Towards MultiMedia Instruction in Safe and Secure Systems

Bernd Krieg-Brückner

Bremen Institute of Safe and Secure Systems, Universität Bremen,
Postfach 330440, D-28334 Bremen
bkb@Informatik.Uni-Bremen.DE

Abstract. The aim of the MMiSS project is the construction of a multimedia Internet-based adaptive educational system. Its content will initially cover a whole curriculum in the area of Safe and Secure Systems. Traditional teaching materials (slides, handouts, annotated course material, assignments and so on) are to be converted into a new hypermedia format, integrated with tool interactions for formally developing correct software; they will be suitable for learning on campus and distance learning, as well as interactive, supervised, or co-operative self-study. Coherence and consistency are especially emphasised, through extensive semantic linking of teaching elements, and through a process model borrowed from the theory of formal software development, enlarging the knowledge base with the help of version and configuration management, to ensure "sustainable development", i.e. continuous long-term usability of the contents.

1 Aims

The aim of the MMiSS project (*MultiMedia instruction in Safe and Secure Systems*), which is supported by the German Ministry for Research and Education, bmb+f, in its programme *"New Media in Education"* from 2001 to 2003, is to set up a multimedia Internet-based adaptive educational system, covering the area of Safe and Secure Systems. Thanks to a consistent integration of hypermedia course materials and formal programming tools, teaching in this area will attain a level hitherto impossible in this form. The system will be as suitable for learning on campus and for distance-learning with its associated management of assignments, as it is for interactive, supervised, or co-operative self-study.

The system is to be introduced step by step, over the duration of the project, into the normal teaching activities of the project partners: Universität Bremen (Krieg-Brückner, Eckert [now at Darmstadt], Gogolla, Kreowski, Lüth, Peleska, Roggenbach, Schlingloff [now at HU Berlin], Schröder, Shi et al.), FernUniversität (Distance-University) Hagen (Poetzsch-Heffter, Kraemer et al.), Universität Freiburg (Basin, Wolff et al.), Ludwig-Maximilians-Universität München (Wirsing, Kroeger, Merz et al.), and Universität des Saarlandes (Hutter, Melis, Siekmann, Stephan et al.). However, as the "Open-Source" model is to be used and teaching materials and tools are to be made freely available, a much greater

national and international take-up is to be expected. To assist this, a MMiSS Forum is to be founded with German, international, and industrial members, to evaluate the emerging curriculum and assist its development and distribution. The Advisory Board shall advise the project from a scientific as well as an industrial perspective, with a view to future applications.

The area of *"Safe and Secure Systems"* has in the last few years become increasingly important. Software is increasingly used to control safety-critical embedded systems, in aeroplanes, spaceships, trains and cars, and electronic trading over the Internet, with its associated security risks, is rapidly expanding; all this requires qualitatively and quantitatively better training. To go with the planned deployment at universities, a number of well-known German companies have already expressed, through the various industrial contacts of the project partners, an interest in measures for further in-house training.

At the core of the system is the hypermedial adaptation of a series of classes or lectures on the development of Safe and Secure Systems. The lecturers should be able to store various sorts of course material, such as overheads, commentary, bibliographies, books, lecture notes, exercises, animations and so on, and retrieve them again for use in teaching. The system provides a formal framework for the integration of teaching materials based on a *semantic structure (ontology)* and enables fast directed access to individual teaching elements. An initial collection of teaching materials is already available and should be further hypermedially developed as part of the project. It covers the use of *formal* methods in the development of (provably) *correct* software. Highlights include data modelling using algebraic specifications; modelling of distributed reactive systems; handling of real-time with discrete events; and the development of hybrid systems with continuous technical processes, so-called *safety-critical systems*. The curriculum also covers informal aspects of modelling, and introduces into the management of complex developments and *security* .

The system will also contain a meta-database, containing methodological, ontological and paedagogical knowledge about the contents. The teaching materials should, where possible, be available in several different variants. It should be left to the teachers, or the students, to choose between variants, according to the educational or application context. For example reactive systems could be modelled with either process algebras or Petri-nets.

An important educational aspect is to teach about the possibilities and limits of formal tools. Tools for formal software development should be integrated in the system, to illustrate and intensify the contents to be taught. Thus students doing assignments can use the system to test their own solutions, while gathering experience with non-trivial formal tools. The integration of didactic aspects with formal methods constitutes a new quality of teaching. It will become possible, for the first time in formal methods, both to present a variety of formal tools as a subject for teaching, and to use them as a new medium. Thus an algorithm can for example be simultaneously developed, visualised, and verified.

The goal of applying the new system in as many universities and companies as possible, and the fact that the area of Safe and Secure Systems will continue

to develop in future, requires the highest level of *flexibility, extensibility and reusability* of the content. It should be possible to incrementally extend or adapt content and meta-data, to suit the teacher's individual requirements, and to keep them up-to-date.

As the individual parts of the curriculum rely on each other, there is a network of *semantic dependencies*, which the system should be able to administer; thus it must at the least handle version- and configuration-management. The ontology additionally allows better support for orientation and navigation within the content. It should also form the basis for adaptation to the user, for example by learning from exercises which concepts the students have understood, and adapting future assignments accordingly.

