Deliverable N°: D10 (Revised version)

Student Model Specification

The LEACTIVEMATH Consortium

June 2005

Version 2

Main Authors:

Eric Andrès, Paul Brna, Nicolas Van Labeke, Manolis Mavrikis, Rafael Morales, Helen Pain, Kaśka Porayska-Pomsta

Project funded by the European Community under the Sixth Framework Programme for Research and Technological Development

Project ref.no.	IST-507826
Project title	LEACTIVEMATH- Language-Enhanced, User Adaptive, Interactive eLearning for Mathematics

Deliverable status	Restricted
Contractual date of delivery	December 31st 2004 (Month 12)
Actual date of delivery	May 20th 2005
Deliverable title	Student Model Specification
Type	Report
Status & version	2
Number of pages	133
WP contributing to the deliverable	WP4
WP/Task responsible	T4.1, T4.2, T4.3
Author(s)	Eric Andrès, Paul Brna, Nicolas Van Labeke, Manolis Mavrikis, Rafael Morales, Helen Pain, Kaśka Porayska-Pomsta
EC Project Officer	Colin Stewart
Keywords	

Contents

Executive summary

This document provides a specification of the Student Model as required by the workplan for LEACTIVEMATH. The specification relates Work Package 4 Student Model (WP4) to the other LEACTIVEMATH components and describes its subcomponents. The Work Package main deliverable, the student modelling subsystem of LEACTIVEMATH, is renamed as the "**E**xtended **L**earner **M**odel" subsystem (xLM) and is composed of four subcomponents named Learner Model (LM), Learner History (LH), Situation Model (SM) and Open Learner Model (OLM).

This specification conforms to the requirements generated by the project, as in Deliverable D5 (LeActiveMath Partners, 2004c). For the benefit of the reader, all requirements referenced in this document can be found in Appendix C. The project used a Claims Analytic approach to generating Deliverable D5 Requirements, and the WP4 team is continuing to work in this way. Several new claims for the Extended Learner Model are included in this specification in order to support design and implementation. This claims generation process will continue and will be extended along low-level design and implementation of the subsystem.

In addition to addressing the issue of conformance of the specification with the requirements, this specification provides a view of the research background, the challenges to learner modelling in LEACTIVEMATH and the methodology employed to design the Extended Learner Model, as well as descriptions of the functionality and architecture for xLM and its various subcomponents.

The work of WP4 is a combined effort by researchers from the University of Glasgow (formerly from Northumbria University), the University of Edinburgh and the University of Saarland. This document is constructed out of sub-teams working across the boundaries between these three Universities. Here are the (approximate) sub-teams for the five core sections of work reported in this document though all have contributed to most, if not all, aspects of the work.

xLM architecture	Rafael Morales, Eric Andrés, Nicolas Van Labeke
LH	Eric Andrés, Nicolas Van Labeke, Kaska Porayska-Pomsta
LM	Rafael Morales, Nicolas Van Labeke, Paul Brna
SM	Kaśka Porayska-Pomsta, Helen Pain, Manolis Mavrikis
OLM	Nicolas Van Labeke, Paul Brna, Rafael Morales

Following the recommendations from the reviewers of the first delivery of this document, the specification of the Learner Model component has been thoroughly rewritten in order to give a complete and detailed description of its internal representation and updating process (sections 2.4 and 4.4). In particular, details are provided of the sources of evidence used to collect information about the learner characteristics represented in the Learner Model (sections 4.4.2.1 to 4.4.2.5), as well as details of how the Learner Model will be updated from the evidence gathered and stored in the Learner History (in section 4.4.4). A revised API for the Extended Learner Model that takes into account this improved level of description is given in Appendix B.

Also following the recommendations from the reviewers, section 3 gives an overview of the various learning scenarios currently specified by WP6 and how they will be taken into account by the Extended Learner Model.

Chapter 1

Introduction

The development of an effective student model for LeActiveMath is a key factor for the success of the project: if the student model works effectively, it will provide significant opportunities for the various components of LEACTIVEMATH to deliver valuable interactive experiences for learners. While the Student Model component provides valuable services to the Tutorial Component (among others), it also provides opportunities for highly dynamic interactions between the learner and the Open Learner Model. These interactions offer potential value that previous covert student modelling systems have not been able to provide. Hence the student model has both significant scientific potential as well as provides the necessary functionality to enable all the components of LEACTIVEMATH to work together effectively.

Some terminological decisions have been made within Work Package 4. The most important one is to avoid the use of the word "student" and replace this with "learner" because, as a more general term, the latter refers to a wider range of possible LEACTIVEMATH users. Hence, in the remainder of this document, the reader will find terms like "Learner Model" and "Open Learner Model" rather than Student Model and Open Student Model. (The title of the deliverable, however, has remained as originally given.)

Another important term introduced here is the "Extended Learner Model"—xLM, for short. One reason for introducing this term is that the project proposal mentions two different kinds of learner model—termed, in the original proposal, as the Learner Model[1] (LM) and the Situation Model (SM). The extended Learner Model is intended to include both these models as well as the Open Learner Model (OLM) and the Learner History (LH), a repository of the events that have occurred and that relate to the learner's activity and the inferences made by the system. Hence the xLM can be seen as the primary concern of Work Package 4.

As stated in the Project Annex, Open Learner Modelling is taken here to mean a scrutable, inspectable, and modifiable learner models. This definition is closer to Dimitrova's notion of an interactive open learner model (Dimitrova et al., 2001), and this emphasises that one of the main research interests within Work Package 4 is the design of the interactions that take place through the Open Learner Model's user interface. This means, first of all, that the design of a Learner Model must supports such interactions and benefit from them. Previous research has shown that there are may ways of taking into account evidence gathered through open learner modelling, and of combining it with evidence gathered by more traditional diagnosis mechanisms. It has also shown there are many important practical, theoretical, social and ethical issues to consider. This means that open learner modelling, rather than simplifying the learner modelling process, makes it more complex. This is the reason why the issues around the design of the modelling components of the Extended Learner Model make the core of this specification.

The overarching design principle within Work Package 4 is user-centred design (Norman and Draper, 1986) and it is influenced by an informant-oriented perspective (Scaife et al., 1997). Design decisions that need to be made for this deliverable are therefore subject to later evidence/information that may change such decisions—i.e. a form of iterative design. From an engineering

[1] Actually, Student Model

and project management perspective, the risks associated with such changes are estimated carefully and decisions made that are consistent with the completion of the project on time.

The Extended Learner Model is a complex intelligent subsystem of LEACTIVEMATH. So, in order to make its description clear, we have organised it following a classical approach for describing intelligent systems that entails three levels of description: knowledge, logic and implementation (Russell and Norvig, 1995). This means that xLM is described first, at the knowledge level, in terms of what it knows to support its behaviour as an intelligent agent. Then, at the logical level, a detailed, more formal and complete description is given of xLM functionality, internal structure and content. Finally, implementation details are provided when judged as supportive of clarity. Since this document is a specification, not a description of a particular implementation, its completeness is not affected by the omision of details that have no impact on the functionality of xLM at the knowledge and logical levels of description.

To explain—as much as possible—the rationale behind the design and implementation decisions that have been taken so far in Work Package 4 has been yet another important organisation principle for this deliverable.

The core of the document is organised in four main chapters. Chapter 2 provides the grounds for the specification, starting with a brief review of the research background that is followed by a discussion of the particular challenges posed to (open) learner modelling by the nature of LEACTIVEMATH. The second part of this chapter is devoted to describe the methodology followed to produce the design of the Extended Learner Model subsystem, which is elaborated on in the final part of the chapter.

Chapter 3 describes the impact LEACTIVEMATH learning scenarios have on the design and operation of the Extended Learner Model, as well as the support given by xLM to this learning scenarios.

Chapter 4 contains a detailed description of xLM functionality as a learner modelling server within LEACTIVEMATH, its internal structure and content, as well as details of functionality, structure and content of each one of the four xLM components: Learner History (LH), Learner Model (LM), Situation Model (SM) and Open Learner Model (OLM).

The conformance of xLM specification to LEACTIVEMATH requirements, as described in deliverable D5 (LeActiveMath Partners, 2004c), is explained in Chapter 5. This is followed by a list of the conceptual and procedural errors (CAPEs) identified so far in LEACTIVEMATH, in Appendix A, a description of xLM API in Appendix B and the list of requirements from deliverable D5 that apply to xLM.

Chapter 2

Design of the Extended Learner Model (xLM)

2.1 Research background

The history of the use computers in training and education started soon after the introduction of the first commercial computers. In time, research and development in this area split into two main currents, commonly referred to as Computer Based Training (CBT) and Intelligent Tutoring Systems (ITS). A central tenet in CBT development has been that information and communication technologies (ICT) are useful tools in improving people's access to learning resources and enhancing their teaching and learning experiences. Consequently, CBT practitioners have been mostly concerned with developing new tools using ever evolving ICT and applying them to training and education.

In contrast, ITS development has been led less by ICT evolution than by research in the so called cognitive sciences—artificial intelligence, cognitive psychology, neurosciences and philosophy of mind, among others. The focus of interest for ITS practitioners have mostly been enhancing the learning experience by making computers as flexible and supportive as human tutors can be, on the grounds of cognitive science theories and techniques. ITS systems have been proudly described as systems that 'care, precisely' (Self, 1999).

Learner models, understood as internal representations of learners held by ITS systems, have been at the core of ITS from the beginning, and are considered by many as the distinctive trait of ITS. They give ITS systems the flexibility and discernment needed to treat learners as individuals. On the other hand, learner modelling is a very difficult task (Self, 1990) that has been a major research topic, not only for ITS but also for artificial intelligence, since the mid 1970's. The difficulty of the problem has brought many approaches to solving it, from grounding learner models in cognitive theories of learning (Anderson et al., 1995) to the use of powerful machine learning and knowledge representation techniques for diagnosing learners' states and characteristics and representing them by computers (Sison and Shimura, 1998; Aziz et al., 1995); from covert learner modelling, ideally unnoticed by the learner, to overt learning modelling that opens the doors to learners' active participation in the modelling process (e.g. via self-reports) and includes facilities for negotiating or direct manipulation of the content of their learner models (Dimitrova et al., 2001).

2.1.1 Open learner modelling

In principle, learners can be very helpful in solving the problem of learner modelling (Self, 1990). Some studies have indeed shown that able learners could learn from accessing their own knowledge (Bull and Pain, 1995; Aleven and Koedinger, 2000; Mitrovic, 2001). This has been one important reason to pursue this area of research that, in contrast to the traditional approach of regarding the learner as an object to be diagnosed as surreptitiously as possible, encourages the *active* and *explicit* involvement of the learner in the modelling process. However, many issues still remain to be investigated; for example, how to represent the evolving set of the learner's beliefs about their own state of knowledge (Self, 1994a,b); how to give psychological credibility

to learner models (Anderson et al., 1995); how to design systems capable of maintaining learner models in collaborative interactions with learners (Dimitrova et al., 1999a,b), and how to convey a convincing image of the system as a collaborator (du Boulay et al., 1999; Dillenbourg and Self, 1992; Hietala and Niemirepo, 1997).

Open learner modelling provides learners with additional learning experiences and equips the system with another channel to interact with them. This is important because it increases the benefits of designing, implementing and running a (costly) learner-modelling strategy. From this viewpoint, a learner model acquires new purposes, beyond the conventional ones of adapting the system to the particular needs of the learner, predicting her behaviour and assessing her knowledge. Self (1990) praises open models for their potential to support, provoke, and even challenge learners to reflect on their own understanding, helping them to 'develop a more favourable self-image and a better view of how knowledge is acquired' (see also Dillenbourg, 1992). Furthermore, designing learner models to be inspectable by learners could be, in Self's words, a 'salutary principle' that 'might benefit ITS design by reducing the temptation to include crude, ad-hoc classifications' in the models.

2.1.1.1 External influences

The shift from covert to open learner modelling parallels the change of focus from tutoring systems to learning environments. Whereas ITS systems care for each learner's individuality, most of them embody direct instruction of a predefined and well structured curriculum. Intelligent Learning Environments (ILE), on the other hand, generally incorporate a learner-centred philosophy that conceives learning to be constructing knowledge from personal learning experiences gathered through independent exploration of the subject matter (Hannafin and Land, 1997). Endowing learners with the ability to take control of their learning experiences makes it harder to maintain a learner model almost exclusively from information gathered in a furtive manner—for example, the system cannot choose the next exercise in order to discern between competing explanations of learner behaviour.

Learners have to be able to deal with their new responsibility as directors of their own learning process; otherwise they will get lost in the middle of rich and flexible learning environments, not knowing which path to explore nor which question to ask, unable to learn anything from either success or failure (e.g. Aleven and Koedinger, 2000). An approach to this problem is represented by the emphasis that some modern educational theories place on *meta-cognition* (Weinert and Kluwe, 1987), which is about acquiring knowledge about ourselves (and other cognitive creatures), processing this "meta-knowledge", and using it for directing, monitoring and evaluating our problem-solving activities. In essence, the idea is that learners should be able to exert control over their knowledge acquisition and problem-solving, and hence increase the efficiency and quality of their learning, by improving their meta-cognitive abilities (Joyce et al., 1997). Open learner modelling fits better in this view of education than covert learner modelling because it gives learners an opportunity to rehearse and improve their self-knowledge.

2.1.1.2 Objectives

Since the publication of Self's paper (1990), many other researchers have also been driven by the belief that open learner modelling encourages learners to reflect on and become more aware of their own knowledge, learning and problem-solving (Kay, 1997; Paiva et al., 1995; Bull et al., 1995). Naturally, open learner modelling has been seen also as a way of constructing better models—an interpretation of Self's recommendation of directly asking learners for information instead of struggling to guess it. The idea has also been explored by many authors (Beck et al., 1997; Kay, 1994b; Bull and Shurville, 1999).

A further source of motivation for open learner modelling has been to decrease the cost to benefit ratio of learner modelling by devising further ways of exploiting learner models. Open models have been either used or proposed as tools for communication between learners and the system, between learners and teachers (Brna et al., 1999; Bull, 1997; Pain et al., 1996) and among learners (Ayala and Yano, 1996).

2.1.1.3 Research issues

Learner modelling can be a very complex process and produce quite elaborate representations of learners. Consequently, it opens several opportunities for the active and explicit involvement of learners. More importantly perhaps, it raises a number of interesting questions such as which of these opportunities are worth taking; to what extent learners can, should or even must be allowed to participate in the modelling process; and the implications of opening a particular aspect of learning modelling for the process as a whole and for the design of other components of ITS systems.

Taking part in the modelling process requires time and attention from learners. Every time they are asked a question about themselves—like, for example, how well they think they understand the concept of limit, in differential calculus—or browse the learner model, they have to stop performing other activities they may regard as more important to their learning. In time, they can become quite frustrated (Beck et al., 1997), irritated or distracted, unless the timing of interruptions, the selection of questions and feedback and the user interface are carefully designed to avoid that.

Learner models can contain information that may be very difficult to present in a clear form, may overload or confuse learners, or may even be too sensitive for learners to know about (e.g. how much the system trusts the learner's level of confidence in her knowledge; Beck et al., 1997). Non-symbolic, mixed learner models or highly complex symbolic ones may pose serious difficulties for visualisation (cf. Zapata-Rivera and Greer, 2000), while presenting in full detail the evidence that supports inferences based on machine-learning techniques may be unnecessary or undesirable. Stereotypes can be quite useful and effective (Kay, 1994a), but may be interpreted in a pejorative sense by some learners.

Self (1990) and Kay (1997) have suggested that considering learner participation in the modelling process is a beneficial principle of design, even if not fully implemented. Moreover, current and future regulations may demand each bit of personal information stored in the learner model be accessible to learner's scrutiny under request, and may also give learners rights to put limits on the availability of that same information to other interested parties, both inside and outside system that keeps it (Vassileva et al., 1999).

A caveat against giving learners voice in the learner modelling process comes from the well-known attack on introspection in the early part of the past century, as described in (Ericsson and Simon, 1984), and from more recent studies on the quality and quantity of learners' level of self-knowledge (Barnard and Sandberg, 1996): learners' self-knowledge can be very little and inconsistent with their performance—e.g. they may think they know how to do something but, at the same time, being unable to do it and to articulate their knowledge—but they also may not be interested in improving it or may find the task too hard. Furthermore, even if some learners do have a good deal of accurate self-knowledge, they may express it using a different set of conventions—'How can we be sure that the introspecting observer uses language in the same way as the interpreting experimenter?' (Ericsson and Simon's phrasing of Watson's (1913) attack on analytic classical introspection, ibid., p. 58). Other researchers have suggested that blindly empowering learners to take control of their learning may be detrimental because they may lack the necessary (meta) knowledge to make proper use of this power (Aleven and Koedinger, 2000).

A viable way of getting around the lack of learners' self-knowledge follows from Ericsson and Simon's insight that useful and well-supported information can be obtained from verbal reports

without having to trust the speaker. For example, the learner may be mistaken when she affirms that she knows how to do something—e.g. how to carry out symbolic differentiation—but her affirmation implies that she is at least aware of the *existence* of such a piece of knowledge—i.e. symbolic differentiation. This approach gets safer at the expense of reducing learners' power to make decisions on the modelling process, and can be accused of being disguised diagnosis rather than proper open learner modelling, and may have a detrimental effect on the motivation of learners to participate in the modelling process. The opposite approach, that of fully trusting learners, goes more on the lines of giving learners more control and responsibility by presenting the system as a peer collaborator but, in these conditions, a learner model cannot be interpreted anymore as a more or less reliable image of the learner, but rather as an image the learner *believes* is faithful, or even as the image the learner *wants* to present to the system. Whether this sort of "learner model" can still be a source of information about the learner, and how a learning environment can make use of it, are open questions (cf. Vassileva et al., 1999). The idea departs radically from the standard aim behind a learner model: that of being a faithful representation of the learner useful for adapting the environment to better support her learning. Other uses of the learner models, as outlined before, still seem to assume there is a certain degree of reliability in the models. Whereas it could be the case for a completely deceived tutor to still provide adequate support to their tutees, it is hard to believe that this would happen frequently. Alternatively, a very clever tutor could "meta-reason" about the model—e.g. 'she wants me to believe that she knows it,' or 'she might believe that I believe that she knows it'—and act accordingly[1]. Some variations of this sort of reasoning have been proposed for learner modelling (Dimitrova, 2002; Self, 1994b).

Better solutions should distribute trust, control and responsibility among the learner, the system, and any other participants in the modelling enterprise. Mechanisms for reaching agreement and conflict resolution among the parties should exist, from recording all opinions and executing predefined conflict-resolution procedures, to allowing for full-fledged negotiation to take place. The UM toolkit (Kay, 1994b), for example, implements the first strategy, whilst MRCOLLINS (Bull et al., 1995) combines recording of all opinions with support for challenge and negotiation. The more recent work by Dimitrova (2002) on STYLE-OLM is an attempt to build a learner modelling component able to collaborate with the learner through dialogues similar in structure to human-human dialogues. STYLE-OLM keeps the beliefs of learner and system separate, and uses formal reasoning to infer shared and conflicting beliefs.

2.1.2 Situation modelling

Situation modelling is an essential part of the system's ability to adapt to the individual learners' requirements because it places them in a specific context of an interaction. Context provides a basis for making meaningful decisions as to the way in which to communicate the content to the learner in a manner that suits their cognitive, emotional and motivational needs best. For human tutors, the ability to extract the pertinent information from the context in order to adapt to the learner's needs is necessarily part of their social as well as pedagogical competence.

The notion of situation modelling is well grounded in the existing research especially in the area of social linguistics. The theory of linguistic politeness, in particular Brown and Levinson's approach (Brown and Levinson, 1987), provides a formal account of the way in which situational context and the speaker's model of the hearer in the specific context may be used to choose from amongst a number of possible strategies (and the specific instances of those), in order to communicate with the hearer most successfully. Successful communication is defined in terms of the achievement of the speaker's communicative goals and in terms of the persisting willingness of the hearer to cooperate with the speaker. Whilst the willingness of people to cooperate with one

[1] That is how The Man in Black defeated Vizzini the Sicilian in *The Princess Bride* (Reiner, 1987)

another constitutes the backbone of successful communication, a selection of possible communicative strategies through which one may want to achieve their goals provides the means for maintaining such cooperation. Essentially, the communicative strategies are conventions which define the boundaries between polite, i.e. socially acceptable, and impolite, i.e. socially unacceptable behaviour. Central to the ability to choose the appropriate strategy and the appropriate action within it, are the notions of *face* and *facework*. Face is a socio-psychological artefact which refers to a person's need for freedom from imposition by others (**autonomy**) and their need for being liked and/or approved of by others (**approval**). Facework is the speaker's ability to recognise the hearer's face needs and to act upon such recognition in an optimal way.

Brown and Levinson's approach forms the basis for a number of recent computer applications in natural language generation e.g. (Walker et al., 1997), (Cassell and Bickmore, 2002) and in the context of tutoring (Johnson and Rizzo, 2004). However, given that in the Brown and Levison's model different *face* requirements and people's attempt to accommodate them are linked directly to people's linguistic actions, their definitions of autonomy and approval are given only in the context of language use – specifically in the context of choosing the appropriate linguistic form. Similarly the situational context which plays an essential role in determining the possible levels of autonomy and approval is characterised in their model in terms of general dimensions of social distance and power relations. Although on a general level these social dimensions are applicable to the educational context, they are too opaque to serve as a meaningful basis for tutorial feedback choices at a specific level, where details about the changes in learners' affective, motivational as well as cognitive states are of crucial relevance.

In contrast to this general, language and domain-of-interaction independent approach, Porayska-Pomsta (Porayska-Pomsta, 2003) argues that in order to operationalise Brown and Levinson's model in educational context both the definitions of autonomy and approval and of situational context need to be adapted to the requirements of the educational domain of interaction. She defines autonomy directly as the dimension of learner's face which refers to their need to be allowed the freedom of initiative to discover knowledge by themselves, with the general rule of thumb being that the less information the tutor gives to the learner regarding a desired answer the more freedom of initiative (or autonomy) she is giving to the learner. On the other hand, approval is defined as a learner's need to have his motivation and emotional balance maintained explicitly by the tutor which may be achieved in her model either by explicit praises such as *"Well done!"*, or implicit expression of approval by the tutor such as the use of an indirect form to tell the learner that he made a mistake, e.g. *"Are you sure about it [i.e. your answer]?"*.

Given Porayska-Pomsta's definitions of autonomy and approval the situational context is defined in her model as a combination of factors which impact on the two dimensions of face. Based on the existing educational literature, real tutorial dialogues and empirical studies with human tutors, she established a set of eight situational factors relevant to tutors feedback decisions. She also built a situation model which combines the eight situational factors in order to calculate autonomy and approval for every combination of possible situational factor values. In this model the situational factor values are only binary, for example the situational factor *learner's confidence* can take either the value *confident* or *not confident*.

The situation model developed as part of the xLM in the context of LEACTIVEMATH builds on the theoretical and computational framework set up by Porayska-Pomsta. Her model provides a feasible theoretical as well as computational methodology for designing a situation model capable of furnishing LEACTIVEMATH with a number of learner-adaptive powers such as the ability to recommend to other LEACTIVEMATH components how to formulate their feedback or instructions in a way which suits best the individual learners in specific situations. However, in order to adapt the original situation model to the requirements of LEACTIVEMATH, a number of issues need to be resolved. First, a set of situational factors and their possible values relevant in the domain of mathematics needs to be established in order to enable the situation model to make appropriate calculations of autonomy and approval. Second, the manner in which the relevant situational factors combine with one another and how they impact autonomy and approval

needs to be determined for the domain tackled in LEACTIVEMATH. Crucially, in order to for the situation model and for other componets of the xLM to be able to provide LEACTIVEMATH with learner adaptive capabilities, a mechanism for diagnosing situations in terms of the relevant situational factor values is needed. In section 2.3 we present the design methodology employed in this context. Specifically we discuss the empirical studies of which one of the goals was to determine the relevant factors, the relations between them and the sources of evidence that human tutors rely on in order to diagnose them.

2.2 Challenges for learner modelling in LEACTIVEMATH

The profound differences between CBT and ITS systems, sketched at the beginning of section 2.1, have been for many years a constant in the use of computers in education. While CBT systems are nowadays widespread in many shapes and flavours produced by all sorts of makers, ITS systems have mostly stayed in their designers' laboratories. While CBT systems can be and have been developed with relative facility and low cost, the development of ITS systems aimed at public usage have always been a costly and complex task. While ITS systems have cared for the particular needs of individual learners and have flexibly adapted to them, CBT systems have been criticised as unflexible tools for lumping together all learners under the same learning scheme.

Growing awareness of the advantages and disadvantages of both approaches have lead to attempts to taking the best from each one and merge them into an single approach. One of such attempts is the current trend towards sharable, reusable and web-abled learning objects (Wiley, 2001; IEEE, 2002). In particular, LEACTIVEMATH fits the description given by the Advanced Distributed Learning Initiative (ADL, 2004) of second-generation e-learning systems that combine a modern content-based approach—web based, reusable and sharable learning objects, book metaphor, encoding standards, etc.—from CBT with adaptive learning strategies from ITS.

This mixture of approaches produces tensions in the design of LEACTIVEMATH in general, but particularly in the design of the Extended Learner Model since this subsystem is to support a wide range of adaptive learning strategies, from coarse-grain book construction to fine-grain adaptive assessment and dialogue in natural language, with a common lack of the support traditionally afforded in ITS systems: tailor designed and dynamically constructed learning activities that provide large amounts of detailed information about learner behaviour. A learner model in this conditions has to deliver "more with less"; that is to say, it has to be able to

- answer questions about the learner on the basis of scarce information
- without pursuing blind over-generalision.

The scarcity of information about the learner can be alleviated by interactive diagnosis as part of open learner modelling.

Among the various open learner models that have built in the past years, some fall into the category of Editable Learner Model, meaning that learners are enabled to directly alter they own model if they believe it to be wrong. In another word, the learners' own statements of what they believe the learner model should believe about them are simply replacing the learner model's own statements of what it believes they should know about themselves .

If such an approach has interesting properties, it nevertheless raises questions about the trust that is so put into the learner's sole hands. The approach we are following in designing the learner model for LEACTIVEMATH is rather to consider that neither the system diagnoses nor the learner own admissions are superior to the other but are both worth of consideration into building an accurate portrait of the learner.

Let's have an example to illustrate this point. The learner, having just solved a difficult exercise about the chain, decides to report his/her feeling about this situation, using the self-report tool (i.e. admissions are made on a voluntary basis rather than on request):

"I'm quite proud of having solved that problem" (Evidence 1)

Later on in the session with LEACTIVEMATH, the learner starts questioning the Open Learner Model about what it believes:

"What do you think of me?"

"I think you are quite proud of your achievements"

"Why do you think so?"

"Because you told me so"

"Sorry, but I don't think I am that proud" (Evidence 2)

What is the difference between (1) and (2)? Both are dealing with the same learner's affective state; both can be seen as self-reflecting activities (after all, if you have to report about yourself, you better have to think about yourself). But both admissions took place in different spatial and temporal context, i.e. both admissions are coming from different sources that cannot be regarded as equivalent.

The latter statement from the learner is in no way contradicting the former, in the sense that there is no ground to justify why it should just replace it. What it does is completing the portrait the OLM already had about the learner's state. The self-report tool and the interactive dialogue that the OLM provides are two different sources of evidence. Both are providing the learner model with streams of evidence about affective state that have to be accumulated and combined in order to build an accurate model of the learner.

And so this point raises one of the main challenges of a learner model in LEACTIVEMATH, which is how to combine evidence coming from various sources - potentially contradicting each others, as illustrated in this example - and with a limited reliability.

2.3 Design methodology

In general the methodology adopted with respect to the design of xLM is iterative and cyclic in nature. Essentially this means that the design decisions are made in stages and are evaluated in and adjusted to the ever developing context of LEACTIVEMATH and are based on the data which is also becoming available in a cumulative and cyclic manner.

In order to inform the design of the xLM with respect to the ditribution of responsibilities between its subcomponents and in terms of the specific processes that these responsibilities may consist of, we designed and ran empirical studies in which human tutors and students interacted with each other in an environment approximating that of LEACTIVEMATH. The aim of the studies is two-fold. First the studies are used to collect natural language dialogues for the purpose of informing the design of the Dialogue Manager as described in the deliverable D11[2]. Second, the studies are designed to provide us with relevant information as to the affective, motivational and cognitive factors that impact tutors' decisions and students' learning. More specifically, as part of our continuing data analysis conducted in relation to the design of xLM we wish to identify:

- **Situational factors that influence tutors' feedback decisions.** This is needed to establish which factors actually matter and may be explicitly and consistently identified as such by the tutors when they engage in a dialogue with their students in the domain of mathematics. The feature which ensures the relevance of the data gathered through our studies is the channel of communication which is restricted in a way that mimics the environment of interaction available in LEACTIVEMATH. While numerous factors to do with students cognitive, affective and emotive states have been proposed by researchers in other domains [3],

[2]Note that because we are responsible for gathering data relevant to both the Dialogue Manager and the xLM, the description of the study presented here may overalp with the one in the D11

[3]These also include the eight factors proposed by Porayska-Pomsta.

it is not clear which of those actually apply in the domain of mathematics and are detectable in the context of typed interactions. Eventually, situational factors will be needed also for establishing the correspondences between an immediate situation and the best possible tutorial feedback that may be produced in that situation.

- **Types of cues (evidence) that tutors may rely on in diagnosing different situations in terms of the situational factors.** In order for the xLM to be able to model the affective, motivational and cognitive states of the student and, in the case of the Situation Model, to make recommendations of the levels of autonomy and approval needed to be expressed through a given feedback, the xLM needs to be furnished with knowledge about the possible *sources of evidence* for diagnosing the relevant states. The xLM also needs to be furnished with an ability to use this knowledge appropriately. We are using the empirical study to find out what cues in student's behaviour do the human tutors use in order to make their diagnosis of specific factors.

The data collection environment was set up with the above goals in mind.

2.3.1 The Study Design

The study was conducted in three stages:

- *stage-1* at which the main design decisions were made and the pilot data collection environment was developed and tested,
- *stage-2* during which the main conclusions drawn from the the pilot study were applied
- *stage-3* where the conclusions from the observations and of the analysis of the data gathered during stage-1 and stage-2 were used to make further changes to the study design.

At stage-3 the changes were made specifically in order to inform the responsibilities between the subcomponents of the xLM, the distribution of which became much clearer after stage-2. Although we are not as yet in a position to comment on the exact results of the data gathered at stage-3, we discuss why the design changes were required based on the analysis of the data collected at stage-1 and stage-2.

2.3.2 Participants

In total 5 experienced tutors tutored in these studies: 4 tutors from The University of Edinburgh, School of Mathematics and 1 school teacher from one of Edinburgh's high schools. 2 in the University tutors participated in all 3 stages, 1 in stage-2 and stage-3, and 1 in stage-2 only. The high school teacher participated in stage-3 only. Between them the tutors engaged in a tutorial interaction with 32 students: 7 students participated in the pilot (stage-1), 7 in stage-2 and 14 students took part in stage-3. All students were from the University of Edinburgh, taking first year level undergraduate Mathematics service courses designed to provide support for their main degree subject: none were intending to continue into Mathematics degree courses. Whilst at stage-1 and stage-3 each student took part in one interaction only, at stage-2 most students took part more than once: 5 students participated in 2 sessions and 2 students participated in 1 session only. Of the students who took part in only one session, one was an excellent student who had no problems with the material taught and therefore was not recommended by the tutor for another session, while the other student simply did not turn up for the second session.

2.3.3 Materials

A number of computer tools were used in the study, some purpose built. The main tool was a chat interface, based on the Wallis system (developed within the University of Edinburgh Mathematics department), which permitted the student and tutor to interact. Any background material required by the student (see *theory frame* below) was also provided by the Wallis system. Additional functionality was added to enable video capture and replay. The situation factors selection tool was purpose built for this study.

For all stages of the study, the materials need to be described in terms of the functionality of the interface given to the students and the functionality of the interface given to the tutors, because the two interfaces differ in a number of respects. Figure 2.1 shows the student's side of the data collection environment which was split into:

- the *theory frame* (marked *Chat rule - Introduction*)in which the students could refer to the basic explanations of the material if they were directed to do so by the tutor (this happened only once during all of the stages of the study and therefore this frame usually remained empty);
- the *text* and *maths* editor (marked *Chat* in figure;
- the *history of the interaction frame* (to the right of the figure), through which the students could scroll at any time during a session.

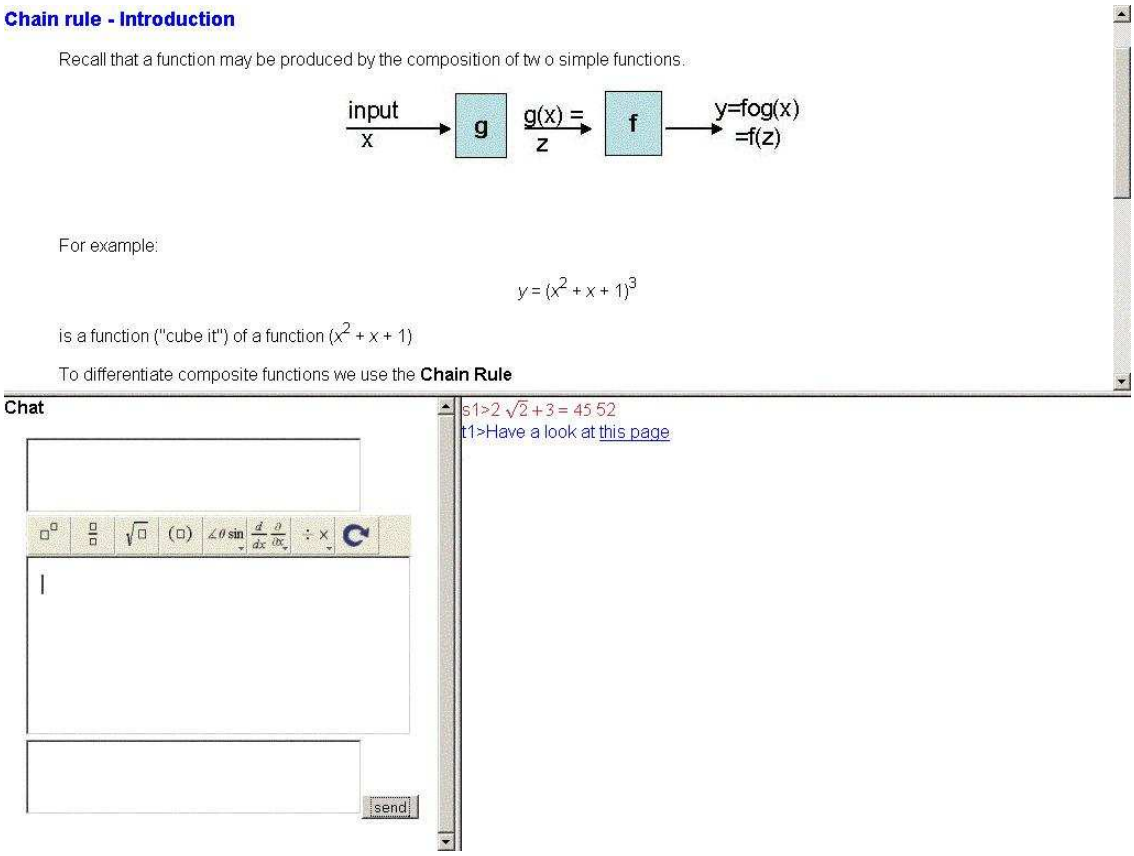


Figure 2.1: Student's screen

The text and maths editor was split into 3 parts:

1. the *preamble text frame* (top panel of *Chat*)in which the students could type the text preceeding any mathematical formulae
2. the *maths frame* (middle panel) in which they could write mathematical formulae. To ease the writing of those, the maths editor was equipped with maths templates from which the students could select a set of pre-defined formulae appropriate for a given excercise. The templates were editable.
3. the *text editor* (lower panel) for the text following a maths formulae.

The tutor's screen shown in figure 2.2 consisted of five parts:

1. the *text and maths editor* as in the student's window;
2. the *preview frame* for viewing the feedback typed by the tutor prior to it being sent to the student;
3. the *exercises frame* (below the *preview frame*) in which a number of chain rule exercises were provided to the tutors for use during the session. The exercises were ordered based on two of the participating tutors' experience and according to their assessment of the difficulty of the exercises relative to the expected overall competency of the participating students. The tutors could click on a given exercise for it to appear in the maths editor frame;
4. the *history of the interaction frame* (lower left panel of the figure), through which the tutor could scroll at any time during a session;
5. the *"current situation"* frame which the tutor used to identify the situational factors and their values relevant to their responses. A set of pre-defined factors was provided to the tutors in order to ease their task. These factors included the set of eight situational factors validated by Porayska-Pomsta (Porayska-Pomsta, 2003), and two additional factors such as student's effort and student's emotional state.

