume a disciplined learner: The learner follows the initial advice and processes KO A, which puts the learner into cognitive position (1, 0, 0, 0). Obviously, the learner is still following both LPs defined in the Cognitive Model. If no other factor (like a pre-chosen LP) will lead the INTUITEL Engine to another conclusion, the learner will be offered two choices by the INTUITEL system:

> "If you want to follow the hierarchical LP, please process B. If you want to follow the chronological LP, please process C."

Conversely, we assume an undisciplined learner who, instead of following the initial advice to process A, has chosen to process D instead. This would put the learner after the first learning step into cognitive position (0, 0, 0, 1). Using the previously described algorithm, the LPM would determine that the learner is close to position (0, 0, 0, 0) which is on both well-defined LP. The learner would then get the advice to process A as the next KO, bringing the learner (if the advice is followed) into position (1, 0, 0, 1). There, the learner is close to position (1, 0, 0, 0), which is on both LP. Therefore, with equal priority (unless other factors come into play) the learner will be advised to process B or C. Suppose B is followed, this would put the learner into position (1, 1, 0, 1) where the learner would finally get the advice to process C leading to the learning goal. The actual LP then is (*D, A, B, C*) where at least three KO are in the sequence which has been termed adequate by the cognitive engineer.

However, our learner might be so undisciplined that instead of following the first advice the learner takes a complete different route processing C after having done the first step of processing D. This would take the learner into position (0, 0, 1, 1) and the closest of the well-defined LP's is the chronological LP with position (1, 0, 1, 0). The learner would therefore get the advice "You are close to the chronological LP, please process B". If the learner follows this advice, the learner is led into position (0, 1, 1, 1) where then the learner would get the final advice to process A and therefore the overall realized LP would be (*D, C, B, A*). In this realization, only the two KOs C and B are in the chronological sequence described in the Cognitive Model. We might also encounter a totally undisciplined learner, who also does not follow the second advice, e.g., processing first D, then C and then A. This would again bring the learner close to the chronological LP, and the learner would receive the final advice to process B. The actual LP then is (*D, C, A, B*) whereas in the previous scenario, only the two KOs A, B are in the chronological sequence described by the Cognitive Model.

3.3.14 Learner-State Ontology: The Output of the LPM

The output of the LPM consists of two parts. Firstly, there is the output of the actual task that the LPM carries out, the set of applicable Didactic Factors. These are defined in context of the Learning Model Ontology as elaborated in Sections 3.1 and 3.2. These Didactic Factors are essential for the individual learner-related rating of learning objects. The LPM prepares the input and particularly prepares the Didactic Factors that are passed to the Engine. It, for instance, checks the connectivity of the learner and ranks it, to state that the learner has a good or bad connection. This represents the learner-specific knowledge of the reasoner and allows the Engine to include this information in the deductive process.

However, the set of actually relevant Didactic Factors alone is not sufficient for the INTUITEL Engine. An OWL reasoner needs an ontology to perform its work and therefore, the LPM must create one that contains all relevant data, i.e., the second part of the LPM output. This basically is a merge of those meta-data that describe the respective course (including its Cognitive Model), the micro Learning Pathway information from the Pedagogical Ontology and the Learning Model Ontology, containing the machine-readable descriptions of Didactic Factors. To combine this information, the LPM creates the so-called Learner-StateOntology. This only temporarily valid collection depicts the current status of the learner in that course with respect to his or her learning history and current environment. Figure 3.9 illustrates the parts that are merged into the Learner State Ontology as the final output of the LPM.

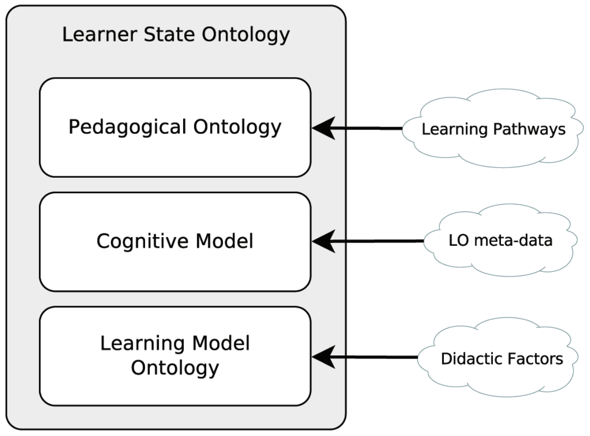

Figure 3.9 The final output of the LPM: the Learner State Ontology.

3.4 Software Architecture

Florian Heberle, Peter A. Henning and Kevin Fuchs

In the previous section we elaborated on the Learning Progress Model as an essential part of the INTUITEL concept. The concrete implementation of the LPM in form of a software component is a central part in the system architecture of INTUITEL. However, it is not the only one and it is embedded into a system landscape that comprises other components like the Learning Management System, a reasoning system and a recommendation component generating user-specific messages. The following section describes the overall system architecture proposed by the INTUITEL consortium.

The architecture of INTUITEL was designed with the focus on modularity and independence of technology. In particular it is insignificant which operating platforms or programming languages are chosen for implementation. Every single component may be implemented completely different from other components as long as the communication between them corresponds to the standard that is elaborated in Section 3.5.

The following describes the sub-components of the INTUITEL system. Note that they do not represent physical components in the first place. In stead they represent logical units that may but do not necessarily need to be reflected in separate software modules. Figure 3.10 shows an overview of the according components.

3.4.1 LMS Integration

The LMS integration component is in charge of tracking a learner within the LMS and informing the INTUITEL system about the learning objects the learner is accessing.

