

Design, Implementation and Application of a Reusable Component Framework for Interactive Mathematical eLearning Sites

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der Rheinisch-Westfälischen
Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der
Naturwissenschaften genehmigte Dissertation

vorgelegt von

Tim Paehler

aus Köln

Berichter: Universitätsprofessor Dr. Ulrik Schroeder,

Universitätsprofessor Dr. Volker Enß

Tag der mündlichen Prüfung: 28. Februar 2005

Diese Dissertation ist auf den Internetseiten der Hochschulbibliothek online verfügbar

Für Petra

Contents

| | |
|--|-----------|
| Introduction | 9 |
| Mathematics and eLearning | 9 |
| Mathematical Literacy | 9 |
| Learning by Doing | 10 |
| The Role of eLearning in Mathematics | 10 |
| Technical and Didactic Foundations of Web-Based Learning | 10 |
| Goals | 11 |
| The Mumie Project | 11 |
| Organisation of this Document | 12 |
| 1 Technical Concepts and Existing Solutions | 15 |
| 1.1 eLearning Sites | 15 |
| 1.1.1 Specific Structure of eLearning Sites | 15 |
| 1.1.2 Interactivity | 16 |
| 1.1.3 Stakeholders and Reusability | 18 |
| 1.2 Analysis of Existing Mathematical eLearning Sites | 19 |
| 1.2.1 MIT OCW | 19 |
| 1.2.2 maths online | 21 |
| 1.2.3 Conclusions and Theses | 25 |
| 1.3 Towards a Reusable Component System | 26 |
| 1.3.1 Technical and Conceptual Reusability | 26 |
| 1.3.2 Levels of Technical Reusability | 26 |
| 1.3.3 Stakeholders and Reusability | 26 |
| 1.3.4 Existing Reusable Frameworks and Scalability | 27 |
| 1.3.5 Need for Didactic Design Process | 28 |
| 2 Didactic Design | 29 |
| 2.1 Didactic Design Process: A Meta-Model | 29 |
| 2.2 Learning Model | 30 |
| 2.2.1 Static and Dynamic Learning Models | 30 |
| 2.2.2 Interactivity: A Constructivistic Approach | 31 |
| 2.2.3 From Learning to eLearning | 31 |
| 2.3 Content Model | 32 |
| 2.3.1 Bruners Theory of Representation | 32 |
| 2.4 Methodic Model | 35 |
| 2.4.1 Taxonomy of Learning Methods: Bloom et al. | 35 |

| | | |
|----------|---|-----------|
| 2.5 | Consequences for Didactic Design | 36 |
| 2.5.1 | Didactic Design Process | 36 |
| 2.5.2 | Content Representation | 38 |
| 2.5.3 | Methods and Tasks | 39 |
| 2.5.4 | Learner Orientation | 40 |
| 3 | The Component Framework | 43 |
| 3.1 | The MathletFactory from the Student's Perspective | 43 |
| 3.1.1 | Mathematical Entities and their Representations | 44 |
| 3.1.2 | Formal Language Processing Capabilities | 44 |
| 3.1.3 | Generic Display and Interaction System | 45 |
| 3.1.4 | Number Handling | 45 |
| 3.1.5 | Animations | 46 |
| 3.2 | The MathletFactory from the Author's Perspective | 47 |
| 3.3 | The MathletFactory from the Application Developer's Perspective | 48 |
| 3.3.1 | MObjects as Models for Mathematical Entities | 49 |
| 3.3.2 | MVC Architecture Supporting Multiple Representations | 49 |
| 3.3.3 | Number Handling | 49 |
| 3.3.4 | Formal Language Processing Capabilities | 50 |
| 3.3.5 | Generic Display and Interaction System | 51 |
| 3.3.6 | Interactivity: Update Graphs | 51 |
| 3.3.7 | Building Mathlets | 51 |
| 3.3.8 | Reusability | 53 |
| 3.4 | The TestletFactory | 54 |
| 3.4.1 | Different Categories of Tests | 54 |
| 4 | Application | 57 |
| 4.1 | Application Scenarios | 57 |
| 4.2 | Applets in the Pyramit System | 57 |
| 4.2.1 | Implementing the Didactic Model | 57 |
| 4.2.2 | Applying the Didactic Design Process | 59 |
| 4.2.3 | Evaluation Results | 62 |
| 4.3 | Other Applications in Mathematical eLearning | 63 |
| 4.3.1 | Mathlets for Numerical Mathematics | 63 |
| 4.3.2 | Additions in the Mumie Project | 64 |
| 4.3.3 | School Applications | 66 |
| 4.4 | Transferability | 67 |
| 4.4.1 | Mathlets for Physics and Engineering | 68 |
| 4.4.2 | TestletFactory | 68 |
| 5 | Conclusions and Outlook | 69 |
| 5.1 | Summary | 69 |
| 5.2 | Towards Critical Mass | 70 |
| 5.2.1 | Integration with Other Platforms | 71 |
| 5.3 | Further Fields of Research | 71 |
| 5.3.1 | Large Scale eLearning Content | 71 |

| | | |
|----------|---|------------|
| 5.3.2 | Navigability and Adaptivity | 71 |
| 5.3.3 | Authoring Support | 72 |
| 5.3.4 | Integration with Server Applications | 72 |
| A | Transformation examples of mathematical concepts | 73 |
| B | Implementation Details | 75 |
| B.1 | MVC Architecture of the MathletFactory | 75 |
| B.1.1 | Requirements | 75 |
| B.1.2 | Fundamental Concepts | 75 |
| B.1.3 | Model | 76 |
| B.1.4 | View | 77 |
| B.1.5 | Controller | 77 |
| B.2 | Arithmetic and Geometric Model | 77 |
| B.2.1 | Number Types | 77 |
| B.2.2 | Vectors, Vector spaces and Matrices | 78 |
| B.2.3 | Affine and Projective Geometry | 78 |
| B.2.4 | Numerical Computing | 78 |
| B.2.5 | Compound Example | 78 |
| B.3 | Algebraic Object Model and Formal Languages | 79 |
| B.3.1 | Lexical, Syntactic and Semantic Analysis | 79 |
| B.3.2 | Introduction to Formal Languages | 80 |
| B.3.3 | Types of Grammars | 82 |
| B.3.4 | From Syntactic to Semantic Analysis | 82 |
| B.3.5 | Formal Languages Used by the MathletFactory | 82 |
| B.3.6 | Tree Architecture | 83 |
| B.3.7 | Basic Tree Model | 84 |
| B.3.8 | Object Model of Operations | 85 |
| B.3.9 | Object Model of Relations | 88 |
| B.3.10 | Applications | 89 |
| B.4 | Arithmetic and Geometric Symbolic View Architecture | 92 |
| B.5 | Algebraic Symbolic View Architecture | 93 |
| B.5.1 | View Architecture of Operations | 93 |
| B.5.2 | View Architecture of Relations | 94 |
| B.5.3 | Metrics of View Components | 94 |
| B.6 | Graphical View and Controller Architecture | 95 |
| C | Design of the Pyramit Platform | 97 |
| C.1 | Requirements | 97 |
| C.2 | Design and Implementation | 98 |
| C.2.1 | Data Model and Presentation Formats | 98 |
| C.3 | Update Cycle of Documents | 100 |
| C.4 | Internationalisation | 101 |
| D | List of Mathlets and their Parameters | 103 |
| D.1 | Generic Parameters for MathletFactory Applets | 103 |
| D.2 | List of Mathlets | 105 |

E Installation CD**131**

Introduction

‘I find myself forced to the conclusion that our survival may one day depend upon achieving a requisite mathematical literacy for rendering the seeming shocks of change into something that is continuous and cumulative.’

– Jerome S. Bruner

Mathematics and eLearning

The Modern Role of Mathematics

Being one of the driving forces for the rapid technological progress, the role of mathematics in society is more and more shifting towards application in almost any context. Mathematics allows not only the objective specification of natural and technical laws, but also their verification and publication in an increasingly distributed and diversified system of research and development. Mathematics may therefore be regarded as being not only the ‘language in which the book of nature is written’¹, but also as building a cultural foundation for the scientific and technological progress.

Mathematical Literacy

The computer era has reinforced this progress by making in-depth simulation and calculation software like computer algebra systems and numerical libraries widely available. These programs take over routine actions from the user, but in turn ask for a new quality of understanding.

For example, computer assisted engineering allows to replace costly experiments with computer simulations, making it possible to calculate parameter changes ‘on the fly’. But in order to validate the results produced by these calculations, a deeper understanding is required, what range of parameters fits the assumed model. Experience shows, that almost any calculation software can be forced to produce nonsense results by feeding it with nonsense (i.e. out of range) parameters. Unreflected use of these tools in construction may thus lead to disaster.²

This means that for vital decisions based on computational analysis, the ability to judge the

¹A saying already attributed to Galilei.

²A prominent example of blind faith in computer calculations leading to disaster is the collapse of the Hartford Civic Center Arena, [Ma99], [Pe85]. On the other hand there have also been disasters predicted correctly by computer simulations being rejected by engineers, like the explosion of the Columbia shuttle, see [Fl03].

adequacy of a mathematical model is as badly needed as the skill to roughly verify its results using simplified (e.g. first-order approximations) calculations and common sense. The OECD Pisa study covers both of these with promoting the concept of *mathematical literacy*:

*'The term literacy has been chosen to emphasise that mathematical knowledge and skills as defined within the traditional school mathematics curriculum do not constitute the primary focus ... Instead, the emphasis is on mathematical knowledge put to functional use in a multitude of different contexts and a variety of ways that call for reflection and insight.'*³

Learning by Doing

If we want to promote the reflected *use* of mathematics we have to transfer this practice-orientation to our didactic model: Learning must be accounted as a self directed activity. This is of course true for all subjects (and also the main axiom of modern learning theories), yet for mathematics it has to be stressed, since many students get stuck in passively reading books or hearing lectures and finally fail to reach the active part. By this they may never experience the reward of mathematics as a useful tool. The reason of this is, that the successful application of mathematics is based on a high amount of prerequisite knowledge and understanding, the acquisition of which often obstructs the path to its universal use.

The Role of eLearning in Mathematics

This is where eLearning comes into play: By providing *interactive* media, the student is given an opportunity to directly engage with mathematics without getting lost in calculations; by giving different representations of mathematical concepts, he may gain and reflect insights on different levels.

On the other hand mathematics seems quite well adapted to eLearning for it offers a consistent structure, multiple representations and a formal language that can be interpreted by machines. In addition to this, eLearning in general offers the opportunity to enforce the propagation of learning content when it is ubiquitous and barrier free, thus strengthening the democratic perspective of learning. This is especially true for the growing field of web-based learning.

Technical and Didactic Foundations of Web-Based Learning

With the WWW being only a little more than 10 years old, web-based Learning is a young discipline, still addressing major technical issues as well as didactic ones. In the technical field, some standards have been designed, some of them still waiting for their reference implementation. But as technical progress seems to be taking the leading role for inventions in eLearning, the didactic design of web-based learning is still very closely connected to (and mostly limited by) the technical capabilities. Therefore, the implementation of modern didactic concepts and scenarios requires a sophisticated technical base.

Learning (and therefore eLearning), on the other hand, is a social activity, its success is mainly determined by 'soft' factors like pedagogic concepts and methods of communication. Until widely accepted (and implemented) standards for didactic models of eLearning will be developed, maybe the most complex problem in eLearning is to resolve the interdependence of

³[OECD99]

technical and didactic issues, where the former require analytic skills and the latter defy them by their dialectical nature. This thesis proposes a solution, that tries to balance the technical and the didactic aspects in a practice oriented design process.

Goals

Analysing existing solutions in mathematical web-based learning (see chapter 1), reveals, that most eLearning sites offer too much static text and too few interactive learning materials. Additionally, the interactive materials offered are not flexible enough for reuse by other authors, thus making their development for a single purpose too costly. This again results from the fact, that there are no architectural patterns, which allow the mass production (more than 100) of interactive units.

We therefore come to the conclusion that mainly two threads of development have to be strengthened to enable web-based self directed learning that leads eventually towards mathematical literacy:

- The *interactivity* of mathematical eLearning has to be strengthened to encourage students to actively play with mathematics.
- The *flexibility* and *reusability* of mathematical eLearning units has to be ensured to encourage authors and teachers to create materials for students.

We will address these needs by implementing a framework that allows the rapid creation of flexible and reusable interactive units and evaluate it by the implementation and application of a large number of these.

The Mumie Project

This document is a product of *Mumie* (Multimedial Mathematics Instruction for Engineers)⁴, a project funded by the German Ministry of Education and Research (BmBF), whose aim is to produce a reusable eLearning platform for mathematics education of engineers. This platform consists of an authoring environment using a specified T_EX dialect for the construction of content (MMTex), an XML and Java based application server (JAPS) and a framework for constructing client side applications, the *AppletFactory*. While another work, [Je04], concentrates on the construction and description of the authoring and server framework, this thesis puts the focus on the client side applications, which are generated by two subpackages of the Mumie AppletFactory: The *MathletFactory* for creating mathematical applets and the *TestletFactory* for creating interactive tests.

⁴[Mu04]

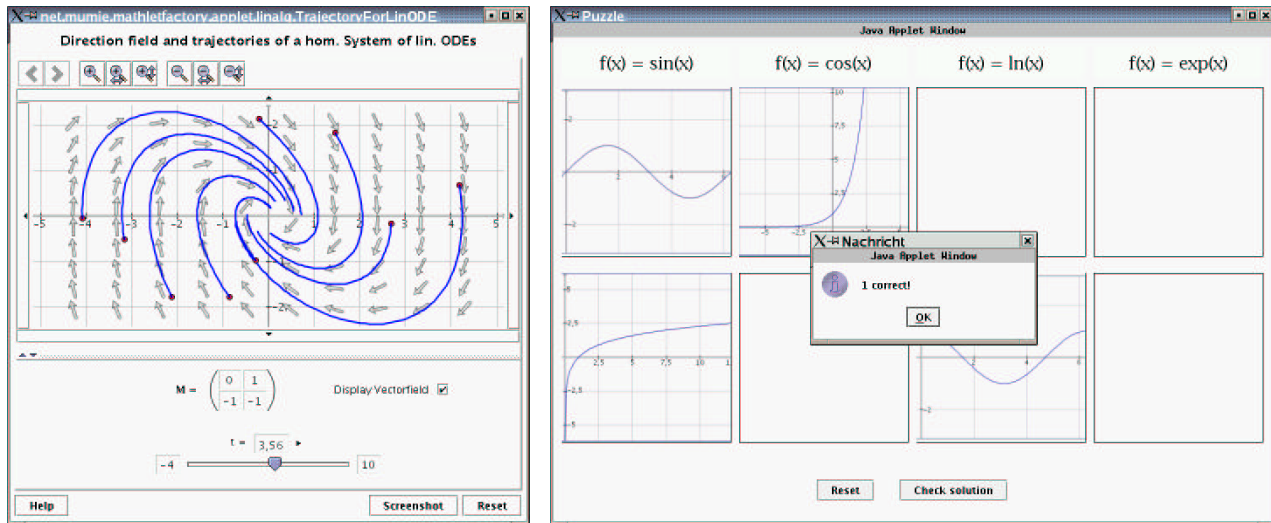


Fig. 1: A mathlet created by the MathletFactory and a puzzle testlet created by the TestletFactory

Though this framework was designed to fit in with the rest of the Mumie components, its reusability within any other server framework has been specifically ensured – even offline uses (e.g. publications on CDs) are possible. This corresponds to the public domain and open-source approach consequently followed throughout the whole Mumie project.

Organisation of this Document

This thesis is the documentation of a complete evolution cycle in the process of constructing a mathematical eLearning site. According to the usual models (such as described by the Unified Process Model⁵), a software development process generally consists of four phases dealing with the requirements analysis, design, implementation and evaluation/testing of the software. Taking into account the considerable work already performed in the field of web-based mathematical eLearning, we regard the process described in this document as a single iteration cycle (a macro cycle) within the evolution of mathematical eLearning site development. We will therefore pick up successful concepts from existing sites, try to improve them – technically and didactically – and make them available and reusable for the next generation of mathematical eLearning. Since this methodic model also applies to each iteration within our own project (a micro cycle, where we improve those parts of our framework that are most needed or promising) the complete process is best described by an evolutionary and object-oriented process model, like the EOS-Model⁶. This model may be regarded as an object oriented improvement of the Spiral Model⁷ and differs from the traditional Waterfall- and V-Model by its adaptability to specification changes (which happen to occur frequently in practice-oriented software design).⁸

⁵[Sa02]

⁶[He02], [He98]

⁷[Bo88]

⁸[Ke98], [GS02]

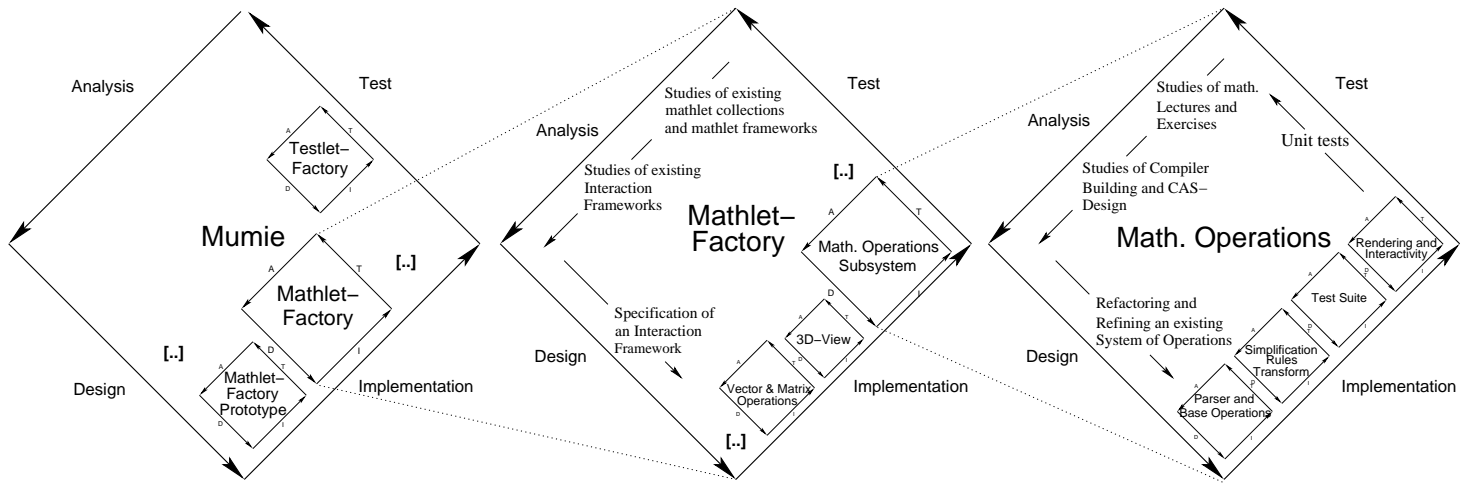


Fig. 2: Developing the Mumie AppletFactory as an evolutionary and object oriented process

This document is thus organised as in the following figure: Each chapter connects two phases of the macro cycle, describing the steps that lead from the phase pointing to it to the next phase.

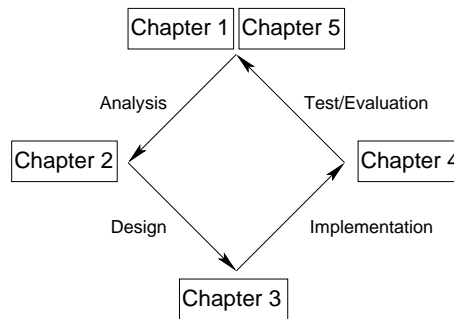


Fig. 3: The chapters of this document in the evolving process

After discussing the specifics of web based learning, we start with the practical evaluation of existing eLearning sites in Chapter 1. This evaluation reveals, that current mathematical eLearning sites mostly offer only isolated spots of interactive content; a fact that results from the absence of a systematic approach for interactive client side applications and from a common disregard of the complete set of stakeholders (namely the authors) involved. To lay the foundation for the solution presented in this thesis, in Chapter 2 a didactic model for interactively learning mathematics is introduced which may serve as a generic process framework for the production of learner-oriented and modular learning content. Together with the practical results of Chapter 1, this model can be used to specify the requirements for sustainable improvements in mathematical eLearning.

The basic concepts of a component framework that meets these requirements are described in Chapter 3. To prove the stakeholder-oriented approach of our solution, we present the component framework from each of three different perspectives (the student's, the author's and the application programmer's perspective) and illustrate its user-specific interfaces. It can be seen, that the process of developing and configuring applets with the framework is simple and flexible. This is also demonstrated in Chapter 4, where we give a wide range of application scenarios and

discuss the improvements and lessons learned following their evaluation. Moreover, the chapter provides a look at the ‘big picture’ of developing, maintaining and extending interactive content for mathematical eLearning sites. To demonstrate its impact on eLearning in general, we also present the transfer of the framework to other fields of use (school scenarios, integration in server frameworks, offline use) and to other fields of science (physics, engineering). A summary of the results of this thesis and an outlook to further extensions is given in Chapter 5, thus completing this cycle and initiating the next one, which might consist of integration steps towards a comprehensive mathematical eLearning/-Teaching/-Research (eLTR) infrastructure as well as towards a converging university-wide eLearning landscape.

Chapter 1

Technical Concepts and Existing Solutions

In this chapter we illustrate the current state in mathematical eLearning by presenting two prominent example sites. We also take a closer look at the specifics of web-based learning.

1.1 eLearning Sites

The need for web-based solutions in mathematics instruction has been recognised by many institutions. For example, the Austrian government funds an international project that aims to construct a website which covers all mathematical subjects from school to university level with interactive examples. A different approach is taken by the Massachusetts Institute of Technology (MIT): It publishes an online version of all its university courses (not only that of mathematical content) in an highly acclaimed open courseware program.¹ In Germany, a wide range of university eLearning projects have been funded by the government in a large scale initiative to improve academic learning. More than six of these projects deal with instructing mathematics.²

To examine the state of these efforts, we will analyse two exemplary mathematical eLearning sites, concentrating on best practice solutions. However, in order to perform this analysis we first need to state the concepts on which we concentrate, making it necessary to take a closer look at the specific structural, technical and social aspects of web-based eLearning.

1.1.1 Specific Structure of eLearning Sites

There are many diverging categorisations of eLearning software.³ Most of these adhere to *didactic* aspects like the cognitive level of the learning content or the tutoring capabilities of the control mechanisms. Unfortunately, for the existing eLearning sites these categorisations do not seem to be appropriate, because their stage of development is still in the phase of solving the technological difficulties, leaving their didactic features yet quite undistinguished.

We therefore use a rather practical categorisation scheme that focuses on the basic technical

¹[MIT03], [NY01], [Mo02], [Ze03]

²For a complete reference see [BmBF04].

³e.g. [Blu98],[Sch02]

and social aspects of an eLearning site and leave didactic considerations for Chapter 2.

Before we specify these aspects, we have to take a look at the technical constraints that make web-based eLearning different to other forms of eLearning:

From the technical perspective, an eLearning site is simply a web server sending HTML content and multimedia objects to a requesting user. For further analysis, we will adopt the common structural view that distinguishes between *macro level* (server platform along with static content and navigation unit) and *micro level* structure (single page content and multimedia objects). For eLearning sites there is an additional logical substructure consisting of courses and course sections. As it makes sense to associate these to the server framework (as subdirectories or hierarchical database tables), the structural levels for eLearning sites are usually as follows:

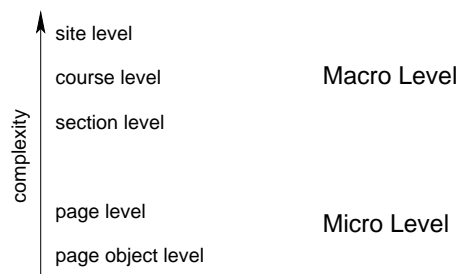


Fig. 4: Structural levels of an eLearning site

Macro Level

On the macro level the system consists of a web server that operates on a content base (files or database) using a navigation structure and a global presentation style. For eLearning sites the macro level structure of the system is closely related with the ‘global’ settings, i.e. organisation and presentation of courses, user tracking, navigation, etc.

Micro Level

On the micro level the system consists of the single page delivered upon a user request along with different media the user might work on to achieve the actual learning progress. For eLearning sites this is often closely related to the content provided by the system. Typical micro level applications are executed on the users computer (e.g. Java applets, Flash programs, interactive PDFs, etc.)

When speaking of certain aspects of eLearning sites, we often have to differentiate between these levels. This is especially true for one of the main topics this thesis is about:

1.1.2 Interactivity

Interactivity is a key concept for eLearning as well as for multimedia in general, since it allows a variety of learning activities that are impossible to realise with conventional media.

Because interactivity connects the technical perspective with the didactic one, we will present a

deeper analysis in Chapter 2. For our practical analysis it is sufficient to regard the interactivity of a multimedia system as a measure for the quality and quantity of message loops in which a user finds himself when working with it.

With respect to eLearning software [MO91] offer a classification of interactivity in terms of *navigability*, *adaptivity* and *reactivity*:

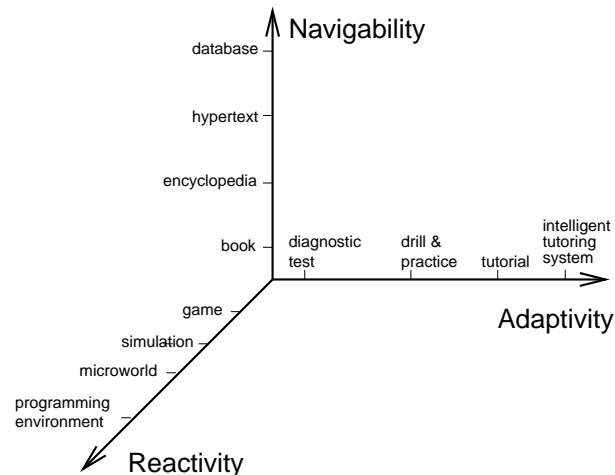


Fig. 5: The interactivity space of Midoro/Olimpo et al.

In this classification, navigability is defined as ‘the capability of retrieving the proper material from a bank of learning material’, adaptivity means ‘the capability of evaluating and executing a set of functions available to the user’ and reactivity is ‘the ability of an instructional system to adapt its behaviour according to the user’s behaviour’.

This means, for example, that conventional learning with a mathematical textbook would have a rather low interactivity (navigability: Low – very often sequential reading is required, a little increase by cross referencing and offering an index; reactivity: Almost none, maybe except the possibility of writing annotations into it; adaptivity: Only little by offering exercises with solutions), whereas a usual Computer Algebra System (CAS) worksheet would have a medium interactivity resulting from a very high reactivity (by offering a programming language), combined with low navigability (no hyperlinks) and low adaptivity (only syntactic errors are reported, no semantic checking by the system).

In general (and as shown by the examples discussed in section 1.2), the navigability of web-based eLearning has already reached a high level (though there are many hypertexts in the web that adopt rather the structure of a book or an encyclopedia), whereas adaptivity and reactivity still remain on a basic level.

eLearning Sites and Interactivity

For further analysis of eLearning sites it makes sense to relate this interactivity categorisation scheme with our structural model and take into account the potential of web-based technology:

| | micro level | macro level |
|--------------|-------------|-------------|
| navigability | low | high |
| adaptivity | medium | high |
| reactivity | high | low |

In this table ‘high potential’ means that an implementation on the specified level is able to provide features that enhance the specified dimension of interactivity, for example, it is easy to enhance the adaptivity of an eLearning site by adding logic to the server framework, since user specific parameters can easily be saved on the server. Single pages and applets only have transient states and are normally not allowed to save data on the client computer. In turn reactions of the server framework can only be initiated by a HTTP request, which always induces the transfer of a file over the internet, slowing down actions below the usually accepted response rate⁴, so reactivity has to be provided mainly on the micro level by using reactive client applications.

1.1.3 Stakeholders and Reusability

Since the web is first and foremost a medium for communication, the role of each participant is a decisive factor in the design and maintenance of websites. In this thesis we therefore regard all social aspects for eLearning sites as the properties and interests of different *stakeholders*. As the following analysis of the example site shows, the reusability of an eLearning site relies strongly on the inclusion of stakeholders into the design process.

Stakeholders in eLearning: A Role Model

eLearning usually takes place not only in a technical environment but also in a diverse social context (e.g. schools, universities). This makes it necessary to analyse the roles of the participants in the specific eLearning scenario.

We can compare the evolution of eLearning with the evolution of education as a whole: Education started with the role of the teacher and a student (e.g. a sophist teaching a young man). Today the role of the student remains almost unchanged while the role of the teacher has diversified into many roles like textbook authors, lecturers, exercises managers, tutors, examiners, etc.

The situation in eLearning is alike: The ‘big brother’ experimentator role from early computer based training experimental studies has diversified with growing system complexity into different roles determining the eLearning scenario. Up to now large scale settings require authors, tutors, developers, administrators, etc.

The roles allocated by an eLearning system differ with its type: Standalone learning programs exclusively address the *student* as a role, larger web-based learning systems additionally provide an *author* role: Someone who creates content with an authoring tool or in a simple markup language (e.g. HTML). To these two roles, blended learning scenarios (i.e. eLearning combined with face-to-face learning) add the role of a *tutor* who watches and supports the interaction of the student with the content.

In practice author and tutor are often the same person (e.g. schools, small university departments), we call this combined role the *teacher*.

⁴[Cr02] recommends that at least 50% of all actions performed in an interactive application should have a response time below 0.5s – a task that is hard to achieve for HTTP applications using low band-width connections.

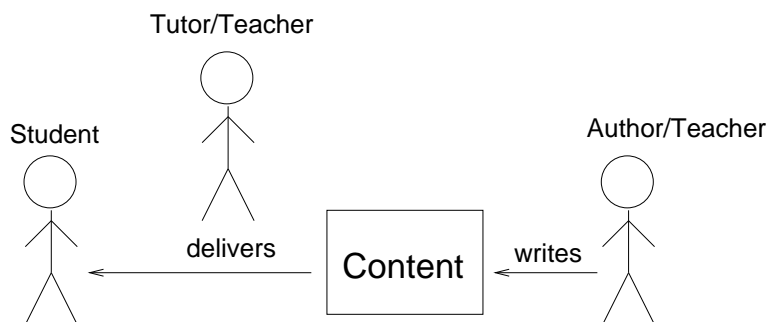


Fig. 6: Minimum role model for a blended learning scenario

Since the goal of our research is to provide a framework that can be used for a wide range of learning systems, we need to consider at least the three elementary roles *student*, *author* and *tutor*.

1.2 Analysis of Existing Mathematical eLearning Sites

In the following we will look at an existing mathematical eLearning site in order to demonstrate the current state and conclude the necessary steps of improvement to be taken.

We will take a look at the – at the time of writing – most prominent mathematical eLearning sites, *MIT OCW* and *maths online*.

1.2.1 MIT OCW

The MIT Open CourseWare System is an eLearning site that offers contents of all MIT's courses, not only mathematics. It describes itself as '*a large-scale, Web-based electronic publishing initiative ... to provide free, searchable, access to MIT's course materials for educators, students, and self-learners around the world*'; in our words this means, that the site aims at blended learning as well as pure distant learning scenarios.

From a structural perspective, OCW is rather a macro level solution, delegating micro level considerations to the individual lecturer. This results in a collection of diverse course materials that vary highly in quantity and quality, using a wide range of media types, from commercial software files to video lectures. Of course it is not possible to analyse this material as a whole – not even for mathematics alone; we therefore take a brief glance on the courses 'Calculus with Applications' and 'Linear Algebra', which cover the standard mathematics that every science and engineering student learns in his first terms. We regard these courses under the perspective of interactivity and the stakeholders involved.