The formalisation of semantic dependencies means that the system can help *maintain the consistency* of the content. Definitions must be coordinated to suit each other; the removal or adaptation of part of the material may force the removal or adaptation of all dependent concepts. In formal software development, a similar problem has to be solved: there are also semantic dependencies between different parts of a development, for example between specification and implementation. Some of the project partners have already developed techniques for the administration of such dependencies as things change, and implemented them in development tools. Here we perceive an important *synergy* between expertise in formal software development – and support tools – and the demands of long-term sustainable administration of consistent multimedia materials in an efficient and productive educational system.

2 Sustainable Development of the Content

The problem of "sustainable development", i.e. how to continuously develop and maintain multimedia educational content, is to a large extent unsolved:

> To help realise complex systems, tools are needed to support the development process from initial design to maintenance. ... Tools are also becoming ever more important, to co-ordinate team-work and guarantee consistency during development and beyond. The existing tools have substantial deficiencies. Support is especially lacking for the early stages of development, as is a suitable methodological framework. ... The commercially available systems ... offer a wide range of possibilities, though the ... results are hardly understandable or maintainable. ... Generation and reuse of previously developed components in a new project is as good as not supported. A further grave problem is the deficient or inadequate support of quality control during development. ... This leads later to maintenance problems, as with current software systems. ... The story is similar with the development of educational systems, for which the development methods and tools in use today correspond to the state of the art 20 or 30 years ago. [36]

In the MMiSS project, the elimination of these deficiencies is a priority.

In teaching practice there is a series of specific problems in the area of formal methods, for example:

- the adequate communication of abstract mathematical concepts;
- the communication of course material which has a complex structure, is often presented in a non-uniform way, and develops dynamically;
- the integration of practical aspects, such as process models [16] and tools.

Teaching Material for Safe and Secure Systems. The area of Formal Methods, the basis for the content to be developed during the project, is established in academia and on the threshold of coming into the industrial mainstream. It is differentiated into a variety of competing alternatives and orthogonal, potentially complementary approaches. Like many mathematical theories, the different methods have a complex internal structure. Due partly to the rapid development of the last 15 years, there also remains a certain lack of uniformity in the presentation of the theoretical foundations, with corresponding consequences for teaching. For the content, the project will address standardisation as a priority in the short term, work out approaches for integration, and devise principles for the comparison and presentation of alternatives. We will now sketch previous work of the project partners in this direction, reflected in the comprehensive teaching material that is already available.

Several project members were involved in the bmb+f project KORSO (*"Correct Software"* [12]), which laid the foundations for this co-operation; of these, several have been working for many years on Algebraic Specification [11, 13, **?**,4] in a rather closed-knit international community, funded for many years by the EU as ESPRIT WG COMPASS [23] and migrating eventually into the IFIP WG 1.3 (Foundations of System Specification) The *Common Framework Initiative for Algebraic Specification and Development (CoFI)* [28, 14] of IFIP WG 1.3, which originated from COMPASS, aimed at the development of an internationally standardised family of specification languages [2, 15, 35, 29]. CASL, the core language of this family, shall be a standard in the project for all teaching content concerned with mathematical foundations, algebraic specification and data modelling; it is well-supported by tools [27, 26], and its development methodology receives increased attention [31–33]; its link to the functional programming language Haskell as a target is well under way [35].

Work at Ludwig-Maximilians-Universität München and at Universität Bremen will be important for integrating the content with which we are concerned here into the whole subject of software development. This work aims to build bridges between Formal Methods, and those informal methods and languages which are in practice now a de facto standard, such as UML and Java. Techniques for specification and verification of object-oriented programs have been developed at FernUniversität Hagen.

The situation is less unified in the area of the formal treatment of concurrent reactive distributive systems, up to and including (hard) realtime and hybrid systems. One possibility is the hierarchy of languages established at Universität Bremen, which stretches from the widely-used language CSP [19, 34, 38] via

Timed-CSP [25] and HybridCSP [1] to hybrid automata and the duration calculus, to be combined with CASL [30]. The foundations of temporal logic have been analysed at Ludwig-Maximilians-Universität München and used there and at Universität Bremen in the teaching of model-checking. At Universität Freiburg, it was demonstrated how decidable monadic second-order logics can be used to model and to reason about such systems (e.g. [9]). Within courses on "Software Techniques", "Testing" and "Proofs and Modelling", a wide range of content on the subject of "Integrating Formal Methods into the Software Design Process" (cf. e.g. [10]) has been created.

The proof system INKA, the VSE-method and its derivatives [6, 5, 7, 8], developed at Universität des Saarlandes (DFKI), combine development methods for abstract datatypes with temporal logic. Educationally, VSE has principally been used in industrial seminars. During the adaption of content in this area, the aim is to further work out existing approaches for integration, and to delimit and classify alternative methods as they apply to particular applications.