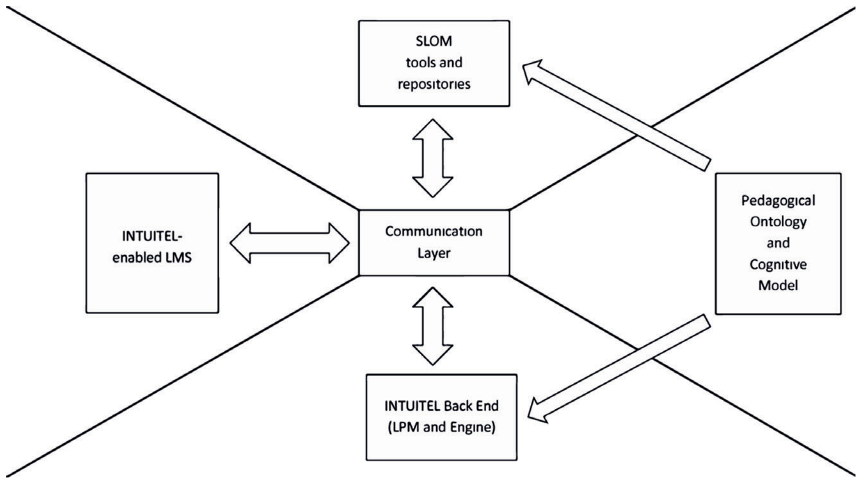
SLOM tools and repositories
INTUITEL-enabled LMS
Communication Layer
Pedagogical Ontology and Cognitive Model
INTUITEL Back End (LPM and Engine)

Figure 3.10 Software architecture.

Moreover the LMS has to provide a web service by which the INTUITEL system can access additional information about the learning environment, the progress of a learner as well as her or his individual characteristics.

Within the INTUITEL project the LMS integration was implemented in the form of plugins for four LMS: Moodle, Ilias, Exact and Crayons. Actually the INTUITEL standard only specifies the web service. It is up to the integrator how the necessary data are retrieved from the LMS and how they are transformed for the web service. In general, you do not even need a classical LMS. It is possible to enhance any learning environment as long as it fulfills the requirements of the web service specification. In Section 4.2, we discuss some example implementations of INTUITEL-compliant LMS enhancements. For example the web service of the LMS plugin must provide the following functionality:

3.4.1.1 Learner update

The INTUITEL back end has to be notified when a learner finishes a learning object or accesses a new one. For this purpose, the LMS plugin must implement a "learner update" mechanism. The LMS may behave in a active or passive way. For this reason, INTUITEL specifies update messages in a push or pull manner. The push scenario means that the LMS sends learner updates in a pro-active way to the INTUITEL back end. In pull mode, the LMS is polled by the back end for updates.

3.4.1.2 User Score Extraction (USE)

The User Score Extraction provides data from the LMS that are related to two aspects of an individual learner:

1. Performance data consist of information about a learner's learning behavior and progress in the past. This includes the completion status of learning objects a learner has attended to as well as specific scores. This data is needed to create personalized learning recommendations based on the learner's learning history and his current cognitive position respectively.
2. Environmental data provide information about the learning environment and the technical conditions under which a learner is working. This becomes especially important in the case of mobile learning which may involve permanently changing noise levels in the surroundings of the learner or temporarily low bandwidth due to mobile network connections. Moreover, environmental data also includes information about a learner's personal characteristics like her cultural background, country of origin, gender, age, physical or mental abilities and disabilities. Consequently such factors have an impact on the subset of recommendable learning objects. All this information may be contained in the database of the LMS or it may be actively prompted from the user via interactive message dialogues.

In concrete, the USE interface provides the data that is needed for the computation of Didactic Dactors (see Section 3.2).

3.4.1.3 Learning Object Recommender (LORE)

The result of the recommendation process taking place in the INTUITEL back end is a set of learning objects a learner is advised to access next. These learning objects are ranked by their levels of suitability according to the learner's learning history as well as her environmental and personal characteristics. The LORE module is responsible for the reception of these recommendations. The LMS integrator can decide freely how the recommendations influence the learner's experience within the user interface of the LMS. The recommendations may be presented as additional information in a pop-up window or an embedded message box. But it may also simply effect the order and priority respectively in which learning objects are listed in the LMS without the user even taking notice of the activities of the INTUITEL system. In Section 4.2 we discuss several examples of LMS integrations.

3.4.1.4 Tutorial Guidance (TUG)

The TUG web service of the LMS is in charge of receiving messages from the INTUITEL system and presenting it to the learner. Like in the case of LORE, TUG messages are the final output of the reasoning process. TUG messages may contain textual learning recommendations processed by a natural language unit that resides within the Recommendation Rewriter Component in the back end. TUG messages are not restricted to plain texts. They may also initiate an interactive dialogue between the learner and the INTUITEL system in order to gather more detailed information for the recommendation process.

For example, as explained before, the User Score Extraction module (USE) may need information about a learner's country of origin or her learning environment. In case that information cannot be gathered from the LMS it can be prompted from the learner directly by sending a respective TUG dialogue. This means that the USE interface can be emulated by the TUG module and that the according implementation of the USE module is not mandatory. The same is true for the previously described LORE interface, which can also be emulated by the TUG module. Therefore, the TUG interface is the minimum requirement that must be implemented by an INTUITEL-enabled LMS while USE and LORE are optional. This allows the implementation of very light-weight LMS plugins. Figure 3.11 illustrates the triggering of a single learner update and the consecutive message sequence between the LMS and the INTUITEL back end.

3.4.2 SLOM Repository

The SLOM Repository contains the Cognitive Models. They are based on the Pedagogical Ontology that we explain in Section 3.1 and describe didactic and pedagogical interrelations between learning objects in a form that can be read and

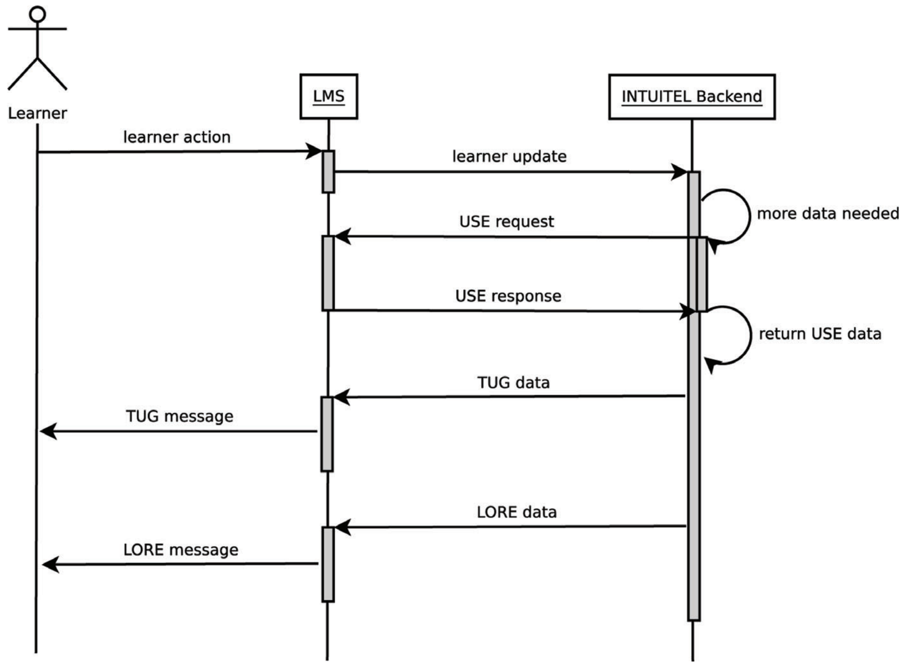

Figure 3.11 Message sequence between LMS and INTUITEL back end.