In addition to the screen within which the tutors could interact with the students and select situational factors, tutors were provided with a second screen which was connected to the student's computer and through which they could observe all student actions regardless of whether or not they were actually submitted by the student.

2.3.4 Procedure and data collection methodology

In all three stages of the study the material concentrated on was symbolic differentiation as a meaningful self-contained subset of calculus, in particular the chain rule. This subset was selected for a number of reasons: (a) it is a significant part of the curriculum at both school and university level; (b) university and high school mathematics tutors had recommended it as an area where students have difficulty and require practise and revision; (c) in addition to differentiation skills, there are a range of skills required for differentiation including algebra, bracketing, handling fractions, square roots and trigonometric functions.

Tutors and students were trained to use their respective interfaces. In both cases the training included an experimenter explaining to the participants the interface and the kinds of facilities that it provides. To ease the cognitive load on the tutors during the interactions, tutors were allowed to get used to their tasks during mock sessions with an experimenter.

For tutors the procedure was the same in all three stages of the study, while for students it changed between stage-2 and stage-3. Both tutors and the students were asked to interact with one another through the chat interface provided to them.

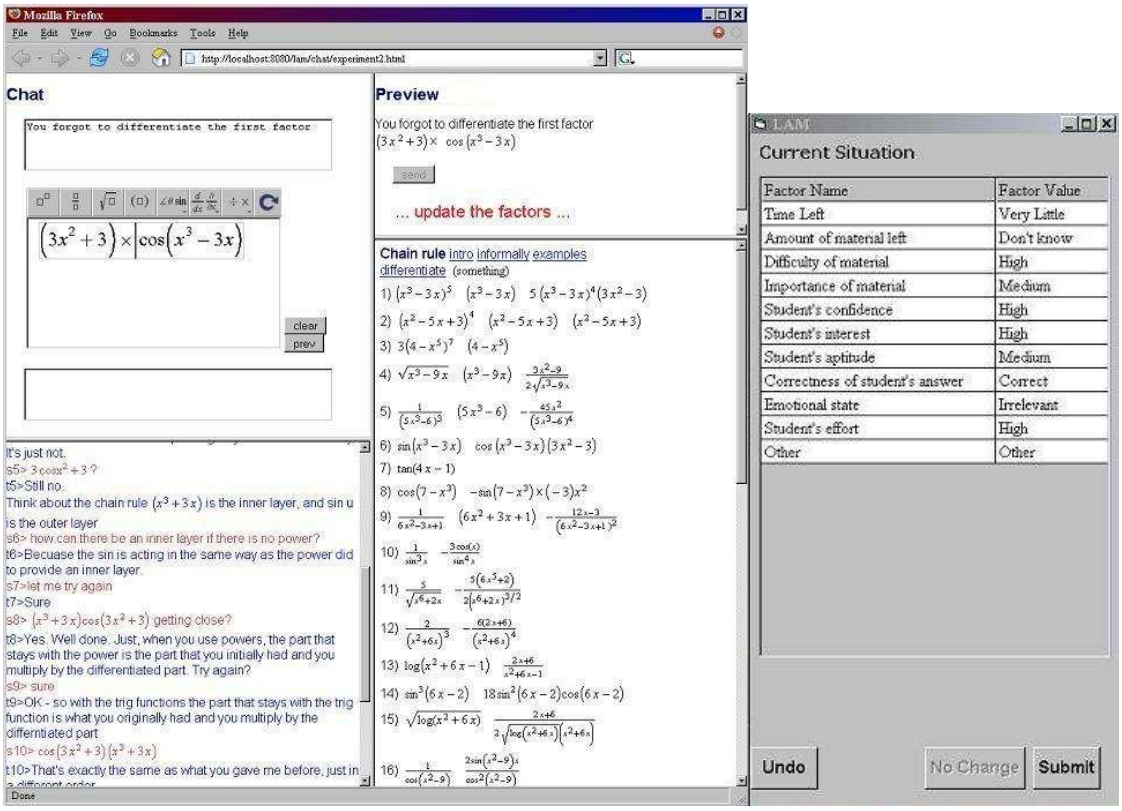


Figure 2.2: Tutors chat interface and situational factors selection tool

During each session, the student screen was video-recorded for the purpose of *replay* and *post-hoc walkthroughs* with the tutors. Additionally, keyboard actions and mouse movements were recorded using a key-capture program. Immediately after each session, both tutors and students were interviewed using a *semi-structured interview protocol*. Different protocols were used for students and tutors. All interviews with the tutors were audio-recorded for the purpose of future data analysis. In the case of the students, only the stage-3 interviews and verbal protocols were audio recorded.

At stage-3, the students were also asked to provide a verbal protocol in relation to the interaction with the tutor. This change was made based on the preliminary analysis of the data gathered in stage-2 which indicated that because of the restricted channel of communication imposed on the tutors in our studies, the tutors found it virtually impossible to diagnose the students affective states. Asking the students to keep verbal protocols was to allow them to report on their states during the interactions, with the view of gathering data relevant to students' affect and cognitive prefences which could then be triangulated with the data obtained from the tutors' side. To facilitate this, students were prompted (by the observer) to comment on what they were thinking and what they were feeling throughout the interaction.

While interacting with the students, the tutors' task was to select the situational factors every time they provided the student with feedback. While in the pilot study this request was made by an experimenter only verbally, in stage-2 the selection of factors was enforced by the fact that the situational factors frame was "hooked up" to the chat frame. This change was introduced based on the observation of the irregularity with which the tutors selected the factors during stage-1 and it also addressed the two pilot tutors' suggestion that this could be improved by having the system "remind" them of the necessity of making such selections. Consequently, in stage-2 and stage-3 the tutors were not allowed to submit their response to the student until they submitted the factors – they were reminded of this by the "...update factors ... " message which was displayed in their preview frame. The tutors could submit the factors either by clicking the "submit" button, or by clicking the "no change" button. The use of the latter indicates that there was no change in the factors between the previous and the current situation.

Although a set of pre-defined factors was provided for the tutor to choose from, the tutors were given the flexibility to add other factors to the existing set. The list of factors provided in the pilot was slightly smaller than the one provided in the stage-2 and stage-3, but the selection process remained essentially the same. The only new factor that we added in stage-2 was *emotional state* which was suggested by one of the pilot tutors as potentially useful. Allowing the tutors the flexibility of adding new factors meant that instead of having to think about the meaning of the factors pre-specified by us, the tutors were given an opportunity to specify situations in their own terms (during the later walkthroughs they were questioned about the meaning of the labels that they chose). For some of the participating tutors, this also eased their overall task of having to use the interface while engaging in a meaningful dialogue with the student, it led to quicker tutor response delivery, and in a more in-depth insight into what may trigger tutor's various responses.

As well as being asked to engage in a tutorial interaction with the student and selecting the situational factors for each response, the tutors were also asked in all versions of the study to talk aloud (i.e. to keep a *verbal protocol*) about any and every possible aspect of the interaction. This was to gain an additional insight to the tutors' thought processes during the sessions, especially with respect to the evidence that they used in determining the values of various factors. For each completed interaction, the tutors were invited back to participate in a post-hoc walkthrough. A *cognitive walkthrough* is a procedure which allows the tutor and the experimenter to see a specific interaction with a student again, to discuss any aspect of the interaction in detail, and for the tutor to indicate any change in their assessment of the situation if necessary. The changes made during walkthroughs were recorded in addition to the original submissions made by the tutor during a given interaction. This consisted of a synchronised replay of:

1. what was happening on the student screen during a given interaction
2. the audio recording of the tutor's verbal protocol
3. the situational factors interface showing the selected situation for each of the resposes that they made to the student

In preparation for every walkthrough the experimenters replayed the corresponding interaction in the same way as it would be shown to the tutor during the walkthrough in order to gather the relevant questions and note points of interest. During the walkthrough the replay was paused as needed and the tutor was asked the questions prepared on the basis of the replay. Although we are still in the process of analysing the data gathered through walkthroughs, it has already proved invaluable for identifying various sources of evidence for determining the individual factor values, for mining the possible rules for combining the sources of evidence to infer factor values, and for understanding why the tutors chose to act in particular ways in specific situations. For tutors, the walkthroughs presented an oppportunity to correct the factor values for situations in which they may have been left unchanged because of tutors' cognitive overload during the interactions, or where the tutors observed additional evidence while listening to their own verbal protocols in conjunction with observing the replays of students' screen.

A number of customised tools were developed to facilitate data analysis.

2.3.5 Preliminary results

The analysis presented here is only of the data gathered in stage-2 where a total of 12 interactions took place. Our aim of the data analysis was two-fold:

1. We wanted to find out **what** factors the tutors consider when deciding on their feedback and **how often** these factors are used across all interactions. The purpose of this was to establish a set of situational factors that should constitute the input variables to the situation model and for which evidence may need to be gathered, as well as to determine other factors relevant to learner modelling.
2. For each factor we also wanted to identify the sources of evidence that tutors typically rely on in determining the individual factor values.

With respect to these two general aims we relied on a number of working hypotheses and assumptions:

- **Hypothesis 1** (H_1) states that there is a number of different situational factors that the tutors rely on when making their feedback decisions.
- **Hypothesis 2** (H_2) states that there are differences in the relative importance of the situational factors to tutors' decisions.
  - **Assumption 1** (A_1) states that relative importance means the relative relevance of a factor to tutors' decisions.
- **Hypothesis 3** (H_3) states that tutors are able to diagnose student affective states relevant to their learning.
- **Hypothesis 4** (H_4) states that mouse movements constitute a reliable source of evidence on the basis of which tutors can infer students' affective states.

– **Assumption 2** (A_2) states that mouse movements are observed by human tutors while they enagage in interactions with students via a chat interface (e.g. de Vicente, 2002). This assumption is also one of the main reasons for the tutors being shown students' screen during the interactions and for the mouse and keyboard actions data being recorded.

Our descriptive analysis led to our acceptance of H_1. In particular, we established that the total number of situations reported for all interactions is 309 with an average of 25.75 situations being reported in each interaction. Out of the total of 309 situations, 144 (46.6% of all situations) are unique situational factor-value combinations. The fact that near half of the situations reported by the tutors were unique suggests not only that tutors are able to determine the situations while engaging in the tutorial interactions, but also that different factors are involved in supporting their decisions. Our further analysis revealed that in total 14 situational factors were used during the 12 interactions and across 4 tutors, with 10 situational factors being set up by us *a priori* and 4 factors being added by the tutors during interactions.

In order to test the second hypothesis H_2 we relied on our first assumption A_1. Amongst the default situational factor values provided in the situational diagnosis tool we included the value "Irrelevant" to allow the tutors to indicate when a factor was not considered important to their decisions. We used this value to perform frequency count to reveal the relative importance of each factor to tutors' decisions across all interactions. Figure 2.3 illustrates our findings.

We identified 5 groups of factors each with a decreasing relevance to tutors' decisions. The most important factors of which the frequency count (FC) is above 200 out of the total of 309 recorded situations include **correctness of student's answer** with FC = 259, **student's confidence** (FC = 250), **student's aptitude** (FC = 222), and **student's interest** (FC = 214).

The second group with the overall frequency count above 150 includes **difficulty of material** (FC = 183) and **importance of material** (FC = 165).

The third group of factors used over a 100 times includes **student effort** (FC = 140) and **amount of session time left** (FC = 119).

Two factors were use more than 50 times: **student's knowledge** (FC = 78) and **amount of material left to cover in the current session** (FC = 76).

The group of the least important factors used less than 50 times by different tutors across interactions includes **student's emotional state** (FC = 46), **relative difficulty (FC = 41)**, **interface** (FC = 38) and **goal** (FC = 37).

The results of the frequency count for the value "Irrelevant" support our second hypothesis H_2 as they show that not all the factors are equally important to tutors' decisions across the board. What these results give us is a systematic, empirically supported way of selecting the factors which are essential to our model. These include most factors in the first, second, third and possibly fourth group, but probably not in the fith group where the factors were used between different tutors less than 50 times in total. Ultimately our decisions regarding the inclusion of a factor in our system rely primarily on whether or not we find further support for or against it in other data sources that we gathered such as verbal protocols, interviews and walkthroughs. For example, the **goal** and **interface** are not only used very infrequently, but they are also used by only one out of five tutors who additionally found it virtually impossible to define the meaning of the two factors in a systematic way or to point to the sources of evidence in students' behaviour that could be used to infer such meaning. Without such information from the tutor it becomes very arbitrary an affair to define the way in which the values of the factors can be inferred, what values are relevant at all and how they may interact with other relevant factors. This is certainly a problem which we also encountered with **student's emotional state** factor which again was used very infrequently by the tutors. This in itself undermines the support for our third hypothesis H_3, which states that tutors are able to diagnose student affective states.

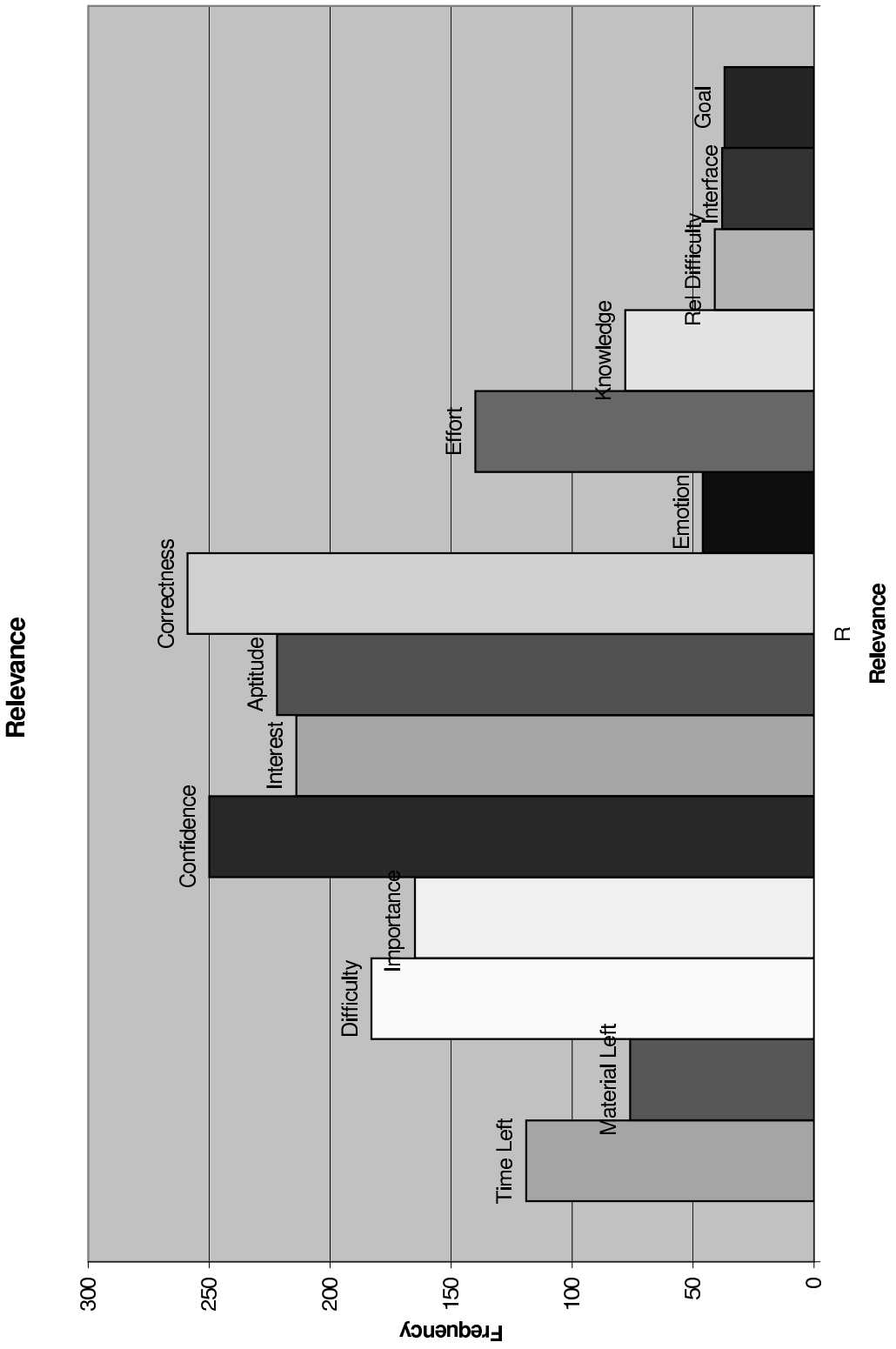


Figure 2.3: Relevance of individual factors to tutors' decisions

In relation to the third hypothesis H_3, an examination of the data obtained from the verbal protocols, the interviews and the walkthroughs reveals that tutors find it virtually impossible to detect students' emotional states through a channel of communication which is restricted to typed interactions. More significantly, in relation to the fourth hypothesis H_4, different tutors pointed out that although mouse actions data is made available to them through access to students' screen and although they do spot "weird", "funny" or "unusual" mouse movements, they do not have any way in which to interpret those in terms of student's emotions or other relevant situational factors. So whilst Assumption 2 holds, it does not in this case appear to aid diagnose. For this reason mouse movement data will not be used for diagnosis within LEACTIVEMATH.

Our analysis to date, together with tutors comments, shows very clearly that the only reliable source of evidence available to tutors is when students self-report on their affective states and even these are typically reports of enjoyment or enthusiasm about the aspects of the interactions. These findings are very relevant to the way which the main responsibilities with respect to diagnosing the situational factors are distributed between the subcomponents of the xLM and the GUI, especially between the Situation Model which relies primarily on the evidence available from observing student's behaviour, and the GUI which provides students with facilities to self-report. Our findings have also led to the changes in the design of stage-3 of the study, which now explicitly includes the collection of students' verbal protocols. The data collected through student verbal protocols is intended to provide support for further design of the GUI with respect to diagnosing students' affective states. Preliminary evidence in stage-3 suggests that students most frequently comment on perceived difficulty of the sub-task, and their confidence in their answer,

2.4 The design

The Learner Model is a collection of beliefs about the learner's states and characteristics specified along five dimensions (see figure 2.4):

1. the subject domain,
2. (mathematical) competencies,
3. motivation,
4. affect,
5. metacognition and
6. CAPEs (Conceptual and Procedural Errors)

Each of these dimensions can be seen as a map which specifies the individual factors or attributes of the processes relevant to learning and considered by the Learner Model. The dimensions also specify how the different factors and attributes relate to each other. For example:

- the subject domain breaks down into *domain topics* such as "Chain Rule", "Function", "Derivative", etc.
- mathematical competencies is decomposed into "Thinking mathematically", "Solving mathematical problems", etc.
- motivation is decomposed into factors that are influencing the learner's motivation such as student's interest, student's confidence or effort that the student puts into learning the material.

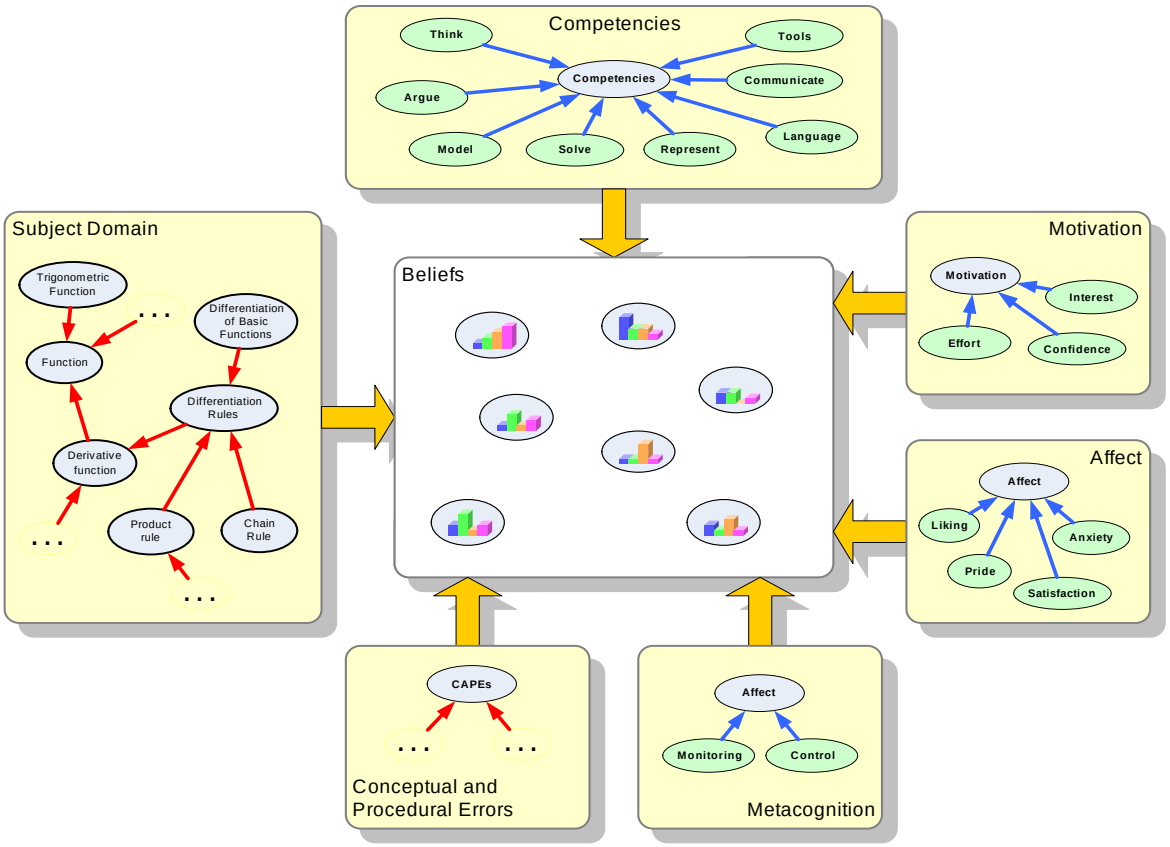


Figure 2.4: The Learner Model is a collection of beliefs specified by several maps defining various learner's states and characteristics.

The exact definition of these maps and what evidence supports the diagnosis of the learner's characteristics will be described in section 4.4.

The five dimensions are used to establish the method of communication between other components of the system and the Learner Model, as well the way in which learners will be able to question the Learner Model about its beliefs. For example, questions like "What is the learner's level of competency in Mathematical Thinking?", "What do you think the learner knows about the chain rule?", "What is the learner's current confidence level?", etc. can be asked to the Learner Model, prompting it to retrieve (and possibly present in a particular form) the beliefs related to the query. The overall dimensions themselves can also be queried, as they represent the combination of all the individual attributes they contain: the Learner Model could therefore answer to queries such as "What is the overall motivation of the learner?" by combining all belief about each motivational factors held in the model.

An extra level of structure is taken into account in the Learner Model, specifying how the dimensions interact with each other (see figure 2.5).

At the bottom of the layer stands the subject domain. It underlines the fact that learning does not take place in a vacuum but relies on domain and content. The Learner Model does not hold any belief about the domain *per se* but does hold beliefs about upper dimensions of the model *in relation to the domain*; domain topics could be seen as placeholders for the system's beliefs about the learner. This is consistent with the principles of a learner model, which represents information about the learner's states and characteristics in a learning situation and not about the domain addressed by the learning activities.

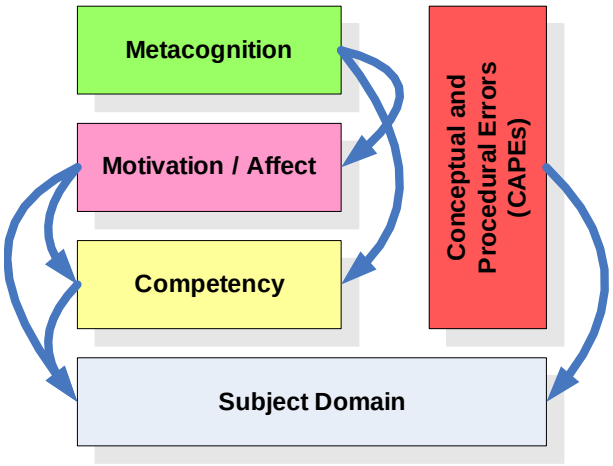


Figure 2.5: A multi-layered model of learner.

On top of the subject domain, (mathematical) competencies, motivation, affect and metacognition stand in a layered fashion, relying on the previous ones for expressing and grounding beliefs about the learner's:

- mathematical competency on the domain (i.e. *"What is the learner's level of mathematical competency on the chain rule?"*, *"What is the learner's overall level for solving mathematical problems?"*);
- affective state about the domain (i.e. *"What is the learner's level of anxiety about the chain rule?"*) or about his/her mathematical competency on the domain (i.e. *"What is the learner's level of satisfaction with solving problems about the chain rule?"*);
- motivational state about the domain (i.e. *"What is the learner's level of confidence about the chain rule?"*) or about his/her mathematical competency on the domain (i.e. *"What is the learner's level of confidence on his ability to solve problems?"*) ;
- metacognitive ability to handle his/her mathematical competency on the domain (i.e. *"What is the learner's ability in monitoring his/her competence to solve problems related to the chain rule?"*);
- metacognitive ability to handle his/her affective state about the domain or about his/her mathematical competency on the domain (i.e. *"What is the learner's ability in monitoring his/her satisfaction when solving problems?"*);
- metacognitive ability to handle his/her motivational state about the domain or about his/her mathematical competency on the domain (i.e. *"What is the learner's ability in controlling his/her confidence when solving problems?"*);

The last dimension of the model, the CAPEs, does not really fit into the model as it includes information that is usually cross-dimensional. Nevertheless CAPEs could be related to the subject domain, in the sense that, whatever the nature of CAPEs taken into consideration (i.e. buggy rules, misconceptions, etc.), they can be clearly identified through the context in which they occur, in this case the subject domain.

Such an organisation of the Learner Model extends significantly its power of expressivity, as it holds beliefs not only about the motivational state of the learner (as a whole) but also it makes explicit extent to which such motivational state is related to the competencies of the learner or to the subject domain. How these beliefs are structured and queried will be described in section 4.4.3.

Finally – and perhaps the most important aspect – the internal structure of the maps (i.e. how their elementary attributes are related to each other) is also used by the Learner Model for controlling the propagation of evidence when beliefs are updated. For example, the structure of the "subject domain" map is used to define how new evidence about the learner's state and behaviour about the "chain rule" will be propagated to neighbour topics such as "product rule", "derivation rules", etc. The propagation of evidence across the dimensions of the domain is a crucial factor of the Learner Model and will be described in section 4.4.4.

The situation model is the component of the xLM which performs situational diagnosis at every point at which a response from the tutor may be required. Based on such a diagnosis, the model also makes recommendations as to the next optimal tutorial feedback. The main high level responsibility of the Situation Model is to account for student motivation by diagnosing student motivational factors such as their *self-confidence*, *interest in the task*, and *effort* that they put into learning. Such diagnosis happens in the context of other situational factors and their values being determined such as the *difficulty* and *importnace* of material, their *aptitude*, *correctness* of their answers as well as the current state of their *knowledge*. The purpose of the situation model is to negotiate and strike a balance between the student's wants and needs and the motivational goals of the tutor, and other situational factors and tutor goals that may arise from those factors and which goals may conflict with the motivational goals. For example, if student confidence is diagnosed as low then according to the situation model the tutor goal will be to boost it. However if correctness of student's answer is diagnosed as incorrect, then the tutor's goal will be to correct it. To provide the student with corrective feedback inevitably means to tell the student that their answer was incorrect which potentially may lead to student's confidence being dented further rather than boosted. The situation model provides means by which to take all of the available situational information into account and to come up with a balanced way in which to inform the student of a problem in their answer. The end result of such a balancing act is to recommend potentially the best way through which the tutor can achieve its goals without compromising the particular students' motivation. Achieving such balance is at the core of successful adaptation to the specifc users in ways which are communicatively, socially and educationally effective.

Thus, the Situation Model has two responsibilities: (1) to diagnose the immediate situation in terms of a number of situational factors including student motivational factors and (2) to recommend the appropriate type of action that may be taken in the situation diagnosed. Immediate situation is a situation which is pertinent to specific feedback decisions made by the tutor as a response to a learner's action at a unique point in a given session. The diagnosis performed by the situational model is therefore local and does not extend to longer term characteristics of the student such as their overall confidence in being able to tackle the material, or their overall interest in the material learned. However, the situation model provides the Learner Model with a basis for inferring the longer terms characteristics in a cumulative way by storing the results of the local diagnosis in the Learner History for all to see and to use. In turn, student longer term characteristics which are inferred by the Learner Model can be used as part of the evidence for the relevant inferences of the situational factors and their values made by the Situation Model at a local level. For example, if the cumulative (i.e. longer term) value for *student's confidence* has been established up to a current point in the interaction as *high*, but based on other evidence in student's current behaviour *the current value of confidence* can be said to be *low* and the *correctness of student's answer* is *incorrect* then given the overall high level of confidence, the tutor may be able to afford to be more direct in the way that it provides the student with corrective feedback than if the cumulative value of confidence for that student were low. The varying the levels of directness of feedback is one way in which the output of the situation model can be use by other components of LEACTIVEMATH, in particular by the Dialogue Manager. Details of both the diagnosis agent, the inputs and the outputs of the situation model are given in section 4.5 along with the description of the way in which the Situation Model outputs contribute to the effect of an adaptive LEACTIVEMATH.

Chapter 3

Learning scenarios

This section gives an overview of how the Extended Learner Model could support the learning scenarios embedded in LEACTIVEMATH. The following list gives an overview of the scenarios currently specified by WP6 (for the coming deliverable D20) and to be implemented by the Tutorial Component (for generating courses).

TeachNew generate an in-depth course that provides to the student all necessary knowledge to fully understand the target concepts. The course contains concepts (definitions and theorems) and all types of satellite elements (exercises, examples, text elements).

Rehearse a course that specifically addresses deficiencies of the student with respect to the target concepts. The course contains concepts and examples that serve to illustrate usage of the concepts. The course does not contain text elements like introductions. Collections of exercises conclude the course and support active understanding of the learner.

Overview a course that provides the student with an abstract and general understanding of the target concepts. The course contains concepts and examples to illustrate usage of the concepts .

trainCompetency a course that trains a given competency with respect to target concepts. The course consists of concepts, examples, and several exercises. Ideally, the learner interacts with the system in order to select the competencies that are trained. E.g., the user selects the scenario, the system then presents competency mastery, and learner selects the competencies on this basis.

Workbook a course that trains all competencies of the user with respect to the target concepts. It consists of a collection of exercises.

ExamSimulation a course that consists of exercises training all competencies and of all difficulty levels.

3.1 Learner Model and Learner History

As described in section 2.4, the LM build a cumulative portrait of the learner's mathematical competency, motivational and affective states, metacognitive abilities, as well as his/her Conceptual and Procedural Errors (CAPEs), whereas the LH maintains an accurate digest of the learner's activities with the system. Both these sources of information could be queried at anytime by LEACTIVEMATH (e.g. "What is the level of motivation of the learner?", "Did the learner perform this exercise and what was its outcome?", etc.), using it for making tutorial decisions.

For example, to initiate the **trainCompetency** scenario, the Tutorial Component requests from the LM an overview of the learner's mastery in each of the eight competencies considered in the Learner Model. Based on this digest, the TC displays at the interface a digest of the LM beliefs for the learner to choose which competency to train (alternatively, the TC could suggest it).

3.2 Situation Model

The following stages reflect the general usage of the Situation Model.

1. The learner specifies a scenario and a course is generated.
2. The learner starts to do the first exercise.
3. In the course of the exercise evidence is collected in relation to the various situational factors, with the outcome being recommendations of specific autonomy and approval values.
4. At various choice points in the exercise, a particular method is selected based on the current autonomy and approval values.
5. In the dialogue based system, feedback selection will be based on the current autonomy and approval values.
6. The value of autonomy is assigned a strength to acknowledge the current scenario, for example, for TeachNew the strength of the recommendation for autonomy is decreased; however, for Exam Simulation it is increased.
7. On completion of the current exercise the next exercise selection is also informed by the autonomy and approval values, as is the directness of any suggestions made to the learner.

The LEACTIVEMATH system will adapt to the learner according to the current situation. For particular scenarios the strength of the recommendations made by the Situation Model will vary. For example, the autonomy and approval values will have little impact on the learner's selection of the particular competency on which to focus in the TrainCompetency scenario, even if high autonomy is recommended: in this case control will be with the LEACTIVEMATH system. In the ExamSimulation scenario any choices that relate to the selection of competency and difficulty levels of exercises will be limited, i.e. the strength of Situation Model recommendations will be increased. In Workbook, TeachNew and Rehearse scenarios the strength of recommendations will be generally decreased.

3.3 Open Learner Model

A generic usage of the OLM can be described in terms of the following stages:

1. At some point during the session, the learner launch the OLM.
2. The learner navigates in the OLM, looking for some topic worth talking about.
3. The learner queries the OLM about its judgment concerning some particular topic (e.g. the chain rule or his/her overall level of motivation) and challenges the OLM about its conclusion.
4. The OLM justifies its judgment by showing how it gradually came to that current conclusion about the learner's abilities regarding the queried topic.
5. The learner could challenge the OLM judgment, for example when being confident that his/her motivation on some exercise, critical as an evidence for the judgment, has been incorrectly assessed. After negotiation, the OLM and the learner could come to an agreement about what was his/her level of motivation. The Learner Model is informed (by way of the Learner History) that a new source of evidence regarding this topic has to be taken into account and starts updating the beliefs [1].

[1] But this is a different story.

From this outline, several elements of adaptativity in relation to the learning scenarios can be envisaged.

A first example concerns the availability of the OLM to the learner. A scenario like **ExamSimulation** could be right not to allow the learner to access his/her OLM. More interestingly, the initiative of launching the OLM could be made by the system. For example, the **trainCompetency** scenario may suggest the learner to use the OLM in order to explore his/her achievements, either after a given time or after a given number of exercises or even after a certain level of competency is attained by the learner.

As another example, the **trainCompetency** scenario implies that the generated course will focus on a particular competency. This will be insured at the level of the content integrated in a book. Obviously, one can expect Learner Model to contain beliefs beyond this competency (after all, the learner may have already used LEACTIVEMATH in different context). But the OLM, as an externalisation tool for the LM, could be ask to hide some of the topics available to discussion with the learner (in this situation, only topics related to the competencies selected by the learner).

Chapter 4

Specification

This part aims to clarify the major decisions that have been made about the subcomponents and how they will operate. The decision has therefore been taken to gloss over the specific, low level details which may change significantly in favour of those higher level constructs which can be relied upon to remain stable over the lifetime of the project.

The architecture of the xLM and its relationship to the other components in the LEACTIVEMATH project is the subject of section 4.1 where various information flows are described. Sufficient detail is provided to ensure that other partners in the project will understand how their components will work with xLM without needing to be concerned with unnecessary detail. The internal architecture for xLM is provided in order that each of the subcomponents in the xLM can be specified — this includes the LM, LH, SM and the OLM.

The Learner History is described in section 4.3. The Learner Model is described in section 4.4. The Situation Model is defined in section 4.5.

The Open Learner Model is the subcomponent that still needs the most further work since the main distinction between the functionality of the LM and of the OLM relates to the interactions held between the learner and the system and the importance of justifying various decisions made by the LM/OLM. The specification laid out in section 4.6 is sufficient for determining both the implications for other components which need to support the OLM and which need to draw on the OLM for its results. The GUI for the OLM will need to be refined but this is mainly of concern to the team working on the OLM.

4.1 The architecture of xLM

This section presents a coarse-grain, logical view of the architecture of the Extended Learner Model in the context of LEACTIVEMATH.

- It provides a framework for discussion of the learner model and its surroundings, by making explicit the components integrating this part of LEACTIVEMATH and the relations between them and with other components of LEACTIVEMATH.
- It makes explicit the expected flow of information between components of xLM, as well as between components of xLM and other components in LEACTIVEMATH, without going to much into the details of the particular of each (type of) information exchange.
- It suggests a grouping of the components of LEACTIVEMATH following the classical architecture of intelligent tutoring systems, providing a general view of the whole system and clarifying the role of each component.

The architecture of xLM described below follows a component-based design and it is based on current ACTIVEMATH architecture plus the new components to be developed for LEACTIVEMATH. It is intended as being broad enough to provide a logical description of xLM in the context

of LEACTIVEMATH, yet to be sufficiently detailed to allow the distribution of xLM's main functionalities among a set of clearly distinguished but tightly related components. The existence (or not) of information flow between components, inside xLM and between xLM components and external components, is made explicit so that any relaxation of the restrictions now imposed in the actual implementation should be justified.

4.1.1 Architecture

Figure 4.1 depicts a coarse-grain view of the LEACTIVEMATH architecture using the classical logical arquitecture of intelligent tutoring systems Burns and Capps (1988) focused on the learner modelling module. Since the components of LEACTIVEMATH are designed with more specific functionalities than the classical modules of intelligent tutoring systems, they are regarded as parts of higher level subsystem corresponding to the classical learner model, domain (knowledge) expert, tutorial and interface modules. Every small box inside the big subsystems represents a component of LEACTIVEMATH, including both data and processes. In particular, the components comprised in Work Package 4 are regarded as serving the shared purpose of providing LEACTIVEMATH with a reliable representation of the learner in the context of their interaction with the system, and hence are put together inside the Extended Learner Model. A brief description of subsystems and components in figure 4.1 is given further below.