Coherent and Consistent Teaching Materials. One problem with the development of a national curriculum on "Safe and Secure Systems" is the wide variety of different and partly competing approaches. Here a unifying approach, at an international level, presents itself, via the specification language CASL. The proposed system may be instrumental in spreading such standardising approaches, and, via New Media education, create an new identity in the field. In an analogous way, the restriction, at first, to a few established and well-supported languages and tools for reactive and hybrid systems should lead to coherence in the curriculum. Initial experience from industrial training has been very promising. Up to now the preparation of content has been done locally from the specialised viewpoint of individual teachers; the comprehensive consistent integration of content will overcome this, and so contribute to a uniform understanding of the whole area throughout Germany, and, as a perspective, beyond.

Semantic Linking of the Content. Many beginners find the subject matter very complicated at first, because of the many dependencies between the various fundamental formalisms. It is tiresome and time-consuming to communicate conceptual dependencies and conceptual analogies to students. The proposed system can play a decisive role, by providing a hierarchy of concepts (an ontology) throughout the whole material, making the complexity manageable for students as well as teachers.

The author of content will be able to assemble teaching units from a structured system of individual modules and elements, by using the structural and semantic relations explicit in the representation, such as pre-conditions, cross-references, related units and alternatives. Thus content can be prepared by different people with different goals and requirements, but together and as part of the same repository, possibly in different variants and views.

Students will also be able to side-step a prescribed order of presentation for course content, navigate by themselves and make use of related materials as their own needs dictate. Rapid access to semantically related concepts and theories will significantly help users in forming an overall picture.

It is expected that this project will influence other areas, such as Mathematics or other areas in Computing Science, in the short run, and so lead to a persistent improvement in the teaching methodology of interrelated theories.

Reusability and Extensibility. One major problem with the preparation of teaching materials of any kind is that the adaptations necessary for each teacher and each year make reuse of older material almost impossible. It is often necessary to completely restructure a course to integrate new developments and results into hand-outs and overheads. Thus a major goal of the proposed system is to guarantee users the highest degree of flexibility, extensibility and reusability of the materials stored within it. It should be easy for teachers to combine different materials, even from different authors, into a whole, and for students to use alternative material. The planned mechanisms to support this, such as version and configuration-management, consistency-preservation, and a tool-supported development methodology, will also be available for other similar systems, and are expected to substantially improve the long-term development of coherent and consistent teaching content in the New Media.

Extensible Knowledge-Base. The speed at which knowledge develops is a special problem in the areas of Computing Science and *Safe and Secure Systems*. It is imperative to be able to continuously extend and modify the stored knowledge. This leads to consistency problems, in particular for multimedia presentations. For example, it must be clearly specified whether a cross-reference (hyperlink) refers to the newest version (whatever it is), or to some specific older version; these could be lost or outdated because of modifications to the referenced content. This is especially important in the context of Formal Methods: a referenced definition must fit into the application context which may not necessarily be compatible. For example, the name of the term defined by the other author may be different, or, worse, the other author may use the same name for an entity that has a subtly different semantics. The system to be constructed shall solve this by keeping track of semantically different entities, independently of their apparent name in a specific context, and by storing additional metadata in the knowledge-base. Semantics and functionality of knowledge are to be separated from representation.

There is also the problem of granularity. An element in the content may be an entire lecture, a particular topic, a overhead (or something structurally equivalent to it) on a subtopic, or indeed a single definition or theorem. This problem is to be solved by structuring the teaching materials into a semantically-based hierarchy, reached via, and defined in their granularity by, the ontology.

Fig. 1. Document Structure and Development Graph

Semantic Relations in the Development Graph. Figure 1 shows, as an example of the structure, a section of a document containing mathematical definitions, theorems and proofs (with connecting texts); a similar situation arises with overhead transparencies for lectures. Texts, on the other hand, contain embedded *formal* components (theorems, formulas, proof-scripts) which can of themselves be processed by corresponding systems. A textual nesting (the *"is contained in"* relation) yields at first a tree structure. This is extended by *semantic relations*, defined explictly by the user or implicitly by the system. For formal components this structure is evident; a formula representing a theorem to be proved *lives in* a theory; a proof-script that proves this theorem within a proof-system is subordinate to this theorem, or in general to a relation containing a proof obligation (*proves*). In the course of development, alternatives arise, for example an alternative proof in the example; this, like earlier versions, must be preserved so that it is possible to return to it. Thus there is in general a *Development Graph* containing one or more *formal* development graphs as subgraphs.

Development Methodology. Semantic approaches from software engineering and, in particular, Formal Methods, can cure the hitherto unsolved problem of how to develop sustainable multimedia teaching content. Here the development methodology of *(stepwise) refinement* is already known (compare with *"is refinement of"* in Figure 1); this could for example be applied to working out the materials with more precision or in more detail. The important point is that a reference to this activity of refinement is preserved. Another concept borrowed

from Formal Methods is the so-called *conservative extension*, which preserves the original content so that a reference to the extended version remains valid for the original meaning. An example of this is a theory whose axioms are kept, but which is extended by further properties derived as theorems (compare with *"is conservative extension of"* in Figure 1). This concept has a well defined verifiable semantics, which naturally cannot be guaranteed for textual, or other multimedia, content. Thus consistency must be preserved through discipline among the developers, rather than formal proofs. In any case we can, as a *semantic* relation, distinguish conservative extension from a *real* change: the latter forces all dependent content to be reworked (this can be automatically recognised and communicated to the authors), while a conservative extension does not do.