processed by a computer. In the particular case of INTUITEL it is the Reasoning Engine component that deduces on these ontologies. Also the LPM which is elaborated in Section 3.3 and further described below processes these Cognitive Models.

SLOM itself is basically a container format describing how cognitive models are represented by files. For a more detailed description of SLOM see Section 3.6. The SLOM repository manages these files and provides them for the other components. In order to create the SLOM files, content creators do not depend on specific software tools. Basically these files can even be created by hand with a simple text editor as they contain OWL ontologies in the form of XML statements. However, for larger contents this may not be feasible. Therefore, for the prototype, we have also implemented an editor with which arbitrary learning material can be imported from an LMS, annotated with the respective metadata and exported into SLOM files. Since all this is based on standard OWL, creators may also use any other OWL-compliant editor. While using standard OWL editors require knowledge in the field of OWL, our INTUITEL editor does not expect the creator to have such particular skills. The creator can create interrelations between learning objects by just drawing them in a drag-and-drop environment on the screen.

3.4.3 INTUITEL Backend

The INTUITEL back end comprises four sub-components which are the Learning Progress Model (LPM), the Query Builder, the Reasoning Engine and the Recommendation Rewriter. In the subsequent sections each of these components is explained.

3.4.3.1 LPM

The Learning Progress Model, as described in Section 3.3 plays a central role in the entire INTUITEL system and is implemented in the form of a respective back end component. It is responsible for all tasks that are needed to retrieve, analyze and transform the entire data set that is needed for the recommendation process. Any action that takes place in a learning environment is first processed by the LPM before passing any other component of the INTUITEL back end. In the following we explain the tasks which the LPM is in charge of.

Tracking of user actions In order to create any recommendations at all, the fundamental first step is to track a user's actions within the LMS. The LPM yields this information from the INTUITEL-enabled LMS via "learner update" messages. For this purpose the LPM can be configured due to multiple "communication styles". Depending on the respective LMS plugin, the required information is either sent actively from the LMS to the LPM each time a learner navigates to a learning object or performs any other relevant action like passing a test. Another method is having the LPM actively poll the LMS while the LMS itself remains in a reactive mode. Furthermore the LPM retrieves performance data from the LMS via the USE interface.

This data includes information about how well the learner has performed on particular KOs. Also using the USE interface, The LPM asks the LMS for additional user-specific information that are useful for the refinement of the recommendation process. This may be environmental data informing about the technical device the learner is working with, information about the surroundings of the learning environment or individual aspects like disabilities, cultural background or the gender of the learner.

Calculation of Didactic Factors The performance data; the environmental data and the information about a learner's individual characteristics form the combined input for the creation of Didactic Factors. Didactic Factors have an important impact on the recommendation process. Briefly recapitulating the elaborations in Section 3.2, the calculation of didactic factors consists of retrieving arbitrary non-nominal data from the LMS and transforming this data into a nominal representation of an entity within the ontologies that constitute the cognitive model. This way, the information is integrated into the ontologies and becomes processable for the reasoning engine.

Determining the learner's cognitive position The cognitive model, as derived from the pedagogical ontology describes the cognitive space in which the learner may move. Within this space the history of learning objects he has attended to combined with respective progress and achievements defines a vector we call the cognitive position of the learner. A key task of the LPM is to determine that cognitive position. Knowing the learner's current cognitive position, the INTUITEL engine is capable of comparing this position to The Learning Pathways that have been predefined within the Pedagogical Ontology by a didactic expert. By this a Learning Pathway can be determined that is closest to the current cognitive position.

Preparing data for the reasoning process Having collected all necessary data the LPM transforms them into a format that can be understood and processed by those components that are in charge of the reasoning process. For this purpose, all the data is packed into an ontology compatible to the pedagogical ontology and passed to the next stage which is the Query Builder.

3.4.3.2 Query builder

After all necessary data has been retrieved analyzed and transformed by the LPM, the Query Builder is the first entry point to the actual reasoning process. The main task of the Query Builder is first building a query for the next stage the Reasoning Engine and second optimizing that query with respect to specific requirements of the reasoner.

3.4.3.3 Reasoning engine

In the preceding, we described the LPM and how it creates an ontology firstly from the pedagogic ontology and secondly from the calculation of didactic factors. In that sense the main purpose of the LPM is to merge two domains of meta-knowledge into one ontology: knowledge about the didactic and pedagogical concept underlying a course and knowledge about a learner's learning history as well as individual characteristics.

Being first processed by the Query Builder, which output ontology forms the input for the reasoning engine. In principle INTUITEL was designed to not only run one reasoning engine but multiple instances in parallel due to scalability. By default, INTUITEL provides one basic reasoning engine to cover the most common scenarios. In case multiple instances of reasoning engines are used, the architecture of INTUITEL allows using a reasoning broker as a preliminary authority before the reasoning engine. That broker is responsible for delegating different queries it receives from the Query Builder to the respective reasoning instance.

The reasoning process itself has to be understood in the following terms. The reasoner reads the ontology originally created by the LPM. Deducing from this ontology a reduction process on all available learning objects is performed. Within this reduction process, the rules and interrelations of learning objects are applied. At the end a subset of those learning objects remains that satisfies the rules encoded by the ontology.