In the diagram, arrows connecting components represent information flow between them. More specifically, an arrow connecting component *S* to component *R* means either that

- component *R requests* information from component *S* and this sends the information back, or
- component *S informs* or *commands* component *R* by sending information.

An arrow starting or leading to a subsystem stands for the corresponding set of arrows starting or leading to every one of the components in the subsystem.

The diagram in figure 4.1 depicts the whole set of information flow occurring between xLM components, as well as information flow occurring between xLM components and other components of LEACTIVEMATH. However, neither the diagram nor the description below are complete in matters completely outside the Extended Learner Model, such as information flow in between components in the Tutorial Subsystem. A more complete and detailed description of LEACTIVEMATH architecture can be found in deliverable *D8: Open Architecture* LeActiveMath Partners (2004b)

4.1.2 Components of relevance for learner modelling

Domain Knowledge Subsystem (DK) The components in this subsystem are responsible, as a whole, for providing a description of the subject domain suitable for learner modelling. They could also provide domain reasoning services to other components, such as the Dialogue Manager, but this is not discussed further here.

Interface Subsystem (IS) The components in this subsystem are in charge of presenting information to the learner and getting information from them.

Dialogue Manager (DM) This is the component in charge of facilitating natural dialogue interaction to LEACTIVEMATH, including natural language understanding and generation.

Tutorial Component (TC) This component is responsible for making the strategic pedagogical moves for optimising the learning process.

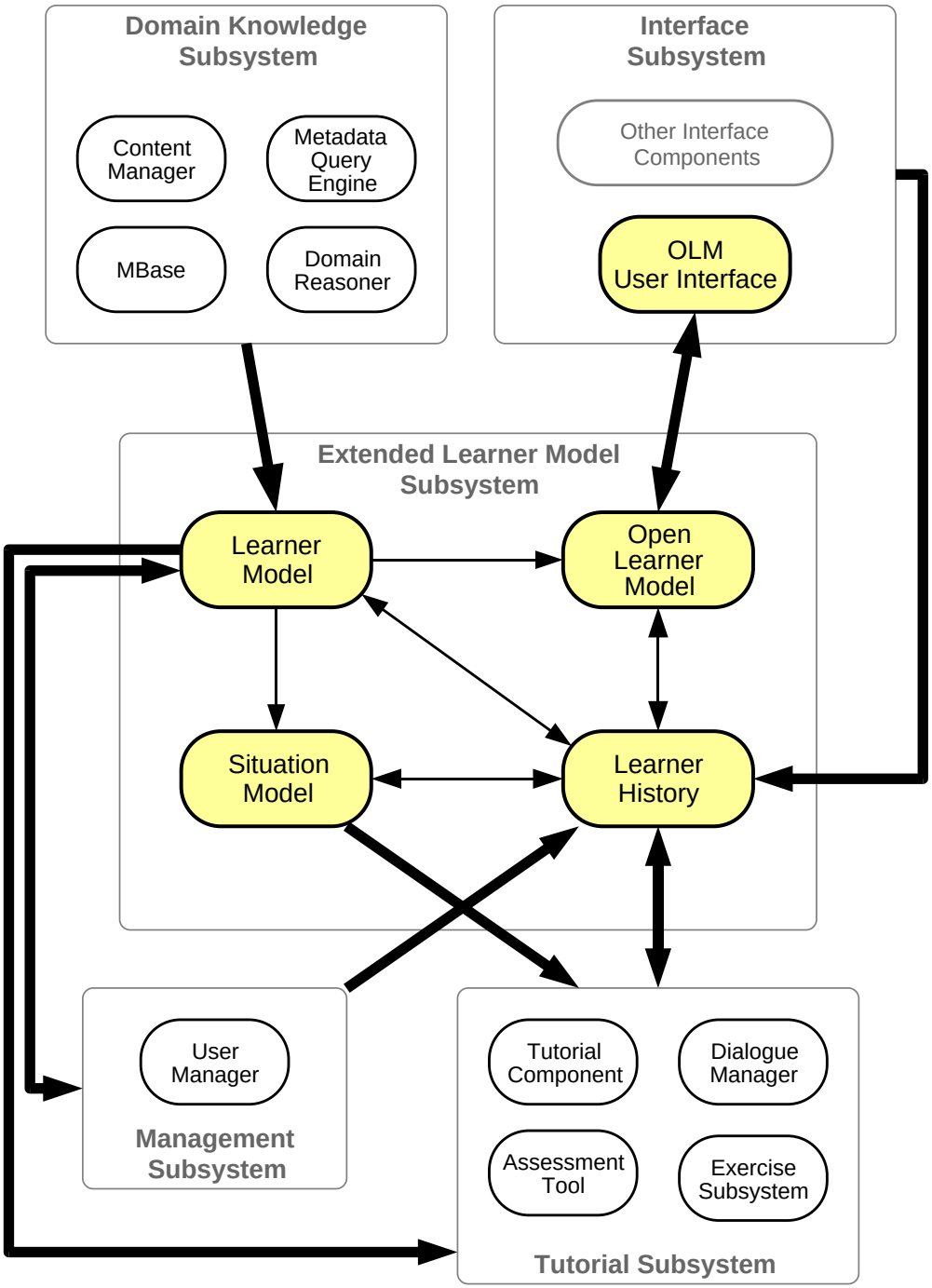


Figure 4.1: Proposed architecture for LEACTIVEMATH in relation to the learner model. Yellow (grey) boxes represent components of LEACTIVEMATH regarded as directly involved in the learner modelling process and hence included in the Extended Learner Model subsystem. Arrows in the diagram represent *information flow* between components.

Exercise Subsystem (ES) This component is in charge of executing exercises for learners to interact with them. As the Extended Learner Model, this is a subsystem that actually exists as such in LEACTIVEMATH.

Learner History (LH) This component manages the storage and retrieval of all historical information that is regarded as relevant for learner modelling.

User Manager (UM) This component deals with users in an application-neutral sense, storing general information about them such as their names and usernames, logins and logouts, as well as general educational information, particularly information that is provided and modified by learners, such as field of interest and educational level.

Learner Model (LM) This is the component responsible for maintaining a faithful representation of the state of every individual learner. It is used for LEACTIVEMATH adaptation of learning experiences to each learner's educational needs.

Open Learner Model (OLM) This component provides the learner with access to the LEACTIVEMATH model of them, including facilities for the learner to alter their model either directly or indirectly.

Situation Model (SM) This component is responsible for capturing aspects of the situational context which are pertinent to tutors adapting their feedback to the individual cognitive and affective needs of the students and for diagnosing the degree to which a given student in a specific situation may need to be given freedom of initiative and may need to be approved of explicitly.

4.1.3 Examples of information exchange

Some examples of information exchange between the distinct components depicted in figure 4.1 are given below. The examples are very simplified, in the sense that they do not consider all information exchanged at a given point in time, which may include time stamping, original source of information, etc.

IS $\rightarrow$

- **OLM** A learner action, such as asking for evidence supporting a belief in the learner model.
- **LH** An event associated with an input device.

OLM $\rightarrow$

- **UI** An instruction to change the presentation of an updated belief (e.g. changing colour from red to yellow).
- **LH** An action taken by the learner (e.g. challenging the system belief on being competent in mathematical modelling).

LM $\rightarrow$

- **LH** A change in a belief on a learner's competency.
- **OLM** The current belief on the learner's knowledge of a topic (e.g. belief on the current learner's knowledge of the chain rule).
- **SM** Current belief on the learner's level of self-confidence.
- **DM** Current belief on the learner's competency of a topic.
- **TC** Current belief on the general learner's interest in the subject domain.
- **UM** The complete learner model for a given learner.

DK → **LM** Structure of the subject domain knowledge (e.g. structure of domain knowledge for calculus) to be used for constructing the learner model.

SM →

LH Change in the learner motivational state (believed to happen, for example, on the basis of learner hesitation in answering a question or in completing an step in an exercise.).

DM Levels of approval and autonomy recommended for the current learner.

TC Levels of approval and autonomy recommended for the current learner.

LH →

LM New event in learner interaction with the dialogue manager (e.g. request for help).

OLM Last change to the belief on the learner's knowledge of a topic.

SM Number of events of a particularly type that occurred in the current session (e.g. number of help requests).

DM Last occurrence of a particular type of event involving the current user (e.g. last time the learner carried out a derivation of a trigonometric function).

TC List of events of a particular type involving the current user (e.g. all exercises started by the learner in the current session).

DM →

LH An action executed by the learner and its evaluation (e.g. the learner has failed at deriving $\sin(x)$ on her own).

TC →

LH Amount of time the learner spent on dealing with a particular exercise.

ES → **LH** Events related to learner interaction with an exercise (e.g. the learner has chosen a wrong answer in a multiple choice question).

UM → **LH** Login events for learners.

4.1.4 Rationale

The justification for the logical architecture of the Extended Learner Model described above is given below following Carroll's approach of describing its causal schemata as a set of claims Carroll and Rosson (1992).

Claim 4.1.1 *LM gets information from DK.*

Supports LM critical activities of initial construction and posterior updating of learner models, and opens the door to learner modelling in several domains. It also promotes independence between components and promises easier decoupling of xLM from the rest of LEACTIVEMATH (e.g. to act as an independent provider of learner modelling services).

Because LM needs to access information concerning the structure of the domain and relevant characteristics of learning activities, in order to build and update learner models. Keeping the description of the domain in DK, and providing it to LM as needed, encourages the use of a sharable domain description in LEACTIVEMATH, and of flexible modelling mechanisms in LM which (within reasonable limits) can be adapted to several domains.

Check rule(s) are (a) the existence of a sharable domain description among the LEACTIVEMATH components in DK; (b) the existence of suitable services in DK and, if needed, corresponding methods in LM; and (c) the consistency between concurrent LM and DK contents.

Issue(s) are the selection of both an appropriate scheme for describing LEACTIVEMATH domain(s), which should be adaptable to other (similar) domains, and of the communication mechanism employed to deliver information from DK to LM.

Claim 4.1.2 *LH is the* single *source of information for updating a learner model.*

Supports the justifiability of a learner model in terms of the corresponding learner history.

Because every *belief* stored in LM will be based on *evidence* stored in LH. Any information coming from any other component, such as TC or DM, must to go through (and be stored in) LH.

An important consequence of this decision is that there must be *no direct manipulation* of LM by its open counterpart (OLM). In order to have any influence on LM, OLM must send information through the LH.

Check rule(s) are (a) the enforcement by the implementation of xLM logical architecture that evidence used for updating learner models is stored in their learner histories, and (b) the enforcement of cosistency between changes in LH and LM by learner modelling algorithms (and their implementation).

Issue(s) is the bandwidth of the communication channel between the Learner Model and the Learner History, which needs to support *practical* dynamic modelling of the learner.

Claim 4.1.3 *LM and SM can send information to LH.*

Supports accountability of learner models and situation models to the view of learners and other stake-holders. Furthermore, it also supports the use of situation models, which are localised to the ongoing situation, as evidence for longer term learner models.

Because changes to a learner and situation models can be registered into the corresponding learner history.

Check rule(s) are (a) the existence of suitable communication mechanisms from LM and SM to LH, as well as

(b) the existence of records in learner histories that are consistent with changes in the corresponding learner and situation models.

Issue(s) is the amount of information that may accumulate in LH as a consequence of many learner and situation models changing over time.

Claim 4.1.4 *SM has access to both LM and LH.*

Supports situation models to be built on the basis of what is believed about the learner (in LM) and past events related to the learner (in LH).

Because access to LM allows SM to build/initialise its models and hence to provide suggestion to other components on the basis of beliefs inferred by LM, whilst access to LH empowers SM to develop its own beliefs about a learner independently to (and not limited by) the learner model.

Check rule(s) is the existence of suitable communication mechanisms from LM and LH to SM.

Claim 4.1.5 *OLM have access to both LM and LH.*

Supports presentation and justification of learner models to learners.

Because LM can provide the beliefs hold about the learner, whereas LH can provide all the evidence supporting such beliefs (claim 4.1.2).

Check rule(s) are the availability of suitable access methods in LM and LH, accessible to OLM.

Issue(s) the existence of a general framework for identifying beliefs and related evidence, so that OLM (and other components) can request LM and LH for specific information.

Claim 4.1.6 *OLM and UI can exchange information in both directions.*

Supports OLM to timely interact with learners, displaying information and receiving feedback.

Because OLM can make use of the interaction facilities provided by UI.

Check rule(s) is the existence of suitable communication mechanisms on both sides.

Issue(s) are the way in which the communication mechanisms should be implemented, taking into account the browser-based nature of LEACTIVEMATH. It is possible for some parts of OLM to be located on the client side (learner's computer) directly controlling the user interface.

Claim 4.1.7 *LM, LH and SM provide access mechanisms to components outside the xLM subsystem, such as UM, DM and TC.*

Supports adaptation of LEACTIVEMATH to the goals, cognitive, motivational and affective needs of learners.

Because other components in the system, especially DM and TC, can make use of the inferences made by xLM about the current cognitive, affective and emotive state of the learner, which are expected to be reliable. UM access to LM makes it possible for UM to act as a proxy of LM to the rest of xLM, providing in that way a single access point to information about the learner state and characteristics.

Check rule(s) will be the existence of such access mechanisms available to LEACTIVEMATH components outside xLM.

Issue(s) are the way in which such mechanisms are going to be available. Given the distributed nature of LEACTIVEMATH, it can be expected xLM to serve components running on distinct computers connected through the Internet[1].

Claim 4.1.8 *UM provides access mechanisms to LM.*

Supports independence between components and promises easier decoupling of xLM from the rest of LEACTIVEMATH

Because it makes possible for LM to serve as the single point of information about the learner state and characteristics inside xLM, an equivalent role to the one played by UM outside xLM. This information can be used by OLM, for example, to tailor communication to each individual learner (e.g. calling her by her name).

Check rule(s) is the existence of access mechanism from UM to LM.

Issue(s) is the definition of a suitable minimum set of management information necessary for xLM internal operation.

4.2 xLM communication architecture

Figure 4.2 depicts the information flow architecture of Extended Learner Model including details of the two ways in which information flows in between components and subsystems of LEACTIVEMATH: method calls and event publication.

[1] Actually, it may even be the case for distinct components inside xLM to live in distinct computers.

- Information flow via synchronous communication, as *methods calls*, is represented using solid arrows. Both a component *R requesting* information from a component *S* and this sending it back, and component *S commanding* component *R* by sending information are represented by an arrow linking *S* to *R*. This form of information flow occur both inside xLM subsystem and, through an xLM API element, in between xLM components and other components of LEACTIVEMATH outside the xLM subsystem.
- Information flow via asynchronous communication, as *event publication*, is represented using dashed arrows, while LEACTIVEMATH components that implement the event publication framework are depicted as square boxes. A local xLM Event Manager (xEM) is responsible for listening to events coming both from inside xLM and from the rest of LEACTIVEMATH, and for publishing the events either locally to each xLM component's listener or to the LEACTIVEMATH Event Manager (EM).

This communication architecture provides all LEACTIVEMATH components with a single front-end to xLM and its components, the xLM Manager. The rationale behind this feature is to provide xLM with a single connection point between xLM and the rest of LEACTIVEMATH, so that xLM remains as autonomous as possible, even when running as part of LEACTIVEMATH. Some important consequences of this design are the following:

- Besides information specific to the subject domain, as provided by the Domain Knowledge subsystem (DK), and information gathered through OLM GUI, all remaining information coming to xLM and used to update both situation and learner models is received through asynchronous communication as events.
- Since event flow inside xLM is controlled by the xLM Manager, it can be made to ensure that specific communication constraints specified in the logical architecture of xLM (section 4.1) are met. That is the case, in particular, of the constraint that every event used to update a learner model should reach and be stored in the learner history.
- The xLM subsystem can define local events, only relevant to its components, which can be distributed locally but are not visible outside xLM.
- Any event published by one of the xLM components (e.g. new evidence for belief revision from the OLM) is treated locally and immediately (e.g. storage of the event in the LH and revision of beliefs by LM). This feature could be a minor optimisation if xLM runs on the same JAVA virtual machine as the system that makes use of its services (as is the case of LEACTIVEMATH), but it would be very important in other cases; for example, in cases xLM where running as an independent web service.

From a more implementational point of view, the xLM manager introduced in Figure 4.2 servers three purposes:

(a) as a facade object for xLM API,
(b) as a publisher of local events inside xLM and
(c) as a forwarder of public events in between xLM and the rest of LEACTIVEMATH.

As a facade object, the xLM Manager is configurable as a LEACTIVEMATH component that creates instances of each xLM component and holds references to them. As a local publisher and forwarder, the xLM Manager subscribes itself to all learner-related events at the central LEACTIVEMATH Event Manager, hence all xLM components subscribe as listeners to the former instead of subscribing to the latter. Later on, the xLM Manager forwards all incoming events from the rest of LEACTIVEMATH to each component that has subscribed to it. The xLM Manager is also the place where each xLM component publish their own local and public events. Local events are published in the same way as public events, but they are not forwarded to the LEACTIVEMATH Event Manager and hence are not visible outside xLM.

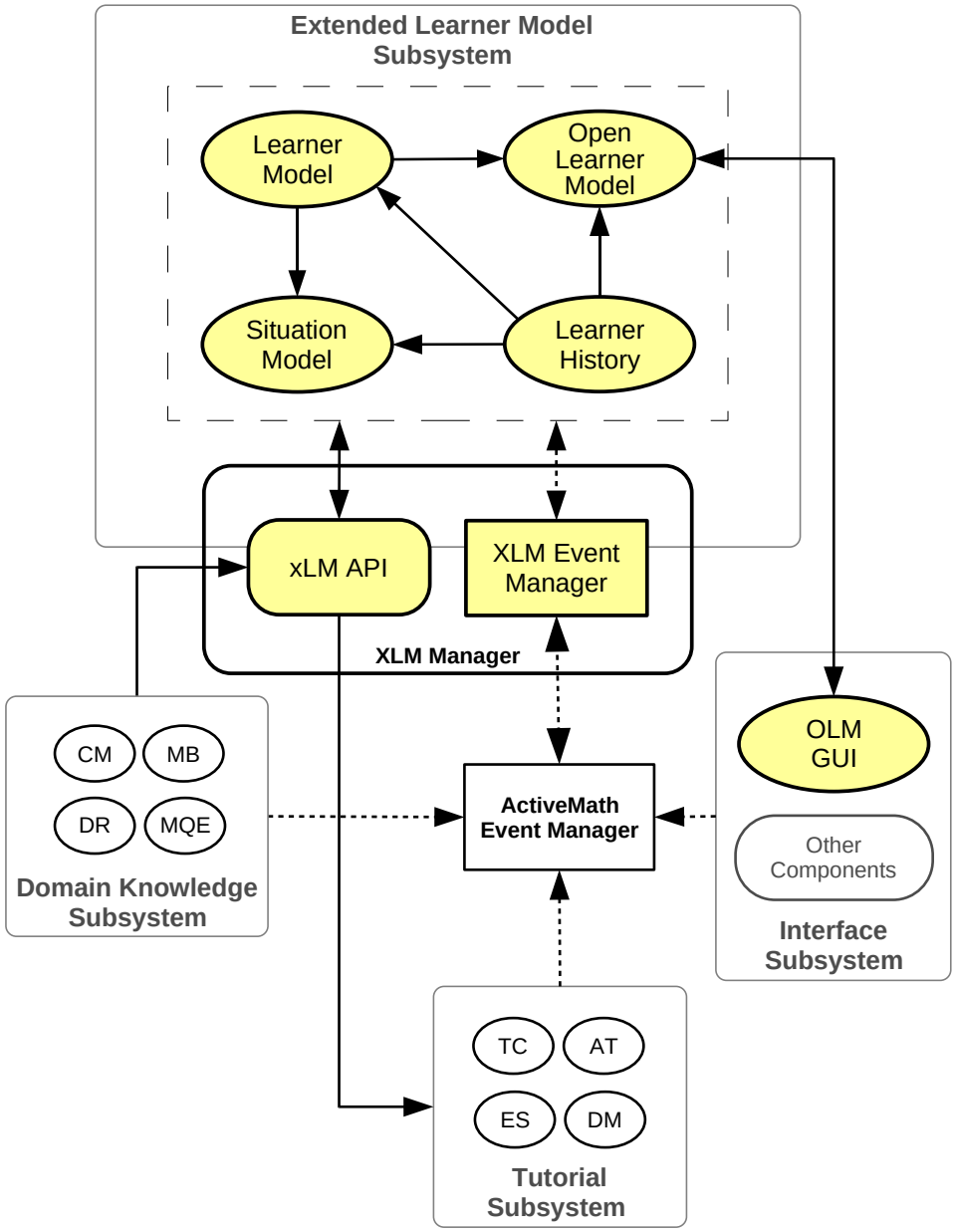


Figure 4.2: Communication architecture for the Extended Learner Model subsystem of LEACTIVEMATH.

4.3 The Learner History

The Learner History (LH) is the part of the extended Learner Model (xLM) responsible for storing events related to the learner that possibly affect the state of the Learner Model, Open Learner Model or the Situation Model. It gets notified of such events by the components in which they occur, e.g. by the exercise subsystem, it stores them, and it propagates them to the relevant components, e.g. to the Learner Model. Such propagation makes sure that all updates and modifications in the OLM, LM or SM are based on the same data, and thereby helps keeping data in the xLM consistent.

The LH will support queries in the form of filters, which can be used to set constraints on event attributes. It will also offer a few predefined query-methods for queries that are frequently used, e.g. whether a learning item has already been seen by the learner.

The Learner History (LH) may affect the:

- Learner Model (LM)
- Situation Model (SM)
- Open Learner Model (OLM)
- Dialogue Manager (DM)
- Tutorial Component (TC)

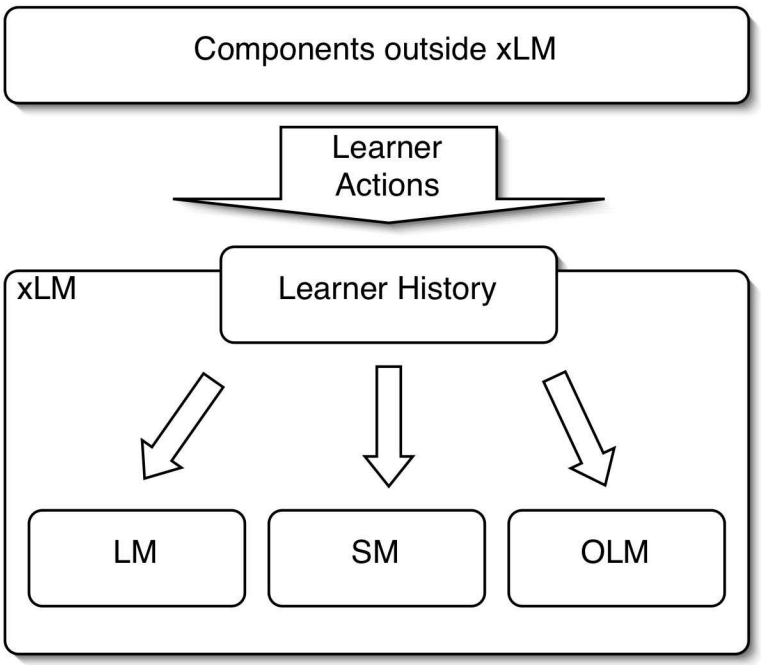


Figure 4.3: LH as Central Information Entry Point

Communication with the Learner History will be possible in two ways:

- Event posting for asynchronous communication.LEACTIVEMATH will offer an event framework for components that want to be notified of interesting events that occur in other

components or services. In this framework, components can choose to *subscribe* to events that other components *publish*. The LH will subscribe to several components that track the learner's actions, thus building up a history of learner's actions (Requirement 5.3).

- Method calls via a traditional application interface (API) for synchronous communication with the Learner History. The LH will expose the query methods in this way.

The LH is the central entry point for information about the learner's actions for xLM components (Fig. 4.3). This means that every learner action that is interesting to the xLM is stored in the Learner History before it is relayed to the other xLM components. The procedure ensures that every inference made in the xLM, e.g., the update of a belief in the LM, is justifiable by the learner's actions that are recorded in the LH. (Requirement 5.4)

4.3.1 Learner History Input

Only events will be used to convey input to the LH. As the LH is the single source of input to the xLM, this will allow for a loose coupling between xLM and other components. When the LH gets an event, it will store it in its underlying database and notify subscribed components. This event will contain the (unique) identifier assigned to the stored information. The other xLM components can therefore tag their inference steps directly with the event's ID to justify them. We believe that the set of events listed in Tables 4.1, 4.2, 4.3, 4.4 and 4.5 offers a good coverage of relevant learner/system-interactions. However, this set of events can easily be modified later on. The current list of events listened by xLM can be found in Appendix B.

4.3.2 Learner History Output

The LH will propagate all information it gets to the registered subcomponents of the xLM, thereby removing the need of these components constantly querying the LH for new events. Furthermore, the LH will offer queries via methods supporting the use of *filters* that set constraints on values in the columns in the LH's underlying database. For most of the columns the constraint will be an equality constraint. For some exceptions, however, equality will not be enough, e.g. for the timestamp value. A list of these special constraints can be found in Table 4.6. The complete interface contains the following methods:

A method needed at least by the Tutorial Component is to check whether a given learner has already seen a given item.

`boolean alreadySeen(String learnerId, String itemId)` returns `true` if the learner has already seen the given item, `false` otherwise.

As some queries might return a large list of Events, we introduce filters in order to restrict the results. Filter objects are sets of properties that are related to the columns used in the storage database. Examples of properties include:

- type (= event Type)
- itemId
- bookId
- maxSuccess

The following methods can be used to query the history:

- `int getNumEntries(String learnerId, List<Filter> includeFilter, List<Filter> excludeFilter))` returns the number of stored events that match the includeFilter but do not match the exclude Filter.
- `List getHistoryEntries(String learnerId, List<Filter> includeFilter, List<Filter> excludeFilter, int startIndex, int maxNum)` returns a List of `maxNum` events for the given learner that match the `includeFilters` but do not match the `excludeFilters`, starting at the index `startIndex` of the full result list.

The following method can be used to retrieve stored events in the LH by using identifiers. This is useful for justifying the current state of the LM.

`Event fetchEvent(String id)` returns the event with the specified ID.

4.3.3 Using Filters to Query the LH

We list a few anticipated usages of filters to query the LH to give a feeling what the queries will look like.

- Query: "All exercises solved at least to 50 percent"
  - IncludeFilter: $type \rightarrow Exercise, minSuccess \rightarrow 0.5$
  - ExcludeFilter: $\emptyset$
- "All exercises solved at most to 50 percent"
  - IncludeFilter: $type \rightarrow Exercise$
  - ExcludeFilter: $minSuccess \rightarrow 0.50$
- "All actions on which the student spent more than 5 minutes (300 s.)"
  - IncludeFilter: $minDuration \rightarrow 300$
  - ExcludeFilter: $\emptyset$

User Administration		
Event Name	Description	Additional Parameters
UserLoggedIn	User logged in	
UserLoggedOut	User Logged out	
UserCreated	User has registered to the system	
UserRemoved	User has 'left' the system	

Table 4.1: LH input events: User Administration Events

4.4 The Learner Model

This section gives a detailed description of the functionality, internal structure and content of the Learner Model. Each of the dimensions used to organise the representation of the learner's states and dispositions, as presented in section 2.4, are described here in detail. An overview is also given of the sources of evidence used to update the set of beliefs that conform a learner model,

User Navigation		
Event Name	Description	Additional Parameters
PagePresented	User has been presented a page	Reference to page Duration
ItemPresented	User has been presented an item	Reference to item Estimate of duration
DictSearched	User has searched the dictionary	Search query
UserBookPlanned	User has requested a course	Reference to course Goals and paramaters
UserBookRemoved	User has deleted a course	Reference to course

Table 4.2: LH input Events: Navigation Events

Exercise and Dialogue		
Event Name	Description	Additional Parameters
ExerciseFinished	User finished an Exercise	State: Abort, Timeout... Overall Rating Reference to exercise
ExerciseStep	User made an exercise action	Action description Reference to exercise Metadata of the exercise step

Table 4.3: LH input Events: NL and Exercise Events

Open Learner Model		
Event Name	Description	Additional Parameters
olmEvent	Interaction occurred in the OLM	Learner Id Interaction Topic

Table 4.4: LH input Events: OLM Events

System Actions		
Event Name	Description	Additional Parameters
BeliefChanged	LM updated a belief	Belief Topic New Value Old Value
UserPropertyChanged	LM changed a property	What property Old Value New Value
StateChanged	SM changed affective state	What state Old Value New Value

Table 4.5: LH input Events: System Events

Special Constraints	
minSuccess	filters out lines with success $<$ *minSuccess*
minTimestamp	filters out entries that occured before *minTimestamp*
minDuration	filters out entries with duration $<$ *minDuration*
maxSuccess	filters out lines with success $>$ *minSuccess*
maxTimestamp	filters out entries that occured after *maxTimestamp*
maxDuration	filters out entries with duration $>$ *maxDuration*

Table 4.6: Special Filter Constraints

as well as indications of any outstanding issues, the alternatives envisaged and the estimated schedule for resolving them.

Then the nature of the beliefs in the Learner Model is presented, as well as the mechanism used for updating the model based on external and internal evidence.

This section will be concluded by some considerations about the computational complexity of the model.

4.4.1 Functionality

In the architecture of the Extended Learner Model (section 4.1) the Learner Model (LM) component acts as a client of the Domain Knowledge Subsystem (DK) and the Learner History, whilst it acts as a server of the Learner History, the Open Learner Model (OLM), the Situation Model (SM) and the Tutorial Subsystem (TS). This means, in particular, that changes to LM negotiated by OLM have to be sent to LM via LH. In this way, any belief stored in LM is justifiable in terms of information stored in LH.

The approach taken to specify the functionality of LM is based on *use cases*, which, roughly speaking, are descriptions of how LM interacts with other components of LEACTIVEMATH under different circumstances. Five use cases are described here:

1. creating a learner model,
2. updating a learner model on-line,
3. updating a learner model off-line,
4. exposing a learner model in full, and
5. exposing details of a learner model.

They embody the classic create, update and select operations in databases, which also apply to learner models.

4.4.1.1 Creating a Learner Model

It is good design, and increases the flexibility of LM, to keep the *definition of the structure of the domain* in DK, where knowledge about the subject domain is kept, and to retrieve it (if necessary) when creating a new learner model.

Description

1. LM gets a request from a component.
    - Learner identifier

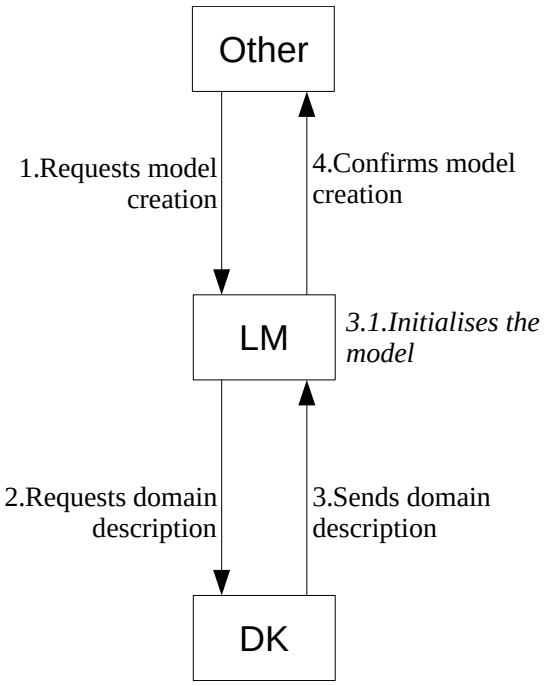


Figure 4.4: Use Case for creating a Learner Model

- Preferences and other traits

2. LM sends a request to DK.
3. LM gets a response from DK.
   - Domain description.

   (a) LM initialises the model for the given learner
4. LM confirms model creation

Claims

Claim 4.4.1 *The definition of the structure of the domain is kept in DK, and it is retrieved by LM (if necessary) when creating a new learner model.*

Supports a better design of both DK and LM, and increases the flexibility of LM to be applied to more than one domain.

Because the need to share the definition of the structure of the domain knowledge between DK and LM — developed by distinct teams in the project — encourages the production of a cleaner definition. If the organisation and encoding of the definition is based on standards, then the same mechanisms can be used by LM to build learner models for other domains.

Check rule(s) are

1. the existence of a specification for the organisation and encoding of definitions of domain structure (DSD's),
2. the existence of a conforming DSD for LEACTIVEMATH's domain,

3. the existence of suitable methods in DK (and LM, if necessary) for recovering (getting) a DSD, and
4. the verification that changes in LEACTIVEMATH DSD produces consistent changes in new learner models.

Issue(s) are, among others,

1. the selection of a good format for encoding DSD's,
2. dealing with interactions between domain-independent and domain-dependent sections in LM,
3. managing changes in LEACTIVEMATH DSD once the system is deployed (e.g. keeping different versions of it).

4.4.1.2 On-line Updating of a Learner Model

Continuous updating of beliefs in a learner model feeds from, and it is triggered by, information about new events made available by LH. However, it may be necessary to complement this information with further information about the learning objects a learner has been interacting with, and this would need to be requested from DK by the LM.

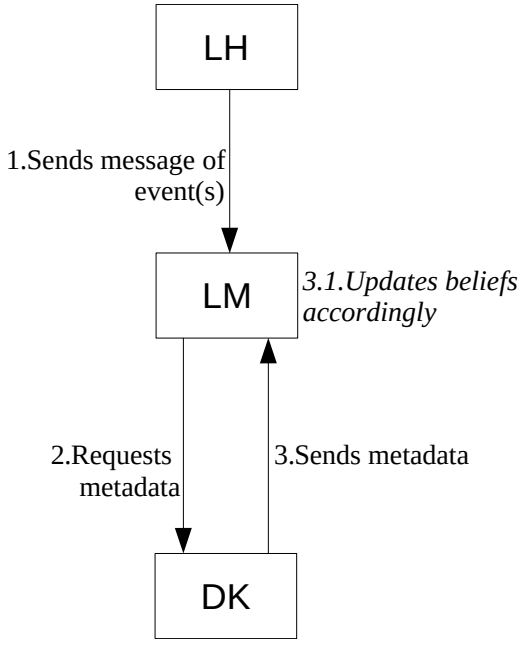


Figure 4.5: Use case for updating a learner model on-line.

Description

1. LM gets a message from LH.
   - Learner identifier
   - Description of event
2. LM sends request to DK.

- Learning object identifiers

3. LM gets a response from DK.
   - Metadata

   (a) LM updates its beliefs accordingly.

Claims

Claim 4.4.2 *Continuous updating of beliefs in a learner model feeds* only *from information on new events made available by LH, complemented with metadata of learning objects made available by DK.*

Supports consistency of beliefs in LM with information in LH and DK.

Because beliefs in LM are calculated exclusively from such information.

Check rule(s) is the existence of an updating algorithm which is sound, predictable, plausible and implemented correctly.

Issue(s) relate to LH capacity of keeping all sorts of information which are necessary for supporting a useful and interesting LM.

Claim 4.4.3 *Continuous updating of beliefs in a learner model is triggered by information on new events made available by LH.*

Supports dynamic updating of a learner model as the learner interacts with LEACTIVEMATH.

Because LM reacts to new events registered in LH (e.g. learner actions, DM utterances, OLM negotiations) by updating beliefs accordingly.

Check rule(s) is the implementation of a mechanism allowing LH to initiate and keep sending information to LM, and corresponding mechanisms in LM to receive such information and act in consequence.

Issue(s) concern the performance of the supporting mechanisms, which need to keep pace with the demands of intelligent adaptation to learner needs.

Claim 4.4.4 *LM is able to request information (metadata) about learning objects that have provided the context for events registered in LM.*

Supports better understanding of events.

Because contextual information is crucial for proper interpretation of events such as learner actions.

Check rule(s) are

1. the presence, in the description of every event, of information about the learning objects that provided its context;
2. the existence of mechanisms in LM and DK for recovering metadata of well identified learning objects.

Issue(s) are, on one hand,

1. the provision of sufficient metadata information for learning objects, so that a good picture of the context can be retrieved; on the other hand,
2. metadata information may not be sufficient for obtaining a good picture of the context in which an event occur, hence other sources of information may be necessary — such as events providing contextual information, and LM internal contextual information.

4.4.1.3 Off-line Updating of a Learner Model

There are situations in which LM may need to take the initiative to update a learner model. For example, LM may need to be synchronized after some maintenance of LH; there may be some events to which LM does not react immediately but only after a critical amount of them have been accumulated in LH, etc.