Version and Configuration Management. An important dimension for Development Graphs is that of versions and their administration. Filtered *views* (realised by the graph-visualisation system daVinci [18, 17]) should help the user. Usually only the *current version* is of interest and all earlier ones are not shown; however, an option should make alternatives to a version visible. It should be possible to select between *variants*, such as the language used (for example "British English" or "German"), the formalism (for example "CASL" or "CSP"), or the level of detail (for example "Lecture Notes", i.e. overheads augmented by comments and explanations, suitable for study after presentation in class), and so on. The notion of a *consistent configuration* is important here; all objects related to a particular selected object should belong to the same version, or at least to a semantically compatible one. The document actually displayed (a subgraph) should in a well-defined sense be *complete*; thus when for example the formalism"CASL" is chosen as a variant, examples should generally be available in the formalism"CASL", and similarly when a particular level of detail is chosen. It is then possible to freeze a configuration as a publicly-accessible *edition*. All these functions, especially verifying consistency and completeness, should be supported by the system.

Scenarios and Roles. The knowledge-base is to be read, enriched or extended by different groups of people, according to the educational context. As sketched above, we recommend a semantics-driven process model for the development of multimedia educational content in which the different scenarios and rôles of those involved are differentiated:

- The *author* provides the initial groundwork (such as overheads), supplements it as required with animations or tool demonstrations, reacts to feedback from students and teachers, adds commentary, and expands it to create hypermedially-related teaching material, such as lecture notes or courses for distance learning (whether tutored or not).
- The *teacher* uses the teaching material stored in the knowledge base for teaching on campus, adapts it to his or her specific requirements (by selection or extension), and stores it back in the system.

- The *tutor* also uses the existing material, but in a different teaching situation; s/he compiles explanatory commentary, answers questions, puts frequently asked questions and answers together, sets and corrects assignments, and so on.
- The *student* uses the (prepared) teaching materials for a review after class; for self-study of the fundamentals or of additional background material; for assignments; and so on. The system helps to navigate or leads through the materials. It also contains (meta) information to support the selection of material according to the student's progress.
- The *system developer* extends the underlying system, incorporating existing and recently developed tools, especially for Safe and Secure Systems, into the development system.
- The *tool developer* works on particular tools, particularly for authoring.
- The *administrator* manages the system and cares especially for version and configuration management both of the content and the system itself. S/he moderates editions and arranges their distribution, including ones for particular user groups (such as authors, teachers, students attending particular courses); creates user groups and manages them; supports the distribution and installation of system versions .

2.1 A MultiMedia Platform for Educational Content

To support didactically worthwhile multimedia training that is genuinely interactive and cognitively adequate, powerful support systems must be developed, particularly in formal areas (Mathematics, Formal Methods); they should be adaptable to the user. Up to now, there are few such systems; the Springer-Verlag's interactive textbooks represent the first steps in this direction. These textbooks all belong to the first generation, which has no KI-methods such as user-modelling and diagnosis, learning, knowledge-representation, distributed architecture (multiple agents), and which makes little use of results from Cognitive Psychology or the theory of Education (cf. also the recent efforts of ActiveMath [24] at Universität des Saarlandes).

The ability to structure theories hierarchically, compare alternative approaches, combine complementary approaches usefully, and abstract away differences in presentation is, sometimes with difficulty, to be found among experts, but hardly among students. Currently, teaching of formal methods is predominantly characterised by being based on (or restricted to) the "local theory environment". Methodologically, classical methods such as lectures (with little or no interaction) and exercise sessions predominate. A comprehensive inter-relation of content is therefore impossible without co-operation and system support during creation. It is often a problem just to combine two, in principle complementary, but in detail differently constructed, textbooks. There is no support for recombination and further development of content, a particular problem given the rapid development of the subject area.

Structure of the Support System. The support system should have an open architecture and accommodate various user models. This requires, in the simplest case, a coarse static classification by user category or rôle in the learning process, such as Diploma or Master's student, student still learning the basics, (external) student in further training, or industrial user; such a classification should be universally introduced. It is also intended to take advantage of the opportunities available for educational systems which dynamically adapt to the progress of the user. Thus the proposed system is divided into components, which are presented to the user in a view depending on the scenario:

- The *authoring system* contains various tools for the preparation, semantic linking and extension of content.
- The *teaching system* serves primarily to support teaching on campus, but is also suitable for tutored distance-learning.
- The *learning system* contains materials for students and supports various learning situations.
- The *development system* for Safe and Secure Systems permits the integrated use of tools for demonstrations and exercises.
- The *assignment system* manages assignments; solutions to exercises are dispatched for correction, the corrections administrated, and the corrected solutions returned to the students.