3.4.3.4 Recommendation rewriter

Once the reasoning output has finished its task, it passes the result to the Recommendation Rewriter. As its name tells it "rewrites" the recommendation created by the reasoner into a human-friendly format. The result of this transformation are the ones previously explained TUG and LORE messages that are sent to the LMS plugin. The LORE messages contain a ranked list of suitable learning objects according to the reduction process of the reasoner. The TUG messages represent text messages that are generated by the help of a special natural language unit.

3.4.4 Data Exchange

The architecture of INTUITEL is formed by a modular design that requires only a loose coupling of the single components. The basis of the architectural design is a clear definition of web services that represent end points for the communication between the previously described components. Provided that these web services are implemented correctly, it is completely irrelevant which technologies are used to implement the single components. As long as the web services facilitate the correct data flow the functionality of the system is guaranteed. The way the INTUITEL system was implemented by the authors is therefore just one possible solution of a prototype and can be considered as a recommendation that may be modified according to particular requirements. Moreover, an INTUITEL implementation may be distributed running each component on its own physical platform or it may be completely or even partially integrated into one physical system. Altogether, this ensures a maximum of interoperability.

In order to decouple the single components from each other a fifth component is specified by the INTUITEL recommendations which are the Communication Layer. The Communication Layer is in charge of receiving and routing messages from components to the respective web services of the other components. Hereby the communication Layer hides the single components from one another and each component only has to address the Communication Layer as a single end point. A typical message sequence to generate a learning recommendation that is triggered by a learner accessing a new learning object from within the LMS is sketched below.

1. By accessing a new learning object, the learner triggers the LMS to send an update message to the Communication Layer (learner update) The Communication Layer routes this message to the component it considers being responsible which – in this case – is the LPM.
2. Through the Communication Layer the LPM contacts the SLOM repository and asks for appropriate SLOM data that are associated with the learning object the learner has just accessed. If needed, the LPM requests additional information from the LMS which includes data about the learning environment, performance data on the learner's progress so far and data concerning individual characteristics and preferences of the learner. This data is then processed resulting in a query that is passed to the next authority: the Query Builder.
3. The Query Builder takes the data it receives from the LPM and prepares an according query for the next stage – the Reasoning Engine.
4. The Reasoning Engine performs the reasoning process on the data it has received from the Query Builder and sends the result to the Recommendation Rewriter.
5. The Recommendation Rewriter is the last authority in the recommendation process. Taking the data it receives from the Reasoning Engine, the Recommendation Rewriter uses a natural language unit to create humanlike recommendation messages including ranked lists of related learning objects fitting most the current cognitive position of the learner.
6. Finally the Recommendation Rewriter sends LORE and TUG messages to the LMS.

Note that the recommendation process is triggered every time a learner accesses a new learning object. This is shown by Figure 3.12 which illustrates the above sketched message sequence. In the next section the data model of these messages is explained in more detail.

3.5 Data Model

Peter A. Henning and Florian Heberle

In the previous section we presented an overview of the INTUITEL architecture. We regard this as a general instruction on how to implement an INTUITEL system without any respect or limitations to technology and concrete software implementation.

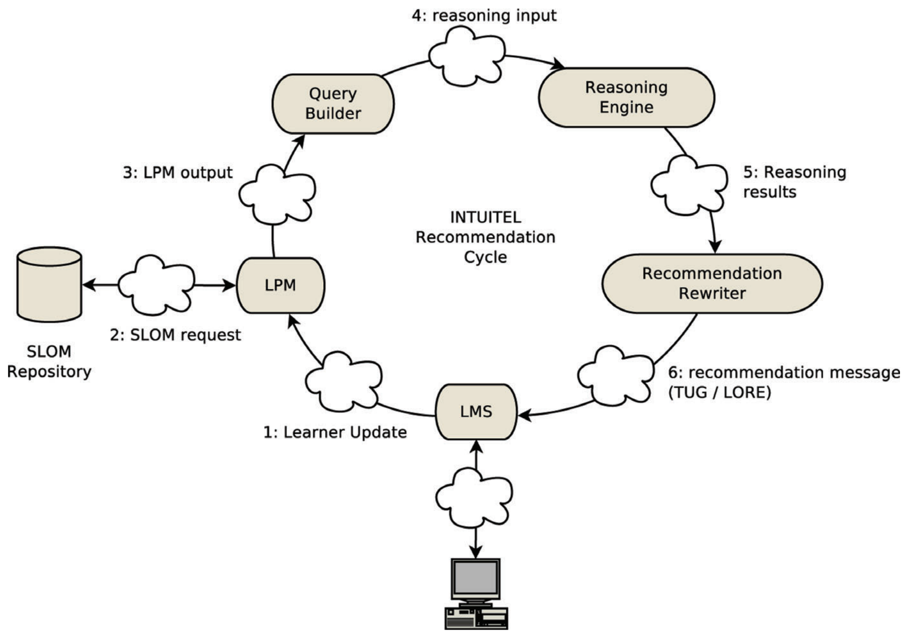

Figure 3.12 INTUITEL recommendation cycle.

The only requirement we claim is the way that the single components communicate with each other. In the following we give a clear definition of the communication interfaces. Any implementation of the INTUITEL system will work as long as these interfaces are implemented correctly.

The communication between the LMS and INTUITEL is implemented as a RESTful web service on both sides. REST (Representational State Transfer) is a series of design guidelines for web services, hence not a strict way to express communication paradigms. Table 3.3 exhibits a list of URLs that is used in this web service. It is

Table 3.3 INTUITEL end points

URL	Description
/lmsprofile	get some basic information about the LMS
/learners	Learner Update: provides information about learner actions
/login	used for annotation purposes: login as a course administrator
/mapping	used for annotation purposes: get a list of available learning objects
/TUG	Tutorial Guidance message
/LORE	Learning Objects Recommendation message
/USE/performance	get performance data for a specific learner
/USE/environment	get environmental data for a specific user

advised to run the communication over a proxy component – the Communication Layer as elaborated in Sections 3.4 and 4.1.

In the sequel we illustrate the data that is exchanged over these end points in more detail. Note that the following elaborations are supposed to give the reader a basic understanding of how the data exchange works. For a more detailed insight – especially if you intend to implement an INTUITEL system on your own – please refer to the technical documentation in the respective Deliverables of the INTUITEL Consortium [27, 28].