In this situations, LM will initiate the process by requesting from LH all events matching a given pattern. Once the set of events is received by LM, the process will continue as in the previous use case (section 4.4.1.2).

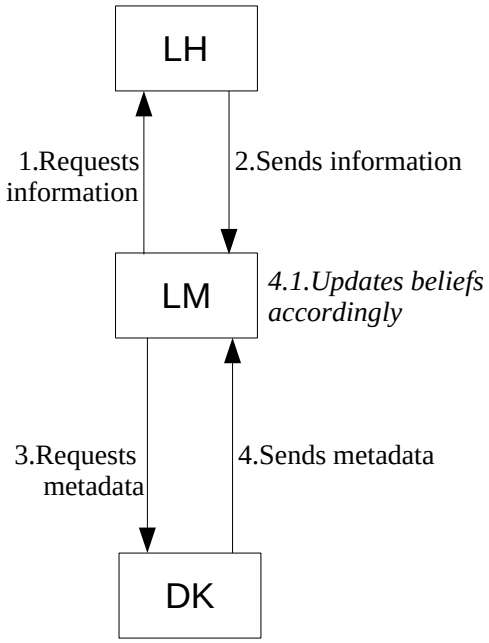


Figure 4.6: Use Case for updating a Learner Model off-line

Description

1. LM request information from LH.
   - Learner identifier
   - Filtering specification
2. LM gets a response from LH.
   - Learner identifier
   - List of events
3. LM sends request to DK.
   - Learning object identifiers
4. LM gets a response from DK.
   - Metadata

   (a) LM updates its beliefs accordingly

Claims

Claim 4.4.5 *In some situations, LM can take the initiative to update a learner model, and initiate the process by requesting from LH all events matching a given pattern.*

Supports flexibility in the process of updating the learner model.

Because in this way LM may reconsider evidence stored in LH, delay the evaluation of events until some criterion is met, or react to indirect requests from OLM, among other possibilities.

Check rule(s) are

1. the provision in LEACTIVEMATH of a representation format for event patterns,
2. the existence of suitable interface methods in LH, and in LM if necessary, which take an event pattern as input and produce the set of all events matching that pattern, and
3. the existence of mechanisms in LM for building such patterns and processing the resulting events.

Issue(s) are the characteristics of the pattern matching facilities provided by LH, which should be easy to use, flexible enough to accommodate interesting behaviour from LM, and efficient (given the number of events that may be stored in LH at one time).

4.4.1.4 Fully exposing a Learner Model

The full content of a learner model may be needed by OLM at the beginning of an interaction with a learner (e.g. to present the model to the learner). Other components of LEACTIVEMATH—such as SM, DM and TC — may also benefit from getting access to a whole learner model. Other possible use is for exporting a learner model (e.g. to be used in another instance of LEACTIVEMATH).

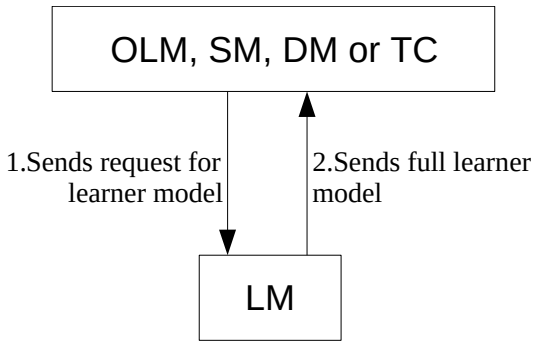


Figure 4.7: Use Case for exposing a Learner Model in full

Description

1. LM gets initial request from OLM, SM, DM or TC.
   - Learner identifier
2. LM answers to OLM or SM.
   - Complete learner model.

Claims

Claim 4.4.6 *LM is able to provide a whole learner model on demand.*

Supports 1. other components of LEACTIVEMATH, such as OLM, to have tailored access to (the public part of) learner models, and

2. a cleaner design of LM.

Because 1. each component can implement a local access mechanism tailored to its own needs (e.g. for a graphical display of the learner model), and

2. LM behaves more like a "glass box" than a "black box".

Check rule(s) are

1. the existence of a specification for the organisation and encoding of sharable representations of learner models,
2. the existence of a suitable method in LM (and in other components of LEACTIVEMATH, if necessary) for producing a sharable representation of a learner model, and
3. the verification that learner models and their sharable representations are consistent.

Issue(s) concern mainly to the selection of a good format for encoding sharable representations of learner models.

4.4.1.5 Partial Exposition of a Learner Model

Depending on the belief updating algorithm employed by LM, a single event may trigger changes to several beliefs in a learner model, many of them not relevant to the next decision made by each component in LEACTIVEMATH. Hence, during a learning session and as a learner model evolves, access to specific beliefs in it is more convenient than requesting complete copies of it.

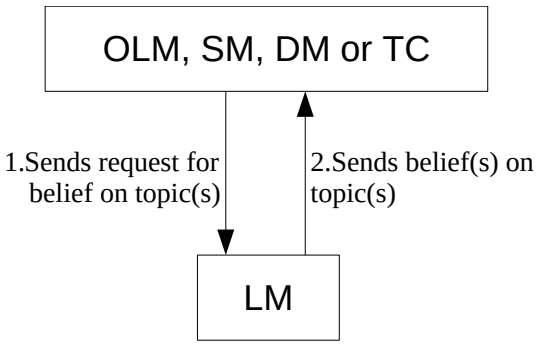


Figure 4.8: Use Case for exposing details of a Learner Model

Description

1. LM gets a request from OLM, SM, DM or TC.
   - Learner identifier
   - Topic identifier(s).
2. LM answers.
   - Belief(s) on topic(s)

Claims

Claim 4.4.7 *LM provides access to individual beliefs in learner models, given learner and belief identifiers.*

Supports adaptation to learner needs through easy access to individual beliefs about the learner by LEACTIVEMATH components other than LM.

Because each component can request the value for each belief directly relevant to its decision making.

Check rule(s) is the existence of an LM interface method implementing this feature.

Issue(s) relate to how other LEACTIVEMATH components may get to know the distinct beliefs in a learner model. One possibility is for each component recovering first the whole model (see section 4.4.1.4), which should include an identifier for each belief. Another possibility is to publish all belief identifiers in a suitable way.

4.4.2 The Dimensions

This section gives a detailed description of each dimension used to organise the representation of the learner's states and dispositions, as presented in section 2.4

4.4.2.1 Subject Domain

The subject domain is not a static map that can be loaded once for all in the Learner Model. This is due to the nature of LEACTIVEMATH and its dynamic and evolutive content-based approach. The *domain topics*, i.e. the individual concepts of the domain on which the modelling process will be based, have to be generated on the fly, as the learner is presented with exercises, examples, etc. Their identification relies on OMDOC, the language used for representing – and marking – content in LEACTIVEMATH.

There is two types of knowledge objects in OMDOC (see Kohlhase (2005); LeActiveMath Partners (2004d)): *concepts* (i.e. symbols, theorems, axioms, definition) which are the principal targets of learning and *satellites* (i.e. examples, exercises, introduction text, etc.) which are used to support the learning of the concepts. Concepts and satellites are inter-connected by typed relations such as for, *requires*, *is_basis_for*, etc.

As target for beliefs, the Learner Model is only considering the knowledge objects at their most abstract level. This is to ensure that evidence coming from the learner's interaction with exercises (i.e. at the content level) is accumulated and generalised at the domain level.

The figure 4.9 illustrates the structure of an authored content, arbitrarily reorganised in order to emphasise the distinction between the "abstract" domain (i.e. a context-independent description of the topics of learning) and the "concrete" contents (i.e. the devices used to formalise, organise and present these topics to the learner).

The central part of the figure (the *deriv_domain* group) consists in domain topics in the context of Differential Calculus: "Chain Rule", "Funtions", "Curve", "Slope", etc. They are connected by associations in order to specify their structural dependence, e.g. the "Chain Rule" is a kind of "Rules for Differentiation", the "Slope" relies on the notion of "Curve", etc. On the basis of this formal description of the "domain", authors can introduce contents (such as in the groups $deriv_rules$, $deriv_numerical$, etc.) by defining theorems, axioms, examples, exercises, etc. They are related to the topic(s) they present (e.g. the theorem X is for the "Chain Rule", the exercise Y is for the "Product Rule") and relations can also be used to express pedagogical dependence (e.g. the exercise Y requires "Polynomial Functions" as prior knowledge).

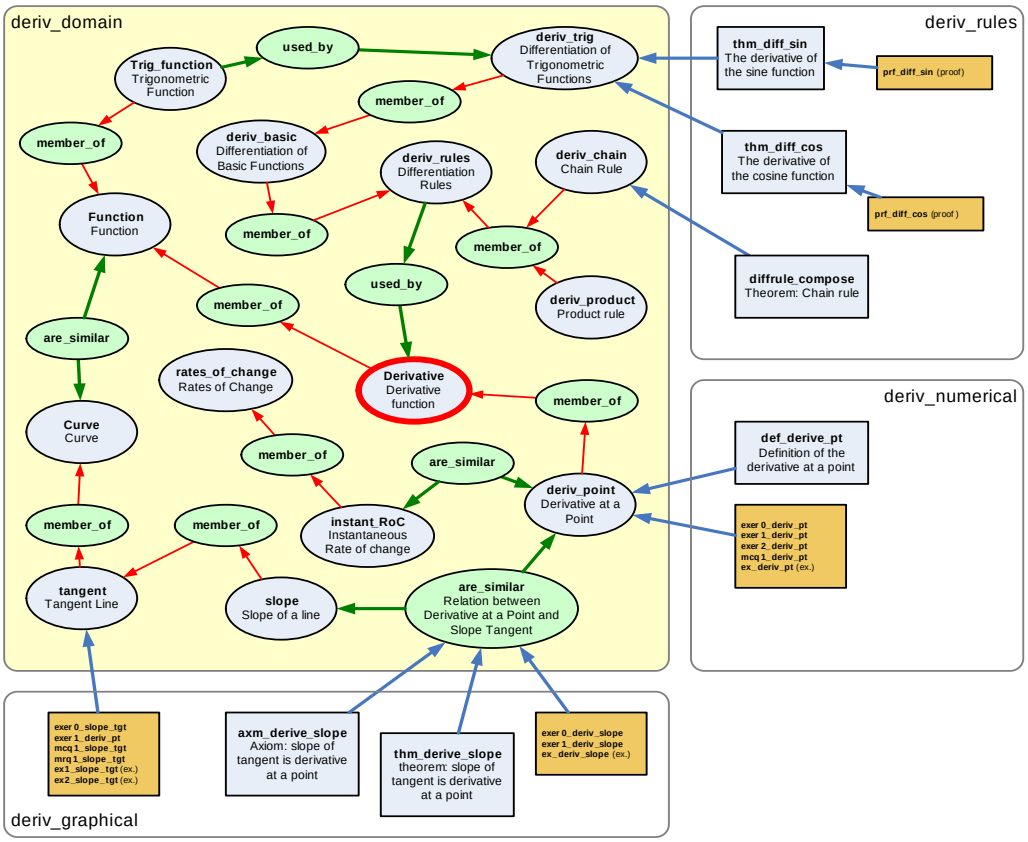


Figure 4.9: Extracting the domain topics from the content presented to the learner.

Accumulating evidence solely at the domain topics level not mean that information related to interaction with satellites (e.g. learner's performance in an exercise, learner reading an introduction) are lost, as they are individually interpreted (e.g. performance evaluation is given, time spent on a text is given) and stored in the Learner History.

Source of Evidence

- Each time the learner accesses a piece of content, all the domain topics related directly (e.g. the symbols supported by this content) or indirectly (e.g. all symbols connected to the content's symbols) to the content are added to the Learner Model, if they are not already present.

Outstanding Issues

- Finding a match between the "domain topics" of the Learner Model and the "knowledge objects" of the content has been a long running issue. The OMDOC symbols are the obvious choice for the abstract domain topics of the Learner Model but they lack relations between themselves to express connectivity (e.g. "chain rule" is part of "derivation rules") and, hence, forbid propagation of evidence by the Learner Model. On the other hand, OMDOC ensures relations between symbols by the way of concepts like axioms or theorems (e.g. using an axiom to signify the existence of a relation between the "slope of the tangent

at a point" and the "derivative at a point") but introducing them in the Learner Model compromises the uniqueness of the abstract domain topics (e.g. the previous axiom could be represented as well by a theorem – with its formal proof – targeting University students). Moreover, representing such relations by symbols on which theorems and axioms could depend presents some obvious conceptual inconsistencies. The alternatives we are currently investigating are:

1. Introduction in OMDOC of extra types of knowledge objects for representing – at abstract level – the *topics* and *associations* that are the targets of learning (i.e. supported by pieces of content like theorems, exercises, etc.). This is the option illustrated in figure 4.9 and is the preferred alternative but still need a consensus within the project as it has implication beyond the Learner Model (i.e. extending OMDOC, slightly reauthoring parts of the content, collaboratively building the domain map).
2. Using only symbols and ensuring that every OMDOC concepts (theorems, axioms, ...) have a top-level symbol to which they refer directly. This is a short-term solution that can be immediately implemented but gives to the OMDOC symbols a semantic that is not consensual. Moreover, pedagogically important relations cannot be explicitly represented.
3. Using both OMDOC symbols and concepts for the Learner Model's domain topics and relying on an assumed consistency of the course generation to avoid duplication of concepts in the LM. By this approach, both topics and association (i.e. by way of theorems and axioms) are represented in the Learner Model but a deliberate care has to be given when authoring the content so that two similar objects are properly dealt with in the Learner Model (like for example, in figure 4.9, the theorem and the axiom in theory $deriv_graphical$, both defining the relation between the derivative at a point and the slope of a tangent at a point but for different public).

The solution 1 is the approach preferred by WP4; a decision will be made and implemented by month 18 .

4.4.2.2 Mathematical Competencies

The mathematical competencies represent the cognitive abilities of learners to deal with the learning material presented to them. The competency framework used in LEACTIVEMATH arise the current international discussion about standards and assessments in mathematics education, led by PISA and NCTM (see for example Niss (2002); Klieme et al. (2004)). They are currently used by WP6 a methodology to develop content.

The following description of the mathematical competencies used by the Learner Model is extracted from LeActiveMath Partners (2004d):

- **Think mathematically:** includes the ability to pose questions that are characteristic for mathematics ("Are there ... ?", "How does... change?", "Are there exceptions?"), understand and handle the scope and limitations of a given concept, make assumptions (e.g. extend the scope by changing conditions, generalize or specify, with reasons), distinguish between different kinds of mathematical statements (e.g. conditional assertions, propositional logic)

- **Argue mathematically:** includes the ability to develop and assess chains of arguments (explanations, reasons,proofs), know what a mathematical proof is and what not, describe solutions and give reasons for their correctness or incorrectness, uncover the basic ideas in a given line of arguments, understand reasoning and proof as fundamental aspects of mathematics

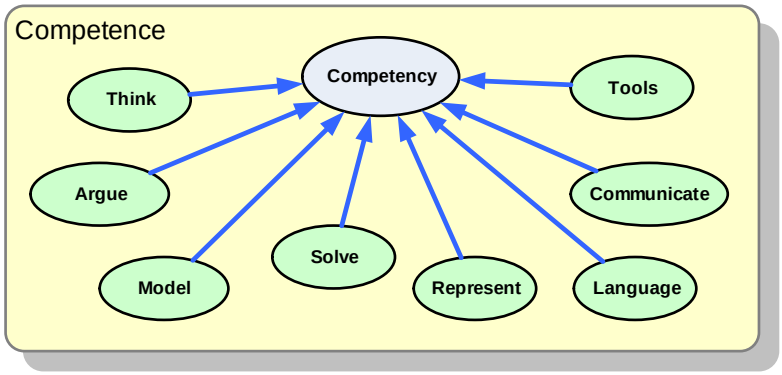


Figure 4.10: The mathematical competencies supported by the Learner Model.

- **Model mathematically:** includes the ability to identify, pose and specify problems, self-constitute problems, monitor and reflect on the process of problem solving, endue strategies / heuristics, solve different kinds of problems (with various contexts outside of mathematics, open-ended exercises)
- **Solve problem mathematically:** includes the ability to translate special areas and contents into mathematical terms, work in the model, interpret and verify the results in the situational context, point out the difference between the situation and the model
- **Use mathematical representations:** includes the ability to understand and utilize (decode, interpret, distinguish between) different sorts of representation (e.g., diagrams and tables) of mathematical objects, phenomena, and situations, find relations between different kinds of representation, choose the appropriate representation for the special purpose
- **Deal with symbolic and formal elements of mathematics:** includes the ability to use parameters, terms, equations and functions to model and interpret, translate from symbolic and formal language into natural language and the other way round, decode and interpret symbolic and formal mathematical language and understand its relations to natural language
- **Communicate:** includes the ability to explain solutions, use a special terminology,, work in groups, including to explain at the adequate level, understand and verify mathematical statements of others
- **Usage of tools and aids:** includes the ability to know about the existence of various tools and aids for mathematical activities, and their range and limitations, to reflectively use such tools and aids

Source of Evidence

- All examples and exercises in the content contain a metadata specifying which the competency (or competencies) they address or train, as well as the competency level a learner is assumed to have in order to successfully perform (see figure 4.15 for an extract of two exercises authored with OMDOC, where the metadata `<competency value="???"/>` and `<competencylevel value="???"/>` are specified accordingly). Based on the performance in the content and on its context (e.g. difficulty of the exercise), evidence about the competencies related to this exercise are accumulated to update the beliefs on the appropriate topics (section 4.4.4 for a description of the update mechanism).

- Every step of an interactive exercise – see LeActiveMath Partners (2004a) – could also contain metadata specifying the competency associated with that particular step (assumed to be one of the competencies of the exercise).
- In the Open Learner Model, the learner could investigate and challenge the various beliefs the Learner Model hold about his/her competencies. These interactive diagnoses are also sources of evidence taken into consideration when updating beliefs regarding competencies

Outstanding Issues

- The mathematical competencies being the core of the Learner Model, it is important that their diagnosis is as accurate and discriminate as possible. We are currently investigating the possibility of deepening the competencies map by adding "sub-competencies". This extra level of information has never been considered before by the project partners and, consequently, need a careful analysis and operationalisation. A first version of this extra-level has been produced, based on an ad hoc review of the content currently authored for LEACTIVEMATH. This initial version still need to be amended and extended by taking into account requirements coming from other parts of the project (in particular the Dialogue Manager and the Domain Reasoner).

  However, this issue is not a problem for the specification or the implementation of the Learner Model: evidence supporting competencies can still be accumulated and belief about competencies updated. A deeper specification of the competency dimension will only improve the reliability of the modelling process. This issue has to be seen as part of the iterative improvement of the xLM.

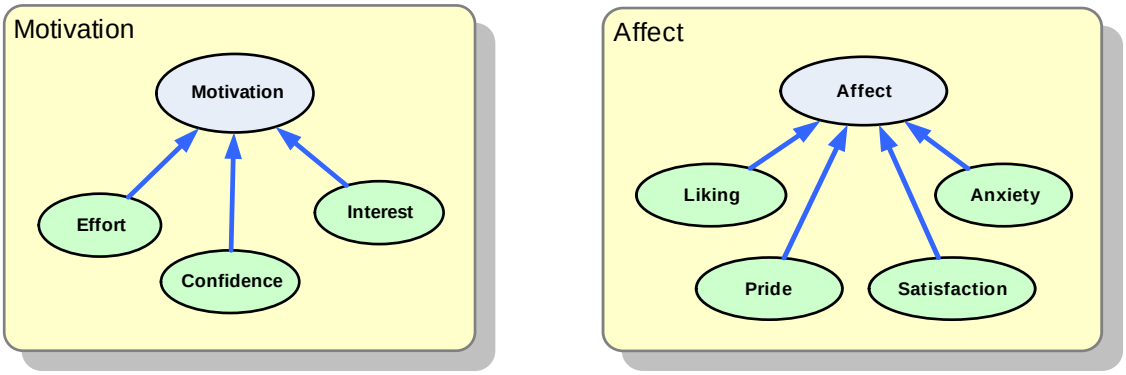
  A decision will be made and implemented by month 18 .

4.4.2.3 Motivation and Affect

As explained in section 2.4, the motivation dimension and the affect dimension of the Learner Model stands at the same level. This is why, in the rest of the document, both dimensions are often referred to altogether, as "motivation and affect". This does not mean that they represent the same dimension of the learner's characteristics: they cannot be combined (i.e. there are beliefs about the learner motivation AND beliefs about the learner's affect) and are diagnosed from different and independent sources of evidence.

The current model of motivation used by the Learner Model is based on some of the situational factors used by the Situation Model (see section 4.5), the difference being on the scope and pertinence of the information stored. Whereas the Situation Model considers only a very local, time-specific situation in order to give advices, the Learner Model on the other hand accumulates the factors discriminant to this situation in order to establish a long-term portrait of the learner's motivational state. Three motivational factors are actually considered:

- **Effort:** an estimate of the amount of work done by the student on the just completed task.
- **Confidence:** the level of student's positive self-belief in relation to their ability to tackle and to solve a given problem.
- **Interest:** the level of student's positive attitude towards the just completed task.



(a) Motivation factors

(b) Affect factors

Figure 4.11: The motivational and affective factors supported by the Learner Model.

The OCC model Ortony et al. (1988), despite is recognised over-simplification, is one of the most common model used for categorising emotions and building emotional agents. Its operationalisation - even partial - in many system (learning or not) gives us a reasonable ground for exploring the issue of the affective state of learners. The rational being to highlight the feasibility of the Learner Model, not to cover every emotions identified in the OCC model, we decided to select only one emotion for each of the perception branches of the model (i.e. events, agents and object). They are :

- **Liking/disliking:** the level of attraction arising from the current object of attention of the learner.
- **Pride/shame:** the level of attribution arising form the current actions of the learner.
- **Satisfaction/disapointment:** the level of prospect arising from the consequences for the learner of the current events.

In addition, and more specifically related to LEACTIVEMATH, discussion among the project partners have led us to consider looking at the issue of *Math anxiety* Richardson and Suinn (1972); Tobias (1995). Together with *Mathematics self-concept*, which refers to perceptions of personal ability to learn and perform tasks in mathematics Reyes (1984), these are the two dimensions of the learners' (negative) attitude toward mathematics that have been the focus of numerous research on mathematical disabilities and have an obvious impact on the.

- **Math anxiety:** the feelings of tension that interfere with the manipulation of mathematical numbers and the solving of mathematical problems in a wide variety of ordinary life and academic situations.

Source of Evidence

- The motivational factors, being a long-term accumulation of some of the situational factors considered by the SM, are evaluated by the Situation Diagnosis Agent (see section 4.5) when an interactive exercise is performed by the learner.
- Affective factors will be provided directly by the learner, using the self-report tool

- In the Open Learner Model, the learner could investigate and challenge the various beliefs the Learner Model hold about his/her affective and motivation states. Each of these challenges is another source of evidence for updating the relevant beliefs.

Outstanding Issues

- The studies currently ran in order to identify the situational factors and their source of evidence (see section 2.4) are not totally over and analysed. It is therefore expected that they may change in the near future and the motivation dimension of the Learner Model will be redefined accordingly. This will have only a minor impact on LEACTIVEMATH as a whole, as it is expected that most of the component requiring information about the learner's state will inquire the Learner Model about the dimension as a whole, e.g. motivation, whatever its internal breakdown.

  Nevertheless, a final decision about the motivational factors to represent in the Learner Model will be made and implemented by month 18 .

- One assumption that still hold at the time of writing is that direct evidence for affective factors will be gathered exclusively using self-reports (i.e. by the self-report tool and by the OLM). This may change, as part of the iterative improvement of the xLM, if further diagnosis mechanisms are suggested and operationalised.

4.4.2.4 Metacognition

Flavell (1976) defined metacognition as follows: "In any kind of cognitive transaction with the human or non-human environment, a variety of information processing activities may go on. Metacognition refers, among other things, to the active monitoring and consequent regulation and orchestration of these processes in relation to the cognitive objects or data on which they bear, usually in service of some concrete goal or objective." Hacker (1998) offered a more comprehensive definition of metacognition, to include the knowledge of one's own cognitive and affective processes and states as well as the ability to consciously and deliberately monitor and regulate those processes and states.

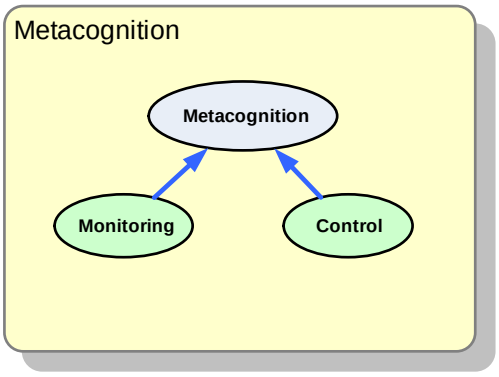


Figure 4.12: The metacognitive factors supported by the Learner Model.

Based on these definitions of metacognition, the Learner Model takes into account the following two metacognitive abilities:

- **Monitoring:** The ability of the learner to *actively* becoming conscious of his/her own congitive and affective processes and states.

- **Control:** The ability of the learner to *consciously* and *deliberately* regulate these processes and states.

The above definition of the factors implies a relation between them, as there cannot be any control if there is no evidence of monitoring from the learner.

Source of Evidence

- The Open Learner Model will be the major source of evidence for the learner's metacognitive abilities. Investigation of the beliefs stored in the model and challenges of the judgement made by the model are collected and generalised as evidence for monitoring. Patterns of interaction identified between the use of the OLM and the content explored or performed in LEACTIVEMATH are generalised as evidence for control.
- Some particular self-motivated interactions with LEACTIVEMATH, such as looking for a term in the dictionary, could be also considered as evidence for monitoring.

4.4.2.5 Conceptual and Procedural Errors (CAPEs)

The Conceptual and Procedural Errors (CAPEs) is a broad category containing all buggy rules, misconceptions, typical mistakes and common misunderstanding that the learner could be faced with when learning.

LEACTIVEMATH being an e-learning environment, it does not support all the capabilities that a more focused Intelligent Tutoring Systems (ITS) could do, notably for identifying and remediating to CAPEs. Therefore, with the notable exception of the Domain Reasonner required by the Dialogue Manager, the use of CAPEs in the current definition of the system is very limited. Therefore, there is not yet a clear picture about what the Learner Model could represent about influence of CAPEs on the learner's abilities, or about what other component such as the Tutorial Component could do should such information be made available in the Learner Model. This is clearly an investigation issue that will be addressed all along the project, rather than a requirement used to unequivocally design the system.

As for the subject domain, the Learner Model does not contains any predefined map or definition of the CAPEs [2]: they are assumed to be defined by the authors in the same way content is. A partial list of the CAPEs identified both from the content currently produced by WP6 and by the studies ran by WP4 can be found in Appendix A. They still need to be reorganised in a useful form (i.e. defining categories, generalising and specialising CAPEs when needed, etc.) and published in a format that could be used by LEACTIVEMATH.

There is two ways the Learner Model is currently using the existence of CAPEs associated with content:

1. By accumulating evidence about the occurrence of any CAPEs and about their eventual settlement by deliberate effort by the learner.
2. By using the presence or not of CAPEs in a particular situation to refine the update of beliefs. For example, we can imagine different treatment of updating beliefs if some CAPES are systematically identified with the learner's interaction with an exercise training this competency. This approach assumes that the list of CAPEs available for the Learner Model is organised according to a more focused taxonomy and that extra information used by the update mechanism are specified.

[2] At least beyond any initial set required by non-authoring-based components such as the domain reasonner.

Source of Evidence

- Individual steps of an interactive exercise reacting to an incorrect or unanticipated answer from the learner could contain reference to relevant CAPEs (as well as defining the feedback delivered to him/her). Such information will be sent to the Learner Model when (or if) that step is triggered by the learner's interaction.

Outstanding Issues

- The whole issue of handling CAPEs in LEACTIVEMATH is an ongoing question that will be explored through the project lifetime. This fact does not affect the implementation of xLM.

4.4.3 The Beliefs

From the previous sections it should be now clear that the Learner Model is a collection of beliefs about several inter-connected dimensions, whose ontologies are distinctly established. So let's now have a look at we we mean by "belief" and "accumulation of evidence".

4.4.3.1 Beliefs and Mathematical Competencies

Competencies and competency level is a perfect illustration of the significance of belief and accumulation of evidence over time. In the context of assessment and content authoring, the competency framework is an holistic approach which relies on a whole sequence of content. But from a learner modelling point of view, the emphasis is this time put on an individual piece of content of these sequence. How can we unite both approach, justifying the Leaner Model usage of the competency framework? By the accumulation of evidence.

When the learner's performance on a piece of content is signaled to the Learner Model, we have no way to know (apart from the beliefs already present in the model) how the learner fits in that sequence. The only thing we can extract from this situation, based on the description of the piece of content (e.g. which competencies it trains and which competency level the learner is assumed to gain or enforce as a consequence of successfully dealing with it), the context of the interaction (e.g. situational factors) and its outcomes (e.g. how well did the learner perform), is new evidence about the current competency level of the learner. This new evidence is combined with the existing one in order to refine the model of the learner. It is by this accumulation of evidence over time that the model will gain in accuracy and reliability.

However, a belief is not merely evidence but an *interpretation* of it. A learner model, as explained in section 2.1 is a *theory* that explains the evidence. Hence a belief on a dimension (e.g. a competency) is the piece of the theory that explains the evidence accumulated in relation to that particular dimension: what the Learner Model *believes* about the current level of the learner on that dimension (e.g. current competency level). The theoretical background behind the representation and management of beliefs in LM is the so called Dempster-Shafer Theory, which is a generalisation of the Bayesian theory of subjective probabilities (Shafer, 1976). This means that a belief on dimension d in LM contains a *confidence interval* $[\mathrm{Cr}(S), \mathrm{Pl}(S)]$ for at least every proposition or statement S of the form 'the learner is at level l on dimension d', where

(a) $\mathrm{Cr}(S)$ measures how certain LM is on the proposition S being true, while
(b) $\mathrm{Pl}(S)$ measures how much LM thinks the proposition does not contradicts the evidence (i.e. the proposition is *plausible*).

We call $\mathrm{Cr}(S)$ the *certainty* of LM on the proposition S being true and we call $\mathrm{Pl}(S)$ the *plausibility* for the proposition S to be true, both of them with respect to the accumulated evidence. Actually, $\mathrm{Pl}(S) = 1 - \mathrm{Cr}(\neg S)$, where $\neg S$ stands for the proposition 'the learner *is not* at level l on dimension d'. The difference $\mathrm{Ig}(S) = \mathrm{Pl}(S) - \mathrm{Cr}(S)$ measures the amount of *ignorance* of LM on the matter, i.e. ignorance on whether the learner *actually* has level l on dimension d.

Example If S stands for 'the learner has competency level III in mathematics' then $\mathrm{Cr}(S)$ stands for how much LM believes this is actually the case, given the amount and quality of the evidence that support the statement S, while $\mathrm{Pl}(S)$ stands for how much LM beliefs that what S states is a plausible fact, given the evidence that stands against it—and the close world assumption: S as to be either true or false, not both neither none. If, say, $\mathrm{Cr}(S) = 0.10$ and $\mathrm{Pl}(S) = 0.95$ this could be phrased as 'LM does not believes much that the learner is really at competency level III in mathematics, yet it reckons that could be rather possible (since it have not got much evidence against it)'.

Before getting into more details on DST formalism it is important to remember the semantics of levels, as described in section 2.4. Since levels are to measure progress they are ordered in a sequence $l_1, l_2, \ldots, l_n$ reflecting the fact that if 'a learner has attained level l_i' then 'the learner has (also) attained level l_j' for all $j < i$. If S_i denotes the former proposition and S_j denotes the latter, then we have that everytime S_i is true then S_j is true also. However, DST requires only one of these propositions to be true at any given time, hence any proposition S_i should be rather interpreted as meaning that 'the learner is at level l_i' or, to be more explicit, that 'the maximum level the learner has attained is l_i'. In this way, if S_i is true then S_j is false for any $j \neq i$.

A simple proposition of the form 'the learner is at level l' as can be represented by a set of levels containing only l; i.e. the singleton $\{l\}$. Actually, the power of DST comes from working not only with simple propositions as shown before but with composite *disjuntive propositions* of the form 'the learner is at level l_i or l_j or ...' where the number of levels put together in the propositon goes from one to the total of levels considered. As for simple propositions, disjuntive propositions can be represented by the sets such as $\{l_i, l_j, \ldots\}$. Now, in the same way that Bayesian probabilistic reasoning generalises on bivalent logical reasoning by introducing degrees of truth, DST generalises Bayesian reasoning by introducing degrees of ignorance. The particular mechanism used to accomplish this is still a distribution of probabilities, but this time not over the possible values (levels in our case) of a variable but over sets of its possible values. This probability distribution is called a *mass function* or *basic probabilistic assignment* (BPA), and it is a function

$$\mathrm{m} : 2^L \to [0, 1]$$

where L is the set of all possible levels for a dimension, 2^L is the power set of L (i.e. the set of all possible sets of levels) and $[0, 1]$ is the closed interval of all real numbers in between zero and one, inclusive. A (normalised) mass function must also satisfy the equations

$$\mathrm{m}(\emptyset) = 0 \quad \text{and} \quad \sum_{S \subseteq L} \mathrm{m}(S) = 1,$$

which mean that some level must be the case (the probabilistic assignment to the empty set is zero) and all possible levels are cover (the overall sum of the assignments is one).

The relationship between the mass function and the certainty and plausibility functions is given by the equations

$$\mathrm{Cr}(S) = \sum_{A \subseteq S} \mathrm{m}(A) \quad \text{and} \quad \mathrm{Pl}(S) = \sum_{A \cap S \neq \emptyset} \mathrm{m}(A).$$

So we can now provide a more complete definition of a belief on dimension d in LM has the collection of confidence interval $[\mathrm{Cr}(S), \mathrm{Pl}(S)]$ for every subset S of the set L of all possible levels the learner could be at on that dimension—i.e. where every subset stands for a disjuntive proposition of the form 'the learner is at level l_i or l_j or... or l_m on dimension d'.

Example If LM knows nothing about a learner's mathematical competency then the best it can do is to believe the learner is at some level on it. This can be achieved by making the mass function equal to zero for all subsets of levels other than the set of all levels $L = \{\text{I}, \text{II}, \text{III}, \text{IV}\}$. That is (see below, for a description of levels of mathematical competency),

$$\text{m}(L) = 1$$

while

$$\text{m}(S) = 0 \text{ for any other set } S \neq L,$$

which leads to

$$[\text{Cr}(L), \text{Pl}(L)] = [1, 1]$$

while

$$[\text{Cr}(S), \text{Pl}(S)] = [0, 1] \text{ for any other set } S \neq L.$$

In words,

- LM is certain that the learner is at some level, since $\text{Cr}(L) = 1$,
- has not idea of which level is because, in particular, $\text{Cr}(\{l\}) = 0$ for all $l \in L$,
- and hence believes all of them are equally and "totally" plausible, since $\text{Pl}(\{l\}) = 1$ for all $l \in L$).

Now let assume the learner has completed very successfully a medium hard exercise that has to do with mathematical competency at level II. This new evidence must be encoded as a basic probabilistic assigment to be merged with the current belief LM has—see section 4.4.4 for details of the updating mechanism. A possible definition for the mass function for the evidence is

$$\text{m}(\{\text{II}, \text{III}, \text{IV}\}) = 0.75$$
$$\text{m}(\{\text{I}, \text{II}, \text{III}, \text{IV}\}) = 0.25$$
$$\text{m}(S) = 0 \text{ for any other set } S$$

Since the mass function for complete ignorance is the neutral element for the updating mechanism, the mass function from the evidence will come the new mass function for the belief, given a set of confidency intervals with the following characteristics:

- $\text{Cr}(\{\text{II}, \text{III}, \text{IV}\}) = 0.75$,
- $\text{Cr}(\{\text{I}, \text{II}, \text{III}, \text{IV}\}) = 1$,
- $\text{Cr}(S) = 0$ for any other set S,
- $\text{Pl}(\{\text{I}) = 0.25$ and
- $\text{Pl}(S) = 1$ for all other set S.

From the results it can be said that the translation given of the evidence as a mass function produces as main result a strong decrease in the plausibility of the learner being currently at level I of mathematical competency, yet the evidence is not enough to be more accurate with respect to which of the remaining levels of competency is the highest level currently attained by the learner. It should be clear also that $[\text{Cr}(L), \text{Pl}(L)]$ is always $[1, 1]$, due to the fact that L is the set of all possible levels of mathematical competency, and hence any other set of levels S is a subsets of it.