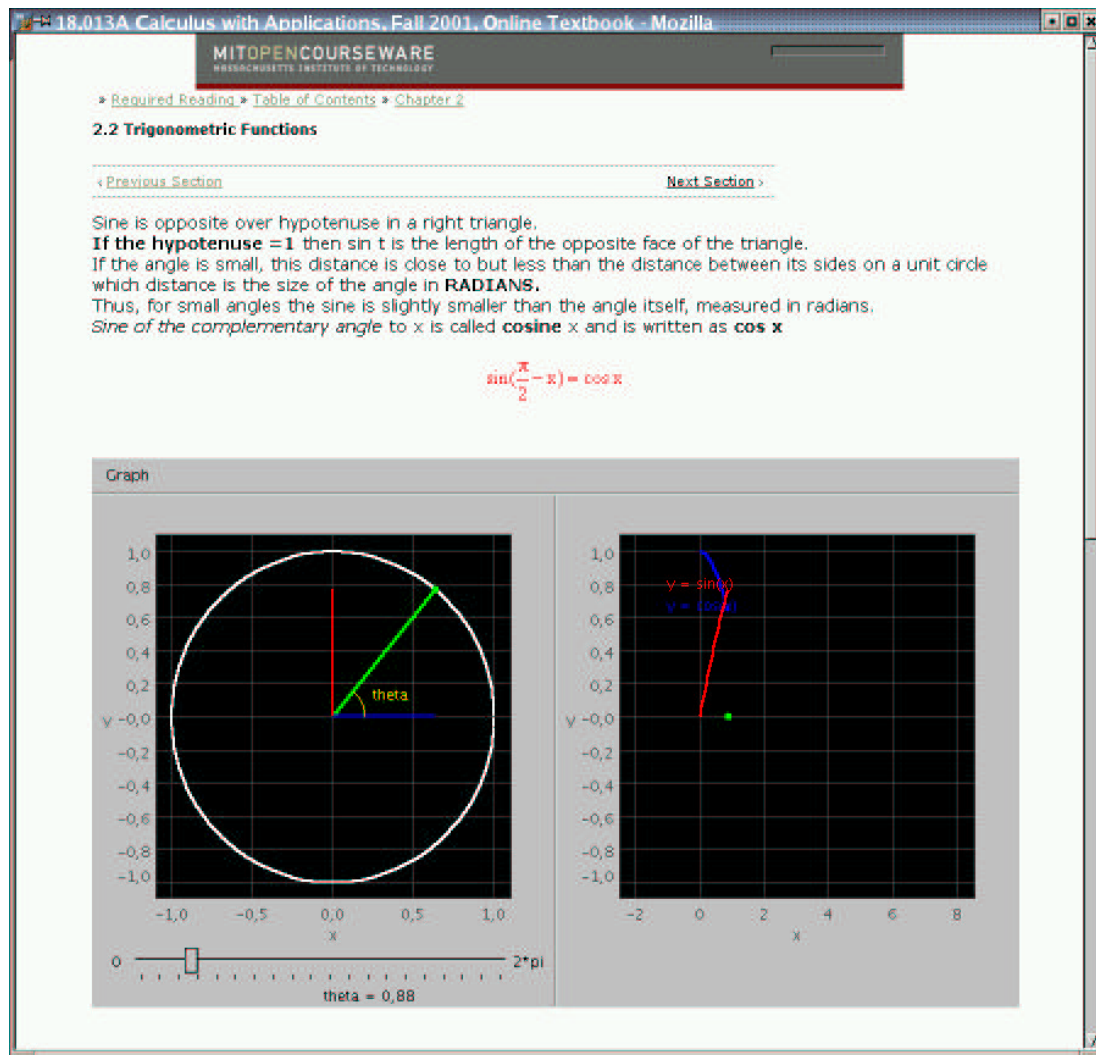


Fig. 7: A page of the analysis course's online textbook in MIT OCW

Interactivity

Since the primary goal of MIT OCW is to make their (sometimes already existing) course materials available online, their structure is still tied to that of textbooks and lectures: The 'Calculus with Applications' course is mainly organised as an 'online textbook' with a linear structure with almost no further navigability. The 'Linear Algebra' course offers in addition to the usual assignment and exam sheets (as PDFs) a link to the book written by the lecturer and a collection of video lectures. This means that the navigability on course level is only marginal above that of conventional media (one is able to search the content, but this applies only to HTML and PS/PDF materials).

In terms of adaptivity there are some assignments with solutions, varying on the specific course (Calculus: questions in the text with links to solutions, Linear Algebra: Problem set PDF sheets with corresponding solution sheets).

Regarding the reactivity of the materials, the calculus online textbook offers an advantage over conventional media by providing more than 20 mathematical applets (spread among the about 180 pages of content). Though named as 'tools' they are each made for visualising a quite

specific topic, allowing the user little interaction with the mathematical concepts presented. A short look onto the other courses with mostly no applets (or any other interactive applications) at all reveals that the chosen example already provides an outstanding amount of reactivity within the OCW materials.

Stakeholders and Conclusions

The stakeholders addressed by MIT OCW are obviously foremost the student and the author/teacher, no explicit support for other roles is offered. The reason for this might be, that (other than most modern universities) the MIT has a low students-per-teacher ratio, allowing small team sizes for the teaching staff.

This almost ‘standalone’ role of the teacher gives him freedom to arrange learning materials to fit his individual course structure on the one hand. On the other hand we observe the static nature of most of these materials, which results from the fact that it is hard to expect additional programming skills from a teacher (who is by the way also occupied with research, publication, organisation of meetings, etc.). This leads us to the thesis, that for creating interactive mathematical eLearning media, the author role should be augmented by addressing an additional *developer* role.

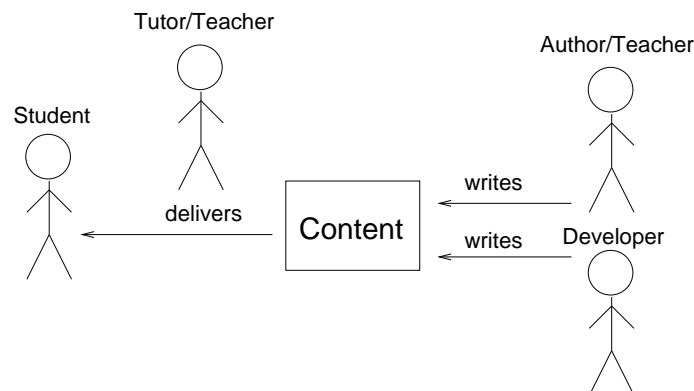


Fig. 8: A developer is needed to assist the author in producing highly interactive content

We will now take a look at an eLearning site, that provides an increased amount of interactivity due to the obvious work of developers.

1.2.2 maths online

maths online is a project based at the University of Vienna, Austria, and is running since March 1998. Its goal is ‘*to contribute to the development of adequate standards for up to date maths education in school, highschool, college, university, and adults’ qualification.*’⁵

A closer look at the site will reveal that these standards concern mainly didactic and content issues. For its authors, the site is characterised by the following elements:⁶

- A section of *mathematical backgrounds* that is regarded as the core element, ‘from which the whole content material may be overlooked and accessed’. These elements are large

⁵[MO03]

⁶[Ob98]. Some of these elements are currently available only in the German version of math online.

pages holding a linear structure with hyperlinks to other – internal and external – pages. There is also an *encyclopedia* for mathematical concepts linked with the backgrounds.

- A *gallery* that puts the focus on interactive multimedia units, mostly applets.
- A collection of *interactive exercises* and *tests* that consists mainly of graphical puzzles and multiple choice test pages. Also a *discussion of common mistakes* is included.

Again, we apply onto these the criteria specified in 1.1.2 and 1.1.3.

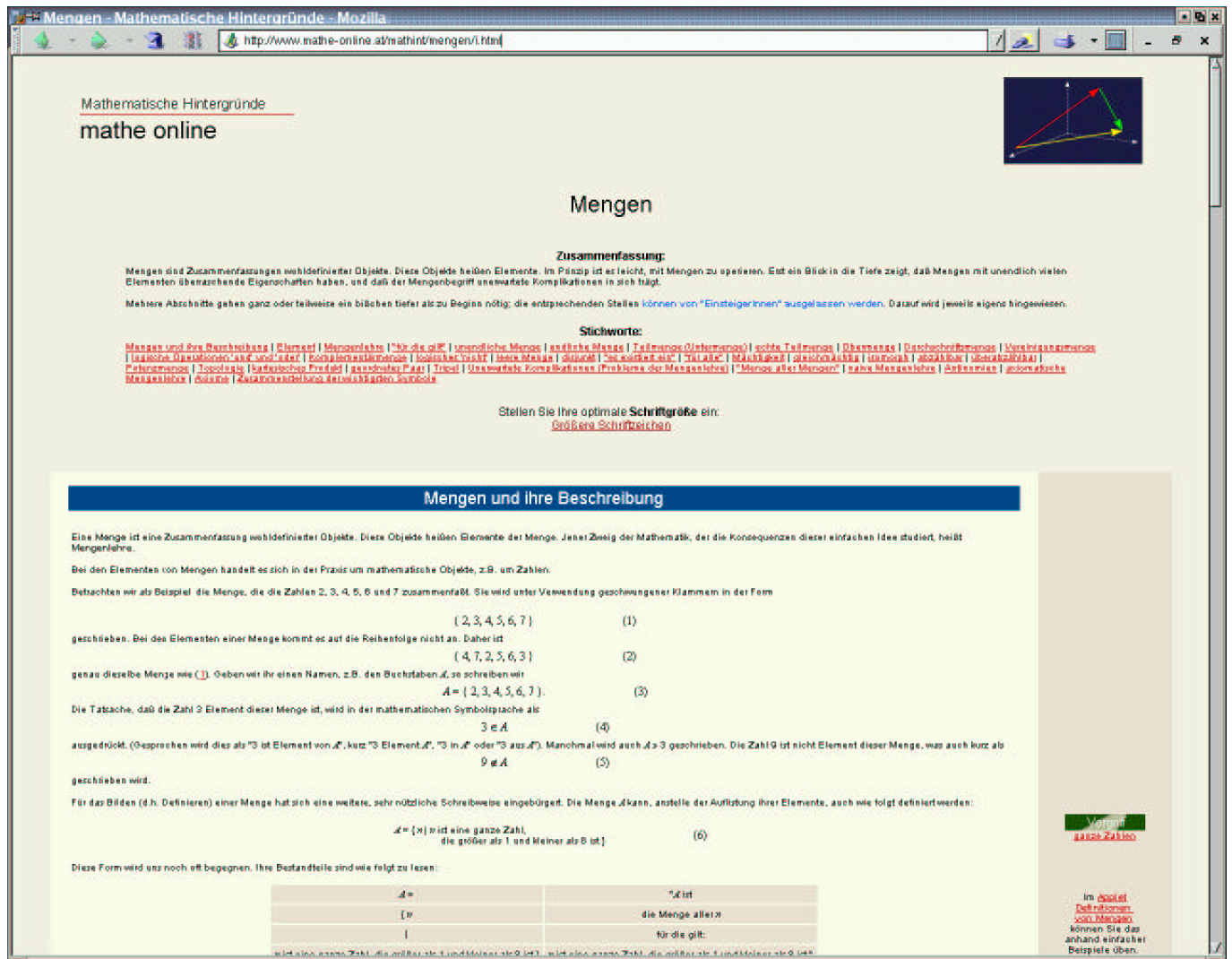


Fig. 9: A section of math online's mathematical backgrounds (note the length of the page indicated by the scrollbar)

Interactivity

One can see that each category of elements listed above deals with the dimensions specified by [MO91]: The elements *mathematical backgrounds* and *encyclopedia* stress the concept of navigability, the *gallery* gathers examples of high reactivity and the *interactive exercises* determine the site's adaptivity.

Navigability While the body of the content presented in *mathematical backgrounds* resembles the sections of a book, a multitude of internal and external links has been added, allowing the user to jump to previous definitions, further explanations or linked topics. Compared to other web pages the single documents are very large (up to 500-1000 lines) and will surely be hard to read for a beginner in mathematics. This raises the question if the advantages in navigability outweigh the difficulties of onscreen-reading compared to using a mathematical textbook, when working with the site.⁷

Reactivity In the gallery math online provides (at the time of writing) 53 Java applets dealing with various mathematical topics. These applets are worth a closer look, since their role as a ‘key concept’ has been confirmed by all evaluation results (see page 24).

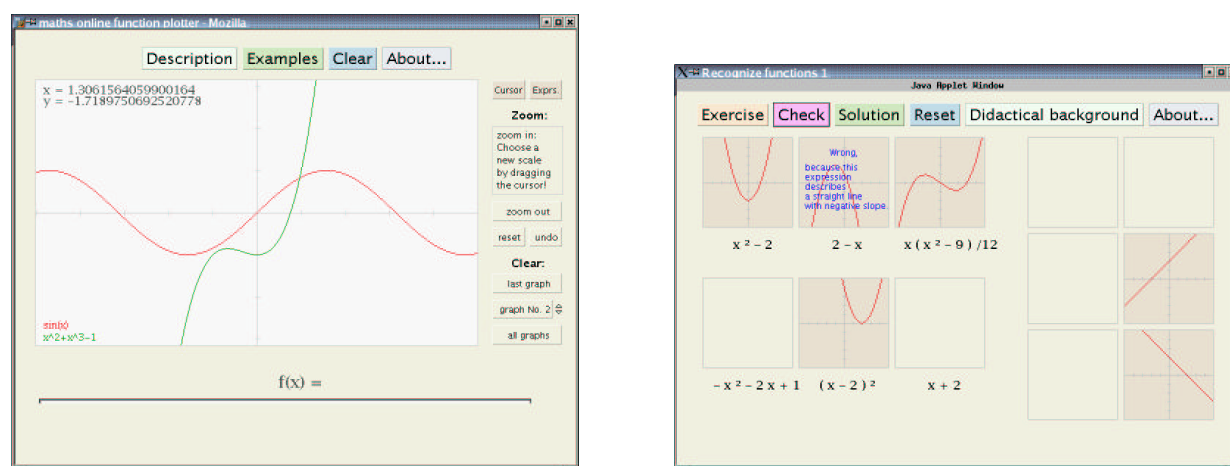


Fig. 10: A function plotter mathlet and a puzzle applet from maths online

The applets can be categorised into those that offer a visual and interactive interface to a specific mathematical topic and those that allow the user to relate different (visual or textual) representations of mathematical entities by drag & drop (see figure 10).

The latter are often called ‘puzzles’, the former comply with the definition of a *mathlet* issued by the Journal of Online Mathematics and its Applications⁸:

A mathlet is a small, interactive, platform-independent tool for teaching math – the equivalent of a good example that you want to haul out, give (or show) to your students, and let them go explore. Some will be more general purpose and of broader use, but the basic idea is that they should be simple to explain and to use.

Adaptivity maths online offers puzzles and multiple choice tests as self assessment programs for the user. Both give feedback in ‘points’ when the ‘evaluation’ button is clicked. The points are not recorded, so it is up to the user to keep track of his learning success.

⁷The authors of math-online also list this as an acceptance problem in their experiences report [Em99].

⁸[JO03]

Stakeholders and Reusability

Originally, maths online was first designed as a CD-Rom standalone application addressing only the role of the student. Over time this aim changed to a distant learning system adding authoring-functionality. Also, some other projects involving blended learning scenarios analysed the role of a teacher in maths online. This gradually increasing range of stakeholders addressed by the system may explain the rather poor support for the author role:

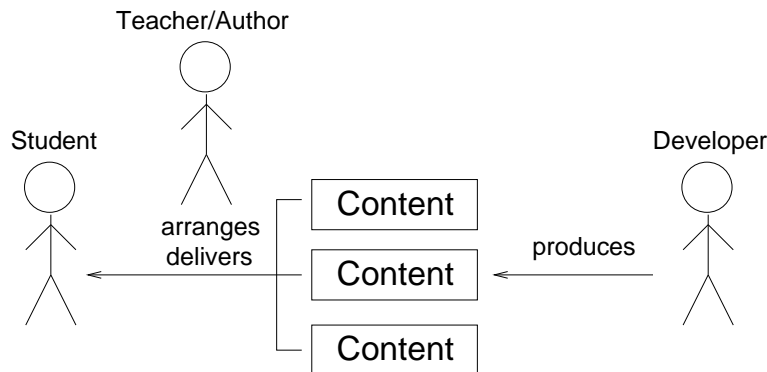


Fig. 11: Stakeholders in the maths online authoring system

The authoring system is designed for linking prefabricated content pieces of the system – thus an author is not supported in creating his own content pages. Instead he may only add comments or exercises to existing learning content, reducing the pages written by authors in many cases to the one containing the links to pages of maths online. Also, most of the mathlets provided by the system are insufficiently configurable. This approach was also criticised by the teachers:

Evaluation Results

maths online has also been chosen as an example for an existing site because of its considerable amount of empirical results.⁹

The project leader Franz Embacher devotes a large part of his report to the role of the teacher. After describing a mathlet that visualises the derivative¹⁰ of a function he states:

However, several teachers asked for the possibility to vary the graph displayed. They did not primarily look at the applet as a visualisation of a concept but as a tool, to be used in various situations. They did not seem to accept intuitive understanding as a primary goal of a learning unit, but were more focused on tools suitable to do something, and to learn by doing.

This need for tools does not only affect mathlets but also puzzles, for which a text-only solution¹¹ is offered:

There was a further series of reactions indicating that teachers wish to have tools at their disposal, even if the adoption thereof for their lessons will cost a considerable amount of time.

⁹The most extensive evaluation can be found in [Os00], the authors sum up their experiences in [Em99]. External evaluations can be found in [Do02], [Kl02]

¹⁰<http://www.univie.ac.at/future.media/moe/galerie/diff1/diff1.html#ableitung>

¹¹<http://www.mathe-online.at/testpuzzle/testpuzzle.html>

... A frequent reaction of teachers was the need for customising their own puzzle applets. We have provided a comfortable possibility to do so (input of the entries by web page).

This corresponds to the way teachers utilise mathlets and puzzles in their own lessons:

The overall approach of teachers towards the applets was seemingly to integrate them as individual units into their courses, separated from the rest of "maths online". Maybe this is a reaction for individuals responsible for their educational activities. So far, we could not verify the well-known phrase that teachers just want material whose use is strictly ruled. (On the other hand, there are already materials in use which do not leave much room for customisation – the maths textbooks. Maybe the situation bears some slight competition between book and web). This reaction may be appreciated – nevertheless, it shows that suggesting a different pedagogical approach of, say, a complete chapter as a whole is a rather difficult enterprise.

In our words this means, that maths online mistook the role of the teacher as that of a mere tutor, depriving it of its authoring capabilities. But inventing the author role would lead to a quite different technical environment: To allow authors to write their own lessons, the content structure needs to be finer. This leads in practice to a shift towards micro level applications ('tools') that can be easily integrated into any course constructed by the teacher. For an eLearning site this means that it should offer a collection of fine grained reusable content objects, like the demanded mathlets.

The favoured interest in applets is also strongly backed by the evaluation performed by Ossimitz¹² in the European/Austrian project 'mathe online im zweiten Bildungsweg' (an adult education study using maths online in a blended learning scenario): 88% of the students interviewed after the course said they were most positively impressed either by the interactive tests or the mathlets, the teachers in turn estimated that more than 70% of their working time with maths online was used for working with mathlets and interactive tests.

1.2.3 Conclusions and Theses

What are the technical solutions that could improve mathematical eLearning for sites like maths online and MIT OCW? In this document we concentrate on the following issues:

1. Improving Interactivity with Micro-Level Applications

The evaluation results of the previous section lead us to the thesis, that a sustainable improvement of interactivity in web-based learning can only be achieved by extending the capabilities of applications on the micro level. This requires the introduction of a developer role that acts as an assistance of the author role.

2. Improving Reusability by Strengthening the Author Role

Why are the widely appreciated concepts of math online not used by other sites and why has its content not been extended over the last years to meet the specified needs? It seems that the standards set by maths online were mostly on the didactic level, but the technical standards – if there were any – were not published to potential authors willing to follow suit. Therefore,

¹²[Oss00]

the author role as a transmitter between developer and student was neglected, reducing the role of the teacher to that of a tutor that had only marginal influence in the content presented (i.e. he could only choose which sections to present to the students).

1.3 Towards a Reusable Component System

1.3.1 Technical and Conceptual Reusability

Though almost all results of this thesis can be reused on a conceptual level, our aim is to develop a system that is also *technically reusable* for a wide range of applications. This means that an author of an eLearning site may use the system and configure it at his wishes. This configuration can be done on various technical levels:

1.3.2 Levels of Technical Reusability

If we wish to reuse portions of an eLearning site with applets in general, we can do this on each of the structural levels defined in 1.1.1.

Moreover, we have seen in 1.2.2 that there is a specific need for flexible micro level applications in mathematical eLearning. For applications written in an object oriented component based language (such as Java) we thus need to extend the structural view below page object level in order to increase the potential for reuse:

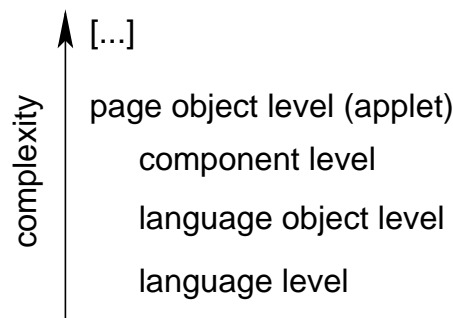


Fig. 12: Refining the structural view for applets (cf. Fig. 4)

This approach is especially promising for mathematical eLearning sites, because it is backed by the fact that mathematics offers a highly formalised language which is independent of specific contexts (see 2.5.2) and that its concepts can be visualised and interacted with in a somewhat standardised way (see 3.3.2).

To demonstrate the practical effectiveness of this approach, we will provide a prototypical author platform that embeds the applets and that also offers reusability on page level (see section 4.2).

1.3.3 Stakeholders and Reusability

For a project aiming at the support of various mathematical eLearning scenarios it is required to specify the roles that could possibly participate. We therefore include all scenarios that

include a subset of the following roles:¹³

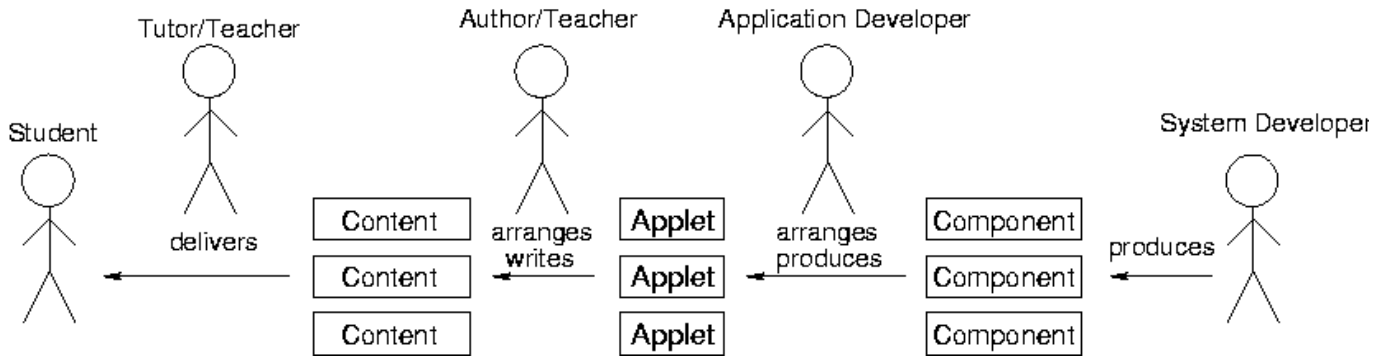


Fig. 13: Possible roles in our eLearning scenario

In order to achieve reusability, we have to state, how a specific stakeholder may reuse the system. This is already implied by the assignment of roles in figure 13: Each stakeholder reuses content entities on the structural level(s) he is working on:

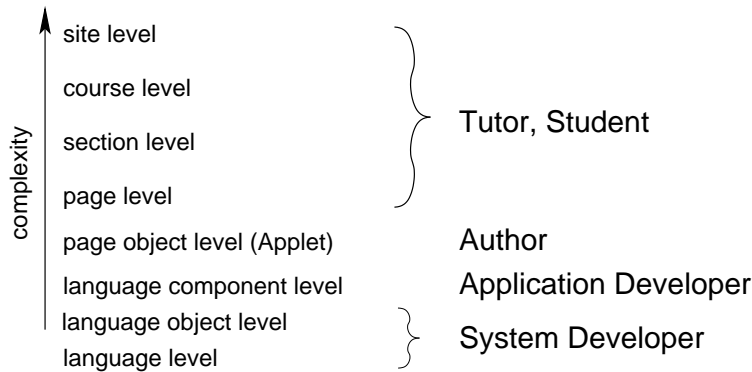


Fig. 14: The structural levels on which stakeholders reuse the system

1.3.4 Existing Reusable Frameworks and Scalability

The need for reusable frameworks in the production of mathematical applets has already been recognised by various institutions, some have even produced solutions for this need: The American National Science Foundation (NSF) funded two projects for the development of a mathematical Java component framework – Java Components for Mathematics (JCM)¹⁴ and Math-ToolKit¹⁵.

As these frameworks were obviously designed for small collections of mathlets (and are technically somewhat out of date) we only give a brief description of them (for further information please refer to the associated web sites).

The two component systems both offer a collection of function types, including a small parsing

¹³Remember that there is no obligation for a person to be limited to a single role, in fact almost all members of our team had multiple roles

¹⁴[JCM01]

¹⁵[MT01], see also [Y01]

unit and 2D graphical representations for calculus and differential equations. One can see from the number of mathlets presented at the associated web sites (not much more than 10), that these frameworks were not intended for scenarios we are aiming at (more than 100 mathlets). In addition to the listed features we need at least representations of linear algebra entities (vectors, matrices, etc.), a generic graphical interaction and navigation system and a 3D representation system. We will present these and other features of our component system in Chapter 3.

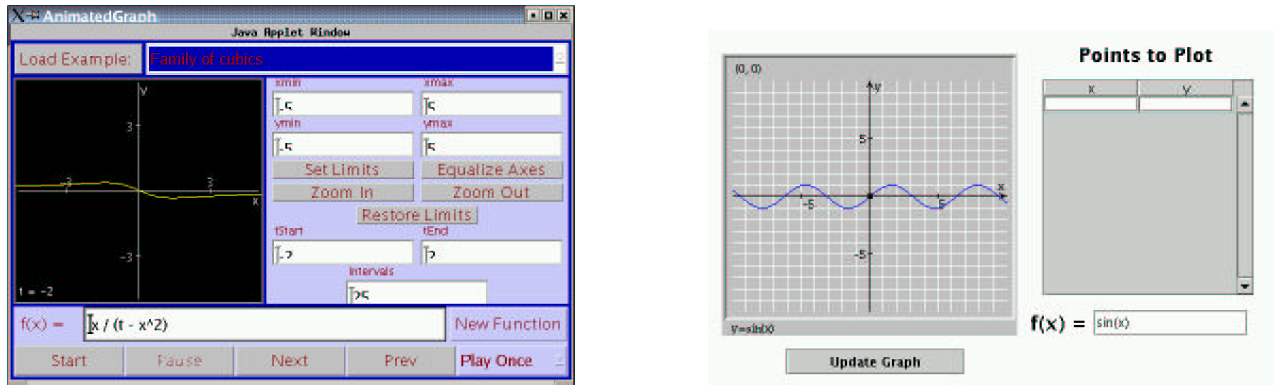


Fig. 15: Examples of mathlets created with the JCM (left) and MathToolKit (right) framework

1.3.5 Need for Didactic Design Process

In the example sites we have seen that the didactic perspective is often superseded by technical constraints: The amount of (didactically desirable) mathlets, was kept low due to the costs of development, the author role was neglected due to problems in delivering an authoring environment. This in mind, we need to carefully analyse the didactic requirements to set up appropriate and useful eLearning scenarios. We do this in the following chapter by taking a closer (specific didactic) look at the concept of ‘interactivity’. This will lead us to designing a component system that allows the construction of didactically helpful applets.

Also, in terms of reusability we need a didactic model that tells us, which parts of a mathlet or lesson are invariant under various use cases (should be reusable) and which parts are specific for a certain use case (should be replaceable). We will return to this issue in 3.3.2.

Chapter 2

Didactic Design

After having examined the technical and the practical aspects of web-based eLearning, it is time to complement these with a didactic perspective that shows, how learning as a subjective activity can be modelled and how eLearning media should be designed to encourage and strengthen learning. We therefore first present a meta-model for the didactic design process and then take a look at theories that might serve as a model for different learning scenarios.

2.1 Didactic Design Process: A Meta-Model

As didactics is made of a wide range of different theories there is also a wide range of didactic processes for the preparation of instruction units. Most of these processes provide a detailed scheme how to take into account all dependencies, like previous knowledge of the learner and social-cultural relations of the subject matter, etc. Aiming at a simple process that guides the development of eLearning media, we start with what [MJ91] call the ‘magic formula’ of consensus of all didactic models and processes: The fact that almost any didactic theory regards an instruction unit under the complementing perspectives of the specific goals, the content taught and the methods used. All these perspectives depend on each other and on the general aims of the instruction.

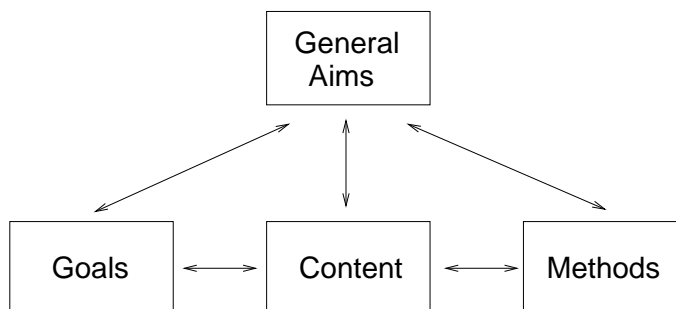


Fig. 16: The fundamental entities in a didactic process and their relations

According to this meta-model, the steps in specifying our didactic process would be to state the general aims of teaching students mathematics and derive from these the necessary contents, methods and specific goals for each learning unit.

But since our aim is to provide a reusable system that should serve a wide range of different target groups and learning scenarios, we cannot state any aims other than that the subject (mathematics) should be taught and presented in a wide range of different contexts. We therefore first have to take a look at how learning can generally be understood by means of a comprehensive learning model. This must be a theory that takes into account the practical results of the previous chapter (e.g. the prominent role of interactivity).

After discussing an appropriate learning model, we will present further components to be used in our didactic process.

2.2 Learning Model

2.2.1 Static and Dynamic Learning Models

Historically, the prevailing concept of learning regards the learner basically as a (more or less complex) input/output machine which processes stimuli by giving according responses.¹ The learning progress was then ensured by comparing the output with the prescribed learning goals. The result of this comparison was fed back to the learner, thus creating a cycle.