Initialization

The INTUITEL system is configured to interact with a certain LMS. When it is started, it will ask the LMS for its capabilities by a message which has the XML format:

```
<INTUITEL>
  <LmsProfile mId="some uuid"/>
</INTUITEL>
```

The parameter *mId* is a globally unique message ID. The LMS will respond with XML formatted data describing its capabilities and settings. The response contains the following data:

- A globally unique message ID, abbreviated as mId. This is the same message ID as in the request.
- A list of attribute-value pairs that provide information about the LMS. for detailed information see Table 3.4.

It will be returned as XML document according to the following template:

```
<INTUITEL>
   <LmsProfile mid="some uuid">
      <Data name="some name" value="some value"/>
      <!-- Repeated if necessary -->
   </LmsProfile >
</INTUITEL>
```

Learner Update

After initialization, the INTUITEL system will react on learner navigation events. This means that INTUITEL will create recommendations (LORE and TUG messages) for learners, when they open a LO. To notify INTUITEL of such a LO transition, the LMS integrators can chose between three different methods. This allows them to select the most suitable one in respect to the technical possibilities and guidelines of their respective LMS.

Table 3.4 Attribute value pairs for the initialization response

Data Item Name	Description
lmsName	Installation name of LMS instance, e.g. “HS Karlsruhe”
lmsType	Type of LMS instance, for example “clix”, “crayons”, “exact”, “ilias”, “moodle”, “other”
lmsId	Unique ID that allows identifying this particular LMS instance, like geolocation of LMS or registration number of LMS instance (may be left empty).
lmsMediaLevel	Level of multimedia support as a collation of characters: v = video, a = audio
comStyle	Specifies the method used for learner updates and how INTUITEL sends LORE and TUG messages: 0 = Pull, 1 = Push, 2 = Push with learner polling (see Section 3.5)

Pull Update

The objective of this scenario is to support LMSs that do not permit the usage of technologies that allow refreshing the LORE and TUG windows in the learners browser asynchronously (e.g., in a LMS that prohibits the usage of JavaScript). The LMS can thus include all necessary data in just one response to the learner. The learner update message sent by the LMS to INTUITEL contains XML-data in the following format:

```
<INTUITEL>
    <Learner mId="uuid" uId="user ID" loId="LO ID" time="access time"/>
</INTUITEL>
```

The Learner-element contains four attributes with the following meaning:

- A globally unique message ID, abbreviated as mID.
- The user ID which identifies the learner in the LMS, abbreviated as uId. This uId is provided by the concrete LMS installation and may vary according to organizational policy.
- The LMS specific Learning Object identifier, abbreviated as loId. This specifies the LO the learner just opened as new LO to work on.
- The point in time the learner accessed the LO in data type long, abbreviated as time.

INTUITEL’s response to the LMS contains XML-data as described in the following template:

```
<INTUITEL>
    <Learner mId="some uuid">
    <Lore uId="some user ID" mId="some uuid"> . . . </Lore>
    <Tug uId="some user ID" mId="some uuid"> . . . </Tug>
</INTUITEL>
```

3.5.1 Push Update

In the push-scenario, the LMS sends a learner update message to INTUITEL when a learner opens a new LO. As in the pull use case, the INTUITEL back end reacts and creates a recommendation. However, the LMS will not get a direct response to the learner update message. It is just a notification for INTUITEL, triggering the reasoning process. The moment LORE or TUG data is available for a learner, the INTUITEL system sends individual messages to the LMS, which can then be forwarded to the learner.

This scenario is designed for LMSs that want to decouple the standard LO processing from INTUITEL. By using this approach, the LORE and TUG data can be loaded asynchronously in the learner's browser (e.g., by using JavaScript and AJAX) after receiving them from the INTUITEL back end. So, at the point in time when the TUG and LORE data is presented to the learner, he or she already started viewing the actual LO content.

The learner update message sent to INTUITEL contains XML-data in the format as specified in the pull scenario above. In order to afterwards assign the responses to the correct learner update message in the LMS, its respective mId is implemented in the LORE and TUG responses as rId (request ID). This ensures that each data item has a unique identifier and that the responses can be attributed to the messages which triggered their creation.

3.5.2 Push Update with Learner Polling

This method is similar to the previously described push approach. The important difference is that INTUITEL actively requests the learner update notifications via polling. Therefore, INTUITEL will ask the LMS in specific time intervals if a new learner update is available. If a LO transition is noticed in that process, the INTUITEL system will carry out the same workflow as in the standard push scenario.

Because the LMS does not have to send the learner updates to INTUITEL, it can act as an only receiving endpoint in the communication with INTUITEL. This allows LMSs that do not support active signaling, to still be able to use the functionalities of INTUITEL. However, this also includes that the learner update messages are adapted to fit to this use case. The messages sent to the LMS by INTUITEL contain XML-data with the following format:

```
<INTUITEL>
    <Learners mId="some uuid"/>
</INTUITEL>
```

As response to this message, the LMS will add a list of XML-elements describing the currently active learners. The mId of the learners-element is the same as in the request by INTUITEL. For each learner logged into the LMS this list will contain:

- A user ID which identifies the learner in the LMS, abbreviated as uId.
- A list which identifies the Learning Objects visited by this particular learner since the last poll. Each entry will also contain the LMS specific Learning Object id (loId) and the point in time the learner accessed the LO as long.

The data will be returned as XML document according to the following template:

```
<INTUITEL>
  <Learners mId="some uuid">
      <Learner uId="some user ID">
          <VisitedLo loId="LO ID" time="access time"/>
          <!-- – Repeated if necessary – -->
      </Learner>
      <!-- – Repeated if necessary – -->
  </Learners>
</INTUITEL>
```

As in the previously described push scenario, TUG and LORE messages contain the mId of this learner update as their rId. This allows the LMS to link the respective recommendation data to the learner update message, initially triggering the respective reasoning process.

3.5.3 Learning Objects Mapping and Inventory

The LMS provides a method to associate its internal content with the external INTUITEL metadata stored in the SLOM metadata repository (see Section 3.6 on the SLOM meta-data model). To this end, the INTUITEL Editor addresses the LMS by a message which has the following XML format:

```
<INTUITEL>
    <LoMapping mId="some uuid">
        <Data name="some name" value="some value"/>
        <!-- –  Repeated if necessary – -->
            </LoMapping>
    <!-- – Repeated if necessary – -->
</INTUITEL>
```

Parameters are:

- A globally unique message ID, abbreviated as mId.
- Optionally: A collection of search strings in the form of attribute-value pairs which are used to narrow down the search results. For a detailed listing of the attribute-value pairs see Table 3.5.