A detailed architecture will be designed on the basis of the process model and the methodology, which in turn serve as basis for the implementation of the system. In particular the architecture will specify the individual components and how they communicate. It will also be necessary to consider how development tools fit into the system, the various supporting formats for encoding overheads or assignments, as well as the technical representations of ontological and paedagogical knowledge.

Knowledge-Base. All the content should be stored in the knowledge-base, which will administer the above-mentioned Development Graph with various views, including version and configuration management. Content is to be stored in its primary format, as well as possibly in automatically derived formats (for example texts should be stored in the LaTeX, XML and OMDoc [20, 21] formats), if possible distributed. In particular, the content developed during the project should be made available on special archive servers, while students are to have personal knowledge-bases, in which examples can be explored, annotations made, or exercises solved before being sent to a tutor. The personal knowledge-base can also serve as a local copy (or cache) for the students, making them less dependent on their local network.

Standards. Standards are decisive for the technical coupling, but also for semantic integration. Therefore current standardisation attempts in Mathematics and Formal Methods should be adhered to.

The educational content, the description of the meta-structures and the internal communication of the software systems are to be based on the new Internet standard XML. Embedded structuring elements are to be tagged with the special formalism used, such as CASL for the integration of structured algebraic specifications, or input formats for the formal software systems and visualisation components involved in the project; this way, these elements can be analysed by appropriate tools. Formal content should be adapted to the XML dialect OMDoc (OpenMath Documents [20, 21]), which is an extension of the OpenMath standards. OMDoc allows materials to be presented in a series of formats, such as LaTeX, DVI and PostScript for printed documentation, HTML for interactive books or browsable presentations, or MathML for special handling of mathematical formulae.

3 Learning Environment and Communicative Elements

For each rôle (among others authors, teachers and students) the system appears as an individual environment. In the following we will consider the learning environment and its elements in more detail. The learning environment supports the students in various situations:

- selecting (or generating) learning materials from the knowledge base;
- studying the course material interactively;
- communicating with tutors and other students;
- perfoming exercises, practicing and experimenting;
- administrating assignments; evaluating progress.

Tutored and Co-operative Learning, Assignments. Experience at FernUniversität (Distance-University) Hagen shows that the New Media are very good at supporting tutored and co-operative learning. This new way of learning has not yet been generally accepted in the other universities, except in isolated experiments. This will change through the continual availability and extensibility of the system to be constructed by the MMiSS project; its extensive deployment should bring about a new quality of learning.

Modern communication technology permits asynchronous and documented discussion of questions, problems and solutions within structured content-specific discussion-forums (similar to newsgroups) which should be integrated with the course material. In particular such forums can be used for efficient tutoring. It is also possible to realise different levels of visibility (for example, visible for a whole group and its tutors, or only for a particular subgroup); this is known to increase the students' willingness for co-operation.

In *tutored learning*, a tutor is available on the net to answer questions about the content, the use of the system and its tools, or assignments – either synchronously ("talk") or asynchronously (via electronic mail). *Co-operative learning* usually implies a group of students who co-operate in studying a large amount of content (for example a major course) together. A number of learning situations, described by different metaphors, should be supported, for example

- *newspaper stand*: latest information from the teaching staff is distributed
- *café*: a few participants meet in unmoderated synchronous conversation
- *market place*: many communicate asynchronously, for example all participants in a course.

The organisation of assignments poses additional problems. Exercises and sample solutions must be handed out (perhaps for quite different areas and levels of expertise of groups of participants in a course); it is also necessary to administer the forwarding of solutions to the tutors, returning the corrected exercises, and so on. At FernUniversität Hagen, Six's team have developed a tool to address these problems, WebAssign which has already been used in a large number of courses with occasionally more than a thousand students each. This system will be integrated into the current project.

The *added value* of this new form of education is that it raises the quality of learning and makes tutoring more efficient and effective, particularly in a subject area with *as many students* as Computing Science. Experience at Universität Bremen has also shown that alternative forums for interaction and communication improve the students' willingness for and enjoyment of co-operation and help them express themselves; optional anonymity can be useful here. Thus new chances for learning arise and the students are *motivated* by their own sense of success. *Women-only learning scenarios* (for example a women-only café) become possible. Another advantage of having support for tutored or co-operative learning is that there is very good feedback about student progress, and criticism about the materials can easily be forwarded to the tutors and authors; this clearly assists *quality-control*. The suitability of such approaches to the area of Safe and Secure Systems and Formal Methods becomes even more plausible as the objects of study (specifications, proofs) are per se of a written nature.

Adaptive Learning Environment. Since the system to be constructed should be available to a large user community, we can expect a variety of different application scenarios: on campus learning, self-study, tutored distance-learning, preparation and subsequent assessments of industrial projects, and so on. The educational environment should therefore contain an adaptive user-modelling component, and be oriented to universally understood metaphors when guiding the user; this will significantly increase user acceptance. The learning environment should be able to generate a personalised document from the knowledge-base, covering a special subject-area and configured according to a particular personal profile, with a variety of interaction possibilities. Document generation is to be based on the ontology and dependencies between the terms and the concepts and methods to be learnt. By this personalisation, the learning materials, examples, assignments and the way in which the knowledge is presented can be adapted to the student's state of knowledge and requirements. This adaptability entails on one hand a more individualised support, on the other a flexible re-use of teaching materials.