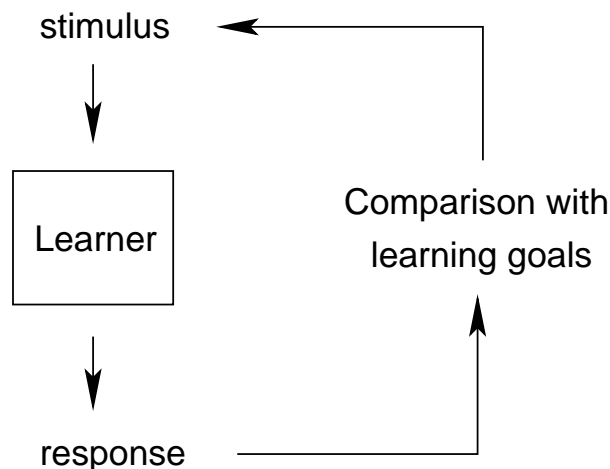


Fig. 17: Traditional (static) learning model

The deficiency of these models was, that they assumed the existence of an objective theory stating how anyone can learn anything in any situation. As a result, the associated theories concentrated on the static entities of learning (curricula, books, tests) without taking into account the factors of learning that in practice have been the most successful: Communication and interaction.

Along with the rise of large scale empirical studies, didactic theories shifted more and more towards evaluating the interaction of learner and learning entities. Though the foundations of the 'objective' theories (behavioural biology and cognitive psychology) were not denied, their lack of providing explanations for practical failures was criticised. This led to a paradigm shift towards theories that focus on describing the dynamics encountered in practical situations.²

¹This is especially true for eLearning, see for example, [Sch02], [Th99].

²[MJ91]

The most striking paradigm that evolved from this process is the concept of constructivism.³ The main difference to traditional learning theories contributes to the inability to objectively measure subjective and social properties. In terms of education this means, that learning should not be regarded as a clearly definable standalone activity of information absorption, but as a dynamic process that is embedded in multiple subjective and social contexts.

For the design of learning media it thus makes sense to focus on its possible uses in a dynamic learning scenario. For eLearning media this brings us back to the concept of interactivity.

2.2.2 Interactivity: A Constructivistic Approach

In his comprehensive structural definition of interactivity, [Ya99] puts the *interaction loop* at the centre of learning:

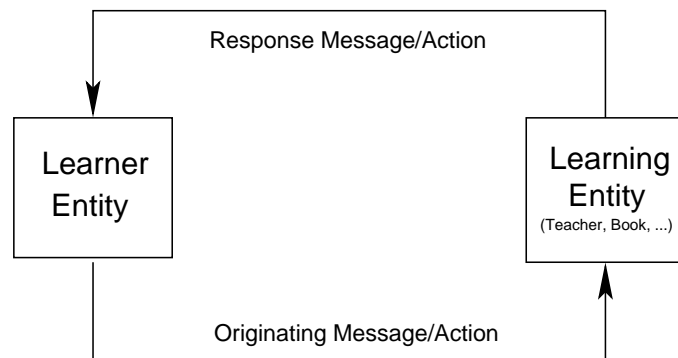


Fig. 18: Dynamic learning model: The interaction loop after [Ya99]

According to this model, a learning person (the learner entity) finds himself in a loop with a learning (or teaching) entity, which might be a teacher, a book, a computer, etc. The learner communicates and interacts with this entity, learning gradually by iteratively completing more or less semantically rich interaction cycles. So what are the differences to the static learning model in figure 17? Primarily, there are no more external learning aims or goals; in contrast, the goals have to be stated by the learner him- or herself and depend on the chosen subject matter. In order to use a dynamic learning model, we therefore will have to modify the meta model in figure 16 for our didactic design process (see 2.5.1).

Yacci stresses, that the interaction between learner and content can be observed within two domains: the cognitive domain and the affective domain. The learner can thus be regarded as having two distinct subsystems that each use a different channel for communication. The clear distinction of these two subsystems is backed by the fact, that biologists even have assigned these two subsystems to different parts of the brain.⁴

2.2.3 From Learning to eLearning

We have seen, that communication has been underestimated in the traditional learning theories. This is even more surprising as historically, learning has always been connected to communica-

³MV98

⁴[Ve78]

tion and social interaction. Platon, for example, considered the dialogue between a student and his teacher as the most important way out of the cave of ignorance. In fact, no other method has proven such practical success in the history of education than the link of learning with human communication regardless of its structure being symmetric (communication between learners) or asymmetric (communication between teacher and learner).

Under this perspective, eLearning takes a new direction by replacing face to face communication with communication over a technical framework, making it much harder to react on any kind of difficulty (which could be possibly handled much easier in face to face situations). Because this is especially true for the affective domain, this document mainly concentrates on the cognitive aspects of eLearning (in fact, our system initially rather aimed at blended learning scenarios, where affective issues can still be handled with face-to-face communication). But the affective component will be kept in mind, for example, when designing user friendly interfaces or using colour schemes that permit association to positive feelings (see 2.5.4).

2.3 Content Model

Having pointed out the key role of interaction and communication in learning we will have to offer a concept of content that makes the interaction between a human and learning content as rich as possible. We will address this issue by referring to a theory that puts the focus on the verbal and non-verbal representations of knowledge.

2.3.1 Bruners Theory of Representation

Jerome S. Bruner illuminated the pedagogic consequences of the results in cognition psychology produced by the works of Jean Piaget. In doing so, he prepared the rise of cognitivistic instruction theories like those of Ausubel and Gagné,⁵ though his philosophical position is rather that of a constructivist.⁶

In his works on instruction theory in the late 60's, Bruner exposes the fundamental role of language in learning as a 'calculus of thought':

'Teaching is vastly facilitated by the medium of language, which ends by being not only the medium for exchange but the instrument that the learner can then use himself in bringing order into the environment.'

The intersubjective quality of language and the availability of different abstraction levels in language makes it even possible to structure learning content for different intellectual levels by the complexity of the language needed for the discussion of it. For example, a basic task for a student could be to explain something 'in his own words', whereas advanced tasks could require to explain it in terms of a scientific vocabulary.

This leads Bruner to a hypothesis that has provoked a lively, yet controversial debate in the field of education:⁷

⁵[Sch02],[Ze95]

⁶[Br66], [Br56]

⁷[Wi81]

‘Any idea can be taught in intellectually respectable form to any child at any age or level of development, provided that you do so in his own conceptual vocabulary.’

We will see in the following sections that the ‘conceptual vocabularies’ can be constructed not only in a verbal (symbolic) representation, but also for non-verbal ones, thus providing us with a model that allows learning on a wide range of channels.⁸

The EIS (Enactive-Iconic-Symbolic) Representation-Scheme

A consequence of the discovered connection between language and intellectual development for instruction could be to make learning completely language centred. In fact this has historically been the predominant education model: Literacy, the ability to read, write and work with texts has always been regarded as a strong measure of ‘higher’ education from the Greek sophists to W. v. Humboldt. As mathematics has a highly formalised language, where each concept has an exact definition, this perspective can be taken even more radically: Teach only the definitions of mathematical concepts and give exercises that are free of applicational contexts.

Bruner takes a different path and leaves the strict cognitivistic model by introducing a scheme that contains not only the *symbolic* language representation of reality but also two other modes of representation. These modes and their effectiveness can best be understood from an evolutionary perspective. We therefore illustrate them by taking a look at how children actually manage to learn and communicate without language:

In the beginning children cannot talk about objects, but only point at them and perform interactions with them. The set of possible actions which can be applied to an object, is what Bruner calls its *enactive* representation. For example, the enactive representation of a tennis ball could be denoted by a mapping of actions to consequences: Touch it and it feels soft; take it in the mouth and it tastes hairy; throw it and it bounces etc. Thus by exploring the enactive representation of an object, the child learns about its properties.

The other mode, in which children are able to learn without words is by using pictures, e.g. by watching a picture-book or by drawing or painting. The ‘picture being sometimes more worth than a thousand words’ stands for the independence of this visual (or more general: perceptual) representations from symbolic or action-based knowledge. In case of the tennis ball, its *iconic* representation would be determined by its geometry, colour and (depending on the level of detail adjusted) by its fuzziness.

Differences to Other Schemes It should be noted that there are also other schemes of content representation, the most prominent of which bases on the Dual-Coding Theory of A. Paivio⁹ that differentiates between visual (‘imagens’) and verbal (‘logogens’) representations. Compared to this scheme, the EIS scheme seems more appropriate for constructivistic learning scenarios, because by adding the enactive representation level, it considers focus the interaction between learner and content in constructivistic learning theories. Other theories, like ACT* differentiate learning mainly in terms of predicative/declarative and and procedural/functional

⁸For a motivation of using different learning channels, see [Ve78].

⁹[CP91]

aspects,¹⁰. By mapping declarative content onto iconic and symbolic representations and procedural content onto enactive representations, the EIS-scheme can also be seen as an extension of ACT*.

Above its didactic comprehensiveness, there is still another reason for choosing the EIS-scheme as a model for the creation of interactive learning content: It offers a concept-centered view on learning content and therefore perfectly harmonises with an object oriented architecture as implementation, which we will see in chapter 3.

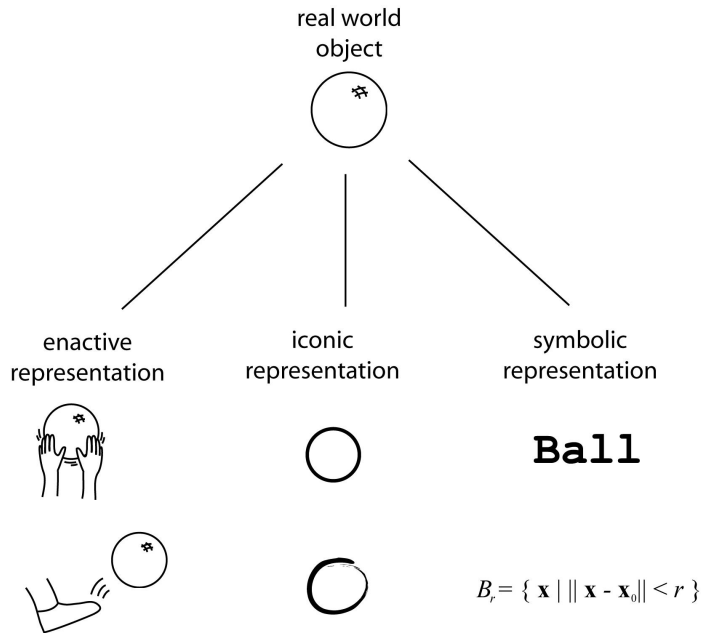


Fig. 19: Modes of representation in Bruners EIS-Scheme

Regarding the different level of abstraction in the representation modes, one can easily guess how learning of a concept (e.g. a ball) takes place: It usually starts with gaining immediate experience by working with enactive representations (playing with a ball), then the visual appearance (real ball or picture of a ball) already brings the set of possible activities into presence. Afterwards the word 'ball' (spoken or written), suffices to refer to the concept of a ball and what can be done with it. The advance in learning and knowledge of concepts up to the highly condensed symbol systems of mathematics can thus be regarded as a pyramid, in which each step relies on a base of more concrete knowledge that has already been learned. In this hierarchy, a more abstract mode of representation cannot substitute its lower counterpart, while the reverse is possible:

For example, one can hardly play football with the picture of a ball and the word 'ball' does not look like a ball, but instead a real ball can replace an image of itself and everybody would understand a picture of a ball within a text as substitute for the word. We will return to these issues in 2.5.

¹⁰[AM98],[Sch03]

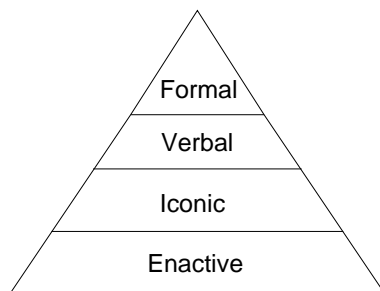


Fig. 20: Hierarchy of abstraction in knowledge representation

2.4 Methodic Model

In this chapter we discuss the methods of interaction between learner and learning entities. In doing so we will tightly connect the two meanings of ‘learning methods’, the methodic knowledge of the subject matter and the methods of learning this knowledge. This connection corresponds to the ‘learning by doing’ paradigm of constructivistic learning theories and its obvious advantage (and, of course, possibly also its disadvantage) is, that it hands the control of the learning process over to the learner. We will therefore use a conceptual framework that helps the learner (and others) to assess his learning progress.

2.4.1 Taxonomy of Learning Methods: Bloom et al.

The EIS-Scheme gives us a powerful structuring method for the presentation of learning content. We now take a look at the level of interaction between learner and learning content by introducing a method to evaluate the learning process in terms of ‘learning value’.

Obviously, the learning value of a lesson or other learning unit varies strongly with the kind of interaction between learner and content:

If, for example, we want to discuss functions, we can do this on very different levels: we could give the student a function definition for a function f and ask him to calculate $f(x)$ for some values of x , we could also ask him to sketch the graph of f or tell us about its properties, we could even ask a student to model a given empirical distribution with a function and interpolate or extrapolate values, making predictions for further measurements.

The complexity level of the approach can therefore be almost independent of the treated content. This corresponds to Bruner’s thesis that any idea could be taught on any level of development (see 2.3.1). A structure that allows to specify different levels of complexity is the taxonomy of Bloom et al.¹¹, a prominent result of the efforts in the 50s to homogenise American school exams and curricula. This taxonomy is designed for the cognitive and the affective domain. For analysing the interactive potential of learning content, we concentrate on the cognitive domain. The taxonomy distinguishes between six levels of cognition:

Knowledge

i.e. the ability to observe and recall information, principles and methods for specifics as well as for universals and abstractions.

¹¹[B156]

Comprehension

i.e. the ability to understand information by being able to translate it into new representations, to grasp meaning by interpreting facts and extrapolate information within a specific subject matter

Application

i.e. the ability to use information, methods, concepts, theories in new situations and to solve new problems using already acquired skills or knowledge.

Analysis

i.e. the ability to see patterns, the organisation of parts, the recognition of hidden meanings and identification of essential components of the chosen subject matter.

Synthesis

i.e. the ability to use of old ideas and principles to create new ones, to generalise from given facts or to relate knowledge from several areas, to predict and to draw conclusions.

Evaluation

i.e. the ability to compare and discriminate between ideas, to assess the value of theories and presentations and to make choices based on reasoned argument, also to verify value of evidence and recognise subjectivity.

This taxonomy fits almost perfectly into the scheme developed up to here:

- The successful application of knowledge is based on its proper comprehension, this corresponds to our thesis, that teaching understanding in mathematics is indispensable to the education of engineers.
- The definition of comprehension includes the ability of translating information from one representation into another; this complies with the representational learning model of Bruner.
- The taxonomy is designed to allow an objective assessment of learning progress. Using it will enable us to design constructivistic learning scenarios, where a user may interactively monitor his or her level of cognition achieved for each specific topic.

We will elaborate these points in the following section.

2.5 Consequences for Didactic Design

In this section we will present the didactic model for our mathematical eLearning site that takes into account the concepts discussed in 2.1-2.4.

2.5.1 Didactic Design Process

According to the considerations in 2.2.2 we replace the predominance of the general aims (cf. Fig. 16) by that of the ‘subject matter’ and the specific goals by learner specific issues (interests of the target group, etc.).

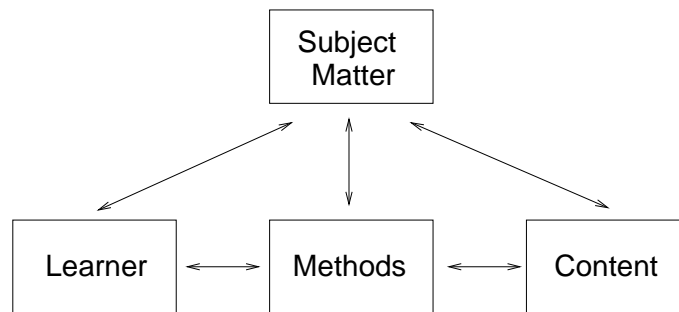


Fig. 21: Fundamental entities regarded in our design process

Having pointed out the complementing relation of content representation, learning methods and context-orientation, we will assume the following didactic design process for the rest of this document:

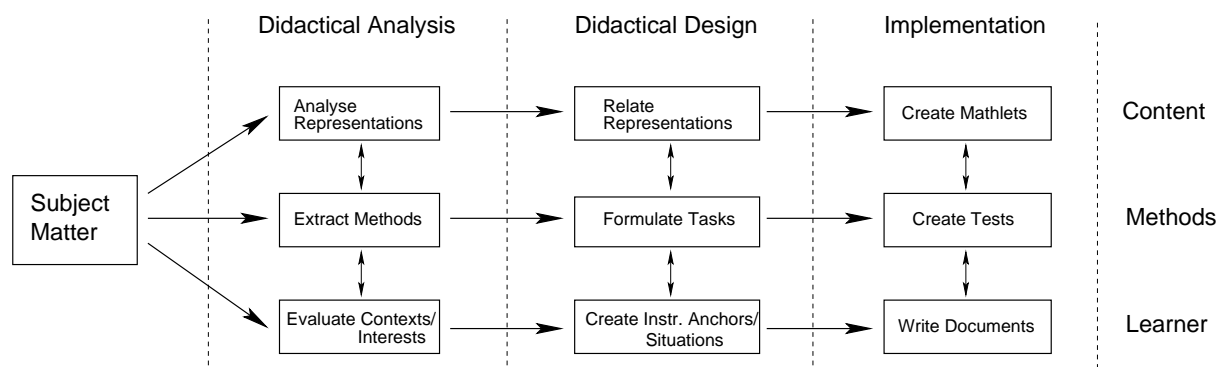


Fig. 22: Activities in the didactic design process

We regard the didactic design process as a sequence of three phases that each address and relate three different levels of dealing with the learning of an arbitrary subject matter. In the didactic analysis phase, the author decides, which representations he deems appropriate for the chosen subject matter, which methods are needed for working with it and in which real life contexts the problems addressed by the subject matter do in fact occur. The results of this analysis determines the choice of content representations, tasks and contexts that are elected in the following design phase. These design decisions are strong guides for creating the learning artefacts in the implementation phase, which are in our case mathlets, puzzles and interactive tests and web pages. For an example of using this process, please see 4.2.2

Though the decisions on each of the levels may be considered quite independent from the other, there is still an important amount of cohesion between them that should not be neglected. For example, for a chosen context, a certain representation and a certain method may be especially valuable, e.g. modelling real world curves by functions would extensively use iconic representations, whereas using functions for analysing oscillations would more rely more heavily on symbolic representations.

2.5.2 Content Representation

Mathematics as Formal Language

Though there are many different representations of mathematical entities there is consensus, that when it comes to producing practical results its symbolic representation is by far the most powerful. Bruner even states that ‘Mathematics is surely the most general metalanguage we have developed’. Learning mathematics thus has to focus on working on its symbolic representations. This is of course also true for eLearning, for which a powerful symbolic computation tool already exist: A Computer Algebra System (CAS) allows a learner to ‘talk in formal mathematics’ to a system by entering arbitrary symbolic expressions that are computed by the CAS. Since this allows real constructivistic self-directed learning (the author used it extensively throughout his mathematics education), a modern mathematical eLearning site should at least offer parsing functionality that works with user defined expressions.¹²

Using the EIS Scheme

Having stated the importance of symbolic representation for working with mathematics it is now time to stress the role of non-symbolic representations in learning. As seen in section 2.3.1 actions and images can be used as a – sometimes more intuitive – substitute for words, we may even call them each an informal ‘conceptual vocabulary’ of which informal ‘sentences’ may be constructed.¹³ This is actually done quite often in everyday life (e.g. the meaning of ‘if you throw away your ball, no one will fetch it for you’ can be taught by words or – sometimes more convincingly – by actions).

Since it is our goal to make as many mathematical concepts enactively and visually experienceable, we need to refer to a kind of pseudo-language for each domain, into which the formal concepts of mathematics can be ‘translated’. Here are some examples of possible transformations, a more extensive list can be found in A in the appendix:

| Mathematical concept | Iconic transformation | Enactive transformation |
|----------------------|--|---|
| Linear independence | n vectors spanning an n -dimensional subspace | Failing in constructing one vector as a linear combination of the other. |
| Derivative of f | Displaying the tangent’s slope in points $(x, f(x))$ | Constructing the derivative by drawing the tangent’s slope for several points of the graph of f |

There is of course often more than one transformation for a mathematical concept and some abstract concepts may not even have a sensible transformation at all, but since there seems to be a need for non-symbolic representations in communicating mathematics, almost all popular mathematical concepts do have multiple representations.

¹²One could conclude to simply centre web-based eLearning about using a CAS; there are, however, several reasons speaking against this, see 3.3.4.

¹³More precisely, we may regard them as what [Wa67] calls an *analogic code*: In contrast to a digital code (human or formal language made of letters or phonemes), analogic codes resemble their referent and allow continuous variations.

Using these transformations we can design and implement mathematical eLearning media with the help of Bruner's EIS scheme.¹⁴ This is done for a chosen mathematical entity by producing iconic and enactive transformations that suit both the mathematical application context and the facilities offered by the technical environment:

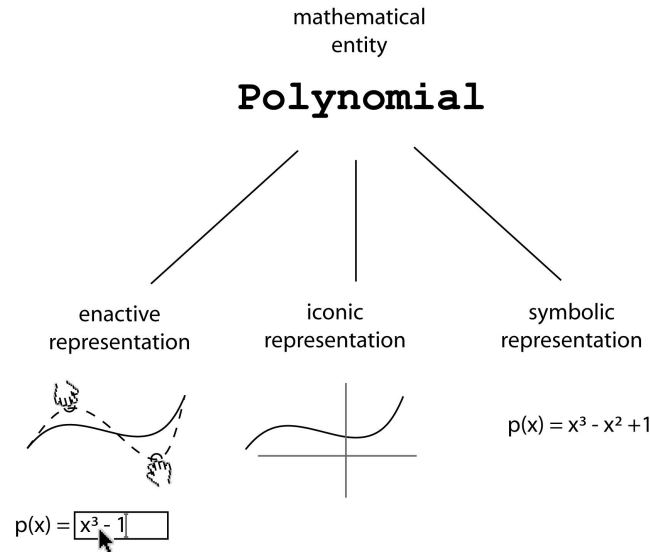


Fig. 23: Representations used in mathematical eLearning media

From this figure it may be clear, that we need an architecture that allows the flexible use and interaction with these different representations. This architecture will be presented in the next chapter.

2.5.3 Methods and Tasks

For the methods and tasks we use the scheme provided by the taxonomy of Bloom. There are already several applications of the taxonomy for mathematics education¹⁵ as well as for eLearning in general¹⁶.

We concentrate on learners working with mathematical expressions (i.e. words of the formal language addressed in 2.5.2), for which example applications will be given in 3.4.

Learning by Relating and Transforming Representations

Bloom regards understanding as the ability to ‘translate knowledge into new contexts’. Since the representation of knowledge is context dependent, this also means that a learner must also be able to relate and transform representations of a concept into another, thus allowing him to perceive the ‘real world object’ behind the different representations. For this reason it is our aim to construct applications that allow the user to initiate and watch interactions with different

¹⁴Note, that in order to do this, we have to put the mathematical object in place of the ‘real world object’, something which is not unimaginable in Bruner’s constructivistic philosophical background.

¹⁵e.g. [Lo03], [Ga00], [Wo02]

¹⁶e.g. [Mu01], [Kn03]. However, there are no applications for mathematical eLearning.

representations (e.g. changing a parameter in a symbolic function definition and watching the graph changing accordingly, see 3.2).

Different Levels of Working with Mathematics

If we regard Bloom's taxonomy discussed in 2.4.1 it becomes obvious, that not every student learning mathematics must do so on all levels. In fact, for engineering students the mathematical repertoire needed primarily consists of practical skills (in contrast to students of mathematics who also need deeper understanding in analysis, synthesis and evaluation of mathematical methods). We therefore concentrate on the first three levels of cognition in order to structure mathematical content for students seeking mainly practical skills (see figure 24).

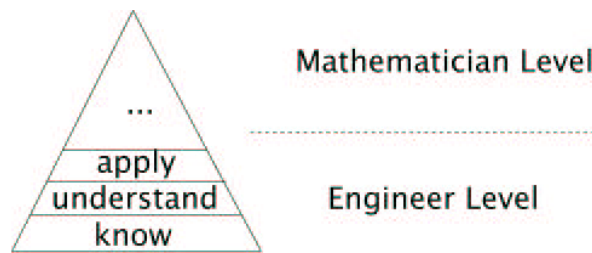


Fig. 24: Classification of cognitive levels in mathematics

Assessment Design

If we want to allow the student to assess his learning progress by himself, we need a model that tells us, what kind of tests supports this. Traditional assessment design (basing on statistical test theory¹⁷) concentrates on the reliability and validity, when designing tests. As from our point of view these are primarily author issues, we focus on the representational system to be used and the degree of freedom a student has. This means we do not exclusively use the classical multiple choice scheme, but also other types like graphical puzzles and 'constructed response' tests. This corresponds not only to the concept of learning stated above, but also uses the assessment model of modern studies like TIMSS¹⁸ and PISA¹⁹, which aim at the evaluation of high level cognitive skills.

2.5.4 Learner Orientation

Affective issues

As discussed above, the affective domain in learning and eLearning is a comprehensive field of research of its own and beyond the scope of this analysis.²⁰ We will, however, consider some of its issues in our implementations, like aesthetics and usability of content representation and choosing the right affective context for a certain subject matter:

¹⁷[Gr03]

¹⁸[ISC03]

¹⁹[OECD03]

²⁰See, for example, [Ve78], [MJ91] for the psychological and pedagogical foundations.

Aesthetics and Usability

In order to increase the affective linkage between learner and learning environment it is necessary to reach a high amount of user-friendliness. This can be achieved by the design of an appealing and reactive interface, that allows a variety of manipulations as well as a simple and elegant access to the mathematical objects.²¹ The user interface presented in the next chapter has been designed carefully and was rearranged in several evaluation cycles. To ensure a maximum of recognition, the component architecture provides a strict layout.

The affective linkage is also increased by using a ‘leitmotiv’ that guides the user through the whole system. For our platform this leitmotiv is the theme of ancient Egypt, where mathematics played a decisive role in achieving cultural heights. As these heights are best symbolised by the Egyptian pyramids, we use a stylised pyramid in the prototype system²² as metaphor for lasting mathematical achievements.

Context Relation

As mathematics is the deductive foundation for all science, it can be presented in almost any scientific context. In fact, mathematics and science didactics in Germany has been stressing the use of contexts in teaching at school in the last decades.²³

Because our research is based on a project for an engineering course, we use contexts and examples from physics and construction technology, especially mechanics (see figure 25).

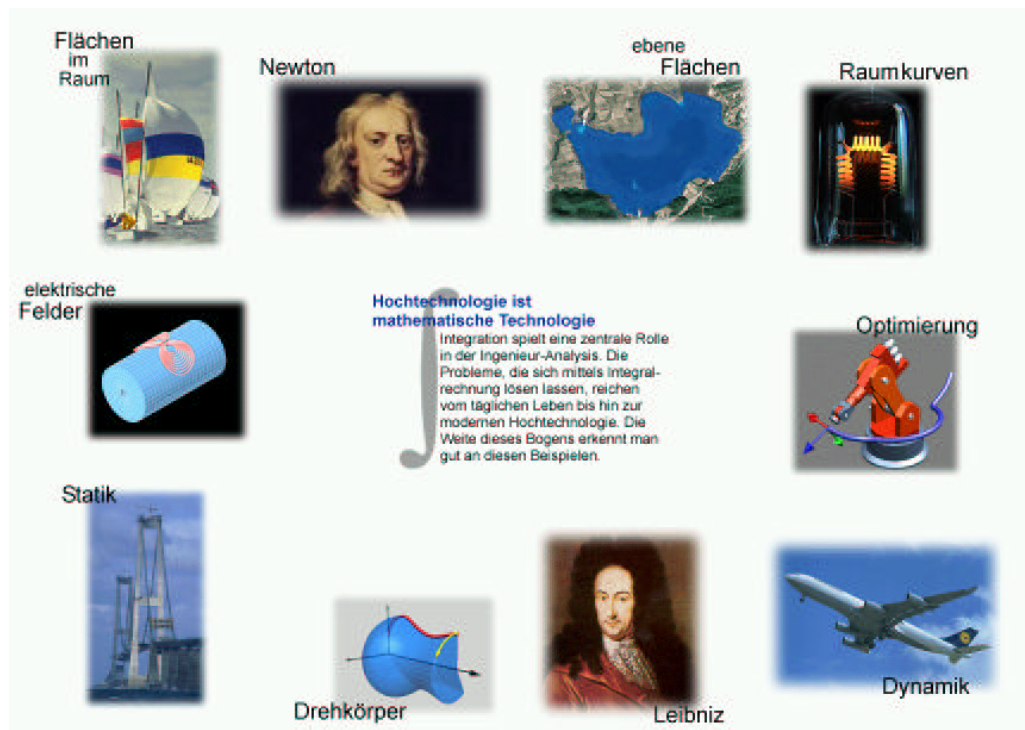


Fig. 25: Engineering contexts for ‘integration’ in a predecessor of Mumie²⁴

²¹For aesthetics and usability of web pages/sites see [Th04], [Si97].

²²See section 4.2.

²³[MSWWF99], [Mu95]

²⁴[TUM01]

Chapter 3

The Component Framework

In this chapter we present a framework that takes into account the conclusions drawn in the previous chapters. It consists of the MathletFactory, a component system for developing mathlets and the TestletFactory, a framework for creating puzzles and tests. The latter addresses authors, teachers and students, while the MathletFactory additionally regards the role of the developer. We therefore give a brief overview from each of the roles discussed in 1.1.3, starting with the student's (and tutor's) perspective, moving over to the author's and application developer's perspective. The system developer's perspective (which requires delving deeper into the architecture) is covered outside this chapter in Appendix B.

3.1 The MathletFactory from the Student's Perspective

For the student all mathlets created by the MathletFactory show some common features that allow him to learn quickly how to work and explore mathematics with them. These are the mathematical entities and their representations used, the formal language processing capabilities of the symbolic representations and the generic display and interaction system allowing the user to interact with all mathlets in a common way.

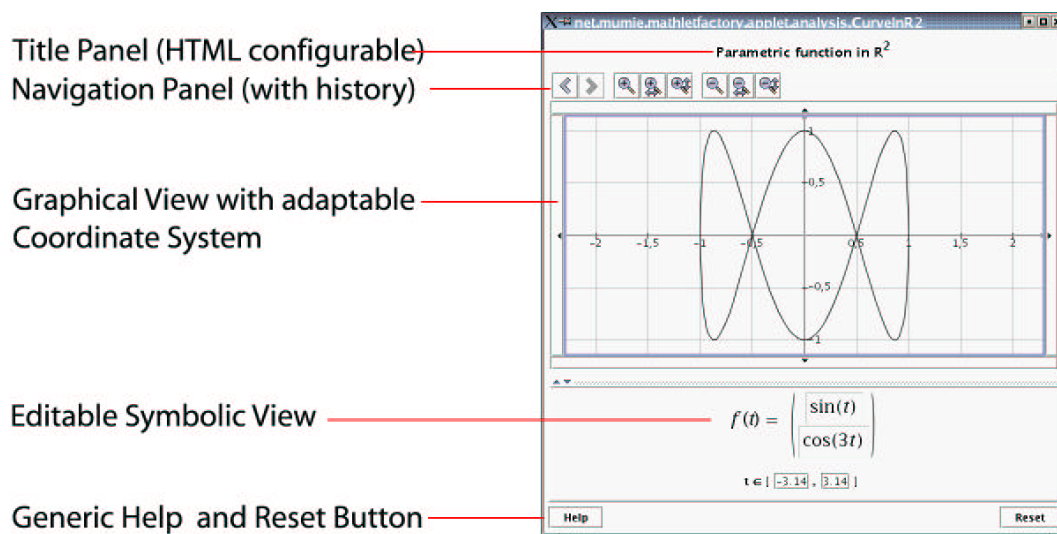


Fig. 26: The common look and feel of all mathlets

3.1.1 Mathematical Entities and their Representations

Using the results of 2.5.2, mathlets put the mathematical entities at the centre of their stage, showing mostly one or two mathematical entities to the user, that he may interact with. This happens on different representation modes. For example, a three-dimensional vector may be represented by an arrow in 3D space (iconic representation) and by a triplet of numbers (symbolic representation). The student now has the opportunity to work on both levels with the vector by either dragging the vector with the mouse or by altering its components with the keyboard, watching the other representation changing accordingly.