Table 3.5 Attribute-value-pairs for the LO mapping

Data Item Name	Description
loName	Learning Object title or name (mandatory for response)
loId	Learning Object identifier as provided by the LMS (mandatory for response)
hasParent (if hierchical)	Learning Object identifier of parent LO as provided by the LMS (mandatory for response)
hasPrecedingSib (if sequential)	Learning Object identifier of preceding sibling LO as provided by the LMS (mandatory for response)
hasChild (if hierarchical)	Learning Object identifier of child LO as provided by the LMS, possibly multiple ones per LO
hasFollowingSib (if sequential)	Learning Object identifier of the following sibling LO as provided by the LMS
lang	language code of the LO
loType	Learning Object type as specified in the pedagogical ontology, e.g. "test"
learningTime	Typical time to work through this object
typicalAgeL	Typical learner age (lower boundary)
typicalAgeU	Typical learners age (upper boundary)
media	Comma-separated list of media types contained in the LO: "text", "image", "audio", "video", "interactive". To be extended if need arises
size	Size of the LO in terms of Kilobyte
getFullCourse	Attribute specifying that a full course mapping including all LO children and siblings shall be returned.

For each LO, the following data will be returned:

- The title or name (loName) and Learning Object ID (loId) of the particular object(s) found in the LMS and matching the search criteria.
- The position of the LO in the internal course structure:
  - *Hierarchical structure*: The LO's parent LO by specifying the attribute "hasParent" (if the LO is not on top level).
  - *Hierarchical structure*: The LO's child LO by specifying the attribute "hasChild" (if the LO is not on bottom level).
  - *Sequential structure*: The previous LO in the given sequence by specifying the attribute "hasPrecedingSib" (if the LO is not first in the LO sequence).
- Further meta-data obtainable from the LMS on this LO. For a detailed explanation consider Table 3.5.

The data will be returned as an XML document according to the following template:

```
<INTUITEL>
    <LoMapping mId="some uuid">
```

```
        <Data name="loName" value="some LO name"/>
        <Data name="loId" value="some LO ID"/>
        <Data name="hasParent" value="some LO ID"/>
        <!-- Repeated if necessary -->
        <!-- Order of data items is not relevant -->
    </LoMapping>
    <!-- Repeated if necessary -->
</INTUITEL>
```

3.5.4 Authentication

Please note that this authentication is not necessary for ordinary learners using an INTUITEL enabled LMS. It is for content creators who want to make use of the INTUITEL system to edit LO meta-data in the INTUITEL Editor.

To log a user into the INTUITEL system, the LMS will receive a message from INTUITEL containing the following parameters:

- A globally unique message ID (128 bit according to UUID standard), abbreviated as mId.
- A user ID which identifies the learner in the LMS, abbreviated as uId. This uId is provided by the concrete LMS installation and may vary according to organizational policy.
- A password abbreviated as Pass.

Each of these messages is an XML fragment enclosed in XML-elements named "Authentication" according to the following template:

```
<INTUITEL>
    <Authentication uId="some user ID" mId="some uuid">
        <Pass>some password</Pass>
    </Authentication>
</INTUITEL>
```

The LMS returns an XML document containing acceptance codes enclosed in XML-Elements as seen in the following template:

```
<INTUITEL>
    <Authentication uId="user ID" mId="uuid" status="OK or ERROR">
        <LoPerm loId="some LO ID"/>
        <!-- Repeated if necessary -->
    </Authentication>
</INTUITEL>
```

In each of these, the message ID and the user ID are identical to those contained in the original request. Contained in the Authentication-tag is a list of all courses that may be edited by this user as LoPerm-elements (Learning Object permissions). If the collection is empty, there are no courses available for editing for this particular user.

3.5.5 Tutorial Guidance – TUG

The TUG interface carries out a dialog between the learner and the INTUITEL system. The primary use case for the TUG interface occurs, when the INTUITEL system sends a message to a learner which contains the following structured information:

- A globally unique message ID, abbreviated as mId.
- A user ID which identifies the learner in the LMS, abbreviated as uId.
- A field containing a message type, which is a 4-digit number, abbreviated as MType. Please refer to Table 3.6 for a listing of the available types.
- A field containing the structured message data in XML format, abbreviated as MData.
- A globally unique request ID, abbreviated as rId. This rId specifies the ID of the message which triggered the creation of this TUG message.

Each of these messages is an XML fragment according to the following template:

```
<INTUITEL>
  <Tug uId="some user ID" mId="some uuid" [rId="some uuid"]>
    <MType>message type</MType>
    <MData>message data</MData>
  </Tug>
  <!-- Repeated if necessary -->
</INTUITEL>
```

Table 3.6 Minimum requirement for message types in the TUG interface

Mtype	Description
1	Simple message, not important. Represented as Text, if necessary with HTML formatting
2	Simple message, important. Represented as Text, if necessary with HTML formatting
3	Simple question, to be answered Yes/No. Represented as Text, if necessary with HTML formatting
4	Single choice question, to be answered with one out of n alternatives. Represented as Text, if necessary with HTML formatting; n different text pieces in structured writing
5	Multiple choice question, to be answered with any number out of n alternatives. Represented as Text, if necessary with HTML formatting; n different text pieces in structured writing

3.5.6 Immediate Response from the LMS

If the XML document containing one or more TUG messages has been received by the LMS, it will return an XML document containing one or more acceptance codes as seen in the following template:

```
<INTUITEL>
  <Tug uId="user ID" mId="uuid" retVal="OK, PAUSE or ERROR"/>
  <!– – Repeated if necessary – –>
</INTUITEL>
```

In each of these, the message ID and the user ID are identical to those contained in the original TUG message from INTUITEL. The acceptance code (retVal) is part of the fragment body and reads:

- OK: The learner addressed by the uId is known to the system and logged in. The TUG message has been accepted.
- PAUSE: The learner addressed by the uId is known to the system but not logged in. He may log in after pausing for some time according to his personal learning schedule. The TUG message has temporarily been rejected. The INTUITEL system will have to resend and possibly update it at a later point in time.
- ERROR: The learner addressed by the uId is not known to the system. The TUG message has been rejected.