4.4.3.2 Beliefs and Levels

Consequently, the definition of a belief about competency is as follow:

- **Belief on Competency:** A belief about *a competency* is a distribution over *competency levels* of the certainty and plausibility of the learner being at those levels (see figure 4.13). The definitions for the four competency levels to be used, as extracted from LeActiveMath Partners (2004d), are as follows:

- Level I: Computation at an elementary level. To achieve this competency level, students have to do apply arithmetic knowledge (factual knowledge, schematic applicable procedures). This level comprises knowledge learned by heart that is easy to recall and can be applied directly in a standard situation. The problem which is solved points to a certain standard form of mathematisation from the outset. Conceptual modelling is not required.
- Level II: Simple conceptual solutions. The simplest forms of conceptual modelling and solutions that include only a few steps are involved as well as the factual knowledge. Either the task is to select the correct solution from several alternatives or the student is provided with structural aids, graphical hints, etc. to develop her own solution.
- Level III: Challenging multi-step-solutions. Students at this competency level are able to perform more extensive operations, and are able to solve a problem with several intermediate steps. Students are also able to deal with open-ended modelling tasks that can be solved in various ways, but that require to find a solution of their own. High level modelling on inner-mathematical connections can be asked for.
- Level IV: Complex processings (modelings, argumentations) Students, who solve exercises of this final competency level successfully are able to work on open-ended tasks, choose adequate models and construct models themselves where necessary. Conceptual modelling at this highest level often includes mathematical justification and proof as well as reflection on the modelling process itself.

The figure 4.13 illustrate the shape that a belief about a competency d could have. All the possible sets of competency levels are represented along the horizontal axis. The set **{I}** represents the LM confidence that the learner is at level I, whereas the set **{I,II,III}** represent the confidence that the learner is somewhere between Level I and level III. Confidence is expressed in term of the confidence interval described above, $\mathrm{Cr}(S)$ standing for certainty and $\mathrm{Ig}(S)$ for ignorance.

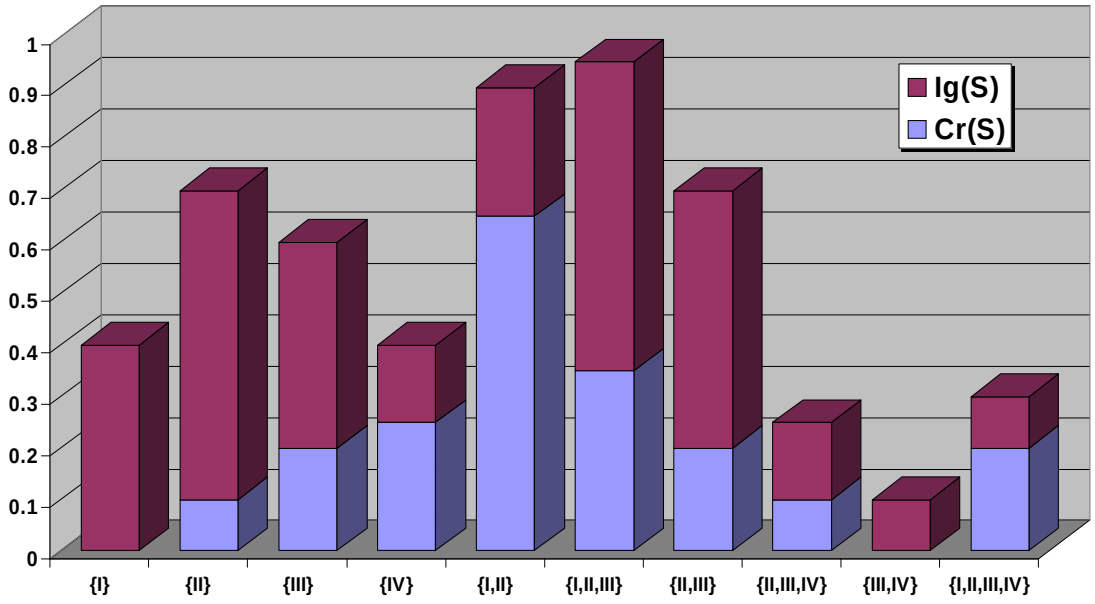


Figure 4.13: A belief as a distribution of certainty and ignorance on ranges of levels.

For consistency purpose, a similar approach is used for all beliefs about the principal dimensions of the model.

- **Belief on Motivation:** A belief about *a motivational factor* is a distribution over *levels of motivation* of the certainty and plausibility of the learner being at those levels. Similarly to the competency levels, the four-level scale represent the degree of motivation the learner is assumed to have, from low (Level I) to high (Level IV).
- **Belief on Affect:** A belief about *an affective factor* is a distribution over *levels of affect* of the certainty and plausibility of the learner being at those levels. Similarly to the competency levels, the four-level scale represent the degree of affective state the learner is assumed to present, from low (Level I) to high (Level IV).
- **Belief on metacognition:** A belief about *a metacognition ability* is a distribution over *levels of metacognition* of the certainty and plausibility of the learner being at those levels. Similarly to the competency levels, the four-level scale represent the degree of affective state the learner is assumed to present, from low (Level I) to high (Level IV).

The integration of CAPES in LEACTIVEMATH and the Extended Learner Model, as mentioned before, being at a too early stage, we are currently excluding them from a similar approach. In the current version of the LM, we are only considering if a particular Conceptual and Procedural Error has been diagnosed or not in relation to a topic.

4.4.3.3 Structuring the Beliefs

The figure 4.14 illustrates graphically the structure of beliefs in the Learner Model and their access from multiple viewpoints.

For example, the Learner Model can hold belief about the mathematical competency of the learner on the chain rule (the node labelled "MT on CR") or about his/her the level of satisfaction regarding the chain rule (node labelled *AonCR*) and regarding his/her level of mathematical thinking on the chain rule (node labelled *AonMTonCR*).

When a component request information about the learner such as the Tutorial Component for deciding which exercise to present next to the learner, specific query regarding these individual beliefs can be expressed (e.g. "What is the learner's level of Mathematical Thinking on the chain rule?").

But more general requests can also be made, such as "What is the overall level of Mathematical Thinking of the learner?" or "What is the overall level of satisfaction of the Learner?". In such situation, the Learner Model gathers all beliefs concerned by the request and combines them under the topic related to the question. Hence, topics like "Satisfaction", "Mathematical Thinking", "Control", etc. have also beliefs associated with them, even if they are not gathered by direct evidence but rather dynamically calculated by aggregating the relevant individual beliefs.

Every belief in the Learner Model can be uniquely identified by its "coordinates" in the four-dimensions space implied by the model. This coordinate can be represented by the following quadruplet,

$$< \texttt{MetacogID}, \texttt{Motiv\&AffectID}, \texttt{CompetencyID}, \texttt{TopicID} >$$

where:

- `MetacogID` identifies one of the ability in the metacogition map, including its top-level dimension;
- `Motiv&AffectID` identifies one of the factors of either the motivation map or the affect map;
- `CompetencyID` identifies one of the competencies in the competency map;

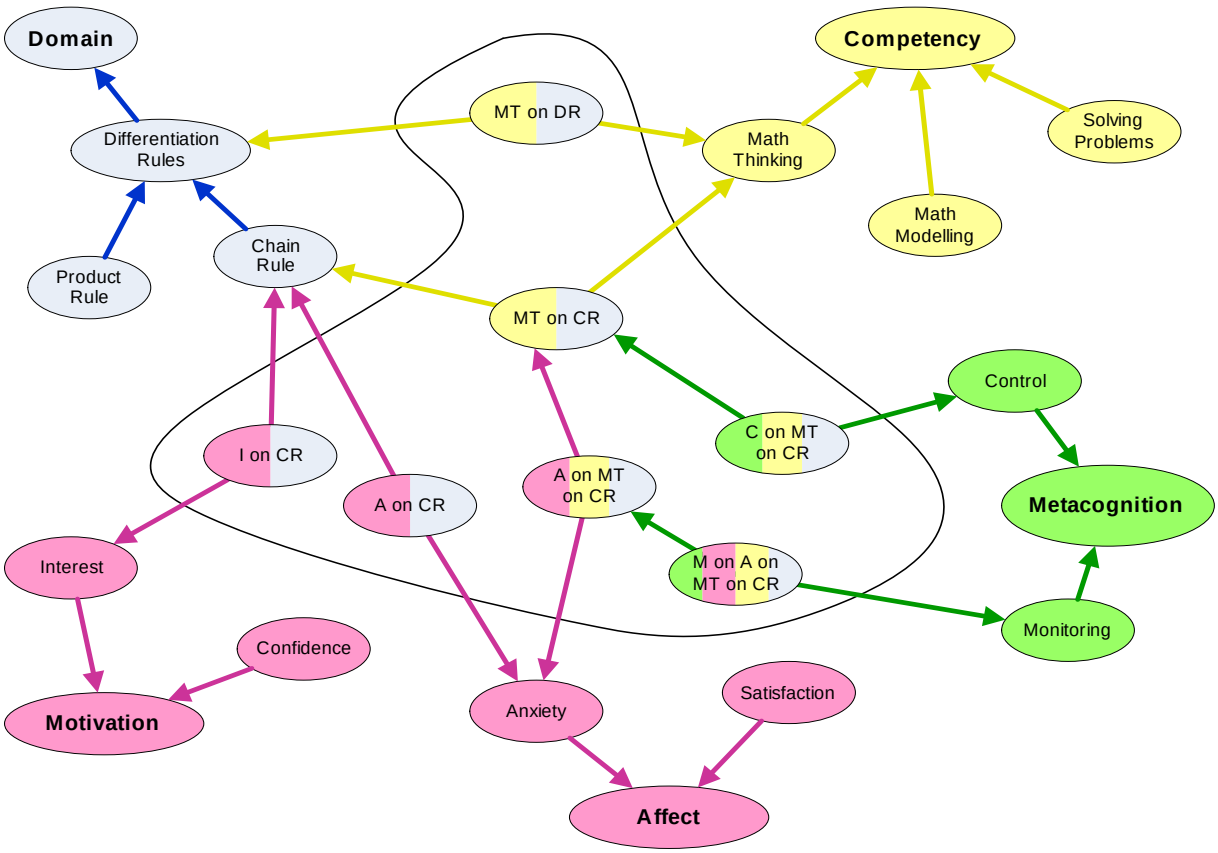


Figure 4.14: beliefs

- `TopicID` identifies one of the topics of the domain map.

Respecting the stack model (see figure 2.5), both individual and overall belief can be referenced by this mean:

- `<none, none, think, chain_rule>` for the belief about the level of mathematical thinking about the chain rule
- `<none, motivation, none, chain_rule>` for the belief about the level of motivation regarding the chain rule
- `<none, satisfaction, solve, chain_rule>` for the belief about the level satisfaction regarding the learner's competency in solving mathematical problems with the chain rule
- etc.

The combination of beliefs raises the question of the scope of the query. When a question like "What is the overall level of Satisfaction of the Learner?" is asked, an implicit parameter is which domain topics have to be taken into account. And there is no obvious answer to this question, as it merely depend on the context and on the motivation of the asker. For example, it could the whole domain (e.g. differential calculus) or the current session (e.g. all the domain topics explorer by the learner since) or the current book (e.g. all topics included in the book currently explorer by the learner), etc.

The learner Model provides a mechanism for dynamically defining such a context. Commonly used definitions of the scope are provided for immediate reference (they are the example given above: the session, the book or the whole domain). When querying the Learner Model about an overall belief, a component such as the Tutorial Component could specify the scope of application

of the combination, using the same coordinate system:

- `<none, none, competency, domain>` for the belief about the *overall* competency level of the learner [as seen as over the whole domain]
- `<none, none, think, domain>` for the belief about the *overall* level of mathematical thinking of the learner [as seen as over the whole domain]
- `<none, motivation, none, session>` for the belief about the *overall* level of motivation of the learner [as seen as over the current session]
- `<none, satisfaction, solve, book>` for the belief about the *overall* level of satisfaction regarding the learner's competency in solving mathematical problems [as seen as over all exercises in the current book]
- etc.

For an implementation of this mechanism, see the API for the Learner Model in Appendix B.

4.4.3.4 Accessing the Beliefs

When querying the Learner Model for information about the state or characteristics of a learner, several degrees of details are supported by the Learner Model to externalise its beliefs.

- **Summary Belief:** At the lowest level of detail, a single numerical value reflecting the dominant level of the distribution. This could be used by any component needing a quick overview of the belief. .

- **Belief:** The complete distribution across the levels (see figure 4.13) could be also retrieved. This will give higher flexibility for a component, for example by making a decision based on the lowest ignorance $\mathrm{Ig}(S)$ or the highest plausibility $\mathrm{Pl}(S)$.

- **Belief Cluster:** When querying about overall belief (e.g. "What is the learner's overall mathematical thinking?"), instead of delivering a combined belief, the Learner Model could instead return the full list of all beliefs clustered around the target of the query (e.g. all beliefs related to the competency Mathematical Thinking, whatever the domain topic they applies, see figure 4.14).

- **Evidence:** At the highest level of detail, the Learner Model could return a belief (or a cluster of belief) associated with all their evidence, i.e. all the events which interpretation and accumulation led to the actual state of the belief. As such a list of evidence could be extensive, a mechanism for organising them (e.g. in terms of strength or pertinence, in chronological order, etc.) will be also provided (see section 4.6.2 about the Open Learner Model).

4.4.3.5 Justifying the Beliefs

As the Learner Model update mechanism is based solely on the events stored in the Learner History, it is always possible to retrieve the list of all interactions that have accumulated evidence for reaching current state of the belief. Moreover, these events being chronological recorded, it is also possible to reconstruct the state of every belief at an earlier stage. Therefore, all beliefs in the Learner Model are explainable. This is a feature which is very useful for the Open Learner Model, as it allows an argumentation with the learner about about why the OLM believes that the learner's state is such or such (and possibly refutation of the evidence).

4.4.4 The Update Mechanism

Basically, the update mechanism could be roughly described as follow:

1. When an interaction between the learner and the system takes place, an event is stored in the Learner History [3].
2. The Learner Model takes this event as a source of direct evidence and build a numerical portrait of it, using a mass function as described in section 4.4.3.
3. The Learner Model proceeds to update all the beliefs related to this event, in light of the built evidence.
4. Based on the maps representing the various dimensions of the model, the Learner Model propagates this direct evidence in order to update all the beliefs connected to the source.
5. The Learner Model sends an event to the Learner History, notifying it (and hence all other interested components) that an update has been made in the Learner Model

Storing an event Phase 1 of the process underlines again the central role of the LH in the Extended Learner Model. The Learner Model does not have to watch every parts of the system in order to pick up the relevant events: as well as acting as a diary (and a digest) of the learner's activity, the LH does precisely this for all sub-components of the XLM. But such a "bottleneck" position of the LH also underlines the importance of the information stored with each event, to allow the LM to reconstruct retrospectively the exact context in which the event took place (see tables 4.1, 4.2, 4.3, 4.4 and 4.5 for a list of all events intercepted by the LH and their associated attributes).

Building an evidence from this event On phase 2 of the process, the LM reconstruct the context of the event in order to specify the nature of the evidence that has just been indicated. A typical example of the reconstruction of the context is when an exercise has been finished by the learner.

From the attributes of the event in the LH (see table 4.3), the LM is able to retrieve the identifier of the exercise, the status of the exercise (e.g. completion), the overall rating of the performance of the learner (evaluated by the Exercise Subsystem), the duration of the exercise. By consulting MBase with the exercise identifier, the LM is able to retrieve the metadata specifying the exercise (for example from the exercise `fib_productderiv` of figure 4.15), i.e. its difficulty (`easy`), typical learning time (`00:02:00`), competencies (`think` and `represent`) and competency level (`simple_conceptual`), etc. By using the relations between this exercise and the domain, the LM finally determines the domain topic concerned by this event, i.e.`product_rule` [4].

The LM can now clearly states that this learner's interaction with the system provides an evidence about his/her level of competency in "Mathematical thinking" and "Using mathematical representations", both in relation to the domain topic "Product rule". Using the coordinate notation introduce in section 4.4.3, they are:

$$B_1 : < \texttt{none}, \texttt{none}, \texttt{think}, \texttt{product\_rule} >$$
$$B_2 : < \texttt{none}, \texttt{none}, \texttt{represent}, \texttt{product\_rule} >$$

[3] It is worth mentioning again that not ALL LEACTIVEMATH events are stored in the LH but only the ones that have a significance for the xLM.

[4] For the reasons mentioned in section 4.4.2.1, the metadata of exercises does not yet contains references to the domain topics.

The parameters of the evidence (i.e. the competency level, the rating of the learner's performance, the difficulty relativised by the exercise learning context versus the learner's education level, the comparison between the typical learning time and the performing time, etc.) are used to quantify the numerical representation of the evidence.

Two basic sets of rules are used. The first one 4.1, in the case the learner has a low performance related to the exercise, states that the LM believes that the learner's level of competency L_i is lower or equal to the level L_e specified in the exercise. The second set 4.2, in the case of the learner's high performance in the exercise, states on the contrary that his/her level of competency is believed to be equal or higher to the exercise's target.

$$\begin{aligned} \mathrm{m}(\{L_1, L_2, \ldots, L_e\}) &= f \\ \mathrm{m}(\{L_1, L_2, \ldots, L_n\}) &= 1 - f \end{aligned} \tag{4.1}$$

$$\begin{aligned} \mathrm{m}(\{L_e, L_{e+1}, \ldots, L_n\}) &= f \\ \mathrm{m}(\{L_1, L_2, \ldots, L_n\}) &= 1 - f \end{aligned} \tag{4.2}$$

In both rules, the function f refers to the evaluation of the context on which the exercises took place, i.e. how the different attributes such as difficulty, abstractness, etc. are taking into account to provide a numerical quantification of the evidence. This is the key element of the whole update mechanism.

In our example, knowing that the learner did succeed pretty well with the exercise `fib_product-deriv` and the attributes of this interaction, one could expect the following evidence to be inferred:

$$\begin{aligned} \text{for } \mathrm{B}_1: \quad \mathrm{m}(\{\mathrm{II}, \mathrm{III}, \mathrm{IV}\}) &= 0.85 \\ \mathrm{m}(\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}) &= 0.15 \end{aligned}$$

$$\begin{aligned} \text{for } \mathrm{B}_2: \quad \mathrm{m}(\{\mathrm{II}, \mathrm{III}, \mathrm{IV}\}) &= 0.75 \\ \mathrm{m}(\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}) &= 0.25 \end{aligned}$$

Updating the beliefs On phase 3, once the evidence have been quantified, the Learner Model combines it with all the previously accumulated evidence in order to update the two beliefs (i.e. mathematical thinking on the product rule and using mathematical representations on the product rule). This is done by using the Dempster-Shafer Theory algorithm on the sets of evidence represented as mass functions (see section 4.4.3).

Propagating the evidence On phase 4, the Learner Model used the topological knowledge explicitly represented in the relevant maps to propagate the evidence on beliefs "similar" to the target one. In our previous example for example, the domain map (see figure 4.9) is used to propagate the evidence from `product_rule` to `deriv_rules`, using the association type `member_of` to specify how much of the evidence applies to the parent topics This association types allows the propagation mechanism to specify that the sibling topics (i.e.`chain_rule`) may not be concerned by the propagation (or in a very weak way).

Notifying the LH Finally, on phase 5, the Learner Model send an event to the LH, stating that beliefs about the learner have been updated. The attribute of this event could be used to pinpoint which beliefs have been updated so that any component could immediately react if the update is relevant to the current situation.

4.4.5 The Updater

The update mechanism of the Learner Model is a process that is not embedded in the model itself but is centralised in an external agent, unimaginatively called the *Updater*. The Updater is in charge of interpreting the result of a learner interaction (as stored in the Learner History), retrieving the relevant beliefs (or building them if they are not in the current state of the model) and updating them according to new sources of evidence.

This independence means that several versions of an update mechanism could be implemented concurrently in the system. On one hand, different approach of numerical methods (such as Bayesian nets, rules system, etc.) could be experimented, without an extensive overhauling of the Learner Model. On another hand, it means that several versions of an update mechanism could be used on the same scenario in order to compare the evolution of the Learner Model. In both situation, this flexibility will be used to improve, from empirical studies, the reliability and the accuracy of the Learner Model.

4.4.6 Outstanding Issues

- As mentioned above (section 4.4.3), DST requires disjoint levels as hypothesis. However, the semantics of competency levels implies inclusion (e.g. if a learner has attained level IV, he has also attained levels I, II and III). Our approach is not compromised by this situation, since the valuation of evidence takes it into account by assigning positive plausibility only to sets of consecutive levels (e.g. to $\{\text{II}, \text{III}, \text{IV}\}$ but not to $\{\text{II}, \text{IV}\}$). The output of DST have to conform with its input and any discrepancies will be used for monitoring problems in the modelling process.
- DST assumes that all possible sources of evidence have to be combined simultaneously in order to produce a new belief. In the LEACTIVEMATH context, evidence will come incrementally along time, reflecting the sequence of learner interactions with the system. We are exploring ways of building an iterative DST algorithm that would be able to combine the current beliefs hold by LM, standing for all past evidences, with new sources of evidence. Having such an algorithm would decrease the computational cost of updating the learner models. This issue will be solved and implemented by month 27.
- The propagation of beliefs across the dimensions of the Learner Model is an important part of the update mechanism and need to be improved. At the time of writing, it is unclear how Dempster-Shafer Theory could be used for that purpose (extensions of DST for propagation has been proposed but their computational complexity is an important question). The current version of the propagation plans to use simple ad-hoc rules for propagating the evidence to neighbour nodes in the maps, basically by increasing the probability of total ignorance (and introducing a threshold for controlling the depth of the process). For example, if the evidence for the belief B_1 `<none,none,think,product_rule>` is as follow:

$$\begin{aligned} B_1: \quad & m(\{\text{II}, \text{III}, \text{IV}\}) = 0.85 \\ & m(\{\text{I}, \text{II}, \text{III}, \text{IV}\}) = 0.15 \end{aligned}$$

  then propagating this evidence to the belief B_2 `<none,none,think,deriv_rules>`, both of them associated in the domain map (see figure 4.9), could result in the following definition of the mass function, to be accumulated with direct evidence for recomputing the belief B_2:

$$\begin{aligned} B_2: \quad & m(\{\text{II}, \text{III}, \text{IV}\}) = 0.85/2 = 0.425 \\ & m(\{\text{I}, \text{II}, \text{III}, \text{IV}\}) = 1 - 0.85/2 = 0.575 \end{aligned}$$

References and implementations of extension of DST for propagation of beliefs are currently reviewed and, if judged applicable, a new version of the updater will be made available by month 27 .

4.4.7 On the Complexity of the Learner Model

Without considering the CAPEs, the size of a Learner Model is $O(T * C * (M + A) * Mc)$ where T is the number of topic in the domain, C the number of competencies (and sub-competencies), M the number of motivation factors, A the number of affective factors and Mc the number of metacognitive abilities.

In practice, every belief in the Learner Model can be uniquely identified by its coordinate in the 4-dimension referential, i.e.<Mc,M&A,C,T>. This flattening out of the belief structure means that an index-based storage with direct access to individual belief can be guaranteed (e.g. both in memory with a hash-table and on a physical drive with a relational database).

If storage capacity is not (anymore) a major concern, time-consuming processes still remains a critical factor in the design of learner models. The most reliable model becomes useless if any update or query jeopardises the use of the system in a learning/teaching situation.

One aspect of the Learner Model that could be time-consuming: the retrieval of evidence regarding a particular belief (or a cluster of beliefs). This operation involves querying the Learner History for every event related to a particular domain topic. But, on the positive side, it is expected that the evidence retrieval mechanism will be used only by the OLM, when the learner ask for the reasons of a judgment on his/her characteristics. Being a GUI front-end on the client side of the system, a (reasonable) delay can be expected before the system react to the learner's request.

Nevertheless, a special care will be put on the implementation of (potentially) time-consuming operations. Several decisions have already been made in that direction. For example the XML-RPC communication, initially envisaged between the xLM and LEACTIVEMATH, have been reconsidered, due to the slowness of the protocol and on the high number of queries to the LM that have been highlighted in previous version of ACTIVEMATH. Furthers will certainly be considered all along the implementation cycles. For example, separating the Updater from the Learner Model opens the possibility for independent asynchronous threads, allowing LEACTIVEMATH components to query the LM while the Updater is still propagating evidence from the previous event (an assumption here is that successive belief updates will not introduce radical shifts in the LM). Lengthy retrieval operations could be shortened by implementing local caches (for examples, LH events identifiers could be stored in a cache associated with each belief in order to ease the evidence retrieval).

4.5 The Situation Model

The Situation Model consists of two main subcomponents:

1. **The Situation Diagnosis Agent (SDA)** which is responsible for diagnosing the current situation in terms of situational factors based on the evidence available from other components of LEACTIVEMATH.
2. **The Situation Modeller** which is responsible for inferring the autonomy and approval values based on the information determined by the SDA.

```
<exercise id="fib_productderiv" for="DiffDeriv/diff_f">
<metadata>
   <Creator role="aut">Christian Gross</Creator>
   <Title xml:lang="en">Derivation of a product</Title>
   <extradata>
     <depends-on>
       <ref xref="DiffDeriv/diff_f/fib_easyderiv"/>
     </depends-on>
     <learningcontext value="higher_education"/>
     <learningcontext value="university_first_cycle"/>
     <field value="all"/>
     <difficulty value="easy"/>
     <competency value="think"/>
     <competency value="represent"/>
     <competencylevel value="simple_conceptual"/>
     <typicallearningtime value="00:02:00"/>
     <representation value="verbal"/>
     <abstractness value="neutral"/>
   </extradata>
</metadata>
<CMP xml:lang="en">
   Please compute the <textref xref="DiffDeriv/diff_f">derivative</textref>
   of the <textref xref="functions/function">function</textref>
   $ap(f,x)=(2/3)*x^2*(7*x^3+4)$.
</CMP>
</exercise>
```

```
<exercise id="K17_TIMSS" for="deriv_maps/thm_diff_poly">
<metadata>
   <Creator role="aut">Christian Gross</Creator>
   <Title xml:lang="en">Find the equation of the function</Title>
   <extradata>
     <depends-on>
       <ref xref="deriv_rules/diffrule_sum"/>
     </depends-on>
     <learningcontext value="secondary_education"/>
     <learningcontext value="university_first_cycle"/>
     <field value="all"/>
     <difficulty value="medium"/>
     <competency value="think"/><competency value="solve"/>
     <competencylevel value="multi_step"/>
     <typicallearningtime value="00:04:30"/>
     <representation value="verbal"/>
     <representation value="symbolic"/>
     <abstractness value="neutral"/>
   </extradata>
</metadata>
<CMP xml:lang="en">
   The <textref xref="functions/graph">graph</textref> of the
   <textref xref="functions/function">function</textref> $f$
   passes the point $list(1,2)$.
   The <textref xref="DiffDeriv/slope">slope of the tangent</textref>
   in any point $list(x,y)$ of the<textref xref="functions/graph">graph</textref>
   is $ap(diff(f),x)=6*x-12$.
   Compute $ap(f,x)$.
</CMP>
</exercise>
```

Figure 4.15: An extract of two exercises authored with OMDOC.

The input to the Situation Model is a collection of evidence corresponding either to directly observable features of the interaction (e.g. overall difficulty of the current task taken from the exercise metadata) or to inferred characteristics of the learner such as the current state of their knowledge or the level of their achievement. Different sources of evidence are relevant to the particular situational factors which are the input variables the situation modeller.

In the current version of the Situation Model, there are eight situational factors. In the context of LEACTIVEMATH, the relevance of all of these factors to situation modelling was established based on the results of stage-2 of the study described in section 2.3. The situational factors along with their informal definitions are provided in table 4.7. The definitions were determined based on the interviews and *post-hoc* cognitive walkthroughs with the tutors during which they were asked to specify the meanings of the different factors explicitly.

Factor Name:	**Informal Definition**
Correctness of student's answer:	The degree of correctness for the just completed exercise.
Student's confidence:	The level of student's positive self-belief in their ability to tackle and to solve a given exercise correctly.
Student's aptitude:	An estimate of the student's ability to solve a given exercise correctly.
Student's interest:	The level of student's positive attitude towards the just completed exercise.
Difficulty of material:	Difficulty of an exercise obtained from the metadata associated with the exercise.
Importance of material:	Importance of an exercise to student's overall understanding of the material.
Student's effort:	An estimate of the amount of work done by the student on the just completed exercise.
Student's knowledge:	An estimate of the student's having mathematical content pre-requisites for the current exercise.

Table 4.7: Situational factors used by the situational model and their informal definitions

The values of the factors correctness of the student's answer, difficulty of material, importance of material and student's knowledge can be obtained from either the metadata associated with an exercise or from the scenario type. Situational factors for which the values cannot be obtained in this way, student's confidence, student's interest, student's aptitude and student's effort, are diagnosed by the situation diagnosis agent.

The output of the situation model consists of two values corresponding to the dimensions of autonomy and approval. As was discussed in section 2.1 according to a number of socio-linguistic theories (e.g. Brown and Levinson (1987)) autonomy and approval are socio-psychological dimensions of face along which people can make their choices that are communicatively optimal. In the educational context they are used to recommend levels of guidance (autonomy) and approval.

4.5.1 The situation diagnosis agent

The situation diagnosis agent is a rule-based sub-component of the Situation Model in which the rules are used to infer the specific factor values for all four situational factors that need to be diagnosed by the agent and for which evidence is available at a point at which the system needs to provide feedback to the learner. Evidence is obtained during a learner's interaction with the system on a particular task.

There are seven main sources of evidence which tutors tend to use to infer the values of the four situational factors. In LEACTIVEMATH all of the sources are obtained through the Learner History. The seven sources of evidence include:

1. **Hesitation level** which is established based on two variables:
   - the elapsed time between the submission of tutor question or instruction and commencement of student response.
   - the expected time for the commencement of a student response which corresponds to the average response time established for that student.
2. **Linguistic cues** of which the specific instances are:
   - use of interrogative forms in student answers (e.g. "?", "...").
   - use of hedges, e.g. "maybe".
3. **Achievement level** of which the estimation depends on three variables:
   - a number of recent student exercises (or steps in an exercise) under consideration (currently, based on our study analysis this number is set to 4).
   - degree of correctness (mark) for each exercise (or step in an exercise).
   - adjacency of the same marks with the number of exercises or steps considered.
4. **Difficulty of material** of which the variable is the rating of difficulty of a particular exercise obtained from the metadata associated with the current and the previous exercise.
5. **Spontaneous admissions** which is established based on presence or absence in student's response of terms that refer their current motivational states, e.g. statements of enjoyment, confusion, boredom, enthusiasm. The presence or absence of specific terms is determined against a look up table established based on the corpus collected in our studies.
6. **Granularity of solution steps** which is determined by comparing two variables:
   - the number of steps taken by the student to present the solution.
   - the number of steps represented in the correct path of the domain reasoner.
7. **Student's initiative** of which the specific (optional) instances include:
   - Student asks a clarification question.
   - Student volunteers to complete next possible step.

Whilst a number of sources of evidence can be obtained without natural language facilities, some sources are inherently dependent on natural language. The latter type includes the linguistic cues, spontaneous admissions and the first instance of student initiative: student asks a clarification question. In the case of spontaneous admissions, alternative means of collecting the evidence will be provided through the GUI in which self-reporting facilities will be made available to the learners. Most of the seven sources of evidence can be obtained in LEACTIVEMATH without any

Table 4.8: Importance of Evidence; 1 means *low*, 5 means *high*

FACTOR	SOURCE OF EVIDENCE	WEIGHT
Student Confidence:		
	Student hesitation	5
	Linguistic cues	5
	Spontaneous admissions	5
	Student initiative	4
	Granularity of solution steps	3
Student Interest:		
	Student initiative	5
	Spontaneous admissions	5
	Granularity of solution steps	4
	Achievement	3
Student Effort:		
	Student initiative	5
	Granularity of solution steps	5
	Difficulty of material	4
	Achievement	3
Student Aptitude:		
	Achievement	5
	Difficulty of material	5

reliance on natural language dialogue. These include hesitation level, achievement level, difficulty of material, granularity of solution steps and the second instance of student's initiative: student volunteers to complete next possible step.

The same source of evidence may be used to contribute to the diagnosis of more than one factor. However, as shown in figure 4.8, the diagnosis of the individual factors relies on a unique combination of the sources of evidence. Additionally, our data analysis suggests that not all sources of evidence are considered to be equally important by the tutors. Some sources of evidence generally contribute more to the diagnosis of certain factors than other sources. This means that each source of evidence may be assigned a weight reflecting the *relative importance* of a source of evidence that should be taken into account during a factor value diagnosis. Although we are still in the process of analysing the verbal protocols and the data gathered during the walkthroughs from stage-2, we can already see certain patterns in the way different sources of evidence contribute to the estimation of the specific factor values. The specific numbers may change as more data is gathered and analysed, but such changes will have no bearing on the overall implementation of the model. In table 4.8 we provide a summary of the four factors for which the values need to be determined by the SDA, the specific sources of evidence that contribute to the diagnosis of each of the factors, and the weights that represent tutors' comments about the relative importance of each type of evidence to the diagnosis of a value for a specific factor. The weights are given on a scale from 1 to 5 where 1 means *low* and 5 means *high*.

In addition to the weight which represents the relative importance of a source of evidence to the diagnosis of a factor value, our study also suggests that tutors take into account the *frequency* with which a particular type of evidence occurs in an interaction. For example, hesitation evidence is taken more seriously by the tutors if it is observed several times in the same interaction and recently in relation to the current task, rather than if it occurs only once or if its occurrence is not recent.

Ultimately, the *overall importance* of each source of evidence is calculated based on a *strength value* which is determined by the frequency of occurrence within a current session and on a *relative importance value* which indicates the relevance of a source of evidence to a factor. The frequecy of occurrence of particular evidence will be established dynamically and cumulatively during a learner interacting with LEACTIVEMATH based on the total number of occurrences of a particular type of evidence in a given interaction and on the recency of such occurrence.

For every source of evidence contributing to the diagnosis of a particular factor we developed a set of *diagnosis rules* which specify the *evidential pre-conditions* for particular factor values. As shown in table 4.8, the diagnosis of all four factors relies on multiple sources of evidence. For every source of evidence available at the time of diagnosis there is an *evidential instance* which in the relevant diagnosis rule is represented as its evidential pre-condition. For example, *difficult* is an evidential instance of the *Difficulty* evidence which also serves as one of the possible evidential pre-conditions in the rules which are used, for example, to diagnose *student effort* and *student aptitude*.

A factor value that is used eventually as the input to the situational modeller is a composite of all the possible values inferred for that factor based on the individual sources of evidence. In other words, the final factor value corresponds to the weighted mean of the results of applying the relevant evidential pre-conditions in a given set of diagnosis rules and it includes the strength and the relative importance associated with each source of evidence. The specific formula used for performing this calculation is given in 4.3.

$$fv(e) = \sum_{i=1}^{n} \frac{(fv(e_i) * W_i)}{n} \tag{4.3}$$

where $fv(e)$ is the composite (final) value of a factor, $fv(e_i$ is the component value derived from an individual piece of evidence e_i and W_i is the overall importance (or overall weight) of the evidence e_i.

The overall weight W_i of an individual piece of evidence e_i is simply the mean of the relative importance R_i and the strength S_i of the evidence:

$$W_i = \frac{R_i + S_i}{2} \tag{4.4}$$

Whilst in some cases a single evidential pre-condition is sufficient to derive a factor value, in other cases two or more pre-conditions are required for this purpose. For example, in the set of sources of evidence for inferring *student confidence*, first the sources are used individually to derive the respective *student confidence* values and then the results of these individual derivations are combined using the weighted mean function. On the other hand, for *student effort* the sources of evidence are combined using an algorithm in which the weighted mean function is applied straight away. There are two reasons for this difference. The first reason is the desire for the rules to be as perspicuous as possible. In the case of *student effort* there are only two sources of evidence used, while in the case of *student confidence* there are five sources. The more sources are available for making a factor value diagnosis the more difficult it is to combine them in a coherent and easy to modify way using single steps. The second reason is the sufficiency of evidence *vs.* necessity of evidence conditions. In the case of the *student confidence* factor, it may be the case that not all evidence is available at the time of diagnosis. For example, *linguistic cues* and *spontaneous admissions* are only available through natural language dialogue. However, these sources are not necessary to allow for an appropriate diagnosis of the factor value to be made and the remaining sources of evidence are sufficient. On the other hand, in the case of *student effort* both sources of evidence are necessary to make the appropriate diagnosis as neither student's

Let's assume:

- eh to be hesitation evidence with $W_{eh} = 1$
- eg to be granularity of solution step evidence with $W_{eg} = 0.77$
- ei to be student initiative evidence with $W_{ei} = 0.3$

Let's also assume that:

- $fv(e)$ is the final confidence value
- $fv(eh)$ is the confidence value derived based on hesitation evidence
- $fv(eg)$ is the confidence value derived based on granularity of solution step evidence
- $fv(ei)$ is the confidence value determined based on student intiative evidence

Finally, let's assume that:

- $fv(eh)$ is medium = 0.5
- $fv(eg)$ is low = 0.25
- $fv(ei)$ is very low = 0.01

We can derive the final confidence value $fv(e)$ by using equation (4.3):

$$fv(e) = \frac{(fv(eh) * W_{eh}) + (fv(eg) * W_{eg}) + (fv(ei) * W_{ei})}{3} \tag{4.5}$$

$$fv(e) = \frac{(0.5 * 1) + (0.25 * 0.77) + (0.01 * 0.3)}{3} \tag{4.6}$$

$$fv(e) = \frac{0.5 + 0.1925 + 0.003}{3} \tag{4.7}$$

$$fv(e) = 0.23 \tag{4.8}$$

Figure 4.16: Example of calculating a final value for **student confidence** given 3 evidence sources

level of achievement nor the difficulty of the material are individually sufficient to allow for the value of *student effort* to be determined.