The foundation for the *user-adaptive generation* of learning elements is a general, partly semantic (and thereby reusable) knowledge representation. To

adapt to the profile, goals and the context of the user, such meta-data are acquired in a user model and can be updated for use in a pedagogical presentation planner. The user model will contain, for example, the user's status as a student or industrial trainee, the level of detail the user requires, or whether the user is preparing for an examination.

Interactive Learning. For the study of a content package, the learning environment provides the technical content in an adequate multimedia presentation, for example texts, graphics, pictures, animations, simulations, audio- and video-sequences, and semantic hyperlinks between them. A *navigational aid* leads the students through the content, based on the onotology. Little exercises provide self-tests to *check* the progress.

Within the learning environment, the presentation of the material can be closely interrelated with other functionality; for example the assignments can be referenced directly from the teaching content. This also applies to the embedding of software-development tools. The learning environment also offers direct access to the course-specific communication and evaluation mechanisms.

Support for Assignments and Practical Work. The learning environment will include an assignment component; this will integrate and simplify *embeddings of software-development tools* specific to the course. It will authenticate students, administrate their solutions and keep track of course marks. Various types of exercises shall be supported:

- multiple choice; exercises with textual or graphical answers;
- creation/modification of specifications, proofs or programs with tool support;
- solution of exercises by dialog with an interactive system.

Evaluation. The learning environment offers, on all those levels we have considered, the possibility of including *support for its evaluation* in the content. As part of the presentation, students can answer questions on the course and add *commentary*. It will also be possible to analyse all the students' responses to an exercise. Points of view publicly put forward as part of communication between students or with the tutors are also potentially useful material for evaluation. The learning environment will provide technical support for managing this data.

4 Embedding of Tools for Safe and Secure Software

The integration of existing formal software development tools shall make teaching more flexible and dynamic. Various interaction levels should be possible, such as "movie-demos" of tools; replays of developments in the tool; completion and extension of developments using the tool; independent working on exercises; working on a project as a team.

Use of Software-Development Tools in Teaching. An important part of training is the familiarisation with computerised tools for developing Safe and Secure Systems. Tools and their methodical use in practical scenarios have, so far, only been integrated into courses in an isolated way, at the moment mostly in the form of complementary tutored exercise sessions. A complete integration into the curriculum has yet to happen, not least because of the general restriction of teaching content to locally-available tools, and the difficulty of providing general methodical integration into the content while avoiding too much detail only of interest in the special case.

As well as educating people in the use of tools, we are also concerned with the methodical improvement of teaching from the point of view of educational theory; an increased use of animations, visualisation and active experimentation can considerably reduce the difficulty of grasping abstract concepts. The user can be aided in make knowledge explicit through an experimental and explorative approach to problem-solving. As is known from Cognitive Psychology, this is an important basis for learning.

As well as integrating tools into teaching content, we should also consider the integration of different tools, and the re-use of developments (represented in a common language) in another (tool-) context; without this coherent teaching is not possible. The project partners have developed (and continue to develop) numerous tools, and demonstrated these in practical applications, up to and including co-operation in large commercial software projects. Above all two approaches that have been developed are relevant. The bmb+f-supported development of UniForM (*Universal development environment for Formal Methods* [22]) at Universität Bremen supports the close interaction and integration of tools. The MathWeb architecture and the OMDoc format [20, 21], both developed at Universität des Saarlandes, support the loose coupling of tools and the common representation of formal development. Both partners cooperate in developing support for development graphs, their visualisation and administration [8].

Integrated Development Tools. Several formal development tools must be integrated in the learning system, if knowledge of how to use them is to be adequately communicated. This entails consistency problems not just between the tools, but also with the other content. The system must hence be configurable, and support input and output in the most popular formats on the basis of new standards.

The tool support for Formal Methods includes editing, syntactic and static semantic analysis and visualisation of specifications, their animation, interactive proof-development, fully-automated decisions procedures and test-procedures. In addition to systems which address one or other of these tasks, there are also development tools which integrate several sub-systems and so provide general support for formal development. The existing tools differ in their functionality and their fundamental formal approach.

This potential should be exploited in teaching, by complementing the passive absorption of teaching content with active explorative components. We see here

the beginnings of a new, decisive approach to improving teaching. Up to now, it is too often the case that what is taught is only of limited applicability, and only limited understanding can actually be said to have been attained. For Formal Methods in particular, current quantitatively and qualitatively limited methods doing assignments (proofs cannot really be worked out) provide no solution.

Because the whole subject area should be covered here, and we aim to be able to adapt to individual special needs and further tool development, the semantically consistent integration of tools and their technical coupling is of high importance; for application-oriented training it is indispensable.

5 Outlook

The project has made good progress during its first year. Many lectures have been converted to the initial LATEX-oriented input format, with good quality output as overhead transparencies in PDF-format. This material is now awaiting further coordination and refinement, as well as semantic interlinking using development graphs in the repository. The Development Manager, and other editing and authoring tools, are well under way towards completion.