3.5.7 Delayed Response from the Learner

If the state of the return message for a particular learner is "OK", the LMS presents on the screen of this particular learner a message window or a similar widget. The exact implementation is left to the individual LMS. This window shall be displaying a message or asking for learner action, e.g., asking the learner to fill in text, click boxes etc.

It cannot be guaranteed, that a particular learner will process a TUG message at all or within a certain response time. If a TUG message has been answered by the learner by pressing the "Submit" button, even after some arbitrary response time, this will be communicated to the INTUITEL system. Therefore, a message from the LMS to the INTUITEL system will be prepared according to the following template:

```
<INTUITEL>
  <Tug uId="some user ID" mId="some uuid">
    <Data name="some name" value="some value"/>
    <!– – Repeated if necessary – –>
  </Tug>
  <!– – Repeated if necessary – –>
</INTUITEL>
```

3.5.8 Learning Object Recommender – LORE

Recommendations are produced as part of the didactic reasoning process in the INTUITEL Engine and are sent to the LMS. The main use case for the LORE interface therefore occurs, when the INTUITEL system recommends to the learner a list of Learning Objects and their priority containing the following structured data:

- A globally unique message ID, abbreviated as mId.
- A user ID which identifies the learner in the LMS, abbreviated as uId.
- A list of elements containing the Learning Object identifier (abbreviated as loId) paired with the allocated priority. This priority is considered to be an integer value from the interval (0, 100) denoting the percentage of relevance.
- A globally unique request ID, abbreviated as rId. This rId specifies the ID of the message which triggered the creation of this TUG message.

Each of these messages is an XML body according to the following template:

```
<INTUITEL>
          <Lore uId="some user ID" mId="some uuid" [rId="some uuid"]>
      <LorePrio loId="LO id" value="a value between 1 and 100"/>
      <!– – Repeated if necessary – –>
  </Lore>
</INTUITEL>
```

All other LOs in that particular course, which are not contained in the LORE element, indirectly get a priority value of zero. So, each time the LMS receives a new LORE message, the previous recommendation can be rated as outdated. If the XML document containing one or more LORE messages has been received by the LMS, it responds with an XML document as described in the following template:

```
<INTUITEL>
  <Lore uId="user ID" mId="uuid" retVal="OK, PAUSE or ERROR"/>
  <!– – Repeated if necessary – –>
</INTUITEL>
```

In each of these LORE elements, the message ID and the user ID are identical to those contained in the LORE message from INTUITEL. The acceptance code (retVal) is part of the fragment body and reads:

- OK: The learner addressed by the uId is known to the system and logged in. The recommendation has been accepted.
- PAUSE: The learner addressed by the uId is known to the system but not logged in. He may log in after pausing for some time according to his personal learning schedule. In this case, no further action is taken because INTUITEL

recommendation should pause as well. The recommendation has temporarily been rejected. The INTUITEL system will have to resend and possibly update it at a later point in time.

- ERRROR: The learner addressed by the uId is not known to the system. The recommendation has been rejected.

3.5.9 User Score Extraction – USE

The USE interface provides the INTUITEL system with personal data from the LMS concerning a particular learner. Note that the USE interface has two important uses, because it may be used to get information about the learner's performance, as well as on the environment.

3.5.9.1 Primary use case: Performance data request

The main use case of the USE interface is to request information about a learner's performance according to specified LOs. This allows INTUITEL to include the learner's current skill level in the reasoning process to create recommendations with a much higher level of personalization.

To receive personal learning progress related data from the LMS, INTUITEL sends a USE request specifying the needed data. Therefore, the UsePerf-element contains the following structured data:

- A globally unique message ID, abbreviated as mId.
- A user ID which identifies the learner in the LMS, abbreviated as uId.
- Optionally: A list of elements (LoPerf) specifying Learning Object IDs, abbreviated as loId.

Each of these messages is an XML body according to the following template:

```
<INTUITEL>
    <UsePerf uId="some user ID" mId="some uuid">
        <LoPerf loId="some LO ID"/>
        <!– – Repeated if necessary – –>
    </UsePerf>
    <!– – Repeated if necessary – –>
</INTUITEL>
```

If the loId-attribute is left blank, the LMS shall return all available LO scores for that particular learner. This shortcut will be used to get all available learner scores in one request when learners start their very first learning session with INTUITEL. If the XML document containing one or more USE messages has been received by the LMS, it will return an XML document containing a body as seen in the following template.

```
<INTUITEL>
    <UsePerf uId="some user ID" mId="some uuid"/>
        <LoPerf loId="some LO ID">
            <Score type="score type" value="score value"/>
            <!-- Repeated if necessary -->
         </LoPerf>
         <!-- Repeated if necessary -->
    </UsePerf>
    <!-- Repeated if necessary -->
</INTUITEL>
```

In each of these, the message ID and the user ID are identical to those contained in the original request. Since there are multiple different types of scores, the contained Score-element further specifies the meaning of the respective value. Which type of score will be relevant for a LO varies depending on the respective LO type and LMS. Not every LMS collects the same amount of data. Consequently, INTUITEL does not require the LMS to fulfill certain requirements concerning these USE scores. A concrete INTUITEL implementation may for example provide the following score types:

- *grade*: The grade a learner achieved in a LO. Values are expected to be within [1, 6].
- *completion*: The completion status of the LO in terms of a percentage value, should be used in context on textual LOs.
- *seenPercentage*: The percentage of the LO which has been seen by the learner. It should be used in context of video and audio LOs.
- *accessed*: Boolean value specifying whether a learner has already accessed a LO (true) or not (false).

It is up to the developers to implement any other score types.

3.5.9.2 Secondary use case: Environmental data request

The second use case of the USE interface is to request information about learners themselves and about their current environmental situation. This allows INTUITEL to further personalize the TUG messages and the recommendation creation. Furthermore, this allows the back end to include the situational aspects (location, device, . . .) of the learner's current learning session in the reasoning process. The environmental USE request to the LMS contains the following structured query parameters:

- A globally unique message ID, abbreviated as mId.
- A user ID which identifies the learner in the LMS, abbreviated as uId.
- Optionally: A data item name specifying the requested data item (see Table 3.7).