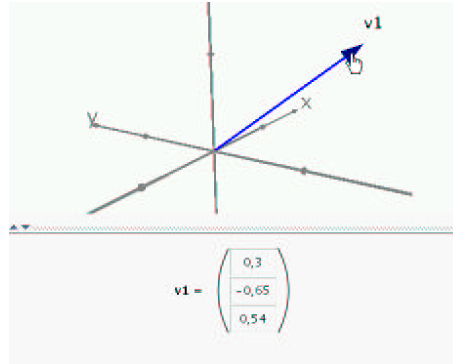


Fig. 27: Iconic and symbolic representation of a 3D vector

3.1.2 Formal Language Processing Capabilities

Since mathematics puts a focus on working with symbolic representations, the MathletFactory's capability of processing formal languages is a vital feature. The MathletFactory allows the student to enter almost any expressions for any mathematical entities that involve mathematical operations or relations like functions, sequences, series, sets, equations, etc. Since the syntax used is almost the same as one would write on paper, it requires no further study of allowed expressions; in cases of doubt the generic help page displays the set of supported symbols.

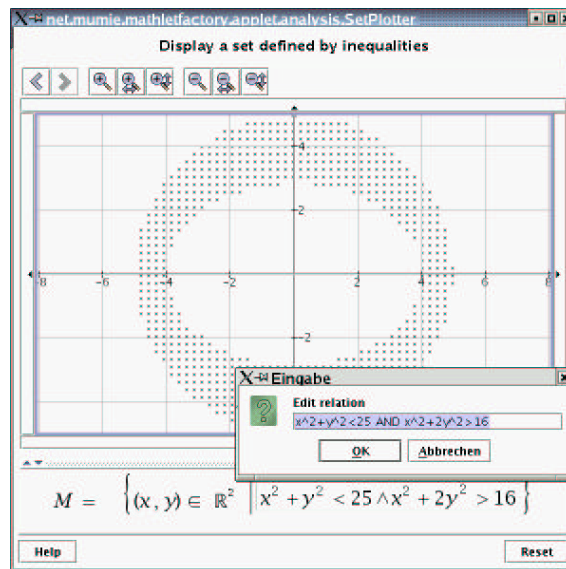


Fig. 28: Entering a user defined relation for a set

3.1.3 Generic Display and Interaction System

Not only the symbolic representations but also the iconic and enactive representations allow an intuitive and standardised way of navigation and interaction with the mathematical entities. This is ensured by the MathletFactory's generic display and interaction system. From the student's perspective it results in a common user interface for all applets – no matter what mathematical entities they are dealing with. For the scene (i.e. the canvas containing the graphical representations of the mathematical entities) we have therefore added a double navigation functionality: navigation by using icon buttons and navigation by mouse drag gestures. The buttons (zoom in and -out, translation of the scene) can be seen in the following figures and are widely self-explaining, making them ideal for beginners.

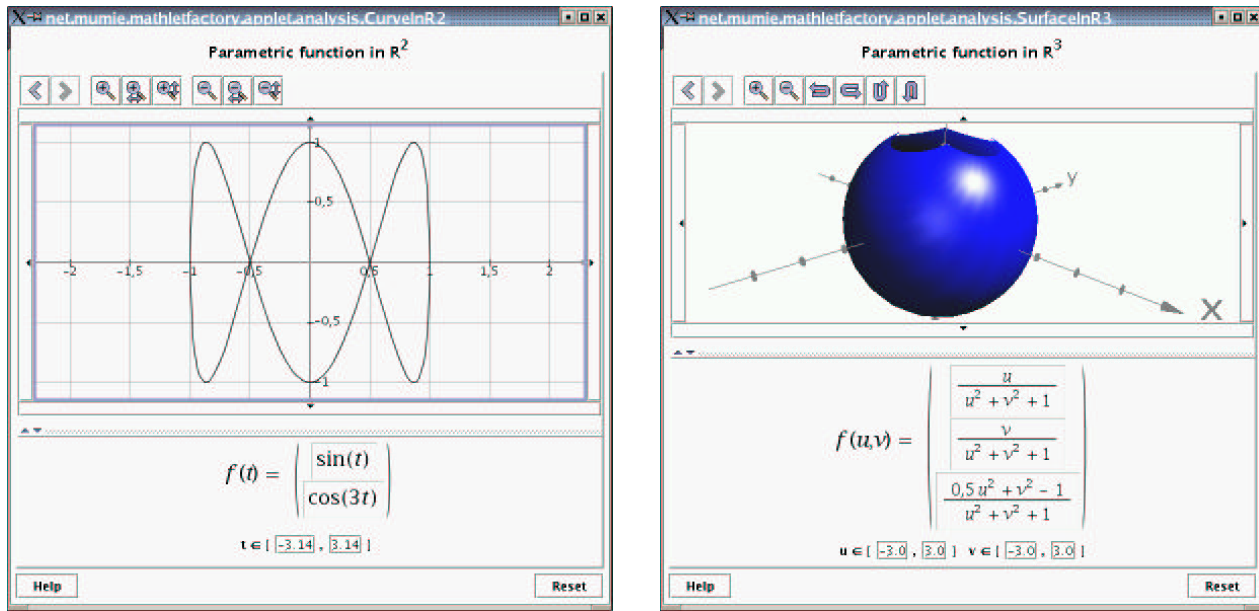


Fig. 29: The generic navigation system for 2D and 3D mathlets

The mouse drag gestures in turn allow a quicker and – for the experienced user – more intuitive way of navigating the scene: by clicking on the scene and dragging, the user's view of the object may change, either translating, zooming or rotating the object.

3.1.4 Number Handling

Due to the fundamental role of numbers in mathematics, the MathletFactory has an optimised user interface for numbers, allowing the student a high amount of comfort and flexibility when handling numbers of different types. This includes the following:

- The system distinguishes between different number classes like integer, rational, real, complex numbers, even the use of finite fields like $\mathbb{Z}/p\mathbb{Z}$ is possible.
- For every number class it is possible to enter symbolic expressions, e.g. $\frac{\pi}{2}$ or $\sqrt{2}$ for real numbers. It is also possible to re-edit these expressions.

- Additionally there is a slider representation for real numbers allowing the comfortable and flexible manipulation of numbers, it is even possible to play an animation of the changing parameter and its consequences.

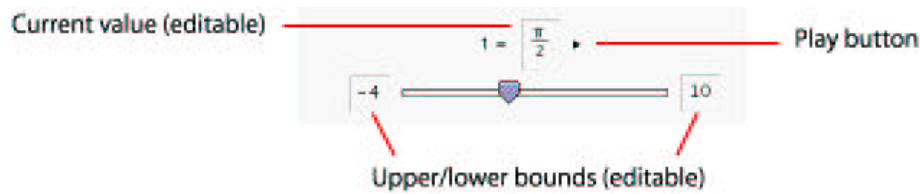


Fig. 30: The generic slider panel for real numbers: In addition to dragging the slider bar with the mouse or editing the values, the user may click on the play symbol to start an animated slide from left to right.

3.1.5 Animations

The flexibility of the mathlets allows their use for both demonstration and exploration purposes, thus making them interesting for pure distant learning as well as augmenting presence learning or blended learning scenarios. There are however cases, when a teacher or author may want to show a certain construction process in mathematics without too much interaction with the mathlet. For these cases an animation framework allows the student to be presented exemplified mathematical concepts ‘live’ and with user defined start conditions. For example, in figure 31 on the left, the vector addition can be animated for any two vectors preset by the student (or teacher). The interface for using the animations is offered in two differing versions: A simple solution embedding the animation into a single panel inside the mathlet window and a comfortable implementation in a separate window, that also allows the user to vary the speed of animation.

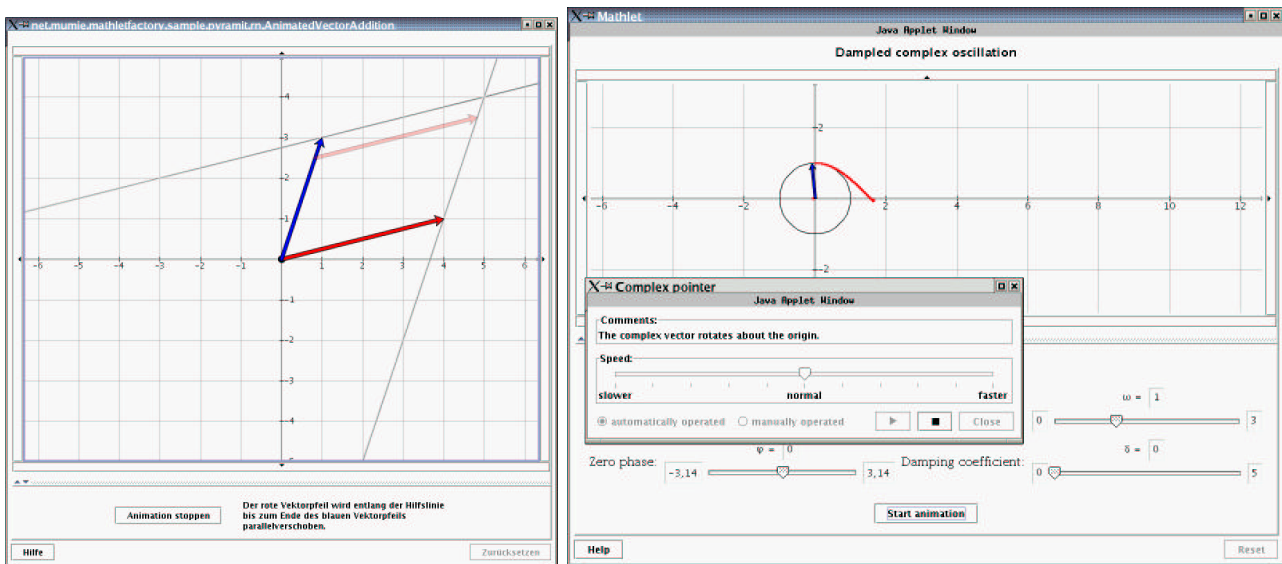


Fig. 31: Examples of mathlets using the animation system

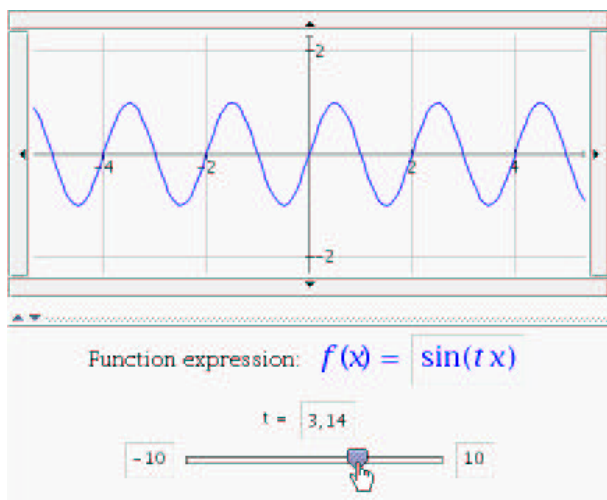
3.2 The MathletFactory from the Author's Perspective

The author's perspective on the MathletFactory mainly comprises that of the student's perspective. But in addition, the author wishes to use existing applets for his own courses, for which they need to offer some features of flexibility and reusability:

Reusability

The flexibility and reusability that is designed for the student of course also benefits the author. For example, he could embed a mathlet in a web page along with some tasks to accomplish, like 'enter the value xy in the input field and move point yz to the origin'. But there is also a separate approach that makes it easier for the author to configure prefabricated mathlets to fit his needs: The use of applet parameters.¹ We consider, for example, a parameterised function plotter, i.e. a mathlet that displays a function depending on some constant t . The user is able to change t with a slider and may therefore watch the resulting 'continuous' change in the function graph. Such a mathlet could be used as a tool to explore any function the user has in mind, but also as a demonstration unit within a specific lesson.

We illustrate this with a simple reuse scenario: For example, an author may want to point out that the frequency of a sine function can be changed by altering t in $f_t(x) = \sin(tx)$. He is then able to do so by adding the mathlet to his instructive page setting the applet parameter **function** to **sin(tx)**. He could optionally add a task for the student to rearrange the term so that t represents the amplitude of the graph, allowing symbolic manipulation technically by setting the parameter **functionEditable** to **true**.²



```
<applet code="../../../FunctionSlider">
  <param name="lang" value="en" />
  <param name="separateWindow" value="true" />
  <param name="appletWidth" value="600" />
  <param name="appletHeight" value="600" />
  <param name="function" value="sin(tx)" />
  <param name="functionEditable" value="true" />
  <param name="parameterName" value="t" />
  <param name="paramLeftBound" value="-10" />
  <param name="paramRightBound" value="10" />
  <param name="worldWidth" value="4" />
  <param name="worldHeight" value="4" />
  <param name="worldCenterX" value="0" />
  <param name="worldCenterY" value="0" />
</applet>
```

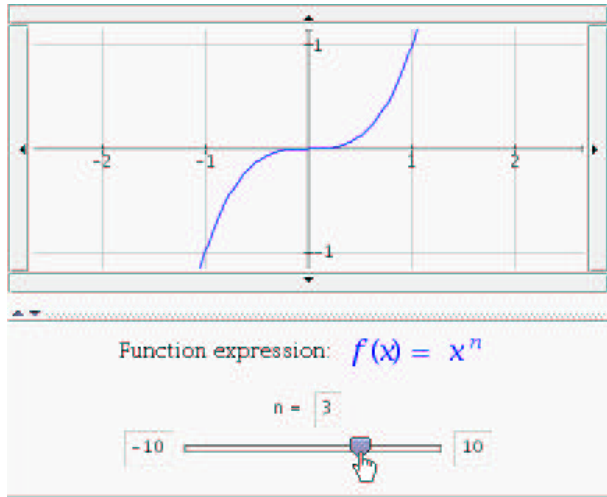
Fig. 32: A parameter configuration of the FunctionSlider mathlet

He may, however, use the same mathlet for demonstrating, how the natural power functions $f_n(x) = x^n$ behave for $n \rightarrow \infty$ in a different page by setting **function** to **x^n**, **parameterName** to **n** and **paramAllowOnlyIntegers** to **true**.

¹[Sun04]

²The generic and specific parameters for mathlets are documented in Appendix D.

We see that the use of applet parameters strengthens the role of the author without programming skills and introduces a new level of flexibility and thus reusability. We will make heavy use of this mechanism in the application part. In addition, we will transfer it to the other parts of the component system that are discussed in the next section.



```
<applet code="../../../FunctionSlider">
  <param name="lang" value="en" />
  <param name="separateWindow" value="true" />
  <param name="appletWidth" value="600" />
  <param name="appletHeight" value="600" />
  <param name="function" value="x^n" />
  <param name="parameterName" value="n" />
  <param name="paramAllowOnlyIntegers" value="true" />
  <param name="paramLeftBound" value="-10" />
  <param name="paramRightBound" value="10" />
  <param name="worldWidth" value="4" />
  <param name="worldHeight" value="4" />
  <param name="worldCenterX" value="0" />
  <param name="worldCenterY" value="0" />
</applet>
```

Fig. 33: The same mathlet with a different parameter configuration

3.3 The MathletFactory from the Application Developer's Perspective

This section describes the application developer's perspective on the MathletFactory, the perspective of the system developer (which is necessary for adding new mathematical objects and representations) is presented in Appendix B. For the developer, the MathletFactory is technically a Java class library of about 800 classes and 150.000 lines of code. Conceptually it is a framework to speed up the production of high quality mathlets for an application programmer. This is achieved by the following structural features:

- a model-view-controller (MVC) architecture that allows the flexible and reusable construction of interactive mathematical exploration scenarios
- a component system that offers support for about 100 multimedial mathematical objects (so called MMathObjects) and their representations
- a generic display and interaction system that is independent of the graphical libraries used
- a communication system that allows the MMathObjects to interact with the user as well as with each other

To elaborate the listed aspects and their origin in didactic design, we will give a short tour d'horizon.

3.3.1 MMOBjects as Models for Mathematical Entities

As a conclusion of the focus on the subject matter (see 2.5.1) and its possible uses, the MathletFactory assigns a key role to the mathematical entities. Their functionality is represented by the *MMOBjects*, multimedial mathematical objects that form the basic components. The core idea of the MathletFactory is that an application programmer should only 'plug together' these components when building mathlets, reducing their production time to minutes rather than hours and days.

3.3.2 MVC Architecture Supporting Multiple Representations

As stated in section 2.3.1 and 2.5.2 we need an architecture that separates the mathematical logic from its representation. The best way to achieve this is known as the model-view-controller pattern³, which claims different modules for logic/data, presentation and interaction code. This allows not only MMOBjects to have different representations (flexibility) but also different mathematical objects to have the same representations (reusability). For example, a function object would have both a symbolic representation and a graphical (iconic) representation, whereas a spline would have the same graphical representation, but a different symbolic representation.

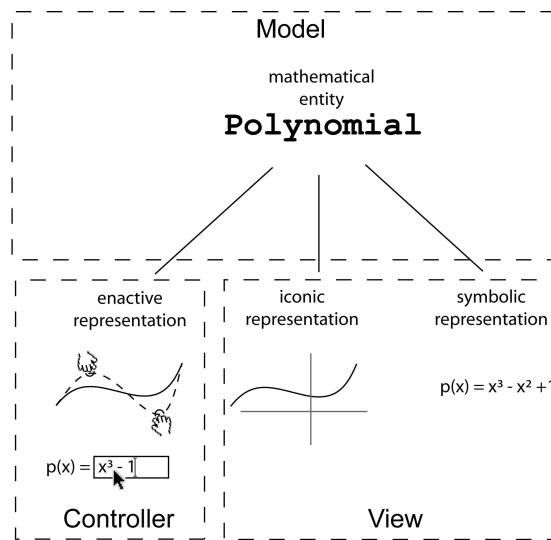


Fig. 34: Mapping the didactic structure to a model-view-controller architecture

3.3.3 Number Handling

As mathematics often offers a common calculation but different representation for different number classes (e.g. rational, real or complex numbers), it is important that mathlets on the one hand support the use of different number classes, but on the other offer common methods for computing with these. The MathletFactory addresses this problem by parameterising all MMOBjects by their number class, allowing them to determine the number class upon initialisation or even at runtime.

For example, figure 35 presents the same mathlet with each a rational and a real configuration.

³[Bu96]

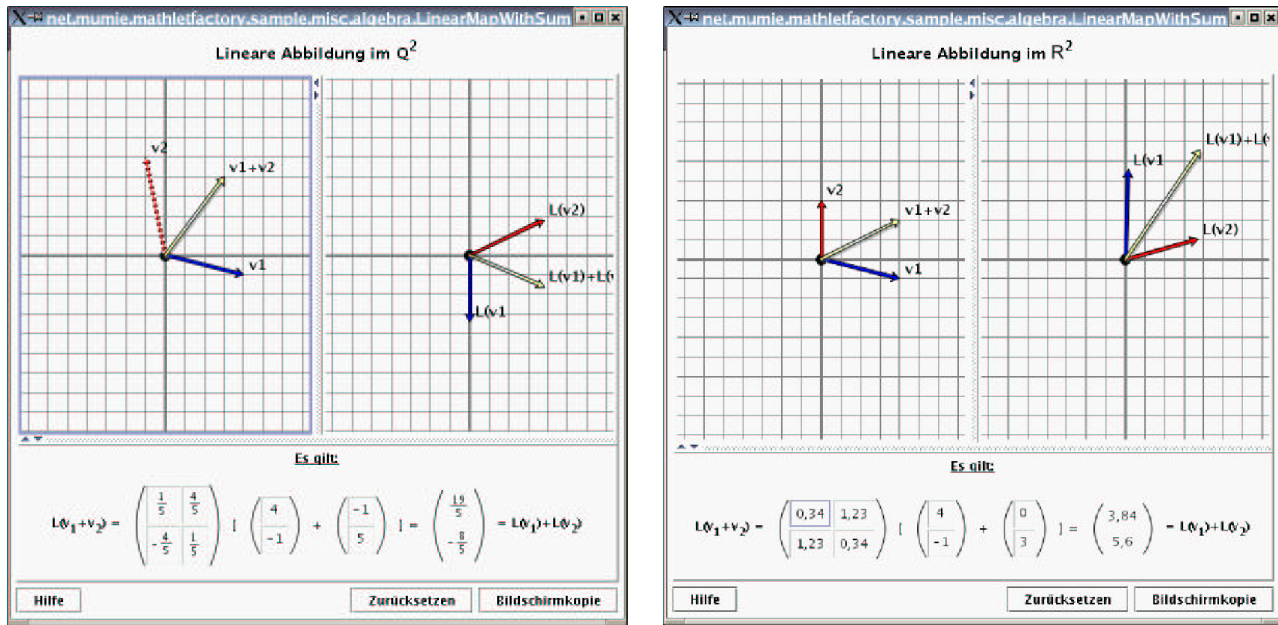


Fig. 35: A linear map mathlet with a rational and a real number configuration

3.3.4 Formal Language Processing Capabilities

In order to achieve a higher level of semantics in communication with mathematical objects we have claimed in 2.5.2 the necessity of a 'mathematical language' processing facility. Existing CAS like Maple, Mathematica and MuPAD already offer this functionality for a wide range of mathematical applications. Why not use them or integrate them into our framework? There are several reasons that argue against it:

- There is no fully operable platform independent CAS⁴, making its use impossible in a pure client side solution.
- Most of the prevalent CAS are not free, almost none of them is open-source, this contradicts our free and open-source approach.
- Almost all CAS were primarily designed for research, thus often lack a lightweight intuitive user interface. Additionally, since there is no standard CAS, there is also no standard way of symbolic input. The standardised protocols MathML⁵ and OpenMath⁶ in turn are too verbose and explicitly not designed for user input.

We therefore implement a small parser of our own, that performs the lexical and syntactic analysis of mathematical expressions. The theory and implementation details of these are covered in Appendix B.

⁴An experimental version can be found under [Hm01].

⁵[W3C04]

⁶[OM04]

3.3.5 Generic Display and Interaction System

From the application programmer's perspective, it is also important that the dimensions of display (2D on Screen, 3D on Screen, Head-Mounted-Displays, etc.) are independent from the specific components. The MathletFactory's display and interaction system is therefore independent of the graphics library actually used.

Instead it offers a generic `MCanvas` class that contains the `MObjects` and bundles all functionality related to displaying and rendering. So if an application developer wants to (iconically) display a mathematical entity, all he has to do is to choose a template applet that contains a specific (2D- or 3D-) `MCanvas` and add the desired `MObject`(s) to it (see code example on page 53). The same is true for symbolic representations, except that they are directly added to the applet, not to the canvas.

3.3.6 Interactivity: Update Graphs

We have seen how the MVC architecture allows the user to interact with the mathematical objects thus allowing the prescribed amount of interactivity exposed by 1.2.3 and 2.2.2. But for interactive learning we need more than this: For example, consider a student who should iconically explore that the altitude lines of a triangle meet at the same point. A triangle is made out of three points (represented by `MMAffine2DPoints`) that are connected by line segments (`MMAffine2DLineSegments`). The user can now interact with the points by dragging them with the mouse. But the system has to be told that each time the points change, the line segments need to change accordingly. Also the `MObjects` representing the altitude lines should be redrawn each time the triangle changes. This means that for implementing interactive learning situations it is crucial to provide a simple yet powerful update system that allows the communication between `MObjects`.

This is achieved by creating a directed update graph that is traversed every time any of the upper objects records a change.

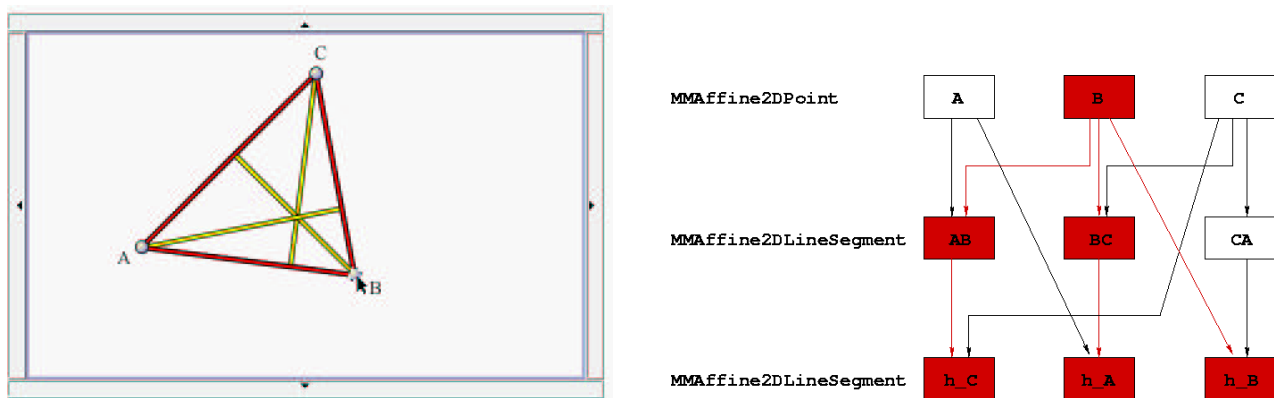


Fig. 36: An interactive triangle with altitude lines and its update graph (marked = updating)

3.3.7 Building Mathlets

We now take a closer look at the main issue of the MathletFactory: Providing a rapid mathlet development process to application developers. This is done by using the following linear

process model:

1. Choose the display type and numbers of displays to be used and extend the corresponding mathlet skeleton.
2. Add the chosen MMathObjects and their iconic or symbolic representations
3. Add necessary handlers
4. Create the update graph by adding updaters and creating dependencies

We demonstrate these steps for a mathlet, that displays an orthogonal vector of a user defined 2D vector:⁷

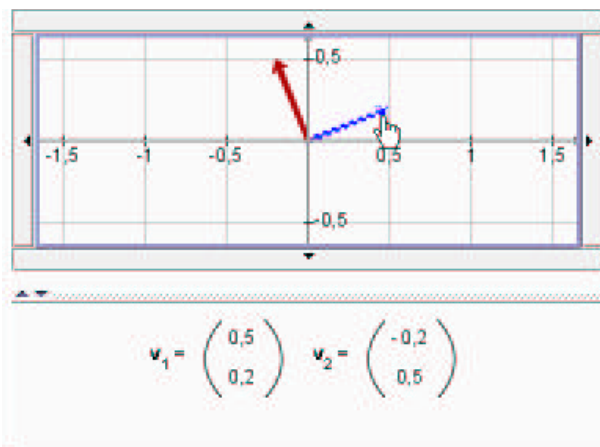


Fig. 37: The demonstration mathlet to be constructed

1. **Choose the display type and numbers of displays to be used and extend the corresponding mathlet skeleton**

We wish to use a single 2D display for both the vectors, so our applet extends the corresponding skeleton:

```
public class OrthogonalVector extends SingleG2DCanvasApplet {
```

2. **Add the chosen MMathObjects and their iconic or symbolic representations**

We create a vector space and two vectors and add their iconic representations to the canvas and their symbolic representations to the control panel of the mathlet:

⁷The complete source of the mathlet can be found under `net.mumie.mathletfactory.test.algebra.OrthogonalVector`.


```
// create MObjects
vectorSpace = new MMDefaultR2(MDouble.class);
vector1 = vectorSpace.getNewFromDefaultCoordinates(0.5,0.2);
vector2 = vectorSpace.getNewFromDefaultCoordinates(); // will be updated anyway

// add the iconic representations:
getCanvas().addObject(vector1);
getCanvas().addObject(vector2);

// add the symbolic representations:
addText("v<sub>1</sub> = ");
addMMObjectAsContainerContent(vector1);
insertHSpace(5);
addText("v<sub>2</sub> = ");
addMMObjectAsContainerContent(vector2);
```

3. Add necessary handlers

We only need to add a mouse handler to the first vector:

```
vector1.addHandler(new DefaultR2VectorMouseTranslateHandler(getCanvas()));
```

4. Create the update graph by adding updaters and creating dependencies

All we need is an updater that sets the coordinates of the second vector to ones that are orthogonal to the first:

```
vector2.dependsOn(vector1, new DependencyAdapter(){
    public void doUpdate() {
        vector2.setCoordinates(vector1.getCoordinate(2).negated()
                                vector1.getCoordinate(1));
    }
});
```

We see that this four step process is sufficient to produce a mathlet that complies with our didactic model: A student may interactively experience the connection between the visual and the symbolic properties of orthogonality.

3.3.8 Reusability

Of course the example above could be altered in various ways, at the pleasure (i.e. the didactic intention) of the application developer. For example, one could ask the student to find out the symbolic relation for orthogonality himself by entering the coordinates for v_2 for given coordinates of v_1 . This would only require to remove the updater code and add a `vector2.setEditable(true)` statement. This demonstrates that for developing mathlets their reusability and flexibility bases on the flexibility of the MathletFactory components.

3.4 The TestletFactory

The mathlets developed with the MathletFactory give the student various opportunities to explore the structure and relation of mathematics, thus supporting his *understanding* of mathematics. But in order to *apply* mathematics the student also needs learning units that demonstrate the methods used in mathematics and give him tasks to practice these methods. One way to accomplish this would be to give assignments that can be solved using mathlets – either directly as a tool for producing the result, or in a more assisting way by helping to generate ideas. We will present examples of these mathlet-based tasks in the application chapter.

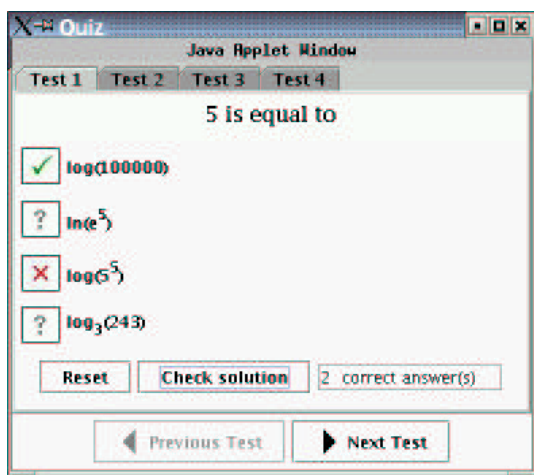
But to put a real focus on the methods of mathematics it is necessary to offer not only components that present the content in various ways but also tools that allow different levels of interaction with mathematical methods. This is also a tribute to the fact that despite all technical progress there is still an urgent need for practising mathematics with pen and paper. We therefore present a framework that allows the student a basic self-assessment of his skills in mathematics for a specific topic with a wide range of didactic purposes.