Acknowledgement

We are grateful to the members of the Advisory Board, V. Lotz (Siemens AG, München), H. Reichel (TU Dresden), W. Reisig (HU Berlin), D.T. Sannella (University of Edinburgh), and M. Ullmann (BSI [Federal Institute for Security in Information Technology], Bonn), for their advice, and to G. Russel for his help with the manuscript.

References

1. Peter Amthor. *Structural Decomposition of Hybrid Systems – Test Automation for Hybrid Reactive Systems*. PhD thesis. Universität Bremen, 1999. Monographs of the Bremen Institute of Safe Systems 13. Shaker.
2. Egidio Astesiano, Michel Bidoit, Hélène Kirchner, Bernd Krieg-Brückner, Peter D. Mosses, Donald Sannella and Andrzej Tarlecki. CASL: The Common Algebraic Specification Language. *Theoretical Computer Science*, to appear 2003.
3. Egidio Astesiano, Michel Bidoit, Hélène Kirchner, Bernd Krieg-Brückner, Peter D. Mosses, Donald Sannella and Andrzej Tarlecki. (eds.). CASL- the CoFI Algebraic Specification Language: Tutorial Introduction, Language Summary, Formal Definition, Basic Data Types. (submitted).
4. Egidio Astesiano, Hans-Jörg Kreowski, and Bernd Krieg-Brückner (eds.). *Algebraic Foundations of System Specification*. IFIP State-of-the-Art Reports, Springer 2000.
5. Serge Autexier, Dieter Hutter, Bruno Langenstein, Heiko Mantel, Georg Rock, Axel Schairer, Werner Stephan, Roland Vogt, and Andreas Wolpers. VSE: Formal Methods Meet Industrial Needs. *International Journal on Software Tools for Technology Transfer, Special Issue on Mechanized Theorem Proving for Technology*, Vol. 3:1, pages 66–77. Springer, 2000. (see also `www.dfki.de/vse/`.)

6. Serge Autexier, Dieter Hutter, Heiko Mantel, and Axel Schairer. INKA 5.0: a logic voyager. *Proc. 16th Intl. Conference on Automated Deduction*, Trento. *LNAI* volume 1632, pages 207–211. Springer, 1999.
7. Serge Autexier, Dieter Hutter, Heiko Mantel, and Axel Schairer. Towards an Evolutionary Formal Software Development Using CASL. In Christine Choppy, Didier Bert, and Peter Mosses (eds.): Recent Developments in Algebraic Development Techniques, 14th International Workshop, WADT'99, Chateau de Bonas, France. *LNCS* volume 1827, pages 73–88. Springer, 2000.
8. Serge Autexier and Till Mossakowski. Integrating HOLCASL into the development graph manager MAYA. In: A. Armando (ed.). Frontiers of Combining Systems, 4th International Workshop. *LNCS* volume 2309, pages 2–17. Springer, 2002.
9. David A. Basin and N. Klarlund. Automata based symbolic reasoning in hardware verification. *Formal Methods in Systems Design*, 13(3):255–288, November 1998.
10. David A. Basin and Bernd Krieg-Brückner. Formalization of the Development Process. In [4]. 521–562.
11. Michel Bidoit, Hans-Jörg Kreowski, Pierre Lescanne, Fernando Orejas, and Donald Sannella (eds.). *Algebraic System Specification and Development: A Survey and Annotated Bibliography*, *LNCS* volume 501. Springer 1991.
12. Manfred Broy and Stefan Jähnichen (eds.). *KORSO: Methods, Languages, and Tools for the Construction of Correct Software – Final Report*, *LNCS* volume 1009. Springer, 1995.
13. Maura Cerioli, Martin Gogolla, Hélène Kirchner, Bernd Krieg-Brückner, Zhenyu Qian, and Markus Wolf (eds.). *Algebraic System Specification and Development: Survey and Annotated Bibliography.* 2nd edition, 1997. Monographs of the Bremen Institute of Safe Systems 3. ISBN 3-8265-4067-0. Aachen: Shaker, 1998.
14. CoFI. The Common Framework Initiative for algebraic specification and development, electronic archives. Notes and Documents accessible at http://www.cofi.info.
15. CoFI Language Design Task Group. CASL – The CoFI Algebraic Specification Language – Summary. in [14].
16. Carla Freericks. Open-Source Standards on Software Process: A Practical Application. In K. Jakobs (ed.). *IEEE Communications Magazine, Vol. 39, N 4* (2001) 116–123. See also `www.tzi.de/gdpa/`
17. Michael Fröhlich. *Inkrementelles Graphlayout im Visualisierungssystem daVinci.* Dissertation. Monographs of the Bremen Institute of Safe Systems 6. ISBN 3-8265-4069-7. Shaker, 1998.
18. Michael Fröhlich and Mattias Werner. *The interactive Graph-Visualization System daVinci - A User Interface for Applications.* Informatik Bericht Nr. 5/94 (1994). Universität Bremen. Up-to-date documentation: `www.tzi.de/~daVinci`.
19. C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall International Series in Computer Science, 1985.
20. Manfred Kohlhase. *OMDoc: Towards an OpenMath representation of mathematical documents.* SEKI Report SR-00-02, Fachbereich Informatik, Universität des Saarlandes, 2000. `www.mathweb.org/ilo/omdoc/`
21. Manfred Kohlhase. OMDoc: Towards an Internat Standard for the Administration, Distribution and Teaching of Mathematical Knowledge. *Proc. Artificial Intelligence and Symbolic Computation. LNAI.* Springer, 2000.
22. Bernd Krieg-Brückner, Jan Peleska, Ernst-Rüdiger Olderog, and Alexander Baer. The UniForM Workbench, a Universal Development Environment for Formal Methods. In: J. M. Wing, J. Woodcock, and J. Davies (eds.): *FM'99, Formal Methods.* Proceedings, Vol. II. *LNCS* Volume 1709, pages 1186-1205. Springer, 1999.