The specific sets of diagnosis rules corresponding to the various sources of evidence are given in tables 4.9, 4.10, 4.11, and 4.12. Whilst the less complex rules are specified in full, for the more complex ones only a subset are provided for the purposes of conciseness and readability. Note that these rules are subject to further revision.

In the rules provided in the tables the values diagnosed for each factor are High and Low respectively. These values are used here for illustrative purposes and they represent High and Low regions rather than discrete values. In practice, the tutors in our studies diagnosed most of the situational factor values at least in terms of one of the five basic fuzzy linguistic descriptions shown in table 4.13. These descriptions will also constitute the possible situational factor values in our implementation. In table 4.13 we also provide the numerical values in terms of which we interpret the corresponding five fuzzy linguistic descriptions. An example of how these values would be used in calculating a final factor value which could be then pass on to the situation modeller is shown in figure 4.16.

Table 4.9: Diagnosis rules for inferring the value of **student confidence**, where **C** means confidence value.

Evidence Source	Rule Number	Evidential Precondition(s)	Result
1. Hesitation:	1.1	IF time elapsed is greater than time expected	THEN C = Low
	1.2	IF time elapsed is smaller than or equal to time expected	THEN C = High
2. Linguistic cues:	2.1	IF specific instance present in solution step	THEN C = Low ELSE C is Unknown
3. Student Initiative:	3.1	IF specific instance present in solution step	THEN C = High ELSE C is Unknown
4. Spontaneous Admissions:	4.1	IF admission of confusion present in solution step	THEN C = Low ELSE C is Unknown
5. Granularity of solution steps:	5.1	IF 1 step and Rule 1.2 applies	THEN C = High
	5.2	IF more than 1 step and Rule 1.1 applies	THEN C = Low
	5.3	IF more than 1 step and Rule 1.2 applies	THEN C = High

Table 4.10: Diagnosis rules for inferring the value of **student interest**, where **I** means interest value and p correct means partially correct.

Evidence Source	Rule Number	Evidential Precondition(s)	Result
1. Achievement:	1.1	IF 1 incorrect + 2 correct + 1 p correct	THEN I = High
	1.2	IF 1 incorrect + 1 correct + 2 p correct	THEN IF ADJACENT (incorrect and p correct) THEN I = Low ELSE I is Unknown
	...	...	...
	1.10	IF 4 incorrect	THEN I = Low
	1.11	IF 4 p correct	THEN I = Low
	1.12	IF 4 correct	THEN I = High
2. Student Initiative:	2.1	IF specific instance present in solution step	THEN I = High ELSE I is Unknown
3. Spontaneous Admissions:	3.1	IF admission of confusion or boredom present in solution step	THEN I = Low ELSE I is Unknown
	3.2	IF admission of enthisiasm or enjoyment present in solution step	THEN I = High ELSE I is Unknown
4. Granularity of solution steps:	4.1	IF 1 step	THEN I = High ELSE I is Unknown

Table 4.11: Diagnosis rules for inferring the value of **student effort**, where **E** means effort value and p correct means partially correct.

Evidence Source	Rule Number	Evidential Precondition(s)	Result
1. Achievement and Difficulty:			
	1.1	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = High	THEN E = High
	1.2	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = Medium	THEN E = High
	1.3	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = Low	THEN E = Low
	1.4	IF 1 incorrect + 1 correct + 2 p correct AND difficulty = High	THEN IF ADJACENT (incorrect and p correct) THEN E = Low ELSE E is Unknown
	...	...	...
	1.N	IF 4 correct AND difficulty = High	THEN E = High
2. Student Initiative:	2.1	IF specific instance present in solution step	THEN E = High ELSE E is Unknown
3. Granularity of solution steps:	3.1	IF 1 step	THEN E is Unknown ELSE E = High

Table 4.12: Diagnosis rules for inferring the value of **student aptitude**, where A means aptitude value and p correct means partially correct.

Evidence Source	Rule Number	Evidential Precondition(s)	Result
1. Achievement and Difficulty:			
	1.1	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = High	THEN A = High
	1.2	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = Medium	THEN A = High
	1.3	IF 1 incorrect + 2 correct + 1 p correct AND difficulty = Low	THEN A = Low
	1.4	IF 1 incorrect + 1 correct + 2 p correct AND difficulty = High	THEN IF ADJACENT (incorrect and p correct) THEN A = Low ELSE E is Unknown
	1.5	IF 1 incorrect + 1 correct + 2 p correct AND difficulty = not High	THEN A = Low
	...	...	...
	1.N	IF 4 correct AND difficulty = High	THEN A = High

Table 4.13: High/Low ranges of situational factor values, the fuzzy linguistic descriptions and the corresponding numerical values.

Range	Fuzzy Description	Numerical Value
High	very high	1
	high	0.75
	medium	0.5
Low	low	0.25
	very low	0.01

.

In the example shown in fig 4.16, based on the combined evidence available for student confidence, we calculated the final value of student confidence to be 0.23 which in fuzzy linguistic terms corresponds to upper-most region of *very low*. This value is now ready to be passed on to the situational modeller which will use it together with other factor values calculated in this manner to calculate the recommendations for autonomy and approval values.

4.5.2 The situational modeller

The situation modeller is responsible for calculating the autonomy and approval values based on specific combinations of situational factor values diagnosed by the SDA. In our approach to situation modelling we are relying on the pre-existing model proposed by Porayska-Pomsta (Porayska-Pomsta, 2003) and initially using its current implementation in order to have a first working situation model by month 18 of the project. Whilst we are concentrating all our efforts on delivering a fully functional SDA which is informed by the data gathered in the context of LEACTIVEMATH by month 18, the new implementation of the situational modeller will be delivered by month 24 of the project. The computational techniques and the main principles by which they were used by Porayska-Pomsta will be retained as we are confident that they are appropriate for the task. However, details of the design of the situational modeller will change based on the data obtained from our studies. Therefore, in this section we describe the techniques and the principles by which those techniques are used and the reasons for using them. We also outline the changes that will be needed to the original model to accommodate the requirements imposed by the context of LEACTIVEMATH and we discuss the methods that we plan to use to deliver the new situational modeller.

4.5.2.1 The basics of the situational modeller design

The input to the situational modeller is a *situation*. Just as is the case in the context of LEACTIVEMATH, in the original model a situation is composed of eight factors. In the original model the factors belong to three possible groups: the lesson oriented factors (LOFs) which are concerned with temporal aspects (amount of time available and amount of material left to cover) and the content taught (difficulty and importance of material), the motivation oriented factors (MOFs) concerned with student motivational characteristics referring to student confidence and interest, and performance oriented factors (POFs) which inform the model with respect to student aptitude and correctness of their actions. In the model this grouping of factors will be retained. However, temporal factors will be removed because they are not of relevance to the context of LEACTIVEMATH, i.e. no limit is imposed on the amount of time that a learner may, or has to, spend on a given material nor is there a set amount of material required to be covered by the learner in a single session. In the new version of the situational modeller, the *difficulty* and *importance* of material will be the only lesson oriented factors retained from the original version of the model. In addition to these two factors *student knowledge* will be added to the LOFs group to reflect the relevance of this factor to tutors' decisions in the studies. Also the MOFs group will be enlarged by the *student effort* factor which was established as relevant in the context of LEACTIVEMATH. The performance oriented factors will remain the same. The groupings of factors that will be used in the new model and its implementation are shown in table 4.14.

In the original situational modeller, each factor may contribute in different degrees to the calculation of autonomy and approval, depending on a given situation in which it occurs. The level of contribution of a factor is also referred to as its *salience*. Salience is an important facet of the situation model which relies on the assumption that although every situational factor has a potential impact on the possible tutorial actions that may be taken, not all factors will have an equal impact. This is being confirmed by our data analysis which indicates that whilst many different factors contribute actively to the choices made by the tutors, only a few of them are crucial in

Table 4.14: Groupings of factors in the new situational modeller.

Group	Factor
LOFs	Difficulty of material Importance of material Student knowledge
MOFs	Student confidence Student interest Student effort
POFs	Student aptitude Correctness of student answer

any given situation. In the model, the salience of each factor in a particular situation depends on other co-occuring factor values. Our data analysis (in particular Pearson's coefficient) revealed a number of different interdependencies between the situational factors which will be used to infer salience values for the factors in the model implemented in the context of LEACTIVEMATH.

Each factor either by itself or by combining with another factor evokes *goals*. Goals reflect the nature of the factors that give rise to them in the first place. Each goal has a salience associated with it which is passed on to it from its parent factor. The situational modeller distinguishes between two types of goals: the *local goals* and the *global goals*. Essentially, the global goals are the goals which map directly onto the the autonomy and approval dimensions which constitute the driving force behind the model. Such goals are expressed either as *Guide* or *Don't guide* (guidance oriented goals – GOGs) for the autonomy dimension, or as *Approve* or *Don't approve* (approval oriented goals – AOGs)for the approval dimension. Some goals give rise to global goals directly. For example, *difficulty of material: difficult* leads to the global goal *Guide*. Other factors give rise to local goals. For example, *student confidence: high* evokes the goal *boost confidence*. The distinction between the local and global goals will be applied in the context of LEACTIVEMATH.

In order to acquire meaning in terms of autonomy and approval, the local goals need to be either translated or recast. The translation is needed for those goals which cannot be paraphrased in a consistent way in terms of the global goals. For instance, it is not clear whether the local goal *boost interest*, which arises as the consequence of the situational factor *student interest* having the value: *low*, means that the student should be provided with guidance or not. In this case the appropriate interpretation depends on other factors such as *difficulty of material* and *student aptitude*, which are used together with their salience to qualify the extent to which the teacher needs to provide the student with guidance. On the other hand, recasting is needed for those local goals which can be paraphrased in a consistent way in terms of the global goals. For example, the goal *boost confidence* may be recast in terms of the global goal as *give approval*. Both ways of interpreting local goals in terms of autonomy and approval will be applied also in the context of LEACTIVEMATH.

Once translated and recast, each goal contributes an equal *vote* towards the final verdict of whether or not to guide and whether or not to give explicit approval to the student. Voting is a way of combining all the relevant gloabal goals together with their salience which is passed on either unchanged from the corresponding situational factor (e.g. the salience of the global goal *Guide* is passed on from the *difficulty: high*) from which it arises, or it reflects the combination of the salience values of two or more factors (e.g. the salience of the global goal *Guide* arising from the *student interest: low* and the modifing factors *difficulty: high* and *aptitude: low*). The result of voting consists of two GOGs: *Guide* and *Don't guide*, and two AOGs: *Approve* and *Don't approve*. Each of the goals has a number between 0 and 1 associated with it. The number – the result of combining all of the relevant salience values – expresses the degree to which each goal applies to a given situation. Because the number must fall in the [0,1] range, when a goal is picked its number also defines the degree to which its opposite applies to a given situation. A small number such as 0 means that a given goal is not recommended in a given situation. Such number also

informs one that the opposite of the given goal is very suitable for that situation. Similarly a high number such as 1 is a definite confirmation that a goal with which it is associated is a suitable one for a particular situation.

Before the final verdict is given by the situational modeller, the votes are modified by the relevant performance oriented factors which simply means that the votes are qualified by the appropriate factor-value and its salience as to the nature of the final goal and the level with which it applies to a given situation. For example, if the result of voting is the recommendation to provide the student with *high* amount of *guidance*, then if *student aptitude* is *low* and the *correctness of student's answer* is *incorrect*, then the recommendation for high degree of guidance is strengthened further and the result is a low value of autonomy given as the final recommendation from the situational modeller.

The interpretation of the final goals in terms of the two dimensions is straightforward. The two GOGs and the two AOGs which result from voting represent the extremes of the Autonomy and Approval dimensions respectively. *Don't guide* and *Approve*, constitute the positive extremes, while *Guide* and *Don't approve* are the negative extremes. In the current model the Autonomy and Approval dimensions are understood in a positive sense: to give Autonomy means *not to guide* which is expressed directly by the GOG: *Don't guide*, while Approval means simply *to approve* and is expressed directly by the AOG: *Approve*. Thus, in order to infer the final values of Autonomy and Approval, the recommended goals need to be exactly *Don't guide* and *Approve* to some degree. The values associated with them express the strength with which they are to be applied to a given situation. For example, a low value of *Don't guide* means that not guiding a student is not only not recommended but that guidance may be a more desirable course of action. Similarly low value of *Approve* means that there is no great urgency in giving the student explicit encouragement.

In the current model the goals are not reconciled in the strict sense of the word; for the Autonomy dimension the goal *Don't guide* will always be chosen as the indicator of the autonomy value. The number associated with it tells one about the force with which it ought to be applied. If the number is small, then effectively this means that quite a lot of guidance should be given to the student. If the number is large – little guidance ought to be given. Finally, if the number is around 0.5, then the goal should be applied with medium force. The same principle applies to the goals *Approve* and *Don't approve*, except that the goal chosen as indicator of the approval value is *Approve*. In that sense neither the Guidance nor the Approval goals respectively are independent from each other in that the salience value of *Don't guide* implies the salience value of *Guide* and, similarly, the salience value of *Approve* informs one about the value of *Don't approve*.

The principles by which the calculations of autonomy and approval are made in the original model will be applied in the context of LEACTIVEMATH.

4.5.2.2 The implemention of the situational modeller

The implementation of the original situational modeller follows its design very closely. This also will be the case once we make our final decisions as to the exact shape of the situational modeller in the context of LEACTIVEMATH. In particular we are as yet to define the way in which the individual factors will combine with one another and we need to determine salience for individual factors in a subset of possible situations. In this section we concentrate mainly on describing and defending the choices of the techniques that are used in the original implementation and that will be used also in the LEACTIVEMATH implementation.

The original implementation of the situational modeller uses a Bayesian Network (BN) technique. The BN is implemented in C++ programming language using the SMILE[5] library which

[5]Both SMILE and GENIE are decision-theoretic software and have been developed by the Decision Systems Laboratory, at the University of Pittsburgh (DSL, 1999).

is designed specifically for the purpose of implementing naive Bayesian Networks. SMILE also comes with a graphical user interface (GUI) called GENIE which allows one to create the basic BN structure without having to code it explicitly in C++, which in the case of the network representing the situational modeller would have been very cumbersome and difficult to accomplish and to inspect. Once the structure of the network is in place, SMILE functions can be used to create code which reads the network, writes prior and conditional probabilities into the individual nodes, sets evidence on the nodes, and runs the network.

The choice of Bayesian Networks to represent the situational component of the model is strongly motivated by its primary purpose which is to represent the process of making decisions regarding the best form of feedback, given a finite set of pre-defined constraints. Specifically, Bayesian Networks are used in the implementation of the situational modeller because they are an efficient method for making decisions based on some evidence from the real world. In that sense they provide an intuitively natural way of reproducing certain diagnostic and decision making capabilities of a human tutor. More importantly Bayesian Networks allow one to model complex dependencies between random variables in an efficient manner. It is precisely the purpose of the situational modeller to represent such complex dependencies between situational factors and to use them to infer the autonomy and approval values. In contrast with rule-based systems, for instance, in which representing such dependencies would not only be very cumbersome, but also very difficult to inspect and to change, Bayesian Networks provide a compact way of describing the entire distribution of the variables in terms of manageable and inspectable probability tables (e.g., Heckermann (1996); Guo and Hsu (2003)) – they facilitate efficient storage of data.

Furthermore, Bayesian Networks allow one to instantiate arbitrary subsets of variables (regardless of whether or not they are fully specified) and to calculate the conditional distributions on another subset in order to make a decision based on those distributions. In other words, just as they provide an efficient way of storing large amounts of complex data, Bayesian Networks have the capability of performing inference in an efficient manner which is ideally suited to modelling situations based on incomplete evidence. In the case of the original model as well as the model developed in the context of LEACTIVEMATH, the conditional distributions of all the relevant stages in the process of making a decision about the appropriate levels of autonomy and approval are based on the instantiations of the situational factors provided as the input to the model.

In the implementation of the situational modeller which we will use initially, there are altogether four levels of nodes which separate the input provided by the system from the final classification of the situation in terms of the autonomy and approval values. In terms of probabilities, the autonomy (in the network: *no-guidance*) and approval values can be also read as the degrees of belief that the autonomy and the approval ought to be granted to the student.

When the evidence is set on the top-level nodes, the values from those nodes are propagated down the network to all of the children nodes. The second level of nodes is used to represent all of the global goals such as GOGs and AOGs evoked by the individual situational factors. Whilst explicitly part of the model, the local goals such as LOGs and MOGs are modelled in the BN only implicitly. This is possible because in the model there is a one-to-one mapping between the situational factor-values and the local goals. For example, the factor-value *student confidence: low* always evokes the goal *boost confidence*. This allows for the local goals to be inferred implicitly from the factor-values, thus reducing the complexity of the network. It also reduces the complexity of inference necessary to propagate the evidence through the network.

The definition of every node at level two relates to at least one input state and two outcome states. The input states represent the factors which contribute to the definition of a node, the factor-values and their individual salience. The outcome states refer to the possible effects of the input states, while the probabilities express the likelihood that these effects will occur.

Essentially, the definitions of GOGs and the AOGs nodes represent the way in which the input states affect the probability of *guide* and *not guide* and *approve* and *not approve* being the case.

Table 4.15: Mapping from fuzzy terms onto numerical values

Fuzzy-term	Numerical Value
very low	0.12
low	0.23
relatively low	0.34
low-medium	0.45
medium	0.56
medium-high	0.67
relatively high	0.78
high	0.89
very high	1.00

The nodes are populated with conditional probabilities using the rules specifically developed for this purpose based on the data collected by Porayska-Pomsta in the context of a different tutoring domain. We are currently analysing the data from our LEACTIVEMATH studies in order to inform the rules for populating the network developed for LEACTIVEMATH. However, the general shape of the rules that will be developed for LEACTIVEMATH will be similar to the ones used for populating the original network.

Thus, the relevant rules will contain the information about those factor-values which affect the outcome states positively and those that affect them negatively. For example, in the case of *GOG difficulty*, the factor-values *student aptitude: low* and DIFFICULTY: *high* will affect the outcome state *guide* positively, while having a negative effect on the state *not guide*.

Given all the information expressed in the rules, the individual conditional probabilities are calculated on the basis of the salience associated with each input state. To determine the salience for the input states (or simply the salience of the individual factors in a given situation) Porayska-Pomsta relied on data collected in the context of another tutoring domain. Determining the salience of individual factors in specific situations is one of the outstanding issues in the current specification. We are analysing the data collected in the context of LEACTIVEMATH in order to address this issue. However, the mechanism for calculating conditional probabilities which rely on salience will be similar to the one used by Porayska-Pomsta. In figure 4.17 we are using one of her examples to show how conditional probabilities are actually calculated based on the rules for combining the situational factors *difficulty of material*, *student aptitude* and *student interest*. The equations in 4.17 calculate the conditional probability of *guidance of interest* based on the salience of the input states *difficulty: high* with a salience value *relatively high* associated with it, *interest: low* with salience value *very high* associated with it and *student aptitude: low* with a salience value *medium* attached to it. In 4.15 we show the numerical translations of the possible fuzzy linguistic description of the salience values used by Porayska-Pomsta to indicate where the numerical values in our example 4.17 come from.

Example 4.17 shows that the probability of having to give plenty of guidance to a given student who is rather bored and not coping well with the material in the circumstances where the material is difficult is very high, 0.89. The example also illustrates the general shape of the rules that we are in the process of determining based on the data gathered in our studies.

The third level in the network consists of two voting nodes: the *Autonomy Votes* and the *Approval Votes* nodes. When evidence is set and propagated through the network, the strength of recommendation is calculated for each outcome state in the *Autonomy Votes* and in the *Approval Votes* respectively. The strength of the recommendation is expressed in terms of the posterior probabilities (i.e., probabilities after running the network) of the occurrence of each of the outcome states in those nodes. The smaller the posterior probability for the positive state, the weaker

- **Assumption:** The input states are *difficulty:high with salience: relatively high*, *student interest: low with salience very high* and *student aptitude: low with salience medium*.
- **Fact:** The table 4.15 indicates that the fuzzy terms *relatively high* corresponds to the numerical value 0.78, term *very high* corresponds to 1.00, while *medium* corresponds to 0.56.
- **Rule:** Rule for calculating *guidance of interest* indicates that: IF *interest: low* and *difficulty: high* and *aptitude: low* THEN Guide with Pr(guidance of interest) being calculated based on the equation 4.9
- **Equation:** Given the assumptions above and the equation below, the calculation in 4.10 will result in the conditional probability of the state *guidance of interest* given the three input states: *interest:low with salience: very high*, *difficulty: difficult with salience relatively high* and *aptitude: low with salience medium*

$$Pr(guidance of interest) = \frac{\left(\frac{S_{interest}+S_{aptitude}+S_{difficulty}}{3}\right)+1}{2} \tag{4.9}$$

$$Pr(guidance of interest) = \frac{\left(\frac{1+0.56+0.78}{3}\right)+1}{2} = 0.89 \tag{4.10}$$

Figure 4.17: Calculating the conditional probability for the state *guidance of interest* given the input states *slow-medium* and *v-little-rel-high*.

the recommendation and *vice versa*, the larger the probability, the stronger the recommendation.

The input states to the *Autonomy Votes* nodes are the outcome states of all of the GOG nodes, while the input states to the *Approval Votes* are the outcome states of all of the AOGs.

In the case of both *Autonomy Votes* and *Approval Votes*, all of the contributing votes have an equal impact on the outcome states which is expressed by the weights attached to them. For example assuming the node *Autonomy Votes* to have five inputs, the weights for each input state which contributes positively to an outcome will be set to 0.2. In the case of both nodes, the conditional probabilities for each outcome state are calculated by summing all of the positive and negative contributions.

The voting nodes constitute the input to the nodes on the fourth and the final level of the network, the *Autonomy* and the *Approval* nodes being the final output nodes in the situational part of the system. The results of the voting performed on the third level in the network need to be modified further by the relevant situational factors. An example of a calculation for the outcome state *guidance*, given the input states *guide* and *correctness: partially correct* with *salience: high* is given in figure 4.18.

Ultimately, the output of the network consists in the posterior probabilities of the outcome states: *guidance* and *no guidance* of the *Autonomy* node, and *approval* and *no approval* states of the *Approval* node. These probabilities should be treated as representing the recommendations regarding the amount of autonomy and the amount of approval that ought to be given to a student in the situation specified as the input to the system. The output of the Situation Model that will be made visible to the interested components of LEACTIVEMATH will be the values indicated for *no guidance* and *approval* states, and will be presented as autonomy and approval respectively.

- **Assumption 1:** the input states are *guide* and *correctness: partially correct with salience:high*
- **Assumption 2:** the weight of the input state: *guide* is 1 indicating that this state is based on the combined votes by five parent nodes each having equal impact on the voting results which is expressed by the weight of 0.2.
- **Assumption 3:** the weight of the input state: *correctness: partially correct* is 0.2 indicating that this state has the same impact on the final Autonomy result as the individual votes of the GOG nodes, but a lesser impact than their combined weight.
- **Fact:** The table 4.15 indicates that the fuzzy terms *high* corresponds to the numerical value 0.89.
- **Rule:** The rule for calculating the probability of *autonomy* based on combining the results of *Autonomy Votes* and the modifying factor *correctness of student answer* indicates that IF *guide* and *correctness: partially correct* THEN *guide* with Pr(guidance) being calculated based on the equation in 4.11.
- **Equation:** Given the assumptions and the equation 4.11, the calculations in 4.12 will result in the conditional probabilities for the outcome state *guidance* which state is the opposite of *autonomy*, given two input states *guide* and *correctness: partially correct with salience: high*.

$$Pr(guidance) = \frac{\left(\frac{(W_{guide}+S_{corr}*W_{corr})}{W_{guide}+W_{corr}}\right)+1}{2} \tag{4.11}$$

where $W_{guide} = 1$ and $W_{corr} = 0.2$.

$$Pr(guidance) = \frac{\left(\frac{(1+0.89*0.2)}{1.2}\right)+1}{2} = 0.9908 \tag{4.12}$$

Figure 4.18: Calculating the conditional probability of the outcome state *guidance* given *guide* and *correctness: partially correct with salience: high*

4.5.3 The output of the Situation Model

The output from the Situation Model are the two values corresponding to the recommended level of Autonomy and Approval respectively. Given that in the context of linguistic politeness, the two dimensions are proposed as directly relevant to selecting natural language feedback in a strategic way we begin by illustrating how they may be used to inform the decisions of the Dialogue Manager.

4.5.3.1 The Dialogue Manager's use of Autonomy and Approval

The Dialogue Manager consists of, amongst others, a collection of communicative strategies each of which are coded for autonomy and approval values in the range between 0 and 1. Once the output from the Situation Model is provided, the Dialogue manager can use it to plan for the most appropriate feedback in the current circumstances. For example, the DM may plan to apply a high level remediation strategy to address a student's incorrect action. Given that there may be many different lower level types of remediation, autonomy and approval recommendations received from the Situation Model may be used in making the choice in a way which attempts to accommodate the cognitive and emotional needs of individual students.

Description of the output requested from SM to DM:

- **DM requests from SM:** void giveAutAppRec(+student, +tutorTurn, +AutAppRec)

4.5.3.2 Tutorial Component's Use of Autonomy and Approval

In line with the Dialogue Manager, the course generation provided by the Tutorial Component (TC) may also benefit from the recommendations of the Situation Model. There are at least three ways in which autonomy and approval values might be used by the TC:

1. In preparation of the book to be presented to a given student. Autonomy and approval values may affect the way in which the book is authored. For example for weaker students, for whom the recommended autonomy may be lower than for good students, a book may contain more and simpler exercises. The exercises may be presented in more steps and there may be more explanations provided. In particular, autonomy and approval values can be used within an exercise to inform the choices between methods.
2. In deciding on the appropriate way in which to present pre-requisites to the student. Stronger students may not require explicit listing of all pre-requisites needed for a given course. Such students may be given higher autonomy and may choose to find the pre-requisites through their own initiative.
3. In relation to suggestions, in deciding on the form to use when inviting the student to check out dictionary entries. The way in which invitations may be issued to the student to check dictionary entries may be affected by the recommended level of autonomy. The more guidance is recommended for a given student the more explicit the instruction should be for the student to check out the dictionary, e.g. "You should have a look at X" instead of a more autonomy giving invitation: "Remember that you can always check out X if you feel stuck".

Description of the output requested from SM by TC:

- **TC requests from SM:** void giveAutAppRec(+student, +tutorTurn, +AutAppRec)

4.5.3.3 Situation Model's Input to the Learner History Component

The Situation Model also provides the input to the Learner History. This input consists of a list of all the factor values determined for a particular situation (i.e. the input to the SM) as well as the evidence used to determine it.

4.5.4 Outstanding Issues

In this subsection we summarise the issues which still need to be addressed in relation to the Situation Model.

- Finish compiling diagnosis rules in the SDA and test the validity of the results of applying them.
- Determine salience for individual situation factors in a subset of possible situations relevant to the context of LEACTIVEMATH. Such subset of situations is already in place and was gathered through the studies described in the earlier sections of this document. This is necessary to provide a mechanism for calculating autonomy and approval values which conform to the context of the LEACTIVEMATH tutoring domain.
- Determine the ways in which different situational factors combine in the context of LEACTIVEMATH. This is needed for defining the basic shape of the Bayesian Network.
- Determine the exact rules that can be used to populate the Bayesian Network with conditional probabilities.

4.6 The Open Learner Model

When a learner is using the OLM, we can distinguish between navigation acts (i.e. selecting the topics to investigate, browsing the model for information, etc.) and interaction acts (i.e. argumentation between the OLM and the learners about performance/behaviour, etc.). The navigation acts will be subject to further elaboration and refinement during the next months as the GUI of the OLM is developed; this document focuses mostly on specifying the interaction acts.

First we briefly address the issue of the learner's "objective" performance and the OLM's "subjective" judgments of these performance and how they are to be introduced and handled in the system. Second, we introduce how the OLM will manage the justification for its judgments by handling "evidence" arising from the interaction between the learner and the system. Third, we provide the set of dialogue moves that the learner and the OLM can use to establish a diagnosis dialogue about beliefs. Finally, we give an overview of how the OLM will be integrated into the LEACTIVEMATH architecture.

4.6.1 Performance and Judgements

Within the OLM, interaction with the learner can be based on a behaviour/performance system (e.g. *"you scored 70%"*) and on value judgements (e.g. *"you did well"*). If the former does not represent a major problem (apart from deciding how to externalise this information), the latter is certainly more complex to handle, in particular when come the question of the nature of the beliefs hold by the Learner Model (see section 4.4.3).

Squarely externalising the numerical representation of a belief, i.e. the distribution of certainty and plausibility over the levels as in figure 4.13, may have some justification [6] but it assumes that

[6] and does open the ground for further interesting investigations.

the learner is familiar with the learner modelling process applied in LEACTIVEMATH. Moreover, it may be detrimental to one of the objectives of an Open Learner Model, i.e. supporting the learner in his/her self-reflection.

When a learner ask the OLM *"How competent do you think I am with the chain rule"*, one could assume that he/she doesn't want to be told *"I'm quite certain that you may be between Level I and III but there is also a good chance that you may be just a Level II but also a none-negligible chance of you being at Level III"* but rather *"I think that you are at Level III"* or even *"I think you are quite competent"*.

This mean that the OLM has to make a decision about which value judgement to present to the learner. And this decision-making is two-fold:

1. The OLM have to decide which is the dominant trait of the belief, i.e. what is the dominant set of levels or even what is the dominant level. One one hand, this decision-making could be relinquished to the LM itself, since it offers a numerical summary of the belief representing its dominant trait. But more sophisticated approaches could be envisaged. For example, the knowledge the LM holds about the learner (e.g. motivation, affective factor like satisfaction, etc.) could be used to help to decide how to externalise this knowledge.
2. The OLM has to value this dominant trait in order to inform the learner about its judgement. We have mapped out a set of such judgements but more work is needed to refine it, in particular by taking into account current work on how to talk about mathematical competencies and competency level with learners.

4.6.2 Evidence

The links between LH events and LM can be used as evidence for justifying a judgment made by the OLM[7], see figure 4.19. Each time some changes are made in the LM (such as when the learner accesses a given concept or when the learner's performance on an exercise is recorded), a relevant event is stored in the LH. Over time, all events related to a given concept accumulate and constitute evidence for the current state of the system's beliefs on that concept. When challenged by the learner, the OLM could present this evidence to the learner, i.e. the events from the LH. But as they come from numerous sources, with various degrees of strength, reliability and relevance, a mechanism for organising them has to be provided.

Let's take an example of a possible dialogue between the student and the OLM:

- **Learner (Inquire):** Show me what you think about my confidence about the Chain Rule
- **OLM (Inform):** I think you are quite confident about the chain rule
- **Learner (Challenge):** I do not understand how you reach your conclusion
- **OLM (Justify):** Here is the evidence I have relating to your confidence about the Chain Rule

In this exchange, the OLM (Inform) move gives its judgement about the learner's confidence on the "chain Rule" topic. This judgement is based on the belief the LM holds about the learner's characteristics, belief acquired by accumulating evidence over time (see section 4.4.4). When challenged about the ground on which the OLM is making its claim, there are many pieces of "evidence" (i.e. events stored in the LH) that could be used to justify such a decision:

[7]Or even by the learner, if we assume the existence of a mechanism that allows the learner to justify his/her own judgment by selecting "adequate" LH events.

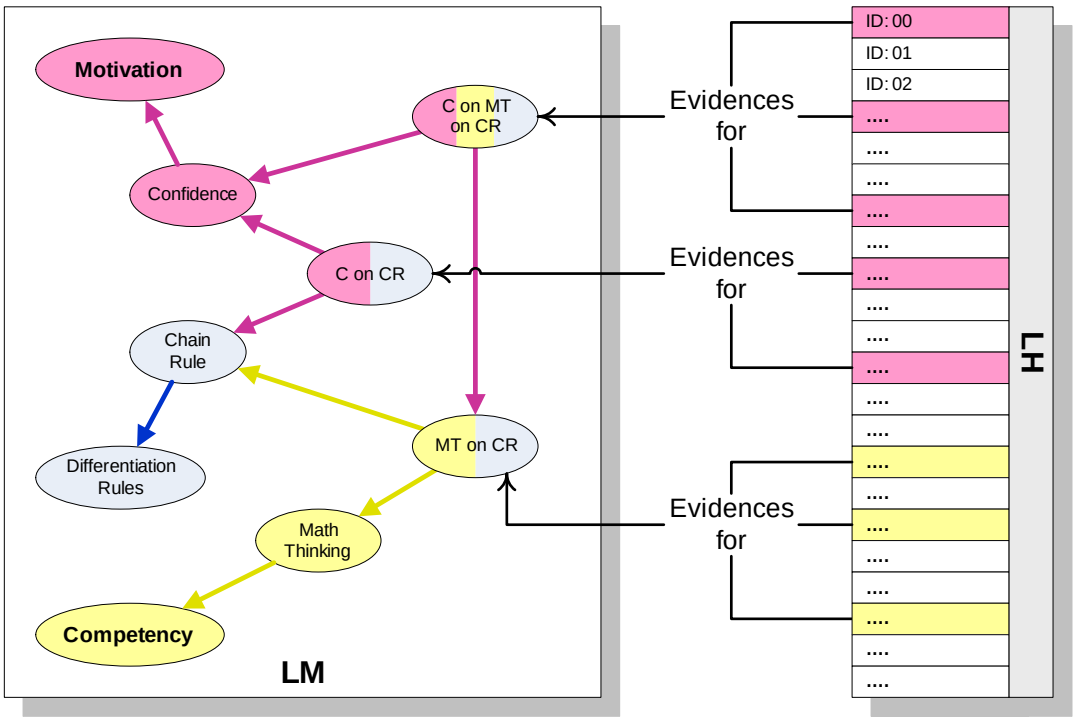


Figure 4.19: Getting evidence from the LH to support the OLM's judgments

- the performance of the learner on the last exercise (e.g. 7 answers correct for 10 questions) that led the LM to establish its current belief about the learner's confidence on the chain rule;
- all the previous exercises and related learner's performance, based on which this belief was reconsidered;
- A previous diagnosis dialogue between the learner and the OLM on this topic of Chain Rule, whose outcome led to further evidence about the learner's confidence;
- similar exercises (possibly on a different topic) to which different performances by the learner are compared and matched for value judgement;
- same exercises with other learners' performances with which peer-to-peer value judgment are performed;
- global phenomena such as interpreted behaviour of the learner across session(s);
- etc.

To reply to the challenge issued by the learner, the OLM has now to provide him/her with the relevant evidence. Several strategies could be used for this purpose.

The first one, and obviously the simplest, will be to simply present the learner with ALL the evidence gathered by the OLM and let him/her draw his/her own conclusion. The advantage of this approach is that it does not require any inference or argumentation engine; the drawback being obviously a lack of control on how to explore the arguments — both for the OLM and the learner.

A more elaborate approach is to define and use an argumentation framework, within which claims and evidence can be presented and argued one by one until an agreement is reach (or not) by both participants. Each round of the argumentation could reinforce, expand or complement previous evidence given by the OLM to the learner as long as he/she still does not accept them or does not understand them or need better explanation for them[8].

We propose to treat evidence and argumentation in terms of a multi-layers structure, within which subsequent layers can be expanded as needed during the dialogue.

There is obviously a chronological dependency between the items of evidence, conveying the temporal evolution of the beliefs. But this dependency may not be adequate to convince the learner of the (ir)relevance of the decision made by the OLM.

A further structuring mechanism is the introduction of a *strength* for the arguments and evidence, conveying for example the importance of an evidence in the judgment or the "confidence" of the OLM in making a value judgment from the learner's performances.

There are good reasons for defining such a framework. First, in a disagree (learner)/justify (OLM) loop, such a structured argumentation could help the system to manage situations such as running out of arguments or endless debates and reiteration of the same (weak) arguments — or, with the opposite purpose, it could help the reinforcement of ideas by addressing which arguments need to be reiterated again and again.

What is needed here is an argumentation framework similar to Toulmin's theory of argumentation (Toulmin, 1959), see figure 4.20.