3.4.1 Different Categories of Tests

Technically we distinguish between three types of assessment units: quizzes (multiple choice tests), puzzles and word tests (also called constructed response tests). Didactically their categorisation also depends on the representations used. For example, a test that lets the student relate the graph of functions with their symbolic expressions (iconic and symbolic representations) operates on a different didactic level than a test relating input-output-sets with function expressions (symbolic representations only).

In the following we present the implementation of the different tests and show an example for each type of testlet.

Quiz



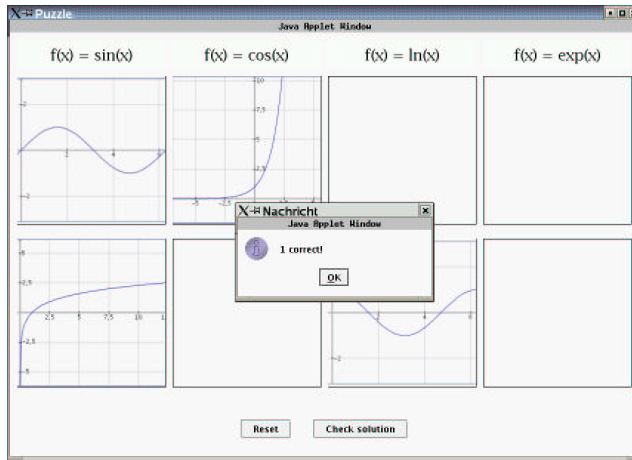
```
<applet code="net/mumie/testletfactory/quiz/CheckBoxTest">
  <param name="lang" value="en"/>
  <param name="fontSize" value="13"/>
  <param name="header0" value="Test 1"/>
  ...
  <param name="main0" value="5 is equal to"/>
  ...
  <param name="question0_0" value="log(100000)"/>
  <param name="answer0_0" value="true"/>
  <param name="question0_1"
    value="ln(e<sup>5</sup>)" />
  <param name="answer0_1" value="true"/>
  <param name="question0_2"
    value="log(5<sup>5</sup>)" />
  <param name="answer0_2" value="false"/>
  <param name="question0_3"
    value="log<sub>3</sub>(243)" />
  <param name="answer0_3" value="true"/>
</applet>
```

Fig. 38: A quiz example and its parameter configuration

The quiz testlet is an applet implementation of the standard multiple choice test. It allows up to 100 multiple choice questions and keeps track on the number of correct and incorrect

answers given by the user. It allows the usage of pictures in questions as well as in answers (individually for each question and answer) thus being open for a wide range of different uses.

Puzzle

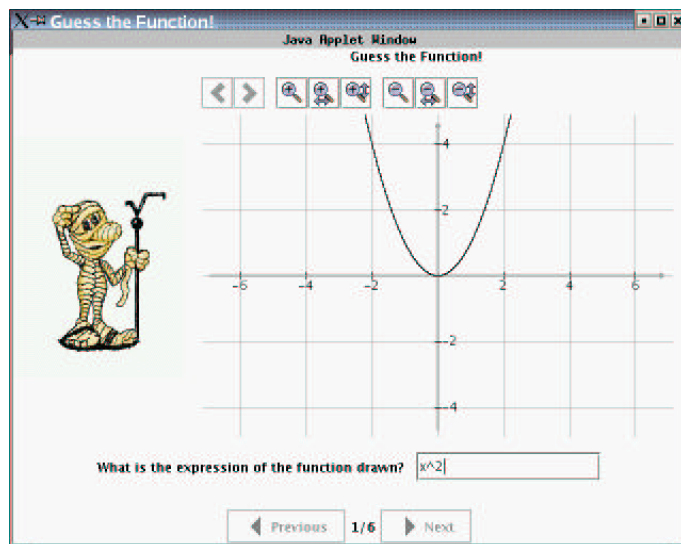


```
<applet code="net/mumie/testletfactory/puzzle/TabularPuzzle">
  <param name="horizontal" value="true"/>
  <param name="fontSize" value="20"/>
  <param name="title" value="Guess the Function!"/>
  <param name="question1" value="f(x) = sin(x)"/>
  <param name="answer1"
    value="http://www.pyramit.de/c_analysis/pre/pic/sinx.jpg"/>
  <param name="question2" value="f(x) = cos(x)"/>
  <param name="answer2"
    value="http://www.pyramit.de/c_analysis/pre/pic/cosx.jpg"/>
  <param name="question3" value="f(x) = ln(x)"/>
  <param name="answer3"
    value="http://www.pyramit.de/c_analysis/pre/pic/lnx.jpg"/>
  <param name="question4" value="f(x) = exp(x)"/>
  <param name="answer4"
    value="http://www.pyramit.de/c_analysis/pre/pic/expx.jpg"/>
</applet>
```

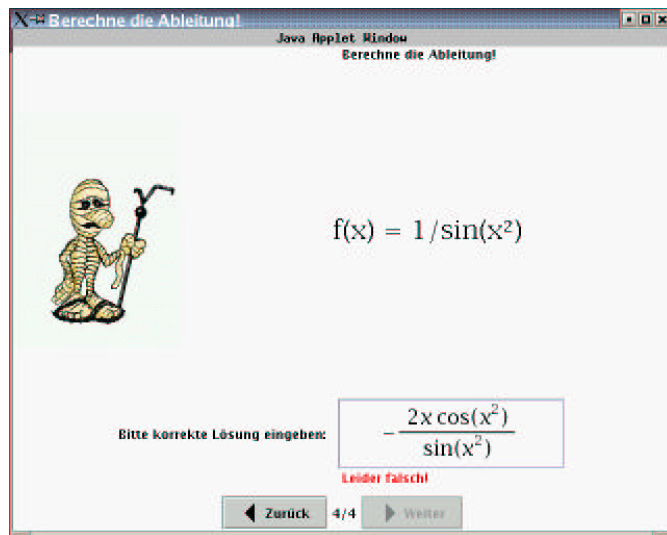
Fig. 39: A puzzle example and its parameter configuration

The puzzle testlet as an applet that allows to ask a user to relate different questions with given answers by moving the answers as puzzle pieces with the mouse (drag and drop). Like the quiz it allows the usage of pictures in questions and answers and the solution can be checked by printing the number of correctly placed pieces.

Word Test



```
<applet code="net/mumie/mathletfactory/.../OpTest">
  <param name="useCanvas" value="true"/>
  <param name="title" value="Guess the Function!"/>
  <param name="answer0" value="x^2"/>
  <param name="answer1" value="cos(x)"/>
  <param name="answer2" value="2x+1"/>
  <param name="answer3" value="x^2-2"/>
  <param name="answer4" value="ln(x)"/>
  <param name="answer5" value="tan(x)"/>
</applet>
```



```
<applet code="net/mumie/mathletfactory/.../OpTest">
  <param name="lang" value="de"/>
  <param name="title" value="Berechne die Ableitung!"/>
  <param name="fontSize" value="25"/>
  <param name="question0" value="f(x) = 3x^2-4x"/>
  <param name="answer0" value="6x-4"/>
  <param name="question1" value="f(x) = cos(x^2)"/>
  <param name="answer1" value="-sin(x^2)*2x"/>
  <param name="question2" value="f(x) = ln(x^3)"/>
  <param name="answer2" value="3/x"/>
  <param name="question3" value="f(x) = 1/sin(x^2)"/>
  <param name="answer3" value="-2x*cos(x^2)/sin(x^2)^2"/>
</applet>
```

Fig. 40: Two word test examples and their parameter configurations

The word test is a sequential test that asks the user different questions, each of which the user answers by typing a mathematical expression. There is a mummy figure commenting on the correctness of the answer. Each time a question has been answered correctly, the user may advance to the next question. Thus it is possible to create tests with an increasing difficulty level. Additionally to posing free text questions the author can also switch to a mode, where the expression asked for is drawn as a function graph – in order to create a ‘reverse function plotter’ game.

Chapter 4

Application

4.1 Application Scenarios

As one might guess from the complexity of the component framework presented in the previous chapter, it has not been developed in a single development cycle, but iteratively in several meso- or micro cycles that arose from different application contexts. The scenarios presented in this chapter therefore form a vital part of our research, because on the one hand they led to a multitude of major improvements concerning the component framework, on the other hand they exhibit its possible uses in teaching and learning mathematics and even its transferability to other subjects.

The application scenarios presented in the next sections are in general chronologically sorted; however, we start with the one that is mostly related to this document, because it demonstrates the application of the component framework in a mathematical eLearning site that uses the design process specified in 2.5.1.

4.2 Applets in the Pyramit System

The Pyramit System¹ is a prototypical eLearning site that has been designed and implemented at the RWTH Aachen university. It serves mainly two purposes: On the one hand it is a functional preview of Mumie JAPS (Java Application Server); on the other hand its aim is the public demonstration and evaluation of mathematical applets within the didactic model specified in Chapter 2. In this section we concentrate on the didactic impact of Pyramit, section C.1 in the appendix gives a brief overview of the technical and practical aspects.

4.2.1 Implementing the Didactic Model

In order to use the didactic process specified in 2.5.1, we need a modular system that states the entities of content, methods and learner orientation.

Concerning the content and methodic structure, Pyramit uses a modular concept. In this structure the *Lesson* (a page represented in the browser) is the smallest independent part of

¹[Py04]

the learning path.²

The process of working through a Lesson is called a *Learning Unit*. A Lesson is widely context independent (and thus also independent from other Lessons) and should be worked through without interruption.

The role of a Lesson in a course can be compared with a school lesson in the context of a teaching sequence, although the average working time for a Lesson should be shorter.

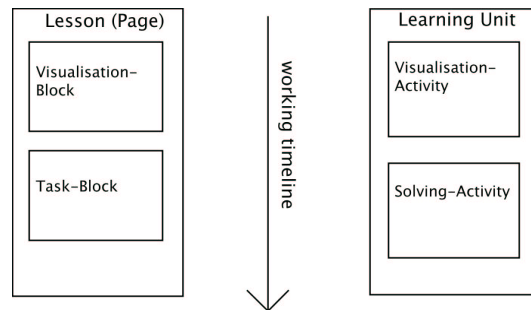


Fig. 41: An example lesson and its corresponding learning unit

Like a School lesson, each Learning Unit is divided into different activity phases. These activity phases are guided by *Building Blocks* in the Lesson. The Blocks are the smallest didactic elements, though they are not context independent, but embedded in a specific learning unit and its underlying Lesson.

In Pyramit, we have the following building blocks:



Element blocks contain definitions, theorems and proofs. To a certain extent they are the mathematical foundations. In contrast to elements inside a textbook, they are only mentioned when they are necessary in the context; otherwise, the lecture notes are referred to as further reading.



Visualisation blocks serve for the depiction of mathematical content. Their role is to connect the formal and the graphical view of a mathematical object wherever possible. By interacting with the visualisation the user may observe and understand the resulting changes on one view by manipulating the other.



Annotation blocks should refer on the one hand to practical meanings of mathematical facts and on the other hand to all sorts of potential sources of misunderstanding.



Routine blocks refer to the practice of a certain computing calculation or algorithm, which may – once internalised – be processed largely automatically. Of course there are additional hints on the importance of basic comprehension of what happens in a routine.

²With respect to the SCORM [ADL04] standard, a lesson can be regarded as a Shareable Content Object (SCO), see also section 5.3.



Brainteaser blocks contain questions that can be answered without the need to perform written calculations. Their aim is to improve the intuitive understanding of mathematical concepts.



Task blocks support the written learning, application and consolidation of the content learned, but also lead to applicational contexts and extensions of the content.



Basic blocks are hints to important knowledge that the student should already have from mathematics at school. They may partly contain short repetitions but more often a request to the student to refer to a class or introductory book.



Link blocks refer to an external resource associated with the subject discussed. They should be used only, if no other block (e.g. annotation, visualisation) seems suitable.

For both the content and methodic model, we apply the taxonomy of Bloom (see 2.4.1, 2.5.3) to structure the building blocks as follow:







| Level | Content Block | Method Block |
|---------------|---|---|
| Application | Annotation  | Task  |
| Understanding | Visualisation  | Brainteaser  |
| Knowledge | Element  | Routine  |

Fig. 42: The content block scheme of Pyramit (From the Pyramit documentation)

4.2.2 Applying the Didactic Design Process

We will now give an example, how the didactic design process developed in 2.5.1 could be applied in practice. We do this by documenting the construction of a lesson (i.e. a single page)

in the Pyramit platform that deals with a core topic in mathematics: Polynomials.

Didactic Analysis Phase

According to our design process, this phase consists of analysing the subject matter with respect to its content, methods and relation to the student. For our example this means, we have to discuss the possible (enactive, iconic and symbolic) representations of polynomials, examine the mathematical methods needed to work with them and look at the role polynomials play for our designated learning group.

Possible representations of polynomials are: Polynomials as functions with their corresponding graphs in \mathbb{R}^2 , as vectors in the polynomial vector space (also symbolic and iconic), polynomials as algorithms performing a sequence of arithmetic operations (e.g. Horner scheme), polynomials as finite power sums, etc.

These representations also come with countless methods: Finding a zero of a polynomial, factorisation and expansion of polynomials, finding a fast algorithm for evaluating polynomials, etc.

Since the learner group consists of engineers in our case, we have to offer contexts that demonstrate the necessity of understanding polynomials. These could be any formula involving polynomial variables (e.g. computing the area or volume of geometric shapes in civil construction), but also dealing with iconic capabilities of polynomials, like describing curves and curved shapes. Furthermore the method of polynomial approximation is a powerful tool when working with more complicated functions. After having examined the technical relevance of the mathematical topic, we also have to keep in mind the special interests of engineers. These are often technology issues related with the ‘hobby-world’ like sports, cars or other things of daily life.

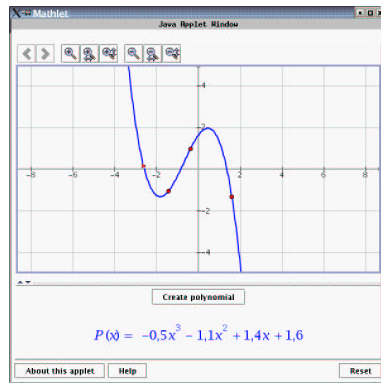


Fig. 43: The polynomial mathlet resulting from the described design process

Didactic Design Phase

After having collected possible representations, methods and contexts for polynomials, we now need to select a subset of each to create a compact lesson that presents a meaningful arrangement of related representations, methods and context applications. In our example, we pick the aspect of curved shapes being described by polynomials, using the creation of car bodies or bridge arcs as instruction anchor. Having selected this context, we need to focus on the polynomial’s representation as function graph. If we want to compute the polynomial function

from a sketched graph we also need a symbolic representation (a formula) and a method how to derive it from specified graphical properties. This means, that a possible task could be ‘compute the polynomial that runs through the following points...’ with the associated method of solving a linear equation system.

Implementation Phase

As a result of the design phase, we implement a mathlet that allows the creation of a polynomial whose graph runs through a set of user defined points (see figure 43) and a page containing a context block³, a visualisation block with the mathlet, a brainteaser block with two exercises concerning the mathlet and a task block containing two exercises that ask the user to compute a polynomial by setting up and solving a linear equation system.

The complete lesson then looks like the following:

The screenshot shows the Pyramit web application interface. The sidebar on the left contains a navigation menu with the following items: Home, Content, About, Analysis, Lin., Algebra, 1. Basic concepts, Functions, Powers, Roots, Polynomials, Exponential function, Trig. Functions, Hyperbolic functions, Conic sections, 2. Number spaces, 3. Sequences and convergence, 4. Series, 5. Continuous functions, 6. Differentiable functions, 7. Integration, 8. Approximation, 9. Applications, 10. Convergence, continuity in \mathbb{R}^* , 11. Differentiation in \mathbb{R}^* , and 12. Extreme. The main content area is titled 'Construction of polynomials' and contains the following text: 'Consider the problem of finding a "smooth" function f for given values $f(x_i)$, such that the graph passes through the points $(x_i, f(x_i))$. The simplest class of functions which solve this problem are the *polynomials*.' Below this text is an image of a silver car and a text block explaining the use of polynomials in engineering. Further down, there is a section titled 'In the following applet you can mark points in the plane by pressing the "c"-key and clicking. By clicking on the button "Create Polynomial" a polynomial will be created, which runs through the points you have selected. After the creation of the polynomial you can still move the individual points with the mouse and observe the corresponding changes of the polynomial.' This section includes a 'Start Applet' button. Below this is a section titled 'Assign in the following puzzle the graphs to the given polynomial expressions:' with a 'Start Puzzle' button. The bottom section contains two exercises: '1. Is there a unique polynomial which runs through a given set of points? What is the role of the degree of the polynomial when determining the formula?' and '2. Create the polynomials $f(x) = x^2 - 4$ and $f(x) = x^2 + x - 2$ with the applet.' The footer of the browser window shows the URL: 'Applet net/nunie/mathletfactory/sample/pyramit/pre/CreatePolynomial started'.

Fig. 44: The lesson resulting from the described design process

³The images for the context blocks in Pyramit were taken from sites that contain either copyright-free or education friendly copyright image sites, see [P4L04], [FI04]

4.2.3 Evaluation Results

As said before, the Pyramit Platform was used for mathlet and testlet evaluation in a minimal eLearning site context in Winter Term 03/04 and Summer Term 04 in the course ‘Mathematik für Bauingenieure I und II’ (mathematics for civil engineers) at the RWTH Aachen University. Because during the course several organisational and technical problems arose (for example, a lot of bugs were found by either the lecturer or the students), we cannot give an evaluation report as comprehensive as the one presented in 1.2.2, but a few important observations have been made by the team:

- As predicted, the students seemed to be mostly attracted by the interactive elements, sometimes neglecting the static texts on the pages completely.
- When examining the applets, students preferred mathlets that had a connection to or helped them for an assignment given in the course over those that were purely made for enhancing understanding. This corresponds to the evaluation results of math online discussed in 1.2.2.
- The lecturer in turn often asked for specialised mathlets that display a certain key concept (but nevertheless including a considerable amount of flexibility, like free choice of functions, points, etc.).
- A common point of critique from the lecturer’s colleagues was the risk of non-reflected use of the applets and the danger of producing incorrect results by pushing mathlets beyond their limitations. To go against this, the lecturer offered an extra weekly lecture in which he showed the proper use of the applets and their limits as an example for the domain limitations of mathematical software. Additionally, several warning dialogues were added to avoid possible misconceptions:

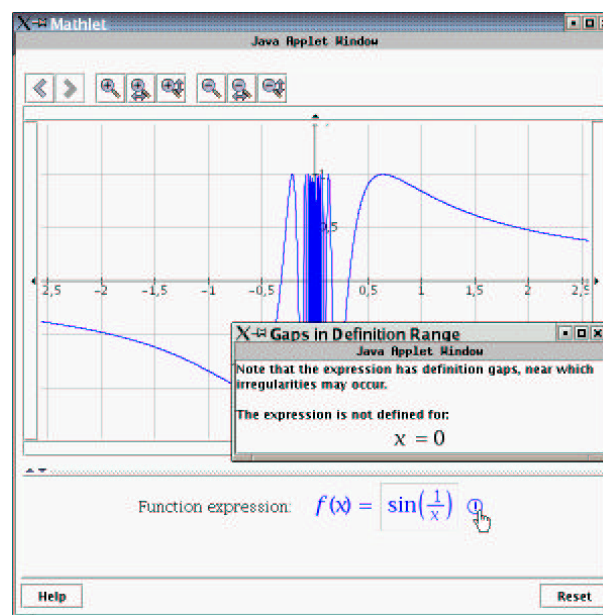


Fig. 45: A generic warning pop-up for operations with definition gaps

These additional checks, again, were made possible by the use of symbolic computation (see B.3.10 in the appendix for details).

4.3 Other Applications in Mathematical eLearning

Since the MathletFactory has been developed in a network of universities, there have been numerous evaluations of mathlets in different contexts and locations. This ensures the practical usability of the framework and led to the specification of further enhancements.

4.3.1 Mathlets for Numerical Mathematics

To test early alpha mathlets in a real life context, M. Holschneider, the head of the MathletFactory project, decided to produce mathlets for his winter term 02/03 numerical mathematics for engineers lecture at Potsdam University to visualise some numeric concepts. These mathlets concentrated on the one hand on the computation and visualisation of polynomials (e.g. an early version of the polynomial mathlet described in the previous section), splines and vector fields, on the other hand on the implementation of fast numerical algorithms that allowed the real time working with mathematical entities like solutions of differential equations etc.

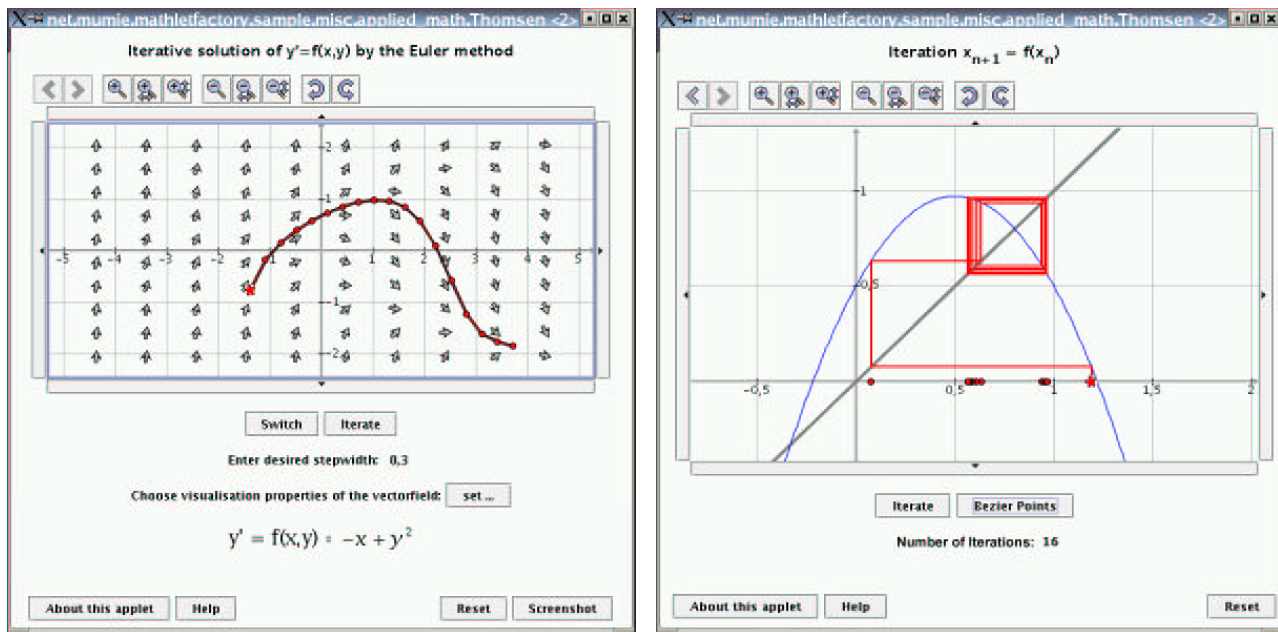


Fig. 46: Two mathlets produced for a numerical mathematics lecture, the highlighted points with their computed poly-lines may be each changed with the mouse without any perceived delay

Evaluation and Development Results

The application of the mathlets produced the following evaluation results:

- The mathlets were mostly presented in the lecture with a reference to the URL for the students to try them at home.

- As part of the examinations of the lecture were in oral form, talks about mathlets were also part of it, thus allowing the students to discuss algorithms and numerics ‘at work’.
- Technically, the evaluation led to a very fast display and computation system that allows the user to continuously move vector fields or multiply iterated systems (see figure above).

4.3.2 Additions in the Mumie Project

For the integration of the Mumie JAPS system, some further additions to the AppletFactory were made, which are also helpful in other scenarios, where a large number of applets has to be handled. These are Quality Assurance and the management of applet metadata.

Applet Quality Assurance

In order to manage and assure the quality of a large number of applets in a distributed content development scenario it is necessary to realise a process of Quality Assurance (QA). This process was designed by the Mumie management group. It uses several states of quality and delegates the responsibility to specific roles:

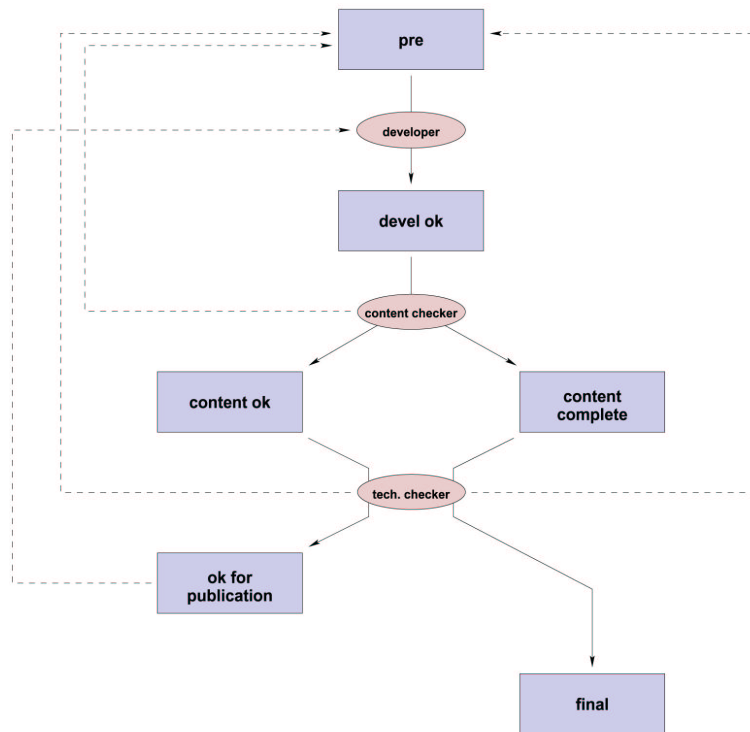


Fig. 47: The applet quality assurance process (from Mumie QA specification)

pre

The applet has been specified by a teacher/author and is in state of development but has not completed any checks for technical correctness and consistency.

devel_ok

The applet has been checked for technical correctness and consistency, but its didactic

properties (usability, colour scheme, initial conditions) and its completeness have not been checked.

content_ok

The applet has been technically checked by the developer and didactically checked by an author or teacher, but requests for enhancements have been made.

content_complete

The applet has been technically checked by the developer and didactically checked by an author and its functionality has been considered complete.

ok_for_publication

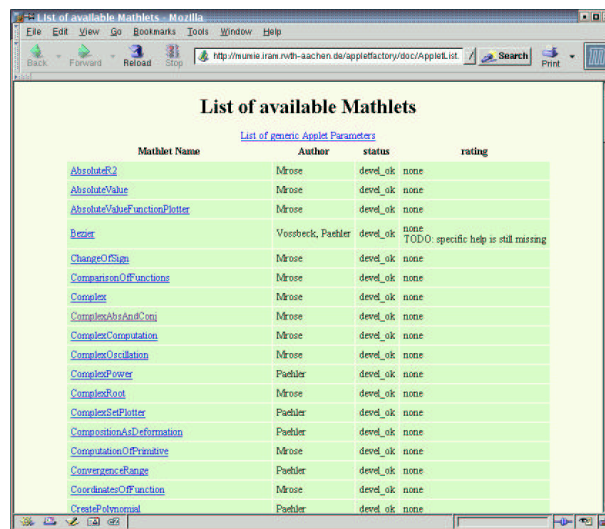
As content_ok, but the applet has additionally been checked by the tech checker, a role that specialises on testing the robustness of applets against irregular behaviour like creating mathematically degenerated conditions (e.g. singular matrices, zero length distances, etc.). This additional QA step has been made necessary by the fact that both author and developer know about the didactic coverage of the applet and thus may not have in mind any student interactions that are outside these considerations. Actually, the Mumie team had often been confronted with technical and didactic errors produced by unexpected student behaviour.

final

As ok_for_publication, but the applet's functionality has also been considered complete.

Mathlet Metadata and Documentation Generation

The QA process is supported on the technical level by the addition of javadoc tags (author, QA status, changelog, todo) to the applet source code. Combined with the implementation specific doclets (Documentation generators using the javadoc API, see [Ja04]) this guarantees a quick overview on the current status of all mathlets developed. Two lists are generated from this metadata, one in HTML format with links to descriptive pages that also contain the mathlet itself (see figure 48) and one in T_EX/PDF-Format (see D.2 in the appendix).



| Mathlet Name | Author | status | rating |
|--|--------------------|---------|--|
| AbsoluteR2 | Mrose | devd_ok | none |
| AbsoluteValue | Mrose | devd_ok | none |
| AbsoluteValueFunctionPlotter | Mrose | devd_ok | none |
| Bezier | Vossbeck, Pachtler | devd_ok | none TODO: specific help is still missing |
| ChangeOfSign | Mrose | devd_ok | none |
| CompositionsOfFunctions | Mrose | devd_ok | none |
| Complex | Mrose | devd_ok | none |
| ComplexAbstraction | Mrose | devd_ok | none |
| ComplexComputation | Mrose | devd_ok | none |
| ComplexOscillation | Mrose | devd_ok | none |
| ComplexPower | Pachtler | devd_ok | none |
| ComplexRoot | Mrose | devd_ok | none |
| ComplexSetPlotter | Pachtler | devd_ok | none |
| CompositionAsDeformation | Pachtler | devd_ok | none |
| ComputationOfPrimitive | Mrose | devd_ok | none |
| ConvergenceChange | Pachtler | devd_ok | none |
| CoordinatesOfFunction | Mrose | devd_ok | none |
| CreatePolynomial | Pachtler | devd_ok | none |

Fig. 48: The list of available mathlets as a web page generated from mathlet source code files

4.3.3 School Applications

Because a lot of the mathlets created for Mumie focused on content also taught at school, the idea of testing them for school applications was brought up. Showing the mathlets to teachers and asking them, if they were relevant to school teaching resulted in a ratio of about 70% of all mathlets. The testlets were even all considered useful for dealing with school math issues, although most teachers at first seemed reluctant to create configurations of their own. But using HTML-editors like FrontPage solved the problem even for teachers that had only basic skills with computers.

Application scenario

To test mathlets in a ‘real life’ scenario, a teacher offered to prepare a series of lessons dealing with numerical methods of finding zeros of a function. These lessons were held at an 11th grade mathematics course in a German secondary school (Gymnasium). In the lessons two applets (one for each lesson) were introduced: One for demonstrating Newton’s method and one for demonstrating the interval bisection method.