Each of these messages is a XML body according to the following template:

Table 3.7 Parameters describing the learning environment of the learner

Data Item Name	Description
lName	Full name of the learner, which will be used in context of TUG-messages
lGender	Learner's gender
lAge	Learner's age
lCulture	Learner's culture. Values are "asian", "caucasian", "african"
lAttitude	Current emotional attitude of the learner. If natively available via LMS; e.g. analysis of facial expression
eNoiseLvl	Average noise level as measured by the microphone of the device
eTime	Current daytime of the learner
dType	Currently used type of device: "desktop", "laptop", "tablet", "smartlet", "phone"
dConType	Type of connection used to access the LMS: "wire", "lte", "hsdpa", "umts", "edge"
dConStab	Number specifying the stability of the internet connection, value between 0 and 100
dRes	Screen size of the currently used device (horizontal x vertical)
dBattery	Current battery state in percentage value

```
<INTUITEL>
    <UseEnv uId="some user ID" mId="some uuid">
          <Data name="some name"/>
          <!-- – Repeated if necessary – -->
      </UseEnv>
    <!-- – Repeated if necessary – -->
</INTUITEL>
```

If the XML document containing one or more USE messages has been received by the LMS, it will return an XML document containing a set of different data items, as described in the following template:

```
<INTUITEL>
    <UseEnv uId="user ID" mId="uuid" retVal="OK, PAUSE or ERROR">
        <Data name="some name" value="some value"/>
        <!-- – Repeated if necessary – -->
    </UseEnv>
    <!-- – Repeated if necessary – -->
</INTUITEL>
```

In each of these, the message ID and the user ID are identical to those contained in the original request. The attribute-value pairs in the "Data" tag correspond to the items listed in Table 3.7. The acceptance code is part of the fragment body, the retVal string reads:

- OK: the learner addressed by the uId is known to the system and logged in.
- PAUSE: the learner addressed by the uId is known to the system but not logged in. He may log in after pausing for some time according to his personal learning schedule. In this case, no further action is taken because INTUITEL recommendation should pause as well.
- ERROR: the learner addressed by the uId is not known to the system. The response is dependent on the retVal and the specified Data-element in the request.
- In case of PAUSE or ERROR, the "UseEnv" element is empty.
- In case of OK and if one or more explicit data items were contained in the request, only data for these items is returned.
- In case of OK and if no explicit data item was contained in the request, all currently available environmental data items are returned.

3.6 The Semantic Learning Object Model SLOM

Stefan Zander and Florian Heberle

In addition to the data model for the communication between the INTUITEL components, which we explained in the previous section, we now briefly introduce the Semantic Learning Object Model (SLOM). SLOM is a new meta-data model developed within the INTUITEL project that provides a semantic web-based knowledge representation framework for the interlinkage of pedagogical and domain-specific knowledge with concrete learning material. SLOM's objective is to complement existing and well-known eLearning formats such as Sharable Content Object Reference Model (SCORM) and IMS-Learning Design (IMS-LD) with semantic information. This allows INTUITEL-enabled Learning Management Systems to process learning material represented as SLOM in more intelligent and personalized way and adapt internal decision making processes to the personal preferences and learning style of learners. It further serves as facilitating data infrastructure for the utilization and integration of externally hosted data in INTUITEL-compliant learning material.

SLOM also provides a data format specification in which the INTUITEL system stores general information about courses. This is a necessary prerequisite for the computation of learning recommendations and personalized feedback for learners. The SLOM format is implemented as a direct extension of the Pedagogical Ontology and provides additional language primitives together with a prescribing schema that determines how course information needs to be described and annotated in order to be compliant to an INTUITEL system. SLOM as a metadata format encloses two different ontologies with varying granularity:

- the Cognitive Map (CM) and
- the Cognitive Content Map (CCM).

While the CM provides terms for the description of topics in a knowledge domain of a course, the CCM serves as description framework for the actual learning material for a given course. A CM is intended to be universally valid and thus can be reused across different courses pertaining to a given topic. CCMs, in contrast, are specific to a given course since it represents the actual learning content.

In its second function as a storage format, SLOM provides a container structure for the original learning material the CCM is based upon. The storage structure specification of SLOM prescribes the structure in which compliant learning material needs to be stored. This entails three main pillars of information that should be compiled into the CM/CCM from the original content format:

1. Topology: The topological pillar contains information about the elements that constitute concrete learning material plus their topological structure and coherence. In terms of an INTUITEL system, a SLOM content package contains definitions for CC and KOs of a given domain of knowledge.
2. Sequences: Sequences are represented as Learning Pathways (LPs) and define the interlinkage of KOs and CCs on different levels of granularity. They are a main pillar of the INTUITEL approach and provide course instructors and designers the opportunity to orchestrate learning content in different didactically meaningful ways. Macro Learning Pathways (MLPs) define the sequence through which learners should work through the CCs of a course. Micro Learning Pathways (mLPs) define sequences how learners should work through KOs – the actual contents of a course. The total amount of possible learning pathways is the result of a combination of Macro and Micro Learning Pathways. While only a manageable set of Learning pathways is modeled by a course instructor or designer, the total amount of didactically meaningful ways to proceed through a given course is the product of them. In technical terms, learning pathways are represented as directed acyclic graphs, otherwise they could not be processed by a reasoner. A detailed description of this was given in Sections 3.1 and 3.3.
3. Technical and Educational Information: The third pillar of information refers to both CCs and KOs likewise: It concerns concrete information about how the content of KOs is to be displayed in terms of technical conditions (e.g., recommended screen resolution, color depth of an image, etc.) as well as educational metadata reflecting pedagogical purpose and background. Such annotations could comprise information about appropriate difficulty levels (e.g., beginner, intermediate level, expert) as well as the knowledge type a KO contains (e.g., varying assignment types).

A combination of these three pieces of information with learner data allows the INTUITEL system to create recommendations for appropriate learning objects and to produce feedback messages in that process. The more information that can be provided on the course, the more information can be integrated in this process. For more detailed information – especially the technical implementation of SLOM – refer to Deliverable 4.1 [29] of the INTUITEL consortium.