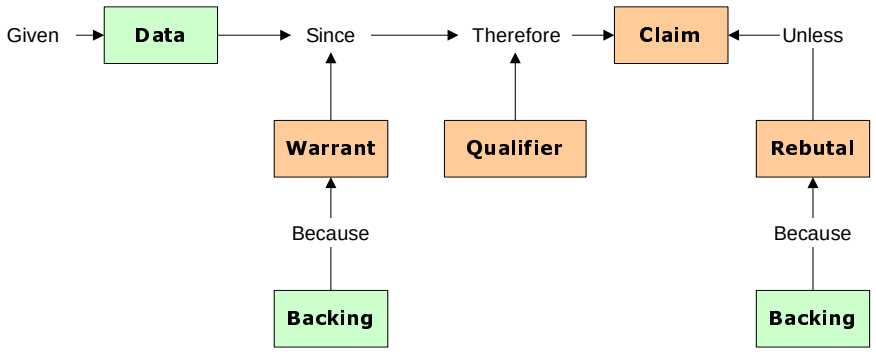


Figure 4.20: Stephen Toulmin's Layout of Argumentation

In brief, *"Data"* (or *"Grounds"*) are the evidence, facts, and information that are the reason for the claim in the first place — a reasoned beginning; *"Claim"* is the position on the issue, the purpose behind the argument, the conclusion that the arguer is advocating; *"Warrant"* is the component of the argument that establishes the logical connection between the data and the claim, i.e. the reasoning process used to arrive at the claim; *"Rebuttal"* is any exception to the claim presented by the arguer; *"Backing"* is any material that supports the warrant or the rebuttal in the argument; *"Qualifier"* represents the verbalisation of the relative strength of an argument, its soundness.

[8]For example, to the question "Why did he die?", one could first answer "Because his heart stopped" — which is indeed an evidence of the claim but its relevance may be not enough to convince the learner. A more in-depth argument, such as "Because the bullet perforated his heart", may do this job — or may not and, thus, requires more and more details).

4.6.3 Dialogue Moves

These dialogue moves[9] provide the initial specification of the language which the learner and the OLM will use to communicate and argue about beliefs and performance. There will be a need to iterate through different versions, expanding and refining them — especially by taking into account the organisation of these moves into a general dialogue framework (see section 4.6.4).

Discussion and negotiation will be based mostly on judgements made by the OLM, regarding knowledge, competence and skills, emotional and motivational states of the learner. We will refer to them as the *topic* of discussion. For each of the moves, we sketch the interface between the various relevant components of LEACTIVEMATH.

4.6.3.1 Learner's Moves

Here is a list of several dialogue moves the learner can use.

"Show me" "Show me" is the basic request a learner could use to enquire about the system's perception of his/her achievement. The focus for the enquiry could be knowledge, self-knowledge, affective state, etc. For example:

- show me what you think I know about differentiation
- show me what you think I know about my own self
- show me what you think I can do in the differentiation domain
- show me what you think I can do with my self knowledge
- show me how competent you think I am in mathematical terms

When the OLM receives such a move, a relevant event is added to the LH (indicating that an inquiry has been made by the learner regarding topic *X*, see figure 4.21).

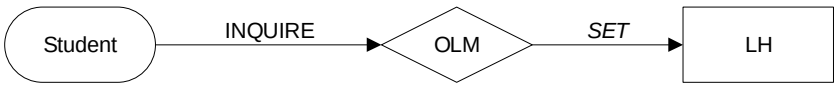


Figure 4.21: The Learner inquires about some Beliefs

"I Agree"/"I Disagree" Such affirmations can be used to seed the values associated with what is being agreed with. This is a response to something like a "Perhaps" move.

For example:

- I don't think I know the chain rule (but you seem to think I can)
- I don't think I am competent with the chain rule (ditto)
- I don't think I can solve problems with the chain rule (ditto)
- I don't think I can improve my motivation
- Yes, I do understand the chain rule

An "Agree"/"Disagree" move could be moderated by the learner to express a certain amount of disagreement, such as:

- I don't think I am *very good* at mathematical modelling

[9] Dialogue Moves refers to the standard logic terminology and do not imply natural language processing. In fact, they could even be deployed by graphical widgets.

- I don't think I am *very well* motivated
- Yes, I am *very* confident about my understanding of differentiation

When the OLM receives such a move, a relevant event is added to the LH (indicating that an agreement or a disagreement has been made by the learner regarding the judgement on a topic X). An indication of the learner's decision is also translated into the LM/SM (see figure 4.22).

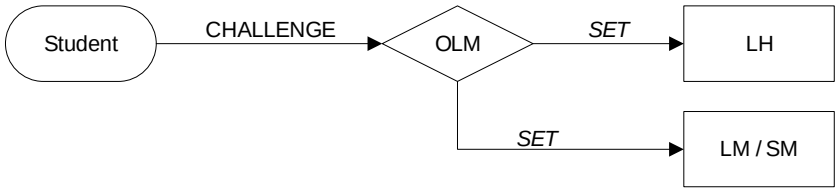


Figure 4.22: The Learner Agrees/Disagrees with some OLM Judgments

"I Confirm"/"I Disconfirm" These moves follow the "Can I confirm" move by the OLM, looking for a compromise position on a particular topic. For example:

- I confirm that I think I understand the chain rule very well
- I cannot accept that I understand the chain rule quite well

When the OLM receives such a move, a relevant event is added to the LH (indicating that the learner accepts or not the compromise on the topic X). Indication of the learner's decision is also translated into the LM/SM (see figure 4.23).

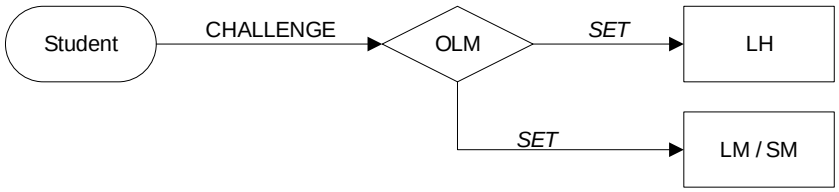


Figure 4.23: The Learner Confirms/Disconfirms some OLM Judgments

"I am Baffled" These moves allows the learner to express his/her confusion regarding what is going on in the discussion with the OLM. For example:

- I do not understand what we are talking about
- I do not understand how you reach your conclusion
- I do not know where you found this information
- I do not understand why we are talking about this
- I do not understand your role

This broad category will have to be broken down at some point, as it includes moves related to both general dialogue management ("I do not understand what we are talking about") and particular judgement such as refutation or challenge ("I do not understand how you reach your conclusion"). This last group is an important one for the framework "judgment-evidence" described in this document (see section 4.19).

When the OLM receives such a move, a relevant event is added to the LH (indicating the learner's confusion on a topic X). Indication of the learner's decision is also translated into the LM/SM (see figure 4.24).

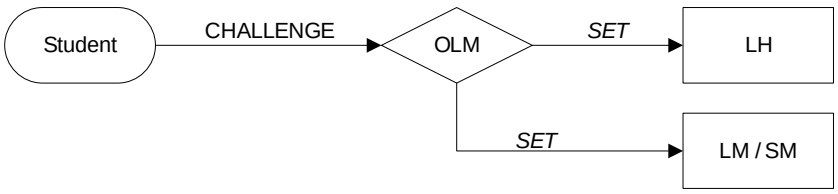


Figure 4.24: The Learner indicates Confusion about some Stage of the Discussion

"Let's Move on" These moves allows the learner to stop the current discussion on a topic and start a new line of inquiry. For example:

- I want to finish with this
- I would like to come back to this later
- I never want to revisit this

When the OLM receives such a move, a relevant event is added to the LH (indicating the learner's confusion on a topic *X*). Indication of the learner's decision is also translated into the LM/SM (see figure 4.25).

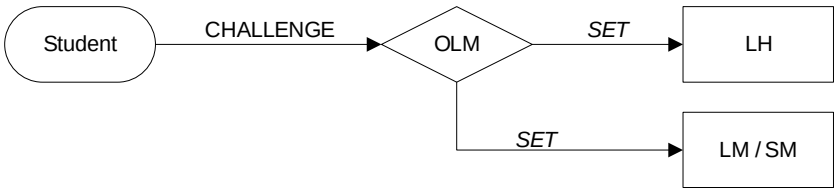


Figure 4.25: The Learner asks for a Change of the Current Discussion Topic

4.6.3.2 OLM's Moves

Here are some dialogue moves that can be made by the OLM.

"Perhaps" This move is the main dialogue mean of the OLM, presenting to the learner its judgement on a topic. For example:

- Perhaps you know the chain rule very well
- Perhaps you are very good at differentiating
- Perhaps you are quite confident about using the chain rule

When asked to give its judgement on a topic, the OLM retrieves the relevant information from the LM/SM and presents it to the learner. A relevant event is also added to the LH (indicating that a judgement has been made on topic *X*), see figure 4.26.

"Here Is" This move is the complement of the "Perhaps" move and is only used when the learner challenges the judgement on a topic or is confused about how the OLM has made this decision. For example:

- Here is the evidence relating to your knowledge of the chain rule
- Here is the evidence I have about what you know about yourself
- Here is the evidence I have about what you can differentiate

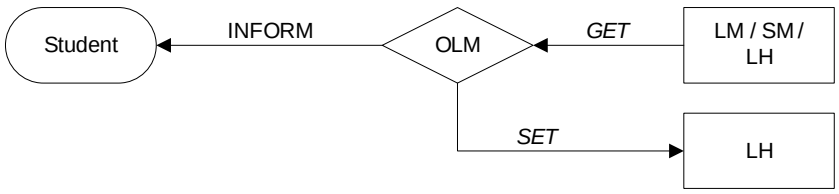


Figure 4.26: The OLM informs the Learner about the Requested Belief(s)

- Here is the evidence I have about your confidence

The OLM then presents to the learner the evidence used to support this judgement. Evidence will be mostly gathered from the events stored in the LH and made understandable by retrieving any relevant information from the LM/SM and the MBase to produce a human-readable account (see figure 4.27).

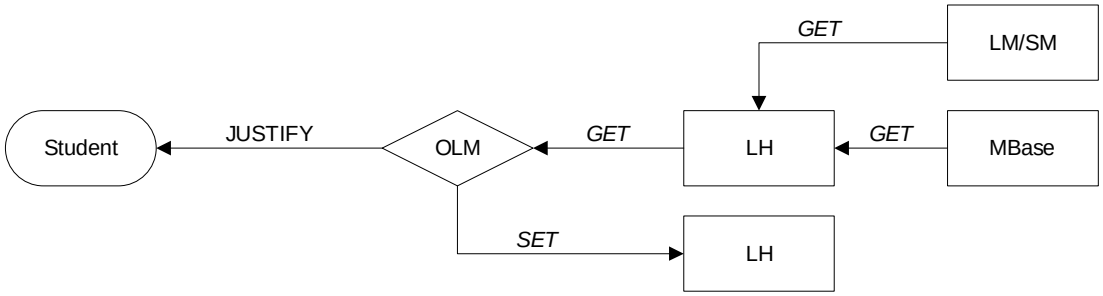


Figure 4.27: The OLM justifies its Decisions by Retrieving the Evidence used to Infer the Judgement

"Can I Confirm" This move is an attempt by the OLM to conclude a line of enquiry by suggesting to the learner a commonly agreed position and by seeking a confirmation of it. For example:

- So we agree that you understand the chain rule well
- So we agree you are quite competent at mathematical modelling
- So we agree you are very good at differentiation

A relevant event is added to the LH (indicating that a proposal for agreement on topic *X* have been presented to the learner), see figure 4.28.

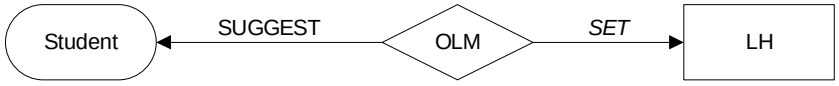


Figure 4.28: The OLM proposes to come to an Agreement on a Topic

"Unravelling Confusion" This move is an answer to the "I am baffled" of the learner. For example:

- We are talking about this since knowing the chain rule is a precursor to being able to differentiate functions of the kind we are being asked to do
- I believe you are quite good at mathematical modelling because you told me so yesterday and your performance is as good as yesterday

- I think you will need to read about the chain rule or ask your teacher but we could shift to discuss ...
- I am your electronic assistant, I am here to help you increase your understanding and learn differentiation more effectively

This move is suffering the same problem as the "I am baffled" move, i.e. too broad a range of issues to deal with. Both moves will have to be broken down further. In any case, a relevant event is added to the LH (indicating that a suggestion has been made to clarify the confusion on topic *X*), see figure 4.29.

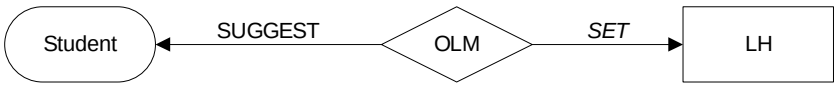


Figure 4.29: The OLM provides Support to the Learner to unravel the Current Confusion

"Finish with Topic" This move is similar to the "Lets move on" of the learner, but under OLM's initiative. We might see this as a compound of "Summary" and "Where we are going". For example:

- So we are agreed that you understand the chain rule quite well - we move on/will revisit some time soon
- So we agree to differ about your competence at mathematical modelling — we move on/will revisit some time soon
- So I'd like to discuss the chain rule now and return to this later
- So I'd like to discuss your confidence now

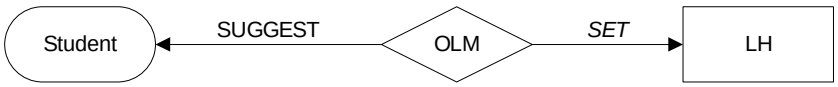


Figure 4.30: The OLM proposes to the Learner to end this Line of Discussion

"Can I Suggest" In some circumstances (such as eliciting a disagreement or address the desire/needs of the learner), the OLM may be in a position to suggest to the learner to perform some exercises or access some content. For example:

- Since we disagree about your competence at mathematical modelling, maybe you could perform this exercise to clarify the situation
- If you want to improve your competency in mathematical modelling, I suggest you have a go with this series of exercise.

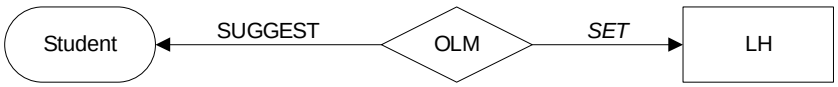


Figure 4.31: The OLM proposes to the Learner to perform an Exercise to clarify a Situation

4.6.4 Interactive Diagnosis

The interactive diagnosis, i.e. a dialogue between the OLM and the learner, is managed by organising the dialogue moves specified above in a Finite State Transition Network (FSTN) or any

equivalent mechanism. This will help us to determine if we have enough (or too many) dialogue moves and how to construct the interface needed for the using the moves. The following is an overview of some parts of such a transition network (figure 4.32).

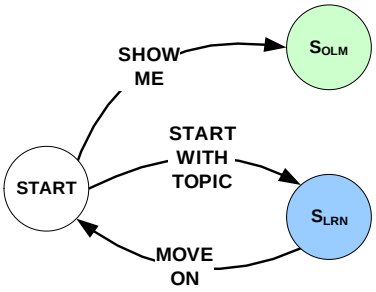


Figure 4.32: Starting a Topic with Mixed Initiative

The central part of the interactive diagnosis is based on the combination of "Perhaps" and "Agree/Disagree" moves (see figure 4.33), i.e. the presentation by the OLM of a judgement on the learner performance and its acceptance or refutation by him/her.

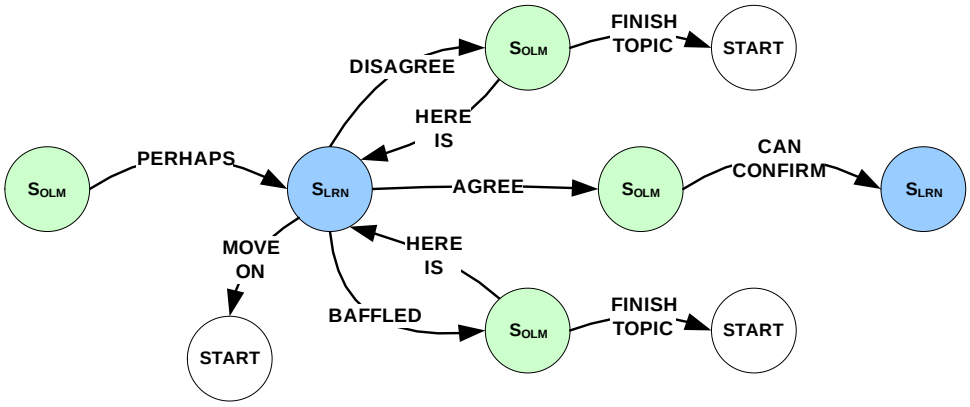


Figure 4.33: Interactive Diagnosis on the OLM Judgement

When the learner refutes the OLM's judgement, two possibilities can be considered. First, assuming the learner is the ultimate decision-maker in the process, the OLM could immediately close this line of discussion in an inconclusive manner and suggest to the learner to come back to it later (expecting further evidence or a change in the learner's opinion).

Another approach (as depicted in figure 4.33) could be for the OLM to submit more refined evidence of its judgment and hope for approval at some point. But it is possible that there will be an "endless argument" between the learner and the OLM. With a "depth-control" mechanism guaranteed by the framework suggested in section 4.6.2, we can ensure that the discussion will reach an inconclusive end after a couple of exchanges, i.e. when the OLM runs out of evidence.

When the learner agrees with the OLM's judgment, the OLM can ask him/her for a confirmation of the judgement (see figure 4.34), close this line of investigation in a conclusive way and move on to a different topic. This topic could potentially be reactivated in the future if further evidence comes up.

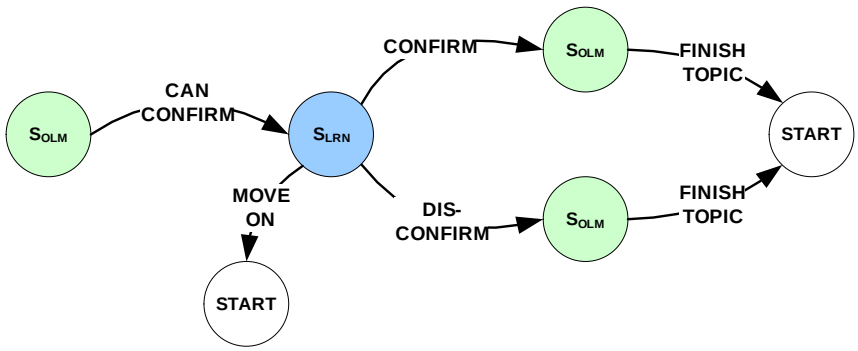


Figure 4.34: Closing a Topic with a Conclusive/Inconclusive Outcome

4.6.5 A Graphical User Interface for the OLM

Due to the complexity of the interactive diagnosis and the requirements for graphical externalisation of the Learner Model, browser-based mechanisms such as DHTML or embedded applet are ruled out for implementing the Graphical User Interface (GUI). Rather, the OLM will be designed as a Java stand-alone application, using extensive GUI libraries like SWING or SWT. The exact content and layout of the GUI will be specified in the next three months but an illustration of its potential can be seen in figure 4.35.

Its deployment and integration within the architecture of LEACTIVEMATH will be insure by protocols such as Java Network Launching Protocol (JNLP) and Java Web Start (JWS), insuring a secure communication with the various components (in particular the LM/SM and the LH) and a smooth and transparent deployment for the users.

Appropriate Java technology will be used to ensure that the OLM GUI meets the requirements on internationalisation of LEACTIVEMATH user interface.

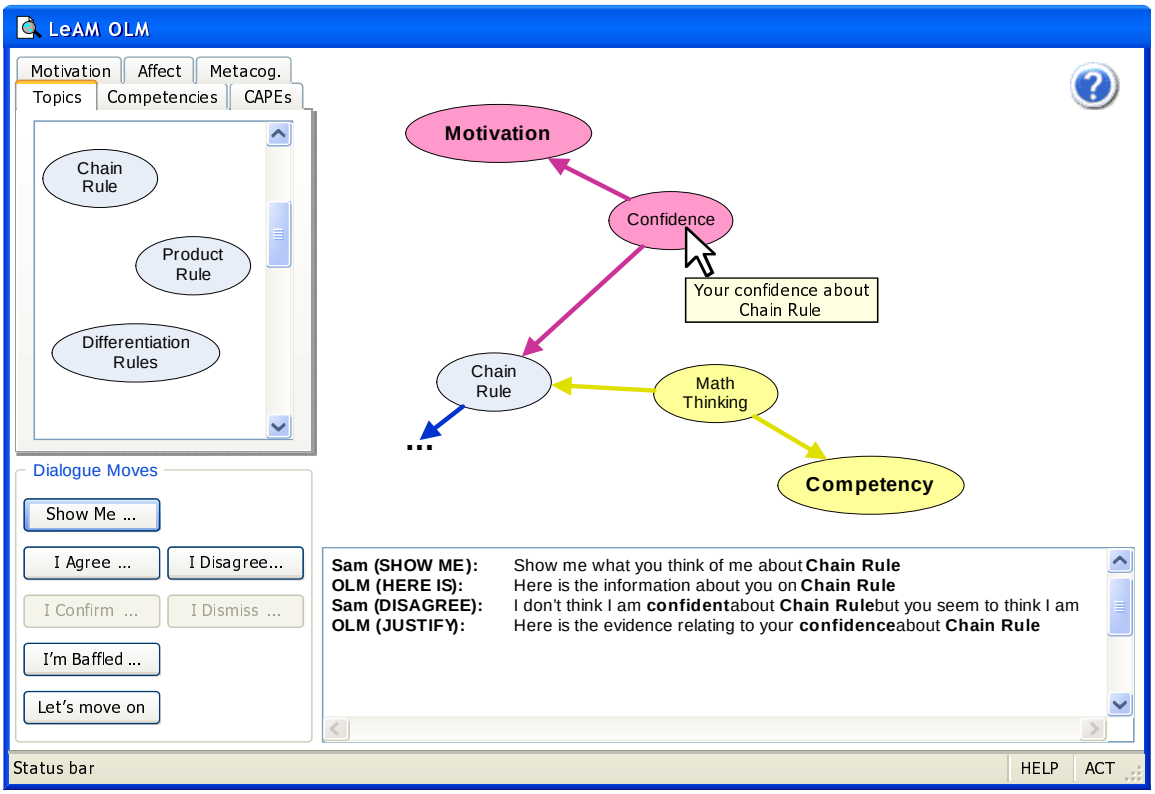


Figure 4.35: A possible GUI for the OLM

Chapter 5

Conformance

The specification of the Extended Learner Model, as required by the workplan for LEACTIVEMATH, conforms to the requirements generated by the project (Deliverable D5 (LeActiveMath Partners, 2004c)). For the benefit of the reader, all requirements referenced in this section can be found in Appendix C.

5.1 Satisfied Requirements

5.1.1 Extended Learner Model

The architecture of the xLM and its communication framework satisfy requirements 3.5, 4.13, 5.3, 5.4, 4.18[1], 4.19 and 6.2[2].

5.1.2 Learner History

The specification of the LH is in conformance with the following requirements:

- The LH will subscribe to several components that track the learner's actions, thus building up a history of learner's actions (Requirement 5.3).
- Every learner action that is interesting to the xLM is stored in the Learner History before it is relayed to the other xLM components ensuring that every inference made in the xLM is justifiable by the learner's actions that are recorded in the LH (Requirement 5.4).

5.1.3 Learner Model

The specification of the LM, described in section 4.4, meets requirements 4.17, 5.9, 7.3, 5.1 and 5.2.

5.1.4 Situation Model

The specification of the SM is in conformance with the following requirements:

- Requirement 5.1 and the related Requirement 4.19 have a contribution from the SM as it utilises information stored by the LM as well as information related to the situation in order to support pedagogical decisions made by the DM and the TC (see section 4.5.1).
- Requirement 5.11 and Requirement 6.2 will be met, and Requirement 5.9 will be partially met, by the provision of the values for autonomy and approval (see section 4.5.3). For Requirement 5.9, this will require that the DM and the TC have strategies/tactics that can make use of these values.

[1] xLM provides the needed access to instructionally relevant information about the learner.
[2] xLM provides the needed access to its components.

5.1.5 Open Learner Model

The specification of the OLM is in conformance with the following requirements:

- Requirements 5.4, 5.7 and 6.6 are met by the definition of interaction acts described in this document (see section 4.6.3).
- Requirements 5.5, 5.6 and 5.8 will be met by the provision of a GUI for the OLM and of navigation acts to the learner.
- Requirement 5.10 is partially met by the nature of the interactions between the OLM and the learner.

5.2 Assumed Requirements

Updating the LM, as described in section 4.4.4, assumes satisfaction — by other LEACTIVEMATH components — of the requirements 4.15[3], 2.3, 2.5, 3.6[4], 4.21, 4.22, 5.3, 6.5, 7.4, 7.5, 7.7 and 7.8.

5.3 Partially Unsatisfied Requirement

- The recording and interpretation of keyboard-stroke and mouse-movement mentioned in requirement 5.3 is no longer considered by WP4. Justification for this decision can found in section 2.3 .

[3] Actually, it assumes the existence of an ontology of learning objects.
[4] xLM requires the reverse functionality; that is, given a set of learning object identifiers, DK can derive the corresponding metadata set.

Bibliography

ADL. *Sharable Content Reference Model (SCORM) 2004: Overview*. Advanced Distributed Learning, 2 edition, 2004.

Vincent Aleven and Ken Koedinger. Limitations of student control: Do students know when they need help? In *ITS'00 - 5th International Conference on Intelligent Tutor Systems*, pages 292–303. Springer-Verlag, 2000.

John R. Anderson, Albert T. Corbett, Kenneth R. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*, 4(2):167–207, 1995.

Gerardo Ayala and Yoneo Yano. Learner models for supporting awareness and collaboration in a CSCL environment. In Claude Frasson and Gilles Gauthier, editors, *Intelligent Tutoring Systems: Third International Conference, ITS'96*, number 1086 in Lecture Notes in Computer Science, pages 158–167, Montreal, Canada, 1996. Springer Verlag.

Normaziah Aziz, Helen Pain, and Paul Brna. Modelling and mending student's misconceptions in translating algebra word problems using a belief revision system in TAPS. In Jim Greer, editor, *Proceedings of AI-ED'95 - 7th World Conference on Artificial Intelligence in Education*, pages 107–112, Charlottesville, VA, 1995. AACE.

Yvonne F. Barnard and Jacobijn A. C. Sandberg. Self-explanations, do we get them from our students? In Paul Brna, Ana Paiva, and John Self, editors, *European Conference on Artificial Intelligence in Education: Proceedings of EuroAIED*, pages 115–121, Lisbon, Portugal, October 1996. Colibri.

Joseph Beck, Mia Stern, and Beverly Park Woolf. Cooperative student models. In B. du Boulay and R. Mizoguchi, editors, *Artificial Intelligence in Education: Knowledge and Media in Learning Systems. Proceedings of the AI-ED 97 World Conference on Artificial Intelligence in Education*, number 39 in Frontiers in Artificial Intelligence and Applications, pages 127–134, Kobe, Japan, 1997. IOS Press.

P. Brna, S. Bull, H. Pain, and J. Self. Negotiated collaborative assessment through collaborative student modelling. In *Proceedings of the Workshop on Open Interactive and other overt Approaches to Learner Modelling at AIED'99*, pages 35–42. 1999. Le Mans, France.

P. Brown and S C. Levinson. *Politeness. Some universals in language usage*. Number 4 in Studies in Interactional Sociolinguistics. Cambridge University Press, 1987.

Susan Bull. See yourself write: A simple student model to make students think. In Anthony Jameson, Cécile Paris, and Carlo Tasso, editors, *User Modeling: Proceedings of the Sixth International Conference, UM97*, pages 315–326, Chia Laguna, Sardinia, Italy, June 1997. User Modeling Inc., Springer Wien New York.

Susan Bull, Paul Brna, and Helen Pain. Extending the scope of the student model. *User Modeling and User-Adapted Interaction*, 5(1):45–65, 1995.

Susan Bull and Helen Pain. 'Did I say what I think I said, and do you agree with me?': Inspecting and questioning the student model. In Jim Greer, editor, *World Conference on Artificial Intelligence and Education*, pages 501–508. Charlottesville VA: AACE, 1995.

Susan Bull and Simon Shurville. Cooperative writer modelling: Facilitating reader-based writing with scrawl. In Rafael Morales, Helen Pain, Susan Bull, and Judy Kay, editors, *Proceedings of the Workshop on Open, Interactive, and Other Overt Approaches to Learner Modelling*, pages 1–8, Le Mans, France, July 1999. AI-ED'99.

Hugh L. Burns and Charles G. Capps. Foundations of intellgent tutoring systems: An introduction. In Martha C. Polson and J. Jeffrey Richardson, editors, *Foundations of Intelligent Tutoring Systems*, chapter 1, pages 1–19. Lawrence Erlbaum Associates, 1988.

John M. Carroll and Mary Beth Rosson. Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Transactions on Information Systems*, 10(2):181–212, 1992.

J. Cassell and T. Bickmore. Negotiated collusion: Modelling social language and its relationship effects in intelligent agents. *User Modeling and Adaptive Interfaces*, 13(1-2):89–132, 2002.

Pierre Dillenbourg. The computer as a constructorium: Tools for observing one's own learning. In *Knowledge Negotiation*, chapter 8, pages 185–198. Academic Press, 1992.

Pierre Dillenbourg and John Self. PEOPLE POWER: A human-computer collaborative learning system. In Claude Frasson, gilles Gauthier, and Gordon McCalla, editors, *Intelligent Tutoring Systems: Second International Conference, ITS'92*, number 608 in Lecture Notes in Computer Science, pages 651–660, Montréal, Canada, 1992.

V. Dimitrova, J. Self, and P. Brna. Applying interactive open learner models to learning technical terminology. In *Proceedings of UM 2001 the 8th Conference on User Modelling*. Springer, Berlin, 2001.

Vania Dimitrova. STyLE-OLM: Interactive open learner modelling. *International Journal of Artificial Intelligence in Education*, 13:35–78, 2002.

Vania Dimitrova, John Self, and Paul Brna. The interactive maintenance of open learner models. In Susanne P. Lajoie and Martial Vivet, editors, *Artificial Intelligence in Education—Open Learning Environments: New Computational Technologies to Support Learning, Exploration, and Collaboration; proceedings of the 9th International Conference on Artificial Intelligence in Education*, number 50 in Frontiers in Artificial Intelligence and Applications, pages 405–412. IOS Press, 1999a.

Vania Dimitrova, John Self, and Paul Brna. STyLE-OLM—an interactive diagnosis tool in a terminology learning environment. In Rafael Morales, Helen Pain, Susan Bull, and Judy Kay, editors, *Proceedings of the Workshop on Open, Interactive, and Other Overt Approaches to Learner Modelling*, pages 25–34, Le Mans, France, July 1999b. AI-ED'99.

DSL. *SMILE: Structural Modeling, Inference, and Learning Engine. Apllication Programmer's Manual*. University of Pittsburgh, 1999.

Benedict du Boulay, Rosemary Luckin, and Teresa del Soldato. The plausibility problem: Human teaching tactics in the 'hands' of a machine. In Susanne P. Lajoie and Martial Vivet, editors, *Artificial Intelligence in Education—Open Learning Environments: New Computational Technologies to Support Learning, Exploration, and Collaboration; proceedings of the 9th International Conference on Artificial Intelligence in Education*, number 50 in Frontiers in Artificial Intelligence and Applications, pages 225–232. IOS Press, 1999.

K. Anders Ericsson and Herbert A. Simon. *Protocol Analysis: Verbal Reports as Data*. A Bradford book. MIT Press, 1984.

J. H. Flavell. Metacognitive aspects of problem solving. In L. B. Resnick, editor, *The nature of intelligence*, pages 231–236. Erlbaum, Hillsdale, NJ, 1976.

Haipeng Guo and William Hsu. A survey of algorithms for real-time Bayesian Network inference, 2003. URL `citeseer.nj.nec.com/552206.html`.

D. J. Hacker. Definitions and empirical foundations. In D. J. Hacker, J. Dunlosky, and A. C. Graesser, editors, *Metacognition in educational theory and practice*, pages 1–23. Erlbaum, Mahwah, NJ, 1998.

Michael J. Hannafin and Susan M. Land. The foundations and assumptions of technology-enhanced student-centred learning environments. *Instructional Science*, 25:167–202, 1997.

D. Heckermann. A tutorial on learning with Bayesian Networks. Technical report, Microsoft Research, 1996.

Penti Hietala and Timo Niemirepo. Collaboration with software agents: What if the learning companion agent makes errors? In B. du Boulay and R. Mizoguchi, editors, *Artificial Intelligence in Education: Knowledge and Media in Learning Systems. Proceedings of the AI-ED 97 World Conference on Artificial Intelligence in Education*, number 39 in Frontiers in Artificial Intelligence and Applications, pages 159–166, Kobe, Japan, 1997. IOS Press.

IEEE. Draft standard for learning object metadata. Technical Report IEEE 1484.12.1-2002, IEEE, 2002.

W. L. Johnson and P. Rizzo. Politeness in tutoring dialogs: "run the factory, that's what i'd do". In J.C. Lester, R.M. Vicari, and F. Paraguacu, editors, *Proceedings of the International Conference on Intelligent Tutoring Systems*, pages 67–76. Springer, Berlin, 2004.

Bruce Joyce, Emily Calhoun, and David Hopkins. *Model of Learning—Tools for Teaching*. Open University Press, Buckingham, 1997.

Judy Kay. Lies, damned lies and stereotypes: Pragmatic approximations of users. In *Fourth International Conference on User Modeling: Proceedings of the Conference*, pages 175–184, Hyannis, MA, August 1994a. User Modeling Inc., The MITRE Corporation.

Judy Kay. The um toolkit for reusable, long term user models. *User Modeling and User-Adapted Interaction*, 4(3):149–196, 1994b.

Judy Kay. Learner know thyself: Student models to give learners control and responsibility. In Z. Razak Z. Halim, T. Ottomann, editor, *Proceedings of the International Conference on Computers in Education, ICCE'97*, pages 17–24, Kuching, December 1997. Association for the Advancement of Computing in Education.

E. Klieme, H. Avenarius, W. Blum, P. Dobrich, H. Gruber, M. Prenzel, K. Reiss, K. Riquarts, J. Rost, H. Tenorth, and H. J. Vollmer. The development of national educational standards: An expertise. Technical report, Bundesministerium für Bildung und Forschung / Federal Ministry of Education and Research (BMBF), 2004.

Michael Kohlhase. OMDoc: An open markup format for mathematical documents (version 1.2). Technical report, Carnegie Mellon University, 2005.

LeActiveMath Partners. Exercise language. LeActiveMath Deliverable D7, The LeActiveMath Consortium, 2004a.

LeActiveMath Partners. Open architecture. LeActiveMath Deliverable D8, The LeActiveMath Consortium, 2004b.

LeActiveMath Partners. Requirement analysis. LeActiveMath Deliverable D5, The LeActiveMath Consortium, June 2004c.

LeActiveMath Partners. Structure and metadata model. LeActiveMath Deliverable D6, The LeActiveMath Consortium, 2004d.

Antonija Mitrovic. Investigating students' self-assessment skills. In *UM'01 - 8th International Conference on User Modeling*, volume LNAI 2109, pages 247–250, Sonthofen (Germany), 2001. Springer-Verlag.

M. Niss. Mathematical competencies and the learning of mathematics: The danish kom project. Technical report, 2002.

D.A. Norman and S.W. Draper. *User Centered System Design*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

A. Ortony, G. Clore, and A. Collins. *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge, 1988.

Helen Pain, Susan Bull, and Paul Brna. A student model 'for its own sake'. In Paul Brna, Ana Paiva, and John Self, editors, *European Conference on Artificial Intelligence in Education: Proceedings of EuroAIED*, pages 191–198, Lisbon, Portugal, October 1996. Colibri.

Ana Paiva, John A. Self, and Roger Hartley. Externalising learner models. In Jim E. Greer, editor, *Artificial Intelligence in Education, 1995: Proceedings of the Seventh World Conference on Artificial Intelligence in Education*, Washington, DC, August 1995. AACE.

K. Porayska-Pomsta. *Influence of Situational Context on Language Production: Modelling Teachers' Corrective Responses*. PhD thesis, School of Informatics, The University of Edinburgh, 2003.

Rob Reiner. The princess bride. Act III Communications, Buttercup Films Ltd. and The Princess Bride Ltd., 1987. Film.

L.H. Reyes. Affective variables and mathematics education. *The Elementary School Journal*, 84: 558–581, 1984.

F. Richardson and R. Suinn. The mathematics anxiety rating scale; psychometric data. *Journal of Counseling Psychology*, 19(6):551–554, 1972.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1 edition, 1995.