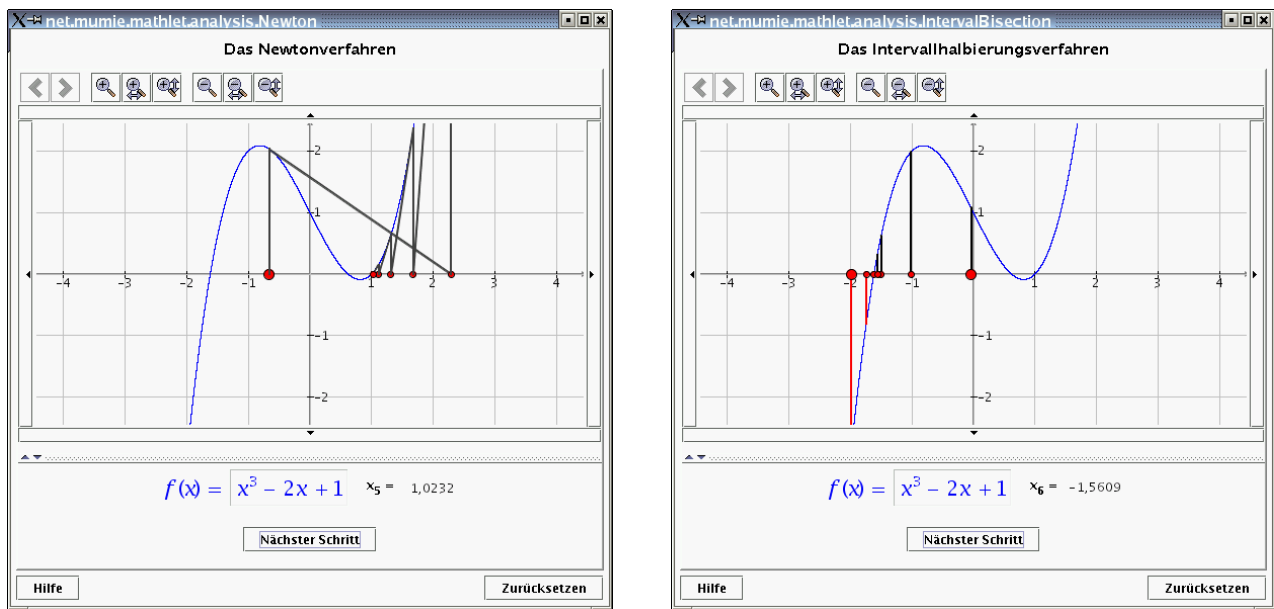


Fig. 49: Newton’s and the interval bisection method mathlets used in the school scenario

Each of them was shown at first to the students, asking them to describe the algorithm that was used by each. Afterwards they had to calculate by hand, what the computer had done and to verify their results with the mathlets. Regarding the concept of reflected computer use (see introduction), this was considered an important if not central part of the lesson. Having completed some additional exercises for each of the methods they were asked to compare them with each other, which resulted in a controversial discussion about the advantages and disadvantages of the methods that was enriched by some examples thrown in by the teacher.

Conclusions

In the evaluation questionnaire⁴, all students (fully or partially) agreed, that the mathlets helped to understand the numerical methods, almost all agreed that it was worth the effort using them. The teacher was also pleased by the results achieved and remarked the motivating effects of the mathlets. Another teacher (owning only basic skills in computers) even reused the series without further technical efforts for a teaching exam ('Lehrprobe'), so that it can be stated that once the basic conditions of having the pages written and installed on the school computer system are met, there are only low technical barriers of using them. This seemed especially to be true for the students, as the lessons revealed, that there were no problems with the technical use of the mathlets and almost all students evaluated the handling of the mathlets as 'simple and intuitive'.⁵

The teachers in turn seemed a bit more reluctant, which is why we offered several training events for teachers and compiled a CD as giveaway for increasing the acceptance of the Mumie system (see Appendix E). This CD is easy to install and use, which is another advantage of the AppletFactory using a purely client side approach.

4.4 Transferability

In order to sketch the transferability of the Mumie AppletFactory to extra-mathematical contexts, we give two short examples, how this has already been done.

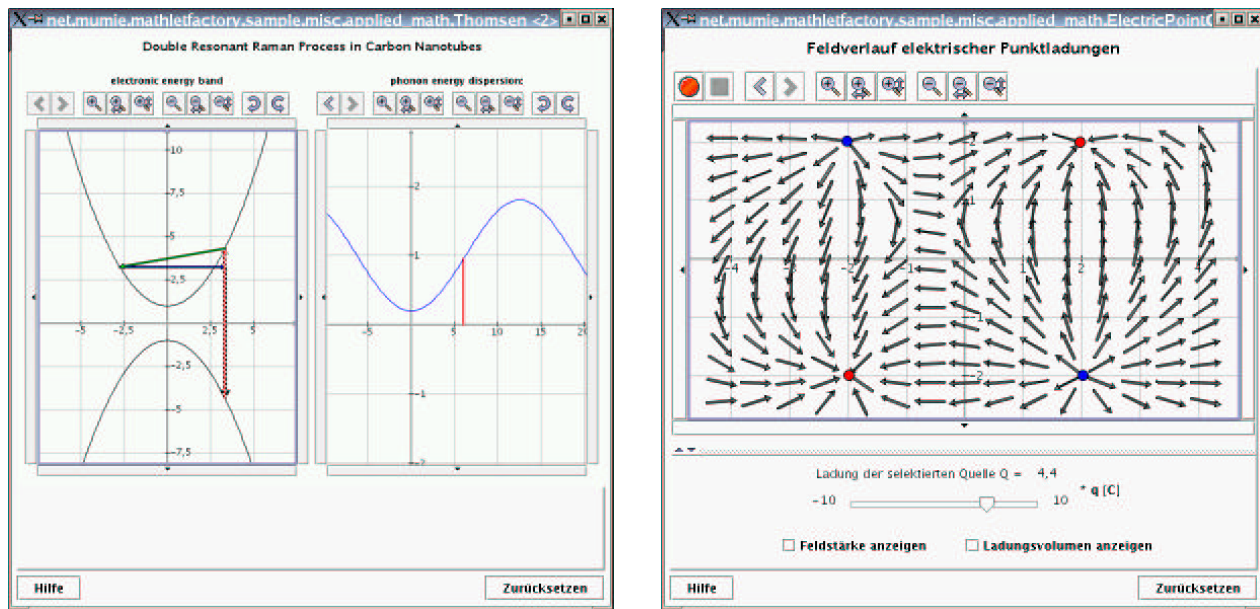


Fig. 50: Two applets presenting subjects of physics and electrical engineering

⁴[Ma04]

⁵Cite URL with html pages and results

4.4.1 Mathlets for Physics and Engineering

In addition to using the mathlets for the creation of various mathematical eLearning sites, there have been some efforts to use them also in other fields of science. This originated from the interdisciplinary structure of the developer team including physicist and engineers as members. For example, a student of basic studies in electrical engineering used the mathematical vector field object to create a mathlet that displays the 2D electrical field of an arbitrary point charge distribution. Another transfer involved a cooperation of the working group of mathematical physics and the Institute of Solid State Physics, creating a mathlet that visualises phonon energy configurations (see figure 50). In addition to these mathlets there are plans to produce mathlets that specifically deal with specific topics of civil engineering (plain structures and their associated linear equation systems, elastic deformation of beams, etc.).

4.4.2 TestletFactory

Transferability is an especially interesting aspect of the TestletFactory, for its design is already quite independent from the content its instances are dealing with; additionally, almost any scientific knowledge may be tested using a puzzle or quiz. This may explain, why there were several requests for the system, ranging from members of psychology and medical faculties to physics and chemistry. The cooperation most advanced at the time of writing was the support of an international project dealing with hydrology engineering. In order to construct an eLearning system for the University of Cairo, the TestletFactory was used as a client side testing framework, configured by dynamically generated server parameters from a database. One critical issue considered, was the feedback of the student's answers to the database. For this reason the TestletFactory was enhanced by a communication protocol between the applets and the enclosing page, using a Java-JavaScript link.⁶ This allows the number of question correctly answered by the student to be sent back to the server, making it possible to save the result of an online examination for a previously identified student.

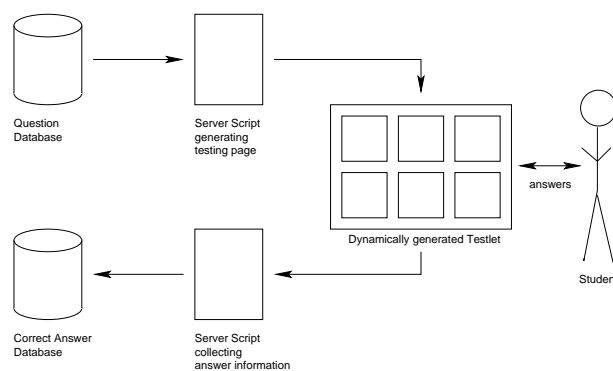


Fig. 51: Integrating testlets into a server side database framework

⁶For a demonstration, see [Te04].

Chapter 5

Conclusions and Outlook

5.1 Summary

Looking back at the previous chapters, we may state that focusing on interactivity and technical reusability has led to a fruitful amount of applets for mathematical eLearning sites. The flexibility of the approach (platform and network independence) and the strong regard for requirements in a role-based scenario model allowed its application in a multitude of different scenarios.

The practical and didactic analysis of mathematical eLearning (Chapter 1 and 2) led us to the development of schemes that are useful for designing mathematical eLearning content and to methods guiding the design process. By differentiating between macro and micro level (Chapter 1), we were able to state a simple concept of reusability and to locate the functionality that promises the highest amount of interactivity. These results were complemented by a didactic perspective on interactivity (Chapter 2) that indentified its key role in a learner oriented and constructivistic learning scenario. In addition to that, the resulting content model specified in 2.3 also offers a perfect base for an implementation of interactive content within an object-oriented Model-View-Controller architecture. Combined with the methodic model based on Bloom's Taxonomy and with some practical aspects of learner orientation we were able to present a comprehensive didactic design process to be applied for the creation of any mathematical eLearning site.

The implementation of the component system presented in Chapter 3 provides a rich and generic interface to mathematical objects and allows the construction of complex mathematical dependencies, thus making it possible for application developers to rapidly implement the visualisation of a specific mathematical scenario. By the use of applet parameters, the author has the ability to configure compiled applets for his needs without having to deal with programming issues. On the other hand the support for changing the applet's behaviour at run time by entering mathematical expression leads to an increase of flexibility and reusability also for students and tutors.

The application scenarios presented in Chapter 4 show a wide range of different uses for the component framework. Their evaluation and the quality assurance process mentioned in 4.3.2 led to substantial improvements of applets as well as the component system itself. For example, the 3D navigation and the animation framework evolved through several cycles of development and feedback from users. The various results of evaluation also give us confidence for the next macro cycle (see page 12) we are aiming at: The convergence with other eLearning technologies

and the propagation of standards and reference systems in web-based mathematical eLearning.

5.2 Towards Critical Mass

The last chapter has shown, that by using the Mumie AppletFactory, the rapid development of complete interactive courses in mathematical eLearning comes into reach. As said before, the framework has been going through an extensive series of tests and applications, thus positioning it as a fairly mature and stable system in the – yet sparsely cultivated – field of mathematical web-based client frameworks.

To tap the full potential of mathematical eLearning systems, however, some crucial tasks have to be accomplished, mostly concerning integration:

- Most authoring tools are still high-technology, requiring specialised knowledge. The AppletFactory alleviates this issue by making high quality applets available to content developers without specialised Java skills (application developers) or none at all (authors), but basic HTML skills are still required for configuration. Though the use of editors like FrontPage or Dreamweaver eases this problem, a more specialised authoring tool would widen the range of potential authors. This problem can, however, only be addressed together with the authoring of static content:
- The market of server platforms and authoring tools is still too fragmented and unstandardised to initiate a wide movement of content creation. Pushing and narrowing standards (XML, MathML, etc.) or de-facto standards (e.g. \TeX) for eLearning would increase content stability with respect to migration to another platform. The Mumie project therefore clings completely to standards and platform-independence, but even this leaves too much room for arbitrary implementation in many cases. However, regarding other fields of software development one might predict that an increasing convergence of platforms will eventually lead to stability and standardisation.
- Though much progress has been made over the past years, the basic conditions of web-based learning are still not at best in German universities. Showing a mathlet in auditorium, for example, required not only the laptop of the lecturer being connected to a beamer, but also hiding all blackboards and thus all relating notes and formulas written onto them. A tighter integration of electronic media and conventional hand writing by using systems like E-Chalk¹ or eCase² has not yet outweighed the advantage of simplicity of conventional chalkboard writing. But regarding the recent success story of electronic presentation usage in lectures might give a guess, that future lecture rooms will be surely adapted to media integration.
- The acceptance of mathematical eLearning (as academic eLearning in general) has not yet reached a critical mass of users to reinforce the demand of further integration.

It is our thesis, that enforcing the latter issue will eventually also remedy the other problems. That is why we tend towards implementing reference systems which demonstrate the state of the art. Apart from the Mumie system this aims also at other platforms.

¹[EC00],[EC04]

²[eC04]

5.2.1 Integration with Other Platforms

At the RWTH Aachen University, a special task force of eLearning experts has been formed to publish exemplary eLearning solutions in order to inspire and support other lecturers willing to follow suit. For mathematical eLearning this could lead to the integration of Mumie content into other platforms. This is actually planned for the Emilea-stat project, which offers an eLearning site dealing with mathematical statistics.³ On the one hand this contributes to a coalescence of contents that are of complementing mathematical subject, but whose publication faces the same technical problems we described in 1.2 and C.1. On the other hand it serves as a demonstration, that the content created is platform independent through the use of standards and de-facto standards (Java, XML and TeX). Beyond this, there are considerable efforts of cooperation in the convergence of mathematical eLearning/-Teaching/-Research (eLTR) technologies⁴ with German and European universities that are documented by the proposals for the projects ‘ULearn(Math)’⁵ and ‘Multiverse’⁶. These activities both contribute to the establishment of stable publishing standards and aim to attract further authors, who are right now waiting for sustainable tools before investing considerable work in creating high quality eLearning courses.

5.3 Further Fields of Research

Now that a framework for the rapid development of mathematical content exists, there are various branches of research that may extend the present work. In accordance with the last section, these are mainly directed at the integration with other frameworks and applications.

5.3.1 Large Scale eLearning Content

To improve the acceptance of mathematical eLearning, practical results as well as research in several disciplines are needed. Evaluation of applications within different courses will be required, from a psychological and educational perspective as well as from a mathematical and technical perspective. By the availability of a framework for the rapid development of interactive content, it is now possible (and has only just begun) to create complete online mathematics courses. This opens fields of further research for the generic structure of eLearning media in contrast to conventional learning material especially with respect to sustainability and maintenance.

5.3.2 Navigability and Adaptivity

The system presented in this document concentrates on the reactivity of mathematical eLearning, there are few theses about the adaptivity and navigability, delegating these issues to the server framework. For a complete interactive system, however, they are nevertheless crucial. We therefore observe studies in this field with growing interest.⁷ Technically, the ADL SCORM

³[Em04]

⁴[JSK04]

⁵[UL04]

⁶[Mv04],[1]

⁷There are, for example, several projects involving AI methods for user adaptivity in Mathematical eLearning, like [AM04], [IM04].

Sequencing and Navigation Specification⁸ offers a detailed standard for Learning Management Systems (LMS), how user navigation and tracking are to be implemented, so there is a good chance, that future LMS might offer a good testing ground for developing content with improved navigability and adaptivity. At the time of writing, the content presented by Pyramit or JAPS does not yet provide SCORM-compliant metadata, but its modular structure will easily allow the addition of metadata and its (re)use in various sequencing models.

5.3.3 Authoring Support

Though the task of strengthening the author role has been a crucial part of our research activities, it is far from being complete. We experienced in our evaluations, that authors need a minimum of technical complexity to be able to concentrate on the production of high quality content. Anything that distracts from this task reduces the author's willingness and commitment, which may prove fatal for the usage of a tool, that is most often done with a voluntary effort to improve the quality of teaching.

In the Mumie project, an authoring environment is in development, that integrates applet testing and content production. In addition to that, we are aiming at a tighter integration with existing authoring tools (e.g. HTML/XML editors).

5.3.4 Integration with Server Applications

Though we have achieved our goal of developing a reusable system for mathematical components, the Mumie AppletFactory is but one contribution to the realisation of a seamless electronic infrastructure in mathematics teaching and research. As one may see from the success of office packages in modern computer usage, the success of a software depends heavily on its integration with other programs. This is especially true in the WWW-server world. We therefore have already added the support for applet parameters in order to let the server dynamically create specific (even user-adapted) configurations of the same applet. But right now, there is no mechanism yet that lets a mathlet tell a server, how the student interacted with it. However, the first steps in this direction have already been taken by implementing a callback mechanism for testlets (see 4.4.2).

Another promising direction would be the outsourcing of computing power to specialised systems like CAS or numerical calculation software, thus allowing even analysis of complex mathematical expressions or the production of high precision results. A first step has also been taken for this by adding Content MathML support to the MathletFactory Algebraic Object System, thus allowing the easy export of mathematical expressions to CAS like Mathematica or Maple (see Appendix B).

⁸[ADL04]

Appendix A

Transformation examples of mathematical concepts

Here is a small list of informal example transformations of mathematical concepts into iconic and enactive representations. It was used to design the mathlets created with the MathletFactory. As said before there may be also no or more than one suitable transformation for each mathematical concept – this list is merely a help for practical purposes.

| Mathematical Concept | Iconic Transformation | Enactive Transformation | Example Mathlet(s) |
|----------------------------|--|--|--|
| Number | Measure (length, area or volume) that is the multiple of a unit | Counting/Increasing a measure | SeriesPlotter |
| Function | Function graph or transformation of an equidistant point grid | Feeding an input/output machine with values, transforming an iconic structure (line, collection of points,...) | FunctionPlotter, FunctionAsDeformation |
| Composition of functions | subsequently transformed point grids | Feeding the input of the outer function with the output of the inner | CompositionAsDeformation, FunctionCompositionPlotter, FunctionCompositionAnimation |
| Inverse function | function graph mirrored across the line $y = x$ | Swapping input and output | InverseFunctionPlotter |
| Limit of a sequence | A point for which every ball (or disc) around it leaves only a finite number of points outside | Coming closer to a point | SequencePlotter |
| Continuity | A contiguous graph of finite length | Drawing a graph without raising the pen (roughly), finding an $\varepsilon(\delta)$ relation (advanced) | EpsilonDelta, DeltaOfEpsilon |
| Derivative of f | Displaying the tangent's slope in points $(x, f(x))$ | Constructing the derivative by drawing the tangent's slope for several points of the graph of f | TangentAndDerivative |
| Integral $\int_a^b f(x)dx$ | Area between graph and x-axis | Approximating the area with different methods (Upper and lower sum, etc.) | StepFunctionPlotter, LeftRightApproximation, TrapezoidApproximation |
| Linear independence | n vectors spanning an n -dimensional subspace | Failing in constructing one vector as a linear combination of the other | GeneratingSystemOfR2 |

Appendix B

Implementation Details

This appendix contains an overview of the MathletFactory from a system developer’s perspective. This perspective is necessary for developing new MMathObjects or display components that extend the set of mathematical entities represented by the MathletFactory.

In the following sections we omit the details, which can be found at the API documentation but give a structural overview that follows the Model-View-Controller architecture.

B.1 MVC Architecture of the MathletFactory

B.1.1 Requirements

Following the didactic model of Bruner, mathematics can be regarded as a system with three complementing representations: The *enactive* representation of a mathematical entity is determined by what you can *do* with it, the *iconic* representation is an image or sketch that *visualises* one or more of its properties and the *symbolic* representation *denotes* it in a formal language system. One of the main conclusions of Bruner’s Theory is, that although professional mathematicians almost exclusively use symbolic representations, the other types of representations play a vital role in learning mathematics.

If we use this didactic principle as a requirement for the architecture of a mathematics learning framework, it makes sense to allow different representations for mathematical objects. This can be done best by using the Model-View-Controller Pattern¹, an architectural pattern that separates the data of an entity from its presentation and application logic.

B.1.2 Fundamental Concepts

A good starting point when describing the MathletFactory is to describe what happens, when a student uses an applet created with the MathletFactory.

If, for example, a student drags the graphical representation of a three dimensional vector on a canvas with the mouse, the canvas generates an event that is sent to the `CanvasController`. This instance checks all objects contained in the canvas if they are meant, by using the mouse coordinates and the canvas’ internal projection parameters. If an object has been picked and it can handle the type of event (by owning an appropriate handler), the event is delivered to

¹[Bu96]

it by calling `MObjectIF.doAction()`. The `MObject` then delegates the event to the specified handler, which processes the event (e.g. translating the end-point of the vector in a plane parallel to the viewport by transforming the new mouse coordinates to world coordinates) and modifies the state of the `MObject` (e.g. setting its coordinates to new values). After this it is checked, if any other objects depend on the vector (for example, the vector could be designated as normal for a plane). For this it is checked, if any `Updater`s are associated to the object and if so, their `update()` method is called, changing the state of any dependent objects (which in turn might also have other updaters and so on). After the update graph traversal has been finished, The views of the objects that were changed, are redrawn (both the graphical and the symbolic representation), by invoking `render()` and `draw()` in all transformers of the `MObject`. By this the student is informed of the result of his action and may proceed with further actions.

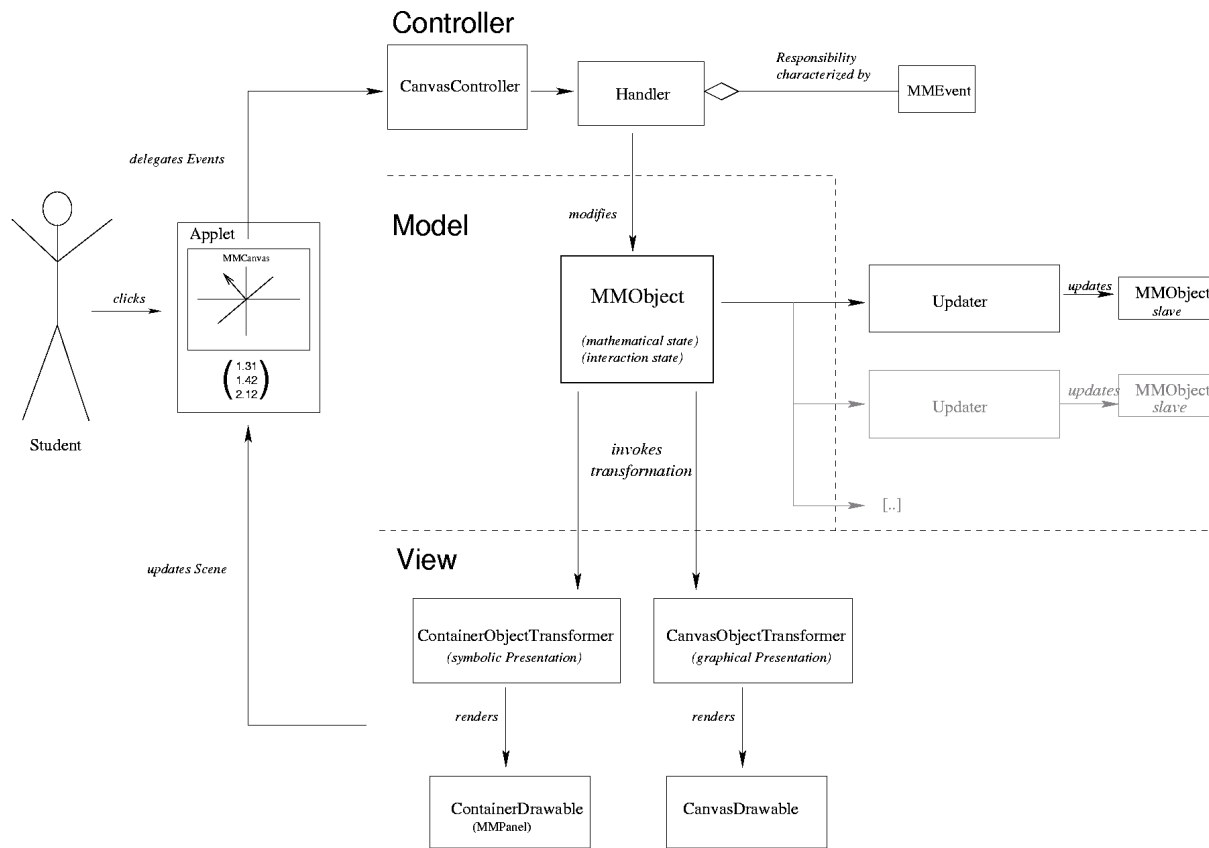


Fig. 52: An action cycle in the MVC architecture

B.1.3 Model

The core model used in the MathletFactory is the *MObject*, represented by a class that implements the `MObjectIF` interface. An instance of this class contains on the one hand the mathematical information of the object, on the other hand it also owns references to the handlers that allow its manipulation and to updaters that connects the object with other `MObjects` in the update graph. It also contains the link to the view components.

B.1.4 View

The view component consists basically of two objects: The *transformer* (represented by a subclass of `GeneralTransformer`) the *drawable* (represented by the interface `Drawable`). While the drawable is the actual displaying unit, the transformer establishes a link between model and drawable and knows how to repaint it, when the `MObject` has changed.

B.1.5 Controller

In the controller section we differentiate between objects that directly manipulate `MObjects` and those that are responsible for constructing the update graph. The directly manipulating objects are represented by two different classes: Handlers (subclasses of `MMHandler`) for manipulating iconic representations in a canvas (e.g. selecting and dragging vectors with the mouse) and panels (subclasses of `MMPanel`) for editing symbolic representations. Note that the panel is also a drawable into which the controller-code (which consists of only a few lines) has been integrated into.

B.2 Arithmetic and Geometric Model

In order to compute a wide variety of mathematical problems, the Mumie MathletFactory offers a flexible and economic model for representing the geometric and arithmetic properties of mathematical entities and allows an accessible interface for easy manipulation. In the following, we briefly describe the model architecture of the basic entities.

B.2.1 Number Types

Starting with the specification that all mathematical objects – directly or indirectly – use numbers, we assign to each `MObject` a specific number type that can be changed upon construction or sometimes even at run time. By this it is, for example, possible to use the same sequence object for displaying both real valued sequences and complex valued sequences. In the first case the object is instantiated by invoking the constructor with the argument `MComplex.class`, the second with the argument `MDouble.class`. The implementation is done using an abstract base class `MNumber` of which all number types are subclasses. Currently, the following number types exist:

| Number Type | Set of numbers modelled | Internal Java type used |
|---------------------------|--------------------------|-------------------------|
| <code>MNatural</code> | \mathbb{N} | <code>BigInteger</code> |
| <code>MInteger</code> | \mathbb{Z} | <code>double</code> |
| <code>MRational</code> | \mathbb{Q} | <code>long</code> |
| <code>MBigRational</code> | \mathbb{Q} | <code>BigInteger</code> |
| <code>MDouble</code> | \mathbb{R} | <code>double</code> |
| <code>MComplex</code> | \mathbb{C} | <code>double</code> |
| <code>MZmod5</code> | $\mathbb{Z}/5\mathbb{Z}$ | <code>int</code> |

The last is an experimental type that demonstrates the use of finite fields in the MathletFactory and will soon be replaced by a generic class modelling $\mathbb{Z}/p\mathbb{Z}$.

B.2.2 Vectors, Vector spaces and Matrices

While the subclasses of `MNumber` provide the base for one-dimensional number computing, the class `NumberMatrix` – representing an $m \times n$ -matrix – is fundamental for the calculation in linear spaces of higher dimension. As `MObjects` it has also exclusively uses the generic number interface allowing each instance of a `NumberMatrix` to represent a matrix of different number type.

The `NumberMatrix` is extended by `NumberTuple`, a class that represents $m \times 1$ -matrices and which is basically used as coordinates for vectors or matrix columns/rows. It therefore offers additional functionality like the norm, the dot product, etc.

Vectors in turn are modelled by subclasses of the abstract `NumberVector`. It is a specific trait, that each vector of the MathletFactory ‘knows’ the vector space in which it exists and is represented by coordinates that are relative to its associated basis. By this it is possible to transform all vectors of a chosen vector space by changing the space’s basis. The `NumberVectorSpace` class therefore provides a wide range of methods to manipulate its basis.

B.2.3 Affine and Projective Geometry

The vector space model is also used in the geometric classes. Like most CAG (Computer Aided Geometry) Modelling software. All internal data is stored in homogeneous (i.e. projective geometry) coordinates. This allows the easy transformation of mathematical entities also for affine geometry. For example, when calculating the intersection of two planes in the three dimensional space \mathbb{R} (i.e. the intersection of 2D affine subspaces within a 3D affine space) simply the projective hyperplanes have to be intersected, reducing the problem to finding the null space of the matrix that contains the projective coordinates of the subspaces’ basis (see extended description of this example below).

B.2.4 Numerical Computing

Numerically, almost all affine and projective geometric operations – like the example above – base on the Gauss algorithm implemented in the class `EchelonForm`. This ensures a high reuse and offers an easy optimisation opportunity: If the Gauss algorithm is made faster, a wide range of computations will be executed faster.

B.2.5 Compound Example

To demonstrate, how all the concepts described above work together, we give a ‘real life’ example: In the mathlet depicted below, the intersection of two planes in three dimensional is computed and displayed (For details refer to the documented source code):

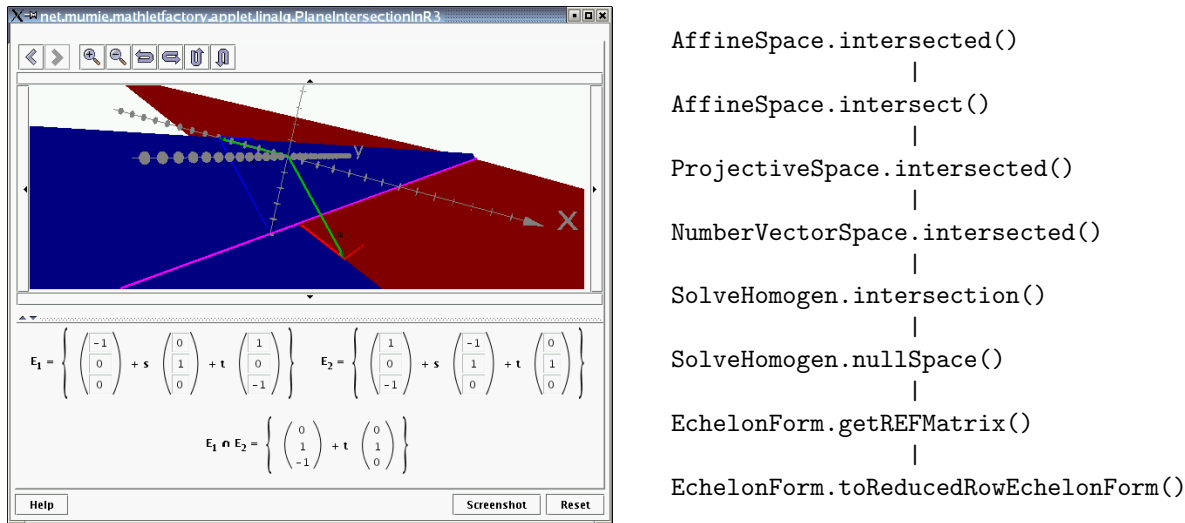


Fig. 53: The intersection of two planes in three-dimensional space and its stack trace to the Gauss algorithm

This starts in by invoking the method `AffineSpace.intersected(AffineSpace with)` in one of the planes with the other as argument (the plane class descends from the affine space class). This method does nothing but the invocation of its projective representation to intersect with the projective representation of the other plane. This is done by giving their two vector bases to the method `SolveHomogen.intersection(NumberTuple[] span1, NumberTuple[] span2)` which in turn calls the `nullSpace(NumberMatrix matrix)` method with a (3×6) matrix as argument that contains all the base vectors as columns. This method in turn uses the Gauss algorithm implemented in `EchelonForm.getREFMatrix(NumberMatrix matrix)` to transform the matrix to reduced echelon form to determine its null space basis. The null space basis is then used as parameters for constructing the basis of the intersecting space, which is returned by the `intersected()` method.

B.3 Algebraic Object Model and Formal Languages

Since often it is not only needed to let the application developer perform computations with the MathletFactory but also the student (or teacher/tutor) himself, the arithmetic and geometric model is complemented by an algebraic model that allows the input of expressions in a formal language. By this, the flexibility of the mathlets is increased and allows them to be used as tools in open learning scenarios.