23. Bernd Krieg-Brückner. Seven Years of COMPASS. In *11th Workshop on Specification of Abstract Data Types, Joint with the 8th* COMPASS *Workshop, Oslo*, *LNCS* volume 1130, pages 1–13. Springer, 1996.
24. Erica Melis, Eric Andres, Georgi Goguadse, Paul Libbrecht, Martin Pollet, and Cartsen Ulrich. ActiveMath: System description. In Johanna D. Moore, Carol Redfield, and W. Lewis Johnson (eds.): *Artificial Intelligence in Education*. IOS Press (2001) 580–582.
25. Oliver Meyer. *Structural Decomposition of Timed CSP and its Application in Real-Time Testing*. PhD thesis. Universität Bremen, 2001. (To appear in Monographs of the Bremen Institute of Safe Systems. Logos Verlag.)
26. Till Mossakowski. CASL: From Semantics to Tools. In S. Graf and M. Schwartzbach (eds.) *Tools and Alogorithms for the Construction and Analysis of Systems*, Proceedings TACAS 2000. *LNCS* volume 1785, pages 93–108. Springer, 2000.
27. Till Mossakowski, Kolyang, and Bernd Krieg-Brückner. Static semantic analysis and theorem proving for CASL. In *12th Workshop on Algebraic Development Techniques, Tarquinia*, *LNCS* volume 1376, pages 333–348. Springer, 1998. (For the Bremen CoFI Tools see `http://www.tzi.de/cofi`.)
28. Peter D. Mosses. CoFI: The Common Framework Initiative for Algebraic Specification and Development. In *TAPSOFT '97: Theory and Practice of Software Development*, *LNCS* volume 1214, pages 115–137. Springer, 1997.
29. Horst Reichel, Till Mossakowski, Markus Roggenbach, and Lutz Schröder. CoCASL - Proof support for co-algebraic specification. In *Recent Trends in Algebraic Development Techniques, 16th International Workshop, WADT'02*, *LNCS*. Springer (accepted for presentation).
30. Markus Roggenbach. CSP-CASL - A new Integration of Process Algebra and Algebraic Specification. In *Recent Trends in Algebraic Development Techniques, 16th International Workshop, WADT'02*, *LNCS*. Springer (accepted for presentation).
31. Markus Roggenbach and Till Mossakowski. What is a good CASL specification? In *Recent Trends in Algebraic Development Techniques, 16th International Workshop, WADT'02*, *LNCS*. Springer (accepted for presentation).
32. Markus Roggenbach and Lutz Schröder. Towards Trustworthy Specifications I: Consistency Checks. In M. Cerioli and G. Reggio (eds.). *Recent Trends in Algebraic Development Techniques, 15th International Workshop, WADT'01, Genova*, *LNCS* volume 2267. Springer. 305-327.
33. Markus Roggenbach and Lutz Schröder. Towards Trustworthy Specifications II: Testing by Proof. In *Recent Trends in Algebraic Development Techniques, 16th International Workshop, WADT'02*, *LNCS*. Springer (accepted for presentation).
34. A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice-Hall International Series in Computer Science, 1998.
35. Lutz Schröder and Till Mossakowski. HasCASL: Towards integrated specification and development of Haskell programs. In *Proc. AMAST 2002*, *LNCS*. Springer (to appear).
36. Forschergruppe SofTecNRW. *Studie über Softwaretechnische Anforderungen an multimediale Lehr- und Lernsysteme*. Sept. 1999. See also: `www.uvm-nw.de`, [37].
37. G. Engels, U. Kelter, R. Depke, and K. Mehner. Unterstützende Angebote der Softwarebegleitgruppe. E. E. Doberkat et al. (eds.). *Multimedia in der wirtschaftswissenschaftlichen Lehre – Erfahrungsbericht*. LIT Verlag, Münster (2000) 27–56.
38. Haykal Tej and Burkhart Wolf. A Corrected Failure-Divergence Model for CSP in Isabelle/HOL. In *Proc. FME 97 - Industrial Applications and Strengthened Foundations of Formal Methods. LNCS*, volume 1313. Springer (1997) 318-337.