M. Scaife, Y. Rogers, F. Aldrich, and M. Davies. Designing for or designing with? Informant design for interactive learning environments. In *CHI'97: Proceedings of Human Factors in Computing Systems*, pages 343–350. ACM, New York, 1997.

John Self. Bypassing the intractable problem of student modelling. In Claude Frasson and Gilles Gauthier, editors, *Intelligent tutoring systems: At the crossroad of artificial intelligence and education*. Ablex Publishing Corporation, Norwood, NJ, 1990.

John Self. The defining characteristics of intelligent tutoring systems research: ITSs care, precisely. *International Journal of Artificial Intelligence in Education*, 10:350–364, 1999.

John A. Self. Dormorbile: A vehicle for metacognition. AAI/AI-ED Technical Report 98, Computing Department, Lancaster University, Lancaster, UK, 1994a.

John A. Self. Formal approaches to student modelling. In Jim E. Greer and Gordon I. McCalla, editors, *Student Modelling: The Key to Individualized Knowledge-Based Instruction*, volume 125 of *NATO ASI Series F: Computer and Systems Sciences*, pages 295–352. Springer Verlag, 1994b. Proceedings of the NATO Advanced Research Workshop held in Ste. Adele, Quebec, Canada, May 4–8, 1991.

Glenn Shafer, editor. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

Raymund Sison and Masamichi Shimura. Student modeling and machine learning. *International Journal of Artificial Intelligence in Education*, 9:128–158, 1998.

S. Tobias. *Overcoming math anxiety*. W. W. Norton, New York, 1995.

S. Toulmin. *The Uses of Arguments*. Cambridge University Press, 1959.

Julita Vassileva, Jim Greer, and Gordon McCalla. Openness and disclosure in multi-agent learner models. In Rafael Morales, Helen Pain, Susan Bull, and Judy Kay, editors, *Proceedings of the Workshop on Open, Interactive, and Other Overt Approaches to Learner Modelling*, pages 43–49, Le Mans, France, July 1999. AI-ED'99.

M A. Walker, J E. Cahn, and S J. Whittaker. Improvising linguistic style: Social and affective bases for agent personality. In E. André, S. Sen, C. Frasson, and J.P. Müller, editors, *Proceedings of the 1st International Conference on Autonomous Agents*, pages 96–105. Montreal, Canada, 1997.

J. B. Watson. Psychology as the behaviorist views it. *Psychological Review*, 20:158–177, 1913.

Franz E. Weinert and Rainer H. Kluwe. *Metacognition, Motivation, and Understanding*. Lawrence Erlbaum Associates, 1987.

David A. Wiley, editor. *The Instructional Use of Learning Objects*. Association for Instructional Technology, 2001.

Juan-Diego Zapata-Rivera and Jim E. Greer. Inspecting and visualizing distributed bayesian student models. In Gilles Gauthier, Claude Frasson, and Kurt VanLehn, editors, *Intelligent Tutoring Systems: Fifth International Conference, ITS'2000*, number 1839 in Lecture Notes in Computer Science, pages 544–553, Montréal, Canada, 2000. Springer Verlag.

Appendices

Appendix A contains a list of the Conceptual and Procedural Errors (CAPEs, see section 4.4.2.5) that have been collected from the content currently developed by WP6 (forthcoming Deliverables D18 and D19). In its actual form, it's a list organised by the domain concept associated with the exercises in which the CAPEs was identified.

Appendix B contains a description of the API for the Extended Learner Model. Its purposes, in line with the architectural decisions detailed in section 4.1, is to highlight to other LEACTIVEMATH components the methods and events available for communicating with the xLM as a whole.

Appendix C is an extract of Deliverable D5 (LeActiveMath Partners, 2004c) and contains, for the convenience of the reader, all the requirements related to the xLM, whose conformance have been addressed in section 5.

Appendix A

Conceptual and Procedural Errors identified in the LEACTIVEMATH content

Topic	Conceptual and Procedural Errors
Difference quotient and derivative	
Average slope of different curves	• Belief that the average slope of two curves measured between the same point cannot be the same
The total average slope of a curve	• Exchanged nominator and denominator of the difference quotient • Wrong sign • Forgot y_P in the difference quotient • Forgot y_Q in the difference quotient • Forgot x_P in the difference quotient • Forgot x_Q in the difference quotient • Difference quotient is not y_P/x_P • Difference quotient is not y_Q/x_Q • Fraction not simplified, in particular: just the definition, without evaluating it • Exchanged y_P and x_Q in the difference quotient • Exchanged y_P and x_Q and nominator/denominator in the difference quotient • Forgot denominator of difference quotient • An overall factor like a in $f(x) = a * g(x)$ is forgotten or only applied to either $f(x)$ or $f(x_0)$
Derivation: Comprehension and application	
Intersection between functions and their derivatives	• The derivative is not considered to be a function as well
Computing the derivatives of various functions	
The derivative of constant functions	• The derivative of a constant function $f(x) = c$ is $f'(x) = c * x^{-1}$ or $-c * x^{-1}$

Topic	Conceptual and Procedural Errors
The derivative of power functions $f(x) = a * x^n$	• Wrong sign, especially for negative a or n • Forgot the factor a: $f'(x) = (n-1) * x^n$ • Forgot to lower the exponent: $f'(x) = a * n * x^n$ • Forgot the exponent at all: $f'(x) = a * n * x$ • Raising the exponent instead of lowering it: $f'(x) = a * n * x^{n+1}$, especially for negative n • Forgot the factor n • Buggy rule: $f'(x) = a * (n-1) * x^n$ • For negative n: $f'(x) = a * (x+n) * x^{n-1} = a * (1 - n/x) * x^n$, being a buggy rule of the quotient rule, due to rewriting f as $f(x) = 1/x^{-n}$
The derivative of *sin*	• wrong sign: $sin'(x) = -cos(x)$ • buggy rule: $sin'(x) = sin^{-1}(x)$
The derivative of *cos*	• wrong sign: $cos'(x) = -sin(x)$ • buggy rule: $cos'(x) = cos^{-1}(x)$
The derivative of *tan*	• buggy rule: $tan'(x) = cot(x)$ • wrong sign: $tan'(x) = -1/cos^2(x)$ • buggy rule: $tan'(x) = cos^2(x)$ • buggy rule: $tan'(x) = 1/sin^2(x)$ • buggy rule: $tan'(x) = -1/sin^2(x)$
The derivative of *cot*	• buggy rule: $cot'(x) = tan(x)$ • buggy rule: $cot'(x) = -tan(x)$ • wrong sign: $cot'(x) = -1/sin^2(x)$ • buggy rule: $cot'(x) = sin^2(x)$ • buggy rule: $cot'(x) = 1/cos^2(x)$ • buggy rule: $cot'(x) = -1/cos^2(x)$
Rules of differentiation	
Sum rule	• forgot one of the terms • forgot to differentiate one of the terms • wrong sign in one of the terms (mostly also contained in the buggy rules for the single terms) • differentiation errors in the single terms

Topic	Conceptual and Procedural Errors
Product rule for $f(x) = u(x) * v(x)$	• forgot one of the terms, e.g. $f'(x) = u'(x) * v(x)$ • forgot to differentiate one of the terms, e.g. $f'(x) = u'(x) * v(x) + u(x) * v(x)$ • wrong sign in one of the terms (mostly also contained in the buggy rules for the single terms), e.g., $f'(x) = u'(x) * v(x) - u(x) * v'(x)$ • multiplying the single derivatives: $f'(x) = u'(x) * v'(x)$ • buggy rule for triple (or more) products: $f'(x) = u'(x) * v'(x) * w(x) + u'(x) * v(x) * w'(x) + u(x) * v'(x) * w'(x)$ • differentiation errors in the single terms, see there
Quotient rule for $f(x) = u(x)/v(x)$	• forgot one of the terms, e.g. $f'(x) = u'(x)/v(x)$ • forgot to differentiate one of the terms, e.g. $f'(x) = (u'(x) * v(x) - u(x) * v(x))/v(x)^2$ or $f'(x) = (u(x) * v(x) - u(x) * v'(x))/v(x)^2$ • wrong sign in one of the terms (mostly also contained in the buggy rules for the single terms), e.g. $f'(x) = (u'(x) * v(x) + u(x) * v'(x))/v(x)^2$ • dividing the single derivatives: $f'(x) = u'(x)/v'(x)$ • differentiation errors in the single terms, see there
Chain rule for $f(x) = g(h(x))$	• buggy rule: $f'(x) = g'(x) * h'(x)$ • forgot the inner derivative, $f'(x) = g'(h(x))$ • forgot the outer derivative, $f'(x) = h'(x)$ • mixing both derivatives, $f'(x) = g'(h'(x))$ • forgot to differentiate one of the terms, e.g. $f'(x) = g(h(x)) * h'(x)$ • combination of the first two buggy rules, e.g. $f'(x) = g(x) * h'(x)$ or $f'(x) = g'(x) * h(x)$ • mixing with the product rule: $f'(x) = g'(x) * h(x) + g(x) * h'(x)$ or $f'(x) = g'(h(x)) * h(x) + g(h(x)) * h'(x)$ • differentiation errors in the single terms, see there
curve scetching	
horizontal functions of a function	• Computation with $f(x) = 0$ instead of $f'(x) = 0$

Appendix B

Interface to the Extended Learner Model

This section describes the interface of the Extended Learner Model. As specified in section 4.2, access to functionalities of XLM subcomponents will be thought a single front-end. The description of events, data structures, input and output APIs conform to the syntax used in the Open Architecture document (Deliverable D8 LeActiveMath Partners (2004b)).

B.1 Implementation details

From a implementation point of view, the xLM manager introduced in Figure 4.2 is a singleton that serves two purposes :

(i) as a facade object for xLM API,
(ii) as a local event publisher and event forwarder for public events (both subscription and publisher).

As a facade object, xLM Manager creates instances of each xLM component and holds a reference to it, and it is configurable as a LEACTIVEMATH component. Since currently many information request about the learner are made through a central User object, this should increase the independency of xLM inside LEACTIVEMATH.

As a local publisher and forwarder, xLM Manager subscribes itself to all learner-related events at the central LEACTIVEMATH Event Manager, whereas all xLM components subscribe as listeners to xLM Manager (instead of subscribing to the LEACTIVEMATH Event Manager). Hence xLM Manager forwards all incoming events from the rest of LEACTIVEMATH to each component that has subscribed to it. Finally, xLM Manager is also the place where each xLM component publish their own local and public events. Local events are published in the same way as public events, but they are not forwarded to the LEACTIVEMATH Event Manager and hence are not visible outside xLM.

B.2 Types and Data Structures

Here is a description of all the public types and data structures used with the API described in this document.

- `learnerId : String`
  represents the (unique) identifier of a learner (i.e. user).
- `contentId : String`
  represents the (unique) identifier of a (MBase) piece of content.
- `topicId : String`
  represents the (unique) identifier of domain topic (or an association) in a learner model.

- `competencyId : String`
  represents the (unique) identifier of a competency in a learner model. It can be one of the following values (following the definition for the exercise metadata, see LeActiveMath Partners (2004d)): `think`, `argue`, `model`, `solve`, `represent`, `language`, `communicate`, `tools`, `none` (indicating no competency selected from this level) or `competency` (indicating the combination of all eight competencies).

- `motivationId : String`
  represents the (unique) identifier of a motivation factor in a learner model. It can be one of the following values: `effort`, `confidence`, `interest`, `none` (indicating no motivational factors selected from this level) or `motivation` (indicating the combination of all motivation factors).

- `affectId : String`
  represents the (unique) identifier of an affective factor in a learner model. It can be one of the following values: `satisfaction`, `pride`, `liking`, `anxiety`, `none` (indicating no affective factor selected from this level) or `affect` (indicating the combination of all affective factors).

- `motivAffectId : String`
  represents the combination of affect and motivation identifiers. This identifier is mostly used when querying the Learner Model for a particular belief, as motivation and affect can never be requested at the same time (see the description of the Learner Model, section 2.4).

- `metacogId : String`
  represents the (unique) identifier of a metacognitive ability in a learner model. It can be one of the following values: `monitoring`, `control`, `none` (indicating no metacognitive ability selected from this level) or `metacognition` (indicating the combination of all metacognitive abilities).

- `capesId : String`
  represents the (unique) identifier of a Conceptual and Procedural Error in a learner model.

- `beliefId : String`
  represents the (unique) identifier of a belief in a learner model. Uniqueness of this identifier is guaranteed by its indexing based on the dimensions the belief is about, i.e.`<metacogId,motivAffectId,competencyId,topicId>`.

- `olmMoveId : String`
  represents the identifier of an interaction move supported by the Open Learner Model. It can be one of the possible value: `showme`, `agree`, `disagree`, `confirm`, `disconfirm`, `baffled`, `moveon`, `perhaps`, `hereis`, `canconfirm`, `unravelling`, `finishtopic`, `suggest`.

- `Scope : Class`
  represents the extent the domain is used when combining factors from the upper dimensions of the Learner Model. It can one of the default values representing common-usage scopes: `session` (indicating all domain topics used since the learner logged in), `book` (indicating all topics available in the currently loaded book) and `domain` (indicating all topics included in the Learner Model) [1].

  In order to ensure a greater flexibility in specifying the scope on which beliefs are combines, `Scope` can also be a list of individual topics (i.e.`List<topicId>`).

- `LearnerModel : Class`
  represents the whole Learner Model (i.e. the accumulation of all beliefs for all topics) for a given learner.

[1] This is an initial list which includes the basic scopes that we think will be commonly used by the system. If the need for further definitions arise, we will obviously amend it.

- `Belief : Class`
  represents the distribution of a learner's level in one of the dimension of the LM (i.e. competency, motivation, affect, metacognition).
- `BeliefCluster : Class`
  represents the aggregation of all `Belief` related to one particular point of view (i.e. one particular domain topic, one particular competency, etc.).
- `Face : Class`
  represents a combined autonomy and approval values.
- `Evidence : Class`
  represents the (organised) evidence supporting a particular belief.
- `Map : Class`
  represents a generic storage device containing nodes inter-connected by associations. It is mainly used to store the various map used to define the dimension of the Learner Model, i.e. the subject domain (`DomainMap`), the competencies (`CompetenciesMap`), the motivation factors (`MotivationMap`), the affective factors (`AffectMap`) and the metacognitive abilities (`MetacognitionMap`).

B.3 Events

B.3.1 Event Tags

- `ExtendedLearnerModelTag [Inherits: Application]`
  Events are associated to the Extended Learner Model (xLM)
- `LearnerModelTag [Inherits: ExtendedLearnerModelTag, User]`
  Events are associated to a Learner Model (LM) for a given learner
- `LearnerHistoryTag [Inherits: ExtendedLearnerModelTag, User]`
  Events are associated to a Learner History (LH)
- `OpenLearnerModelTag [Inherits: ExtendedLearnerModelTag, User]`
  Events are associated to a learner's interaction with the Open Learner Model (OLM)
- `SituationModelTag [Inherits: ExtendedLearnerModelTag, User]`
  Events are associated to the Situation Model (SM) of a given learner in a given topic

B.3.2 Public Published Events

Public events are broadcasted to all LEACTIVEMATH components.

- `XlmTopicAdded (Attr: learnerId,topicId) [Tags: LearnerModelTag]`
  A new domain topic has been added to the LM of a learner (this does not mean that beliefs on this topic have been already defined or revised, see `XlmBeliefUpdated`)
- `XlmBeliefUpdated (Attr: learnerId,beliefId) [Tags: LearnerModelTag]`
  The beliefs about a particular topic have been revised

B.3.3 Private Published Events

Private events are restricted to the components within the xLM and are mainly used for internal usage.

- `XlmStarted (Attr: ) [Tags: ExtendedLearnerModelTag]`
  The xLM has been started
- `XlmStopped (Attr: ) [Tags: ExtendedLearnerModelTag]`
  The xLM has been stopped
- `XlmUserAdded (Attr: ) [Tags: ExtendedLearnerModelTag]`
  A new user has been added in xLM (i.e. LH and LM)
- `XlmUserRemoved (Attr: ) [Tags: ExtendedLearnerModelTag]`
  A user has been removed from xLM (i.e. LH and LM)
- `XlmOLMMove (Attr: learnerId,beliefId,olmMoveId) [Tags: OpenLearnerModelTag]`
  The OLM send this message to indicate that a dialogue move `olmMoveId` has been played by the learner `learnerId`, regarding the belief `beliefId`. The nature of the move is specified by `olmMoveId`.

B.4 Listened Events

The following are events listened to by the xLM.

From LeAM Front-End:

- `PagePresented` keeping track of the user's position, analyzing total time spent on page
- `ItemPresented` updating the LM.
- `UserBookCreated` tracking topics of interest to the user
- `UserBookDeleted` cleaning operations.

From the User Manager:

- `UserCreated` creation of a new learner model
- `UserDeleted` tidying up the LM/SM, e.g. removing saved learner models from disk
- `UserLoggedIn` caching/loading operations in the LM/SM, e.g., loading a learner model from disk.
- `UserLoggedOut` storage operations in the LM/SM, e.g., writing a learner model to a file.
- `UserPropertyChanged` tracking property changes in the User Profil

From the Exercise Manager:

- `ExerciseStarted` Updating SM, e.g., start watching the time spent on the exercise.
- `ExerciseFinished` updating the LM/SM, e.g., update the LM's beliefs about the learner

- `ExerciseStep` Updating the LM/SM, e.g., updating the LM's beliefs about the learner
- `ExerciseHelpRequested` Updating the LM/SM, e.g., updating the LM's beliefs about the learner

From the Dictionnary:

- `DictSearch` Tracking user's interests

B.5 Requirements from LEACTIVEMATH

- `getDomainTopics(learnerId)` ⟶ `Map`
  This method is used by the LM for retrieving a map of the subject domain including all domain topics and associations associated with content the learner is currently accessing (see section 4.4.2.1 for an explanation about the domain topics and their relation with the content).

B.6 API

- `createLearnerModel(learnerId)` ⟶ `void`
  This method needs to be called when a new user `learnerId` is created, so that a LM is immediately initialised for the user.

- `destroyLearnerModel(learnerId)` ⟶ `void`
  This method needs to be called when a user `learnerId` is deleted from the system.

- `doesLearnerModelExist(learnerId)` ⟶ `boolean`
  This method is called to check if a LM does already exist for the user `learnerId`.

- `getLearnerModel(learnerId)` ⟶ `LearnerModel`
  This method return a complete copy of the LM for the user `learnerId`.

- `alreadySeen(learnerId,contentId)` ⟶ `boolean`
  This method returns true if the user `learnerId` has already seen the item `contentId`, false otherwise.

- `getNumEntries(learnerId,includeFilter,excludeFilter)` ⟶ `boolean`
  This method returns the number of events stored in the LH that match the `includeFilters` but do not match the `excludeFilters`.

- `getHistoryEntries(learnerId,includeFilter,excludeFilter,`
  `startIndex,maxNum)` ⟶ `List`
  This method returns a List of `maxNum` LH events for the given learner that match the `includeFilters` but do not match the `excludeFilters`, starting at the index `startIndex` of the full result list.

- `fetchEvent(eventID)` ⟶ `Event`
  This methods returns the event with the specified ID from the LH.

- `getFace(learnerId,aut,app)` ⟶ `Face`
  This method returns the current values of autonomy and approval calculated by the situational model based on the relevant information from the Learner Model of the user `learnerId` and based on the relevant interactions. `aut` and `app` refer to the recommended amount of autonomy approval that the learner should be given in the current situation.

- `getCAPEs(learnerId,topicId)` ⟶ `List<capesId>`
  This method returns the list of CAPEs identifiers that have been diagnosed for the learner `learnerId` on the domain topic `topicId`.

- `getSummaryBelief(learnerId,metacogId,motivAffectId,`
  `competencyId,topicId)` ⟶ `float`
  `getSummaryBelief(learnerId,metacogId,motivAffectId,`
  `competencyId,Scope)` ⟶ `float`
  This method return the numerical summary of a belief (see section 4.4.3) about the learner `learnerId`. Which belief to request is determined by using the appropriate quadruplet for `<metacogId, motivAffectId, competencyId, topicId>`.

  For example, `getSummaryBelief(learnerId, none, none, think, chain_rule)` retrieves the summary of the belief about the mathematical competency of the learner `learnerId` on the chain rule, where as `getSummaryBelief(learnerId, none, motivation, none, chain_rule)` retrieves the summary of the level of motivation of the learner `learnerId` on the chain rule.

  Alternatively, a `Scope` for combining beliefs over the domain dimension can be given instead of a `topicId`.

- `getBelief(learnerId,metacogId,motivAffectId,`
  `competencyId,topicId)` ⟶ `Belief`
  `getBelief (learnerId,metacogId,motivAffectId,`
  `competencyId,Scope)` ⟶ `Belief`
  Same principle as the method `getSummaryBelief` above, except that `getBelief` returns the full four-scale distribution of the belief (see section 4.4.3).

- `getBeliefCluster(learnerId,metacogId)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,motivAffectId)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,competencyId)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,topicId)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,Scope)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,metacogId,motivAffectId,`
  `competencyId,topicId)` ⟶ `BeliefCluster`
  `getBeliefCluster(learnerId,metacogId,motivAffectId,`
  `competencyId,Scope)` ⟶ `BeliefCluster`
  This method returns all individual beliefs connected to the node specified by the parameter(s).

  For example, `getBeliefCluster(learnerId, none, none, none, chain_rule)` returns all the beliefs that are connected to the domain topic "chain rule", whereas `getBeliefCluster (learnerId, none, none, think, chain_rule)` returns those only connected to "mathematical thinking about the chain rule".

- `getEvidence(learnerId,beliefId)` ⟶ `Evidence`
  `getEvidence(learnerId,BeliefCluster)` ⟶ `Evidence`
  This method is called to retrieve the evidence on a given belief `beliefId` about the learner `learnerId` (or on a belief cluster `BeliefCluster`, by combining the evidence of all beliefs in the cluster).

- `launchOLM(learnerId)` $\longrightarrow$ `void`
  `launchOLM(learnerId,topicId)` $\longrightarrow$ `void`
  This method is called to launch the OLM for the learner `learnerId`. Alternatively, it is possible to specify which negotiation subject `topicId` the OLM will be initialised with.

- `postSuggestion(learnerId,contentId)` $\longrightarrow$ `void`
  `postSuggestion(learnerId,Criteria)` $\longrightarrow$ `void`
  Calling `postSuggestion` send a suggestion to the TC that a particular activity associated with a specific piece of content should be presented to the learner as the next best choice. The content could either be specified directly (by its identifier `contentId`) or the Tutorial Component can be left to choose using some `Criteria` such as difficulty, type of content, subject, etc.

Appendix C

Requirements related to the xLM

C.1 Knowledge Representation

Requirement 2.3	Metadata have to characterise competencies - learning activities such as model, compute and test.
Supports	• Adaptation to competency-based learning scenario. • Generation of well-balanced courses in terms of competencies.
Because	TC can choose and assemble appropriate learning objects.
Check-rule	Check exemplary learning objects and overall content.
Issues	• More annotation means more work for authors. • Complex exercises may have to be split. • Not yet in all standards.

Requirement 2.5	Metadata should characterise representation forms (verbal, graphical, numeric, symbolic) and abstractness.
Supports	Adaptation to cognitive variables.
Because	If learner does not understand one form, another form can be chosen for or by her.
Check-rule	Check content.
Issues	• Work for authors. • Full automatic adaptation to learner style is questionable. • If abstractness will be used it has to be exactly defined (maybe it depends on learning context, etc.).

C.2 General Technical Requirements

Requirement 3.5	Components need to be accessible by standard communication protocols such as XML-RPC.
Supports	Communication between separate modules and reliability of the components.
Because	Standards should be used.
Check-rule	Proof of existence.
Issues	Performance issues when communication of the net.

Requirement 3.6	Query functionality: components of LEACTIVEMATH can query the knowledge base(s) for items with particular metadata, types, (multiple) relations. It returns an ID or a list of IDs.
Supports	Separation of the functionalities of the different LEACTIVEMATH components.
Because	Different components (TC, DM, evaluation) will need information from the knowledge base(s) and LM/SM.
Check-rule	Check architecture.
Issues	Mapping between ontologies may be necessary.

C.3 Tutorial Component

Requirement 4.13	Allow for reactivity: tutorial component should react to learner's progress and problems.
Supports	Individual needs of the student.
Because	Course generation (which happens before the student accesses the course) may be based on assumptions that change during learning.
Check-rule	Test with small group of students (reactivity vs. non-reactivity).
Issues	• Changes to the course have to be indicated to the learner. • Reactivity may create navigation problems.

Requirement 4.15	Need ontology of instructional objects.
Supports	• Integration of third-party content. • Provides adequate level of abstraction for authors/learners to talk about learning materials.
Because	There exist many knowledge representations of e-learning content and using a shared ontology supports integration. Users do not want to know about internal knowledge representations.
Check-rule	Check: can the learning material be described by the ontology? Do users understand the ontology?
Issues	• Semantics of ontologies is partly not properly defined. • Mapping of ontologies may be ambiguous.

Requirement 4.17	The TC offers scenarios that target competencies.
Supports	Competency-level pedagogy as used in PISA and other studies.
Because	LEACTIVEMATH should be able to conform with didactic standards.
Check-rule	Compare results with national curricula and courses that comply with such standards.
Issues	• Formalisation and encoding of competency standards. • Needs appropriate content. • Variation in different European countries (has to be discovered by Augsburg in the first place).

Requirement 4.18	The TC needs to have access to all instructionally relevant information about the user.
Supports	Adaptivity.
Because	Without information about the user, the TC cannot provide adequate adaptations.
Check-rule	
Issues	• The relevant information includes the learner's current knowledge state, her preferences, her history, her learning/cognitive style, misconceptions, static profile and traits. • We may need for an ontology similar to the ontology of instructional objects in case we want to switch between different learner models that use different representations of individual information.

Requirement 4.19	Action selection and presentation is determined by the TC (and DM) which in turn are informed by the LM/SM.
Supports	Adaptation of content and form of interaction.
Because	• Tutorial actions should depend on the current situation. • Guidance and encouragement (autonomy and approval) depend on such variables. • Factors such as exercise difficulty, achievement and confidence are not independent of each other.
Check-rule	Sensitivity study of the effect of varying factor values on content choice, expert evaluation by teachers.
Issues	• Operational definitions of factors; defining dependencies between factors. • Determine appropriate set of pedagogical and communicative strategies and other actions. • Ratings of possible actions and strategies. • Generation of actions and application of strategies by TC.

C.4 Assessment Tool

Requirement 4.21	Assessment tool should be able to run a variety of exercise types and exercises with several competencies.
Supports	Diagnosis of different competencies.
Because	If MCQs are used only, then assessment of knowledge and competencies is limited.
Check-rule	
Issues	• May require major changes to Siette. • Currently, Siette is envisioned as assessment tool only rather than as 'the exercise system' of LEACTIVEMATH.

Requirement 4.22	Integration of assessment tool into LEACTIVEMATH and communication with other components (learner model, tutorial component).
Supports	A single learning environment with several services.
Because	Initialises learner model, test can be requested by tutorial component.
Check-rule	Architecture and test.
Issues	• Translation of data structures. • Communication protocol. • Changes in Siette. • Assessment may cause replanned curriculum.

C.5 Related to the Extended Learner Model

Requirement 5.1	The learner model (LM/SM) will feature beliefs about the learner's knowledge, skills, competencies, competency levels, academic interests, media competencies, affective and motivational states.
Supports	The selection of pedagogical strategies and tactics to support the learner's progress.
Because	Other components of the system, such as the DM and the TC can utilise these beliefs to adapt the available options and to discuss matters with the learner more appropriately.
Check-rule	The existence of the LM, whereas the quality of the beliefs will be checked by some appropriate empirical method.
Issues	• The selection of particular aspects to hold beliefs on. • The usual problems of maintaining consistency and updating the learner model.

C.6 Related to the Learner History

Requirement 5.2	The LM will need to access the LH to analyse the past history of interaction.
Supports	Looking for patterns in the interaction.
Because	We want to detect the learner's shifts in behaviour and determine, if these are indicate learning or something retrograde.
Check-rule	The proof will be in terms of the patterns that can be detected reliably so checking could be via simulating students learning with LEACTIVEMATH.
Issues	Some patterns may be time-dependent so we need the LH with sufficiently accurate time stamped data.

Requirement 5.3	The LM will have access to time stamped records of learner behaviour inside sessions, ranging from some low-level interface events to high level interpretations of learner behaviour (such as content review and performance at exercises).
Supports	Derivation of relevant features and patterns in LM which, in turn provide support for system adaptation to learner needs.
Because	Learner's shifts in behaviour needs to be detected as well reasons for the beliefs in LM.
Check-rule	Empirical tests to find out whether records are appropriate and sufficient to infer relevant information and patterns.
Issues	• Low-level of action may need to be recorded at client side. • Interfaces needed for the LM to access this information. • There may be problems with translating from low-level events to higher-level ones.

Requirement 5.4	Updates to the LM beliefs should be supported by information stored in the LH.
Supports	Justification of the learner model beliefs by learner behaviour.
Because	Transparency is very important for acceptance by students and teachers.
Check-rule	The explicit representation of the interpretation of the information in the LH as well as the provision of this evidence when necessary (in the open learner model).
Issues	• The design of justifiable interpretation. • Depends on how much explanations the (open) learner model is supposed to provide.

C.7 Related to Open Learner Model

Requirement 5.5	The OLM will be able to present the learner with beliefs about the learner's knowledge, skills, competencies, competency levels, academic interests, media competencies, affective and motivational states.
Supports	The learner thinking about what they know, their interests, preferences, competencies, affective and motivational states, promoting in this way the development of metacognitive skills.
Because	A learner can reflect on the domain being learnt, as well as on their affective state and motivation for learning, acquiring in this way metacognitive knowledge and skills. In addition, the open learner model may be a way for learners helping the system to improve its learner models.
Check-rule	The existence of the corresponding interfaces, as well as qualitative semi-structured interviews which may show the effects on learners.
Issues	The possibility that the learner would consider the learner model too inaccurate, it has to be decided what should be accessible by learners and the extent and mechanisms by they can alter the learner model.

Requirement 5.6	The open learner model provides the learner with a user interface(s) for displaying and manipulating each of the various aspects of the learner's model properly (e.g. student's domain understanding, affective state, motivation, etc.).
Supports	Focusing the student on different aspects of her learning.
Because	It would be confusing to mix these aspects together and each of these aspects may require a particular mode of representation.
Check-rule	It would be possible to do an experimental study based on two different interfaces
Issues	There are a number of issues in defining how to present the different parts of the OLM and what is an adequate representation.

Requirement 5.7	The OLM user interface provides mechanisms supporting negotiation with the learner about the beliefs stored in the LM.
Supports	Accountability of the learner model on the face of the learner, as well as its accuracy, helped by the learner. It also supports metacognition.
Because	The system will need to provide evidence justifying the beliefs in the learner model, and vice versa, the learner will need to justify their claims by providing their own evidence. In addition, negotiating the system beliefs, as stored in the learner model, may encourage learners to think more deeply about the nature of domain knowledge and their comprehension of it.
Check-rule	The existence of the facilities for negotiation should be complemented with empirical studies of their effect on the learner model accuracy and trustworthiness, as well as the level of learner engagement achieved by the negotiation and its effects.
Issues	Negotiation is a complex activity, which demands some sort of natural dialogue structure, even if not carried out in natural language. There is also the issue of who is going to have the last word, and the consequences of it.

Requirement 5.8	OLM user interface may hide some of the beliefs found in the LM.
Supports	Preserving the "face" of learners and preventing overload.
Because	Too much “honesty” might be depressing/demotivating.
Check-rule	Experimental study based on two different interfaces.
Issues	Research problem: this is a serious and difficult issue - lots of problems in deciding, in a principled way, what to hide. Also the hiding must be adaptive, but how?

C.8 Related to Communication and Interfaces

Requirement 5.9	The LM information helps initialising some properties of exercises for the TC and DM. For instance, the LM will help to derive estimates of the appropriateness of the exercise for the learner.
Supports	Selection of the next task (sequence of tasks) by the TC or DM that might be offered to the learner.
Because	The student may need a particular trajectory for her learning.
Check-rule	Proof by existence.
Issues	Decisions about choice of metadata in exercises.

Requirement 5.10	Self-assessment can be used to provide input to LM at various stages.
Supports	Initialising the learner model.
Because	The student should notice the adaptivity as an advantage.
Check-rule	Small lab experimental design.
Issues	The student's attitudes and preferences may change during her learning. By asking too many questions the students may feel disturbed.

Requirement 5.11	The SM provides the DM with values of student situation. for guidance and encouragement (autonomy and approval).
Supports	Adapting the dialogue? generation to the learner.
Because	• We need to communicate with learners in different ways to respond to their *affective* needs, e.g. less confident students may need more encouragement (i.e. more explicit approval). • We need to communicate with learners in different ways to respond to their *cognitive* needs, e.g. students achieving less may need more structured and detailed guidance (i.e. less autonomy).
Check-rule	Sensitivity study to show that the recommendations of values for autonomy and approval change according to the factor values in different situations.
Issues	Ratings of possible actions and strategies in terms of levels of autonomy and approval values.

C.9 Natural-language enhanced dialogue

Requirement 6.2	DM needs access to the LM and SM information for realising the local tutorial dialogue.
Supports	The generation of student-adaptive and situation-adaptive hints.
Because	Dialogue has to take the student's prior performance and situation into account.
Check-rule	Test how student-adaptive and situation-adaptive dialogue affects student motivation and learning gains
Issues	• Definition of relevant information. • Mapping the values of the LM/SM to appropriate dialogue strategies and their corresponding/appropriate NL verbalisations.

Requirement 6.5	NL-Dialogue subsystem interfaces with and informs the LM and SM and the exercise component.
Supports	Proper update of information about the learner's progress, actions, goals.
Because	Interactions, progress, interpretable judgements, etc. take place during dialogue.
Check-rule	A small scale pilot study using qualitative methods could be used to check that the LM is being maintained in a plausibly accurate and timely manner.
Issues	• Determine exactly which information is available and can be passed. • It is highly likely that there will be inaccuracies. • LEACTIVEMATH has to function properly also without a DM which produces English only.

C.10 Inspection of OLM through NL Dialogue

Requirement 6.6	Communication about LM through natural-language enhanced dialogue.
Supports	Student's inspection and modification of the learner model.
Because	NL-enhanced inspection of LM might be more effective than without NL-enhanced dialogue.
Check-rule	Test how NL dialogue can be more effective than LM inspection without dialogue.
Issues	• Pointing may be more effective than natural language. • From the NLU point of view, lots of complexities involved in using NL dialogue to help LEACTIVEMATH to diagnose the student's state of knowledge or expertise. May require the capability to reason about how the OLM came up with its values in the first place. • Need corpus study to inform the design of such as learner model NL-enhanced dialogue component, and to determine the form of dialogue that we can realistically support. • The diverse content of the learner model will need a wider range of dialogue strategies than normally considered within ITS. • Strategy selection given dialogue context and learner model.

C.11 Exercises

Requirement 7.3	Feedback has to be presented.
Supports	Supports discovery of errors.
Because	Empirical evidence.
Check-rule	Check with students and/or authors whether the feedback is suitable.
Issues	• Diagnosis is a hard issue in general. • Different types of feedback may be necessary. • DM needs to cope with the information underlying the feedback as well.

Requirement 7.4	Diagnosis of student's input in exercises.
Supports	Provision of elaborate feedback and tutorial dialogue.
Because	The correctness has to be judged.
Check-rule	Few tests comparing the system's diagnosis with teacher's diagnosis.
Issues	• Requires input from authors/teachers on frequent mistakes and possible solutions. • A correct diagnosis may be very difficult to compute. Therefore, it simplifies diagnosis, if the learner is asked to concretise her problem or to make more detailed steps. • Relevance of steps can only be judged approximately in some cases.

Requirement 7.5	A history of a student's activities in an exercise has to be stored.
Supports	Diagnosis and feedback.
Because	
Check-rule	Test which of this information is actually needed.
Issues	The local history has to be stored separately in order to be accessible to different components of the system.

C.12 Exercise Repository

Requirement 7.7	The repository is accessible and searchable by computational systems. In particular, it is well integrated with LEACTIVEMATH.
Supports	Usability and interoperability of the system.
Because	Availability of the exercises to the TC and DM to integrate them into interactive courses and suggestions.
Check-rule	Check whether all exercises can be selected in tutorial choices of LEACTIVEMATH.
Issues	• Exercises need to be encoded in the OMDoc exercise format with ActiveMath extensions (structure and metadata). • Need to display the copyright statement attached to each exercise. • Useful to have a single naming scheme for mathematical notions. • The repository should distinguish between a user who directly access and a user who access repository via LEACTIVEMATH. • Service approach may be well-suited.

Requirement 7.8	The exercise subsystem communicates with the LM.
Supports	• Adaption of exercises and feedback in exercises. • Learner model update.
Because	Both directions of communication are needed.
Check-rule	Proof by existence.
Issues	Determine the relevant information on both sides.