B.3.1 Lexical, Syntactic and Semantic Analysis

In order to analyse and interpret a formal language, we need structures that operate on three different levels of language: the first is the lexical analysis which analyses the alphabet of symbols being used, the second is the syntactic analysis that applies the rules for building words and sentences. On the third level we have the semantic analysis that analyses the meaning of the words.

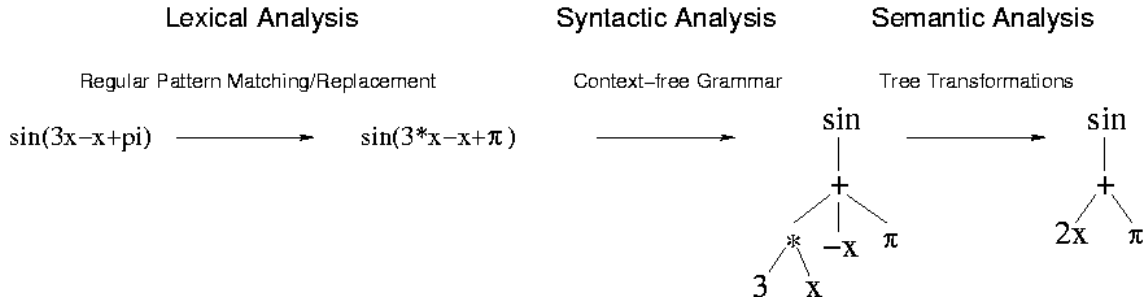


Fig. 54: The different stages in mathematical expressions analysis performed by the MathletFactory

In the MathletFactory, each stage of language processing is performed by a specific unit: The lexical analysis is performed by a set of regular expressions (which also do some replacements that allow an increased robustness like $2x \rightarrow 2*x$) and a small scanner unit. The syntactic analysis is done by a parser that implements a context-free grammar (see below) and the semantic analysis is left to a rule-based tree automaton that operates on the operation tree generated by the parser.

Note that it is quite easy to reproduce a string out of the tree representation by doing a depth-first order traversal, thus closing the sequence to a loop. This is very important for interactive work with mathematics, where the user enters an expression, watches the response of the system and may want to re-edit his input for a receiving a different result.

B.3.2 Introduction to Formal Languages

The main idea of the MathletFactory's algebraic object model takes advantage of the formal languages used in mathematics. Computer science has developed a rich set of methods to interpret these formal languages of which we can only give a short introduction, see [HU79] for details.

A formal language can be defined as a concatenation of symbols from an alphabet. These concatenations are called the *words*. The formal language $\mathcal{L}(\Sigma)$ over an alphabet Σ that consists of all possible words can thus inductively be defined as:

1. $\sigma \in \mathcal{L}$ for all $\sigma \in \Sigma$.
2. $w \in \mathcal{L}$ for $w = u.\sigma$ with $u \in \mathcal{L}, \sigma \in \Sigma$.

with $.$ being the concatenation operator. The language that consists of all words over Σ is also called Σ^* , where the $*$ -operator means, that the resulting set contains all finite concatenations $\sigma_1.\sigma_2 \dots \sigma_n, n \in \mathbb{N}$ of symbols $\sigma_i \in \Sigma$, and the empty word ε .

Of course we are more interested in languages that allow only certain words. For example, we may want $(x+1)$ to be a word of our language, but not $+)1)x$. We thus need a higher structure that tells us, which words belong to the language, a *grammar*.

A grammar is defined by the accepted alphabet (the *terminals*) and by a set of explicit rules how a word can be decomposed into these. For example, one could state the following rules for simple arithmetic expressions of numbers represented by NUM:

1. $+, -, *, / \in \mathcal{L}$, $\text{NUM} \subset \mathcal{L}^\dagger$
2. $w \in \mathcal{L}$ for $w = u.*.v$ or $w = u./v$ with $u, v \in \mathcal{L}$,
3. $w \in \mathcal{L}$ for $w = u.+v$ or $w = u.+v$ with $u, v \in \mathcal{L}$.

We have already differentiated between products and sums of rational numbers, because for the semantic analysis (i.e. evaluating the arithmetic expressions) it is necessary to consider the precedence of operations. One could define grammars like we did in the example above, but for handling more complex grammars it is useful to state a grammar in the Backus-Naur form (BNF), which regards grammars as a tuple (T, N, s, R) , where $T = \Sigma$ is the set of terminals, N is the set of nonterminal symbols (i.e. variables that may contain concatenations of terminals), s is the name of the starting variable (i.e. the variable whose content is tested to be a valid word of the specified language) and $R \subset (N \cup T)^*.N.(N \cup T)^* \times (N \cup T)^*$ is the set of rules that need to apply for a word of the specified language:

$G(T, N, s, R)$

```

T:    NUM, '+', '-', '*', '/'
N:    expr, term, fac
s:    expr
R:
(1)   expr    ->    term { '+' term } | term { '-' term } .
(2)   term    ->    fact { '*' fact } | fact { '/' fact } .
(3)   fact    ->    NUM.
```

We see, that the operator precedence is ensured by using different variables for summands and for factors. The expression $3*4+1$ could thus be tested by the grammar as follows (we enclose the symbol with its type in parentheses):

$$\begin{aligned}
(\text{expr } 3*4+1) &\xrightarrow{(1)} (\text{term } 3*4).+(\text{term } 1) \\
&\xrightarrow{(2)} (\text{fact } 3).*(\text{fact } 4).+(\text{fact } 1) \\
&\xrightarrow{(3)} (\text{NUM } 3).*(\text{NUM } 4).+(\text{NUM } 1).
\end{aligned}$$

After this none of the rules can be applied to the expression anymore which means that $3*4+1$ is a word specified by G . This testing method also gives an idea, how a parser that accepts words of $\mathcal{L}(G)$ could be implemented: By a recursive reduction algorithm, where each rule is modelled by an according method.²

[†]Note, that for avoiding trivial rules it is better to regard a number like **1234** to be represented by a single symbol, not as a concatenation of symbols as one would presume by the fact that it is a concatenation of digits in our common writing system. In the following we will also regard function identifier like **cos** as a single symbol in order to keep our grammar compact.

²This is called the Top-Down approach, which is mainly used in functional or rule-based language implementations, imperative language implementations often also use a Bottom-Up approach, where for each step only as much symbols are read in as needed for applying the next rule.

B.3.3 Types of Grammars

Noam Chomsky has provided a hierarchy of grammars where each type produces a language that is a subset of a language produced by a lower type. The type of a grammar depends completely on the specified rules R :

| Type | Name | Constraints for R |
|------|-------------------|--|
| 1 | context sensitive | $ u \leq v $ for all $R : u \mapsto_G v$ |
| 2 | context free | $R \subseteq N \times (N \cup T)^*$ |
| 3 | regular | $R \subseteq N \times (\{\varepsilon\} \cup T \cup N.V)^*$ |

This means for example, that regular grammars produce only a subset of context free grammars, which in turn produce a subset of context sensitive languages. Which type suits our needs? There are some features for which we need at least a context free grammar. For example, an expression containing parentheses is only valid in mathematics, when every opening parenthesis has its closing counterpart. On the other hand we want to use parentheses on all levels. This means we need a rule $r \in R$ that has the form $\text{prim} \rightarrow '(\text{ expr })'$, which is not possible for regular grammars, which allow no terminals on both sides of a non-terminal. Context sensitive grammars in turn would allow rules with non-terminals and terminals mixed on the left side thus allowing a higher semantic analysis already in the syntax phase, one could, for example, add something like with the following rule:

$\text{NUM } '*' ' (' \text{ VAR } '+' \text{ VAR } ') ' \rightarrow ' (' \text{ NUM } '*' \text{ VAR } '+' \text{ NUM } '*' \text{ VAR } ') ' ,$

thus implementing the distributive law for words containing two variables **VAR** multiplied with a number **NUM**. But as the reader might guess, a program that parses context sensitive languages is hard to implement, so we do things like this in a separate semantic analysis step (see B.3.8).

B.3.4 From Syntactic to Semantic Analysis

After the parser has accepted the mathematical expression, we need to transform it into a data structure that allows easy semantic analysis. In our case it is suitable to represent it as a syntax tree or as a word of a regular tree language [Co03]. For example, $\sin(2x+\pi)$ is interpreted as $(\text{sin } (+ (* 2 \text{ x}) \text{ pi}))$. The expression can then be symbolically transformed by using tree automata (see below) or numerically evaluated, which is done by a recursive procedure that evaluates the tree from the leaves (numbers or variables and parameters with assigned values) up to the root node.

B.3.5 Formal Languages Used by the MathletFactory

The MathletFactory uses two formal languages, which are both context free and read by a recursive descent parser [ASU86]: The operation language *Op* and the relation language *Rel*.

The Operation Language *Op*

The language *Op* is used for modelling algebraic operations as they occur in functions, equations, etc. It can be used by any mathematical object that can be characterised by a numerically

evaluable symbolic expression. The grammar of Op is as follows:

$G(T, N, s, R)$

T: NUM, VAR, SIN, COS, SINH, COSH, EXP, ASIN, ACOS, LN, SQRT, '+', '-', '*', '/', '^',
N: expr, term, fac, pot, prim
s: expr
R:
(1) expr \rightarrow term { '+' term } | term { '-' term } .
(2) term \rightarrow fact { '*' fact } | fact { '/' fact } .
(3) fact \rightarrow SIN pot | COS pot | SINH pot | COSH pot | EXP pot | LN pot | ABS pot
| ASIN pot | ACOS pot | TAN pot | ATAN pot | SQRT pot | FLOOR pot | pot.
(4) pot \rightarrow prim { '^' prim } .
(5) prim \rightarrow VAR | NUM | '(' expr ')' | '|' expr '|' .

The Relation Language Rel

The language Rel is used for modelling algebraic relations as they occur in set definitions, propositions, etc. As one might already guess from the fact that relations consist of operations, words of Op are part of the letters of Rel . More precisely, the Rel terminal **SIMP** is a simple relation that contains a left and right hand side $\text{expr}_i \in Op$, which are separated by a relation sign ($=, \neq, \geq, \leq, >$ or $<$). The grammar of Rel is as follows:

$G(T, N, s, R)$

T: SIMP, NOT, AND, OR, NOT
N: rel, cla, sub, prim
s: rel
R:
(1) rel \rightarrow sub { OR sub } .
(2) sub \rightarrow cla { AND cla } .
(3) cla \rightarrow NOT prim | prim.
(4) prim \rightarrow SIMP | ALL | NULL | '[' rel ']' .

Note that in Rel we use square brackets for overriding precedence, whereas in Op we use parentheses, allowing to parse a complete relation containing operations in a single pass.

B.3.6 Tree Architecture

After the syntactic analysis (i.e. the construction of the operation/relation tree from a user/application provided string), the semantic analysis (i.e. transformation or evaluation of the tree) takes place, again initiated by user or application action. We demonstrate this in a short example:

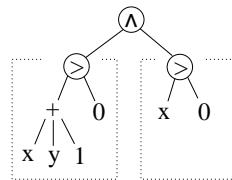


Fig. 55

In the figure above the relation $x + y + 1 > 0 \wedge x > 0$ is displayed in its tree representation (e.g. as a parsed result of the string 'x+y+1>0 AND x>0'). This tree has two levels: a relation level (the encircled nodes) and an operation level (the nodes below the relation nodes). The

relation consists of \wedge -Conjunction as root node with two *simple* relations ($x + y + 1 > 0$ and $x > 0$, the dashed boxes in the figure) as leaves. The relation leaves contain the operations $x + y + 1$, 0 and x , 0. After binding x and y to certain numeric values, the relation can be evaluated, returning either **true** or **false**, depending on the evaluation results of the operations.

The approach of using tree representations for mathematical expressions is adopted by all modern Computer Algebra Systems (e.g. [Wo91], [Wa91], [Mo93]), but their almost purely functional model allows neither typing (e.g. no type distinction between operations and relations)³ nor the integration in an object oriented graphical user interface.

For our requirements, we will adopt the concept of a functional representation, but merge it with an object oriented model. This means for the architectural design that we use the different types **Operation** (any symbolic expression that can be numerically evaluated) and **Relation** (any symbolic expression that can be either true or false). We choose these entities because they are closed under the most transformations we will apply onto them (for example, opposed to the set of equations – the set of relations is closed under equivalence transformations, the set of analytic operations is closed under derivation), so we can still use a functional model when transforming trees of each type.

Tree Representation vs. Flat Representation

Apart from being a construct of formal languages over which human-computer interaction is possible, a tree architecture also grants a greater flexibility than flat structures. For example, if we want to model a finite representation of a Borel σ -algebra in \mathbb{R} this could be implemented by a single class, owning a list of intervals, upon which the operations **intersect**, **join**, **complement** etc. are resolved. This can be quite costly if we want to compute the join or intersection of two Borel sets that are widely ‘scattered’, though the user will usually test inclusion in the result only for some points or a single interval displayed on the screen. Also, we could not model the special case of infinite intervals like \mathbb{Z} , $\mathbb{R} \setminus 2\pi\mathbb{Z}$, etc. which we need when displaying periodic behaviours.

The solution to this problem is to implement the Borel set as a tree structure that has intervals and ‘periodic intervals’ as leaves and operations upon Borel sets as inner nodes. When the user wants the set to be displayed on the screen, the observed interval and an ε for precision (e.g. pixel width) is given to the Borel set and a distribution of points and lines satisfying these parameters is computed.

As this example suggests, constructing trees of mathematical entities allows a higher degree of generalisation without losing flexibility and simplicity. We will also see in section B.3.10 that trees are easily transformable, adding a lot of functionality to tree-structured mathematical entities.

B.3.7 Basic Tree Model

All tree-organised implementations of mathematical entities share a common base class, the **AbstractTreeNode**, which offers abstract tree functionality regardless of type, like adding or removing children, searching and replacement of descendants, etc. From these, the basic nodes

³For example, in Maple variables can hold relations as well as boolean or numerical values, making expressions like `y = (y=1);` or `plot(sin(true), x=0..Pi);` computable.

for mathematical entities are derived (at this time `OpNode`, `RelNode` and `SetNode`), the subclasses of which are the concrete implementations of mathematical operations and operands. The tree of these nodes is referenced by the model of the mathematical entity (`Operation`, `Relation`, `BorelSet`), which serves as static container and fixed reference when performing tree transformations:

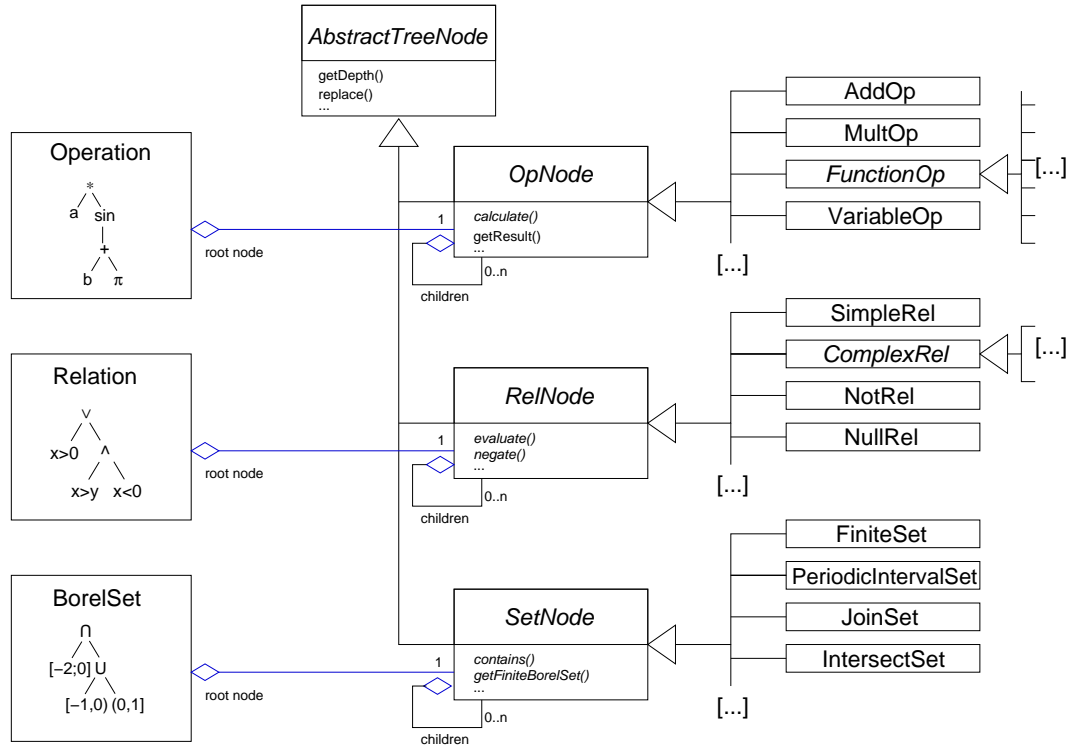


Fig. 56: Inheritance and aggregation model of tree nodes

B.3.8 Object Model of Operations

Generation and Structure of Operations

The Language *Op* is syntactically analysed by an recursive descent parser. When parsing an expression it creates a corresponding operation tree that is contained in an `Operation` object. The operation tree consists of nodes that are of type `OpNode` and whose inner nodes are functions and operations while the leaves are basically variables and numbers. For example, parsing the expression " $\sin(2x+\pi)$ " generates the operation tree (`sin (+ (* 2 x) pi)`):

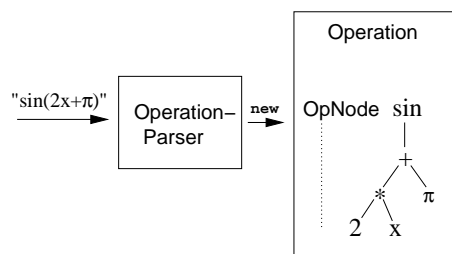


Fig. 57

The `Operation` class bundles the functionality for any operation that can be found in a function or on the left or right hand side of an equation, etc. It contains a reference to the operation tree and keeps track of state information like the variables used and normalisation status (see below).

The operation tree is made of instances of the class `OpNode`, which is an abstract class that provides the common functionality for all operation nodes. It models an elementary operation with a factor and an exponent so the above example could also have the form `(sin (+ 2x pi))`. From the functional view the factor and the exponent are not essentially necessary, since there are also nodes for power and multiplication operations, but with addition and multiplication being the most common operations, this reduces the depth of most trees. This again decreases the costs of analysis and synthesis of larger trees and allows a higher amount of order within the trees (for example, children of a multiplication node can be ordered by power, making fraction handling easier). Adding factor and exponent fields to an operation node is also a common technique used in CAS [Bau02].

Below, a part of the definition of `algebra.op.OpNode` is shown:

```
...

public abstract class OpNode implements Cloneable, Comparable, NumberTypeDependentIF {
    ...
    /** The base value, which is calculated in {@link #calculate}. */
    protected MMNumber m_base;

    /**
     * A numerical factor is stored for each operation, to reduce the tree complexity
     * and allow group actions.
     */
    protected MMNumber m_factor;

    /**
     * The exponent is stored for each operation, to reduce the tree complexity
     * and allow group actions.
     */
    protected int m_exponent = 1;

    /**
     * The children of the node, may be null (leaves), one node (e.g.\ functions) or
     * an arbitrary number of nodes (e.g.\ multiplication).
     */
    protected OpNode[] m_children;

    /** The direct ancestor of this node in the Operation Tree. */
    protected OpNode m_parent;

    /** The number class being used. */
    protected Class m_numberClass;

    ...
}
```

Operation Transformations and Normal Form

As mathematics often consists of transforming expressions, operations can be transformed in multiple ways. For example, the addition of the number 3 to an existing operation can be done by creating a new addition node that has the old operation tree and a '3'-node as children and taking the addition node as the root node for the new operation. There are of course many other possible transformations like substitution, separation of variables, factorisation, derivation, etc. To implement these in an object model, a two level approach is used: Transformations that

create a new operation tree with specific rules for each node (like calculating the derivative or the inverse) are handled by the node object itself, whereas transformations that alter the existing tree structure (expansion, simplification, etc.) are handled by a transformer facility called `OpTransform`. This two-level approach allows to combine the power of recursive algorithms with the structural advantage of object-orientation.

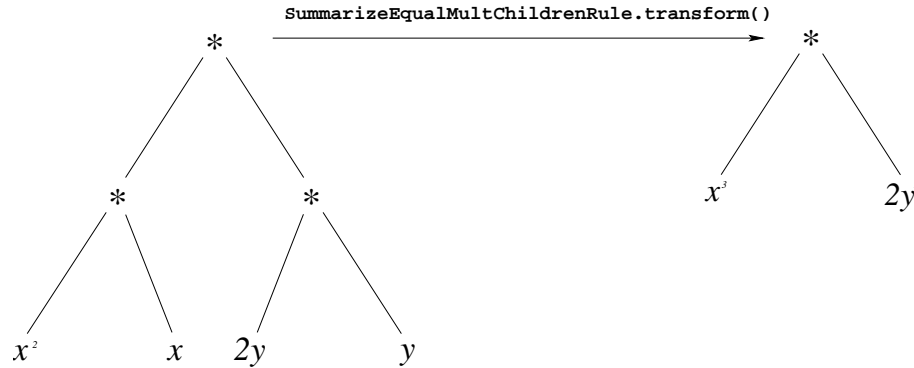


Fig. 58: Transformation of a subtree by a rule

The `OpTransform` object is a tree automaton that transforms trees by using a set of rules (located in the subpackage `algebra.op.rule`), which specify a certain subtree pattern and a transformation that is performed if the tree matches the pattern.

For example, in the tree $(* (* x^2 x) (* 2y y))$ (see figure) the condition for a multiplication node containing two or more equal children (without regarding their factors or exponents) applies to both '*' children of the root node. The rule object therefore raises the power of the first child by the number of the other children, transforming the tree to $(* x^3 2y^2)$. This would also apply to the root node if a previous application of `CollapseEqualOpsRule` had flattened the tree to $(* x^2 x 2y y)$.

In order to apply a rule like in the example above, a certain structure of the tree has to be assumed. For example, the tree $(* (* (^ x 2)) x)$ is mathematically equivalent to $(* x^2 x)$, but it is harder to analyse and transform. Because the rules should be kept as simple as possible (since there are many of them and their set should be easily expandable), it is reasonable to specify a normal form for operation trees.

The normalisation rules are located in `algebra.op.rule.normalize`. Here are some examples:

| Name | Application Example |
|---|---|
| <code>NormalizeMultRule</code> | $(* 4x 6y) \rightarrow 24(* x y)$ |
| <code>NormalizeExponentsRule</code> | $(* x^4 y^6) \rightarrow (* x^2 y^3)^2$ |
| <code>CollapsePowerRule</code> | $(^ (^ x y) z) \rightarrow (^ x (* y z))$ |
| <code>HandleFunctionSymmetryRule</code> | $(\sin -x) \rightarrow -(\sin x)$ |

This set can be easily expanded, most of the rules consist of less than 20 lines of code. All the rules are applied in a deterministic order and since they always produce defined results, it can be ensured that after none of the rules can be applied to any node in the tree anymore, the operation has a unique defined form. This form is defined as the normal form of the

operation (for the specific rule-set). Trees having the same normal form are considered to be mathematically equal by the system.

B.3.9 Object Model of Relations

Generation and Structure of Relations

Analogous to operations, relations are usually generated by parsing a word of *Rel* (see the grammar in B.3.5) and by constructing a relation tree composed of **RelNodes**. The inner nodes of the relation tree are the logical operations \vee , \wedge and \neg , while the leaves are either simple relations (equations or inequations) or special nodes like **AllRel** and **NullRel**, which either relate everything (making interpretation of the used domain necessary) or nothing.

Transformation of Relations, Normal Form

Like operations, relations can be transformed in multiple ways, the most used transformations are the logic transformations that negate, conjunct or disjunct subtrees. But there are also complex transformations that occur when a simple relation is transformed. For example, if the inequation $x^2 - x > 0$ is divided by x this leads to the equivalent complex relation $x - 1 > 0 \wedge \frac{1}{x} > 0 \vee x - 1 < 0 \wedge \frac{1}{x} < 0$ (see figure).

The functionality for handling relation transformations and its special cases are implemented in the class **RelTransform**.

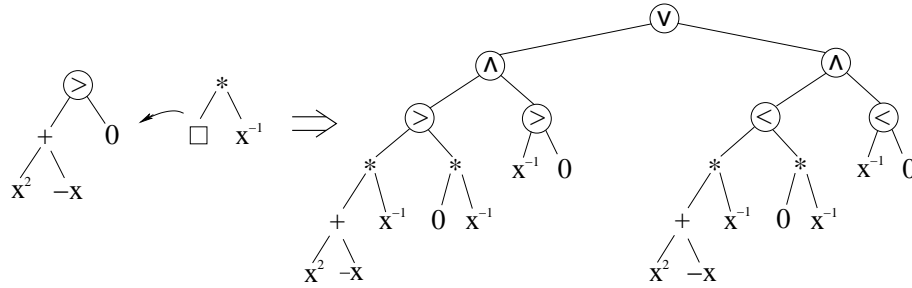


Fig. 59: Transformation of a relation

For comparison and simplification of relations, again, a higher order is needed for the trees to facilitate the development of tree analysis and transformation code.

For example, the relation $x = y \wedge y = x$ should be recognised as redundant by the system and be simplified to $x = y$. As for operation trees, this is done by defining a normal form for relations and implementing rules for normalisation.

The normalisation rules are located in `algebra.rel.rule.normalize`. Here are some examples:

| Name | Application Example |
|----------------------------------|--|
| <code>MoveOrUpwardsRule</code> | <code>[[x=0 OR y=0] AND z=0] → [x=0 AND z=0] OR [y=0 AND z=0]</code> |
| <code>CollapseComplexRule</code> | <code>[[x=0 AND y=0] AND [z=0] → [x=0 AND y=0 AND z=0]</code> |
| <code>RemoveNotRelRule</code> | <code>[NOT [x>=y]] → [x < y]</code> |

B.3.10 Applications

To demonstrate the practical use of the algebraic object model, we give two application examples.

Symbolic Derivation

The symbolic derivative of a differentiable function can be computed automatically by simply applying the well known rules for elementary functions and chaining them together for compound functions, using the derivation rules for sums, products and compositions.

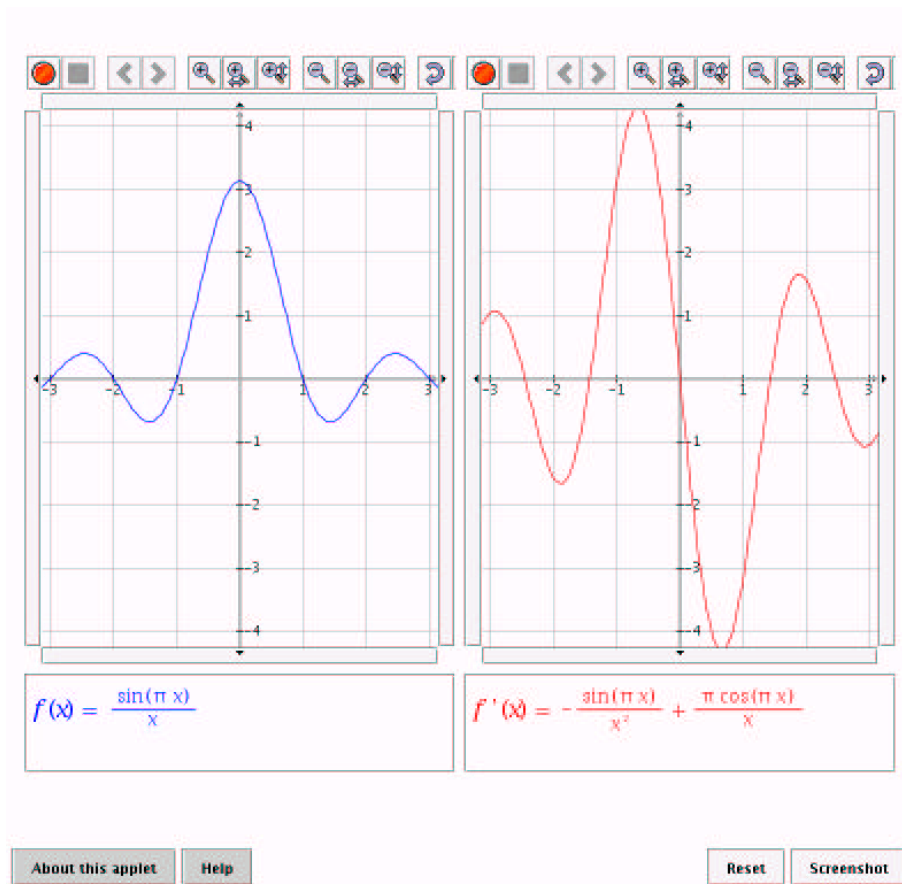


Fig. 60: The function and derivative plotter

The application presented to the user is an applet that plots an arbitrary function typed in by the user and additionally computes and displays the derivative (if any exists).

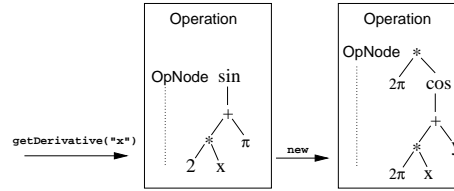


Fig. 61: Object model for deriving operations

From a technical perspective this is done by using an `MMFunctionDefinedByOperation` object from which another instance is created with an `Operation` that is returned by the `getDerivative()` method of the first operation.

Deriving an operation can be implemented completely on a per-node basis, therefore the method `Operation.getDerivative()` simply delegates the calculation to the root node of the operation tree. The derivation code of an operation node has two parts: a generic part that does the derivation for the internal factor and exponent and a specific part that varies on the node type. Below is a code snippet containing the node specific derivation code in the class `SinOp`:

```
/**
 * Implements <i>(sin(f(x)))' = cos(f(x)) * f'(x)</i>.
 */
public OpNode getDerivative(String variable){

    if(getMaxPowerOfVariable(variable) == 0)
        return new NumberOp(m_numberClass, 0);

    // cos(f(x))
    OpNode cosOp = new CosOp(m_numberClass);
    cosOp.setChildren(new OpNode[]{(OpNode)m_children[0].clone()});

    // f'(x)
    OpNode derivedChild = m_children[0].getDerivative(variable);

    // cos(f(x)) * f'(x)
    MultOp derivedCosOp = new MultOp(m_numberClass);
    derivedCosOp.setChildren(new OpNode[]{cosOp, derivedChild});
    return deriveNode(derivedCosOp);
}
```

The generic part implemented in the abstract class `OpNode` is as follows

```
/**
 * Implements <i>(m*a(x)^n)' = (n*a(x) ^ (n-1) * m*a'(x)</i>.
 * @param derivedNode a'(x)
 */
protected OpNode deriveNode(OpNode derivedNode){
    if(m_exponent == 1){
        derivedNode.setFactor(m_factor.copy());
        return derivedNode;
    }
    MultOp derivedPower = new MultOp(m_numberClass);
    OpNode newPower = (OpNode)clone();
    newPower.m_factor.mult(NumberFactory.newInstance(m_numberClass, m_exponent));
    newPower.m_exponent--;
    derivedPower.setChildren(new OpNode[]{newPower, derivedNode});
    return derivedPower;
}
```

Definition Range of Operations

Another useful application of the MathletFactory algebraic object model is the ability to compute the definition range of an operation. We have used this functionality to add an extra warning when displaying mathematical entities that use operations which are only partially defined (see 4.2.3).

When determining definition gaps, it poses a serious implementation problem to numerically calculate the complete definition range and display it graphically for any function (as the example $f(x) = \frac{1}{\sin(\frac{1}{x})}$ in the figure below shows).

The MathletFactory addresses this problem by displaying the definition range of a function as a symbolic expression. Though this expression often has an implicit form (showing only a relation of the used variables that must be satisfied) the user is warned of existing definition gaps, and he is principally able to find out where they are.⁴

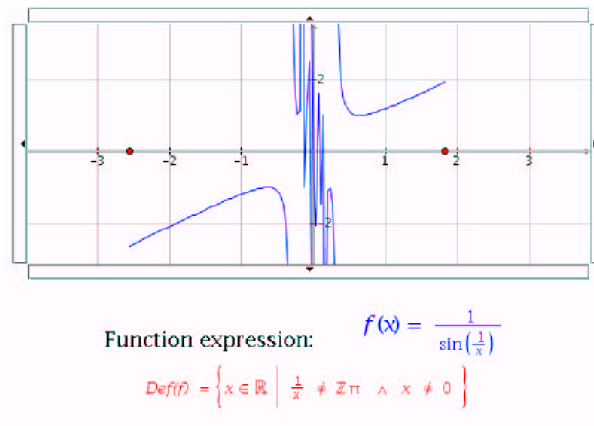


Fig. 62: The function plotter with definition range detector

From the technical perspective the computation of the definition range for an arbitrary operation is also solved on a per-node basis. By calling the method `getDefinedRel()` the `OpNode` object creates a `RelNode` object which represents a relation that a member of the definition range for the operation must satisfy. The `RelNodes` of the children of an `OpNode` are connected by an `AndRel` conjunction, forming a relation tree that is at last anchored in a `Relation` object by the enclosing `Operation.getDefinedRelation()` method. For the `OpNodes` there is – like the computation of the derivative – a generic and a node-specific implementation. The generic part, implemented in `OpNode` is as follows:

⁴Also, by using the `Relation.toContentMathML()` export method, an application programmer is able to link it with a computer algebra system or to implement a numerical solution of his own.

```

/**
 * Returns the relation for which the operation represented by this node is defined.
 * @see #getDefinedRel(OpNode operand)
 */
public RelNode getDefinedRel(){

    // by default the operation is defined totally for any variable
    RelNode definedRel = new AllRel(m_numberClass);

    // retrieve the definition range of this node with the child as argument
    // (non unary operations overload this method)
    if(m_children != null)
        definedRel = getNodeDefinedRel(m_children[0]);

    // for nodes with a negative exponent the base may not become zero
    if(getExponent() < 0){
        OpNode nodeWithoutExponent = (OpNode)clone();
        nodeWithoutExponent.setExponent(1);
        definedRel = new AndRel(definedRel, new NotRel(nodeWithoutExponent.getZeroRel()));
    }

    // intersect the definition range of this node with the definition range of the children
    if(m_children == null)
        return definedRel;
    else
        return new AndRel(definedRel, getChildrenDefinedRel());
}

/**
 * Returns the relation subtree for which this operation with
 * <code>operand</code> as child is defined. It does not consider the
 * exponent of this node, which is checked in {@link #getDefinedRel}.
 */
public abstract RelNode getNodeDefinedRel(OpNode operand);

```

The specific part is implemented by the subclasses of `OpNode` in `getNodeDefinedRel()`. Here is an example for `TanOp` returning a relation that says that the cosine of its operand may not be zero:

```

public RelNode getNodeDefinedRel(OpNode operand){
    return new NotRel(new CosOp(m_numberClass).getZeroRel((OpNode)operand.clone()));
}

```

B.4 Arithmetic and Geometric Symbolic View Architecture

One pattern that has been consequently used in the development of the symbolic view architecture is, that every mathematical inclusion is transformed to a ‘containedness’ relation on the view component level. This allows an easy development and support for a multitude of symbolic representations. We illustrate this with the example of constructing the parametric view for an affine plane in \mathbb{R}^3 :

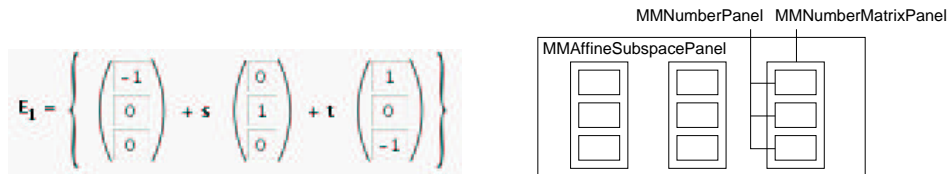


Fig. 63: The symbolic view of a plane and its structure

On Java level, the view is simply a subclass of `JPanel`, namely an `MMAffineSubspacePanel`. This class acts not only as a view for a plane in \mathbb{R}^3 , but also for a line or point in \mathbb{R}^3 or in \mathbb{R}^2 , thus making it possible to use it as a view of a dynamically generated affine subspace (like the intersection or join of other affine spaces). But let us first consider, that it displays a regular (non-degenerated) plane. In this case, we need to display three vector displays: one displaying the origin vector and two for the direction vectors. This is simply done by adding three `MMNumberMatrixPanels`, which is the standard display component for a vector of any dimension, but also of course for any number matrix of arbitrary form. These panels in turn contain all entries as `MMNumberPanel`, the symbolic representation for numbers of any type.⁵ A simple update mechanism using the Java property support ensures, that changes performed by the user are recorded by the master `MMAObject`. On the other hand, if the mathematical state of the `MMAObject` changes (e.g. by updating or user interactions on the graphical level), the changes are immediately displayed, allowing the user to continuously watch the symbolic perspective of his actions.

This applies not only to changing the vectors of the affine subspace, but also to changing its dimension: If the plane was the result of an intersection of two other planes (which were geometrically identical) and one of these planes changed, making the intersection a line or an empty space, the symbolic view adapts to these cases without complaints.

B.5 Algebraic Symbolic View Architecture

In the following, we will also sketch the view architecture used by the MathletFactory's algebraic object model, for details the commented source code and API documentation should be consulted.

The architecture for symbolically displaying algebraic entities is closely related to the structure of the algebraic object model: A tree of operation nodes is mapped on a tree of view nodes that recursively draw the expression on a panel, whereas a tree of relation nodes is mapped on a tree of panels with each parent containing its children.

B.5.1 View Architecture of Operations

The implementation of the symbolic view for an operation is the `OperationPanel`. This is a GUI component that draws the expression string on its screen area when asked to repaint. This is done by view nodes (an analogon to the `TeX nodes`⁶), each of which corresponds to an operation node of the `Operation`. Apart from the reference to its operation node, a view node keeps track of its metrics which is determined by the metrics of its children (if any) and the font of the panel. For example, the view node for a square root must know the width and height of its child expression in order to fully enclose the radicand (see figure).

So when a repaint event is sent to the operation panel, the panel asks its root view node to draw itself on the panel. If the root view node has children, it asks them to calculate their metrics (which in turn may depend on their children's metrics, etc.) and then draws the expression it

⁵Note, that the inclusion does not end there, for the number panel itself contains an operation panel (see below) to allow the display of constants like $\frac{3}{2}\pi i$, etc.

⁶[Kn82]

represents accordingly.

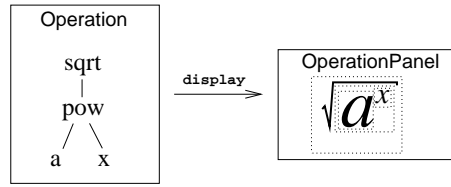


Fig. 64: Displaying operations, the dashed lines mark the OpViewNodes that draw on the OperationPanel.

B.5.2 View Architecture of Relations

The inclusion of operations in relations can also be transferred to the view architecture: The view component for a simple relation, a `SimpleRelationPanel` is merely a panel containing two `OperationPanels` with a relation sign label between them. Complex relations are displayed by instances of `RelationContainer`: Panels that contain either `SimpleRelationPanels` or other `RelationContainers`. The root node itself is contained in a component called `RelationPanel`.

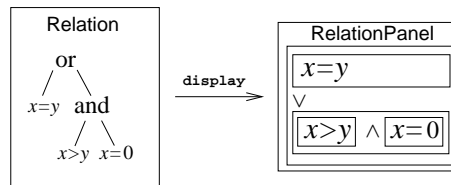


Fig. 65: Relation trees are rendered in a container hierarchy rooted by an RelationPanel.

B.5.3 Metrics of View Components

The MathletFactory uses the standard metrics model of typography⁷: The metrics of each glyph is characterised by its width, ascent and descent; for the rendering of fractions, sub- and superscripts the baseline must also be recorded. Operation view nodes keep their metrics parameters in an object called `ViewNodeMetrics`. On the panel level (anything that uses `OperationPanels` or container trees of these) this is done by implementing the interface `Alignable`, which declares methods for retrieving the metrics parameters. By using this interface it is ensured that two operations can be horizontally aligned (i.e. having the same baseline), even if they have different heights or one of them has a border (e.g. for marking it as editable).



Fig. 66: The baseline, ascent and descent of a font.

⁷[He93]

B.6 Graphical View and Controller Architecture

At the time of writing, implementations for the Java2D system library and the Java3D API exist, but previous prototype implementations have also been tested with the third party graphics library JavaView⁸. The architecture works as follows: For each different display type there exists a specific canvas (a subclass of `MMCanvas`) that can be added to an applet like any other GUI component. If an application programmer wants to display a certain `MObject`, he simply calls `addObject(MMObjectIF object)` in the canvas with the `MObject` as argument and the systems automatically assigns the appropriate (or a previously chosen) visual representation.

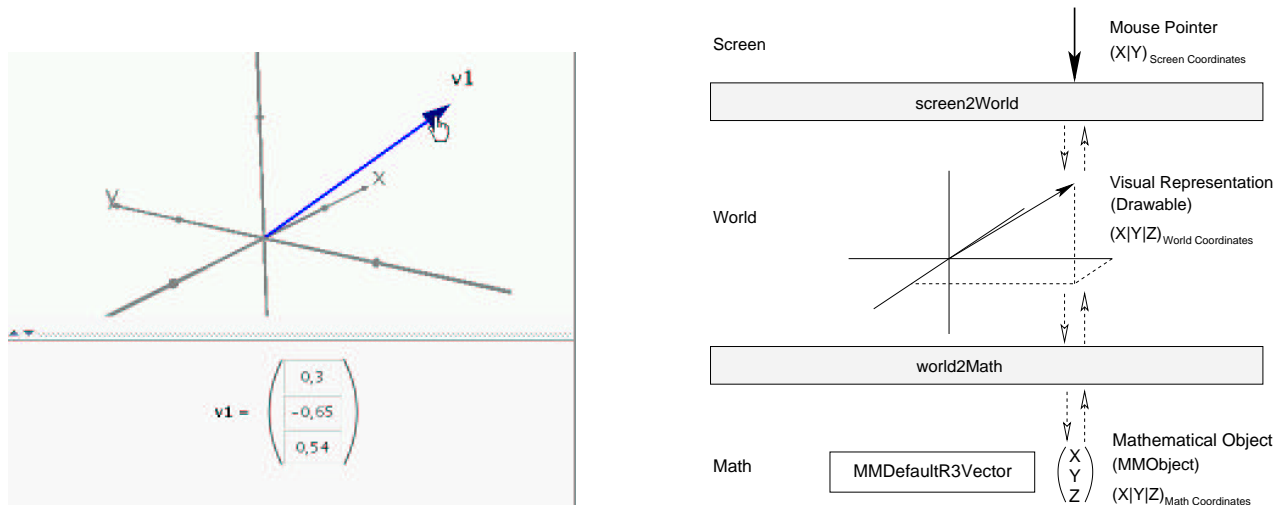


Fig. 67: Dragging a 3D vector: How the display and interaction system works

For example, when a user drags the end point of a 3D vector with the mouse, the mouse coordinates (measured in pixels) are transformed by a matrix `screen2World` into world coordinates. These describe a virtual space, where the visual representations of mathematical objects ‘live’. The mathematical objects themselves reside in a separate coordinate space called the math space. This allows them to be independent of the world’s geometry and dimension, thus allowing, for example, projections from a spherical geometry into the euclidian world coordinate space. For euclidian math spaces the transformation is simply done by another matrix called `world2Math`. So when the user drags the vector, the mouse coordinates are transformed into world and math coordinates. The object’s coordinates are changed and the graphical view updates accordingly, allowing further interaction. For 2D (or 1D) vectors the mechanism is the same.

We have given only a short overview of a complex system; yet, an application developer needs not know about all this, because there is a large set of prefabricated *handlers* that implement almost all desired functionality for manipulating `MObjects`. So if an application developer wants to construct a mathlet, where the user may drag a vector (or any other `MObject`), he only has to add the appropriate handler to it.

⁸[JV04]

Appendix C

Design of the Pyramit Platform

This section contains a brief overview of the technical and practical aspects of the Pyramit platform. It also illuminates some key issues for creating a server platform of mathematical eLearning sites. Most of these requirements (and many more) are also met by the Mumie JAPS¹.

C.1 Requirements

In order to realise the intended functionality of Pyramit (see 4.2), the following requirements were specified for the design and implementation of the platform:

- To reach a wide audience, different presentation formats have to be developed. On the one hand, the platform should be multilingual to allow an international target group for different scenarios, distant self-learners as well as local students. On the other hand we need a multi-platform solution to support the most common browser types and operating systems. For this it is necessary to support both a MathML/XHTML version (which is the W3C-compliant standard for displaying mathematics on the web, but is currently rendered only by Mozilla/Netscape or by MS Internet Explorer using a plugin) and a plain HTML version (for any other browser). The use of Java as implementation language already ensures that the applets can be viewed on almost any type of platform.
- The issue of different formats of mathematics shown to the student also raises the question which representation to choose for the author: Both MathML and HTML symbols are almost impossible to edit for large formulas. A comfortable solution would be to use the widely accepted \TeX notation.
- In order to support the didactic model specified in Chapter 2 we need a framework that helps the author to distinguish between different levels of representation on the one hand and between cognitive levels on the other by mapping these concepts onto the content structure.
- To achieve a maximum of reusability, a modular concept for the content should be chosen. This would make it also attractive to link to single modules from other sites.

¹See [Je04].

- To serve practical evaluation in an evolutionary software design, the platform needs to offer quick update functionality. For example, an author should not have to check in multiple objects into a database, when something in the content has changed, but issue maximally one or two commands for the changes to take place.

C.2 Design and Implementation

In the following, we take a look at the implementation aspects of the requirements listed above. Note that the didactic issues are covered in section 4.2.

C.2.1 Data Model and Presentation Formats

The need to have a single content to be presented in multiple formats is a common requirement in web publishing. The latest solution to address it have been XML web publishing frameworks², which work by transforming XML data sources with XSL transformation style sheets. We use the most prominent of these frameworks, Apache Cocoon, as base for the platform, allowing authors to edit the content in a single XML file and students to view it either as MathML/XHTML or as HTML.

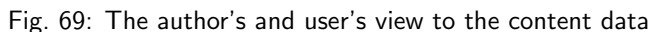
Technically, the transformation to different views has been done by using the free software tools TTH³ and itex2MML⁴ for \TeX to HTML and \TeX to MathML translation. The process of translation is executed ‘on the fly’ upon user request within a Cocoon processing pipeline containing wrapper components for each of the tools.

²[ML01]

³[TTH04]

⁴[I2M04]

Fig. 68: The author's view to a lesson: an editable XML file



The mechanism of transformation upon user request has a decisive advantage, when it comes to frequent updates of documents. In the traditional model of \TeX to web translation the author had to start the (sometimes painstakingly slow) transformation process by hand each time he changed the document. The Pyramit system does this automatically when a user requests the document (and it discovers a change in the original data, otherwise it caches the output). This mechanism is especially useful when performing a lot of small updates (which is the way most authors are used to work since the emergence of WYSIWYG-tools).

C.4 Internationalisation

In order to offer an international and a local version, the MathletFactory and the TestletFactory were internationalised, now allowing their localisation in any language without recompilation. Also the Pyramit platform is multilingual, using the same logic and presentation style for producing HTML pages, but storing different XML source files for each language.

Appendix D

List of Mathlets and their Parameters

The following is a list of the currently implemented parameters and the mathlets available of the time of this writing (October 2004). This list is automatically generated and the most current version can be found under <http://www.mathletfactory.de/doc/AppletListDoc.pdf>.

D.1 Generic Parameters for MathletFactory Applets

Parameters recognised by BaseApplet

The following parameters are recognised by all MathletFactory applets:

| Parameter | allowed values | Description |
|--|---|--|
| <code>lang</code> | language codes: "en", "de", etc. | The language to be used (for internationalised applets). |
| <code>separateWindow</code> | "true", "false" | If set to "true", display applet in a separate Java frame otherwise the applet will be embedded in the page. |
| <code>buttonText</code> | arbitrary string (including html tags) | The text to be displayed on the button. |
| <code>appletWidth</code> , <code>appletHeight</code> | an integer number | The initial width and height (in pixels) of the applet when displayed in a separate window. |
| <code>title</code> | arbitrary string (including html tags) | The title of the window, displayed in a centred label on top of all other window content. |
| <code>helpUrl</code> , <code>aboutUrl</code> | a reachable URL | URLs for the file that is displayed, when the help- or about-button is pressed. |
| <code>MumieTheme.defaultTheme</code> | At the time only "/resource/MumieClassicTheme.theme" | The default theme (Color of GUI-Elements) to be used. |
| <code>MumieTheme.appletSizeTheme</code> | "tiny-applet", "small-applet", "big-applet" | The theme (Colour of GUI-Elements) to be used for a specified size (tiny, small, big). |

Parameters recognised by SingleG2DCanvasApplet

The following parameters are recognised by all 2D MathletFactory applets with a single canvas:

| Parameter | allowed values | Description |
|---|------------------|---|
| <code>worldWidth</code> , <code>worldHeight</code> | a decimal number | The height and width of the "mathematical world" to be initially displayed in the canvas (default is 1x1). |
| <code>worldCenterX</code> , <code>worldCenterY</code> | a decimal number | The x and y coordinate in the "mathematical world" that are initially at the center of the canvas (default is (0,0)). |

Parameters recognised by SideBySideG2DCanvasApplet

The following parameters are recognised by all 2D MathletFactory applets with two horizontally aligned canvases:

| Parameter | allowed values | Description |
|---|------------------|--|
| worldWidthLeft, worldHeightLeft, worldWidthRight, worldHeightRight | a decimal number | The height and width of the "mathematical world" to be initially displayed in the left or right canvas (default is 1x1). |
| worldCenterXLeft, worldCenterYLeft, worldCenterXRight, worldCenterYRight | a decimal number | The x and y coordinate in the "mathematical world" that are initially at the center of the canvas (default is (0,0)). |

Parameters recognised by UpperLowerG2DApplet

The following parameters are recognised by all 2D MathletFactory applets with two vertically aligned canvases:

| Parameter | allowed values | Description |
|---|------------------|---|
| worldWidthUpper, worldHeightUpper, worldWidthLower, worldHeightLower | a decimal number | The height and width of the "mathematical world" to be initially displayed in the upper or lower canvas (default is 1x1). |
| worldCenterXUpper, worldCenterYUpper, worldCenterXLower, worldCenterYLower | a decimal number | The x and y coordinate in the "mathematical world" that are initially at the center of the canvas (default is (0,0)). |

Parameters recognised by SingleJ3DCanvasApplet

The following parameters are recognised by all 3D MathletFactory applets with a single canvas:

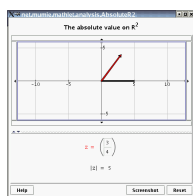
| Parameter | allowed values | Description |
|----------------|--|--|
| worldPosition | a triple in the form (x,y,z) , where x,y and z are decimal numbers | The position in 3D space (default is "(0,-2.0,0)"). |
| worldDirection | a triple in the form (x,y,z) , where x,y and z are decimal numbers | The direction in 3D space in which the viewer faces (default is "(0,1.0,0)"). |
| worldUpwards | a triple in the form (x,y,z) , where x,y and z are decimal numbers | The direction in 3D space that is "upwards" for the viewer (default is "(0,0,1.0)"). |

Parameters recognised by SideBySideJ3DApplet

The following parameters are recognised by all 3D MathletFactory applets with two horizontally aligned canvases:

| Parameter | allowed values | Description |
|--|--|--|
| worldPositionLeft, worldPositionRight | a triple in the form (x,y,z) , where x,y and z are decimal numbers | the position in 3D space (default is "(0,-2.0,0)") for each canvas. |
| worldDirectionLeft, worldDirectionRight | a triple in the form (x,y,z) , where x,y and z are decimal numbers | the direction in 3D space in which the viewer faces (default is "(0,1.0,0)") for each canvas. |
| worldUpwardsLeft, worldUpwardsRight | a triple in the form (x,y,z) , where x,y and z are decimal numbers | the direction in 3D space that is "upwards" for the viewer (default is "(0,0,1.0)") for each canvas. |

D.2 List of Mathlets



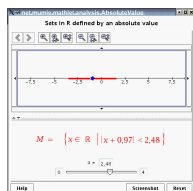
AbsoluteR2

Author: Mrose

This applet visualizes a vector in \mathbb{R}^2 and its norm. The user may alter the vector by changing its graphical or symbolic representation.

Parameters:

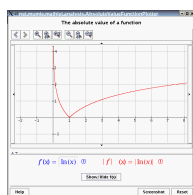
vCoords: Initial coordinates of vector, as '(x,y)' with double values x and y, default is '(3.0,4.0)'



AbsoluteValue

Author: Mrose

This applet visualizes a set in \mathbb{R} given by an inequality of the form $|x - x_0| < a$. The user may alter a or x_0 graphically.



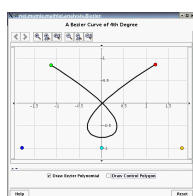
AbsoluteValueFunctionPlotter

Author: Mrose

This applet visualizes a user defined function and its absolute value. The user may alter the function expression to be displayed.

Parameters:

function: An arbitrary expression of x



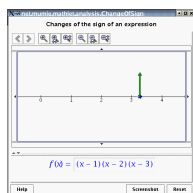
Bezier

Author: Vossbeck, Paehler

This applet draws a bezier curve of degree 4 and its control polygon. The user may drag the control points with the mouse and watch the curve change accordingly.

Parameters:

coords[1-5]: Coordinates of the i-th control point as '(x,y)' with double values x and y



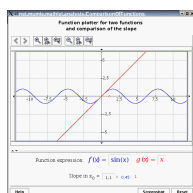
ChangeOfSign

Author: Mrose

This applet allows to determine the changes of sign of an expression and visualizes them. The user may inspect the sign for arbitrary values and alter the expression.

Parameters:

function: An arbitrary expression of x



ComparisonOfFunctions

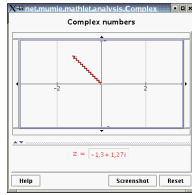
Author: Mrose

This applet allows to compare two function graphs and the gradient of the functions in a point. The user may enter arbitrary function expressions for both functions and may also move the point in which the gradient is computed.

Parameters:

function1: An arbitrary expression of x

function2: An arbitrary expression of x



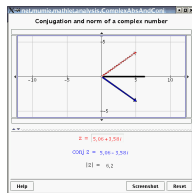
Complex

Author: Mrose

This applet visualizes the complex numbers as vectors in the complex plane. The user may alter the number by dragging the vector with the mouse or by editing the value in a textfield.

Parameters:

zValue: a complex number of form 'a+ib', where a and b are double values



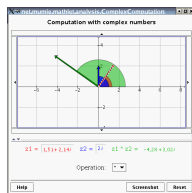
ComplexAbsAndConj

Author: Mrose

This applet visualizes the conjugation and the norm of a complex number in the complex plane. The user may alter the number by dragging the vector with the mouse or by editing the value in a textfield.

Parameters:

zValue: a complex number of form 'a+ib', where a and b are double values



ComplexComputation

Author: Mrose

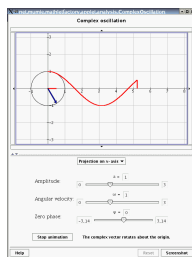
This applet visualizes basic arithmetic operations of complex numbers in the complex plane. The user may alter the numbers by dragging the vectors with the mouse or by editing the values in a textfield.

Parameters:

zValue1: a complex number of form 'a+ib', where a and b are double values

zValue2: a complex number of form 'a+ib', where a and b are double values

operation: '+', '-', '*', or '/'



ComplexOscillation

Author: Mrose

This applet visualizes a complex harmonic oscillation. The user may alter each of the parameters by using a slider.

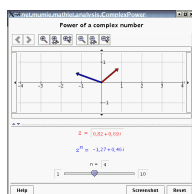
Parameters:

amplitudeParameterName: label for the amplitude, default is 'a'

angularFrequencyParameterName: label for the angular frequency, default is 'omega'

zeroPhaseParameterName: label for the zero phase, default is 'phi'

see HarmonicOscillation for other parameters



ComplexPower

Author: Paehler

This applet visualizes the n-th power of a complex number in the complex plane. The user may alter the number by dragging the vector with the mouse or by editing the value in a textfield. Additionally he may increase or decrease the power by using a slider.

Parameters:

zValue: a complex number of form 'a+ib', where a and b are double values

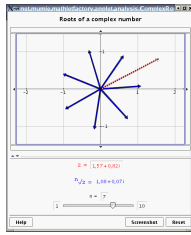
nValue: the power to be displayed, an integer number

nLeftBound: the left bound of the n slider

nRightBound: the right bound of the n slider

ComplexRoot

Author: Mrose



This applet visualizes the n -th roots of a complex number in the complex plane. The user may alter the number by dragging the vector with the mouse or by editing the value in a textfield. Additionally he may increase or decrease the power by using a slider.

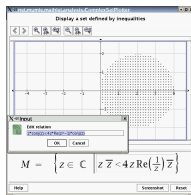
Parameters:

zValue: a complex number of form 'a+ib', where a and b are double values

nValue: the roots to be displayed, an integer number

nLeftBound: the left bound of the n slider

nRightBound: the right bound of the n slider



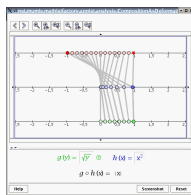
ComplexSetPlotter

Author: Paehler

This Applet visualises a subset of the gaussian plane defined by a relation. The user may edit the set by entering an arbitrary relation.

Parameters:

relation: The relation of z defining the set



CompositionAsDeformation

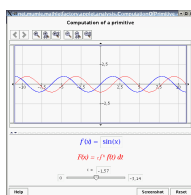
Author: Paehler

This applet demonstrates the composition of functions by using the 'function as deformation' metaphor. The user may move the set 'deformed' by the functions and may additionally enter arbitrary expressions for each of the function.

Parameters:

gFunction: An arbitrary expression of x

hFunction: An arbitrary expression of x



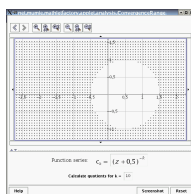
ComputationOfPrimitive

Author: Mrose

This applet visualizes the primitive for a given function. The user may edit the function by entering an arbitrary expression, additionally he may alter the integration constant by using a slider.

Parameters:

function: An arbitrary expression of x



ConvergenceRange

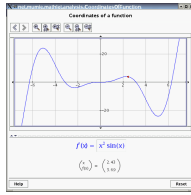
Author: Paehler

This applet visualises the convergence range of a complex function series by calculating the quotient c_k/c_{k+1} and drawing a dot if it is less than 1. The user may enter arbitrary expressions for the series, additionally he may change the index for which the quotient is checked.

Parameters:

series: An arbitrary expression of z and k

calculateTo: The value for which the quotient is computed, default is 10



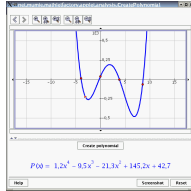
CoordinatesOfFunction

Author: Mrose

This applet allows to determine the coordinates of a function in a point. The user may move the point by dragging it with the mouse or enter an arbitrary function expression.

Parameters:

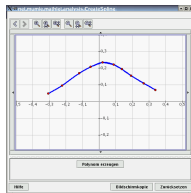
function: An arbitrary expression of x



CreatePolynomial

Author: Paehler

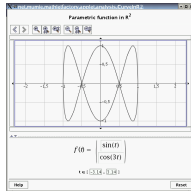
This applet allows the construction of a polynomial of degree lesser than 10 by letting the user create points in the plane and creating a polynomial that runs through these points.



CreateSpline

Author: Paehler

This applet allows the construction of a cubic spline running through up to 10 points created by the user.



CurveInR2

Author: Paehler

This applet draws a parameterized curve in \mathbb{R}^2 . The user may edit each of the coordinate expressions and the boundaries of the domain interval.

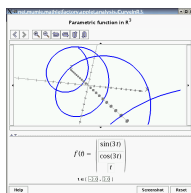
Parameters:

xfunction: An arbitrary expression of t

yfunction: An arbitrary expression of t

ParamLeftBound: Left bound of the parameter t

ParamRightBound: Right bound of the parameter t



CurveInR3

Author: Paehler

This applet draws a parameterized curve in \mathbb{R}^3 . The user may edit each of the coordinate expressions and the boundaries of the domain interval.

Parameters:

xfunction: An arbitrary expression of t

yfunction: An arbitrary expression of t

zfunction: An arbitrary expression of t

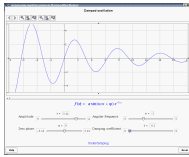
ParamLeftBound: Left bound of the parameter t

ParamRightBound: Right bound of the parameter t

DampedOscillation

Author: Mrose

This applet visualizes a damped harmonic oscillation. The user may alter each of the parameters by using a slider.



Parameters:

dampingCoefficientParameterName: label for the damping coefficient, default is 'delta'

dampLeftBound: left bound of damping coefficient slider, default is 0

dampRightBound: right bound of damping coefficient slider, default is 5

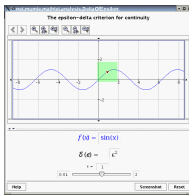
dampInitValue: initial value of damping coefficient slider, default is 0

see HarmonicOscillation for other parameters

DeltaOfEpsilon

Author: Mrose

This applet visualizes the epsilon-delta-criterion for continuous functions, where delta is a function of epsilon. The user may edit the function expression, the expression for delta(epsilon) and the value of epsilon to be displayed.



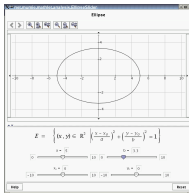
Parameters:

function: An arbitrary expression of x

EllipseSlider

Author: Liu

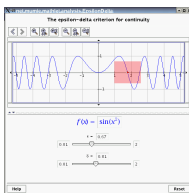
This applet shows an ellipse of which the user may alter the parameters by using sliders.



EpsilonDelta

Author: Mrose

This applet visualizes the epsilon-delta-criterion for continuous functions. The user may alter the expression of the function by editing the text field or change the values of delta and epsilon by using sliders.



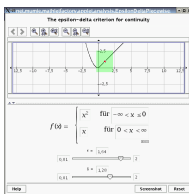
Parameters:

function: An arbitrary expression of x

EpsilonDeltaPiecewise

Author: Mrose

This applet visualizes the epsilon-delta-criterion for piecewise continuous functions. The user may alter the expressions of the function by editing the text field or change the values of delta and epsilon by using sliders.



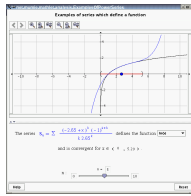
Parameters:

function1: The first function expression of the piecewise function

interval1: The interval for the first function expression

function2: The second function expression of the piecewise function

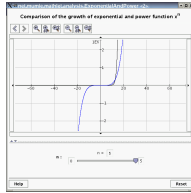
interval2: The interval for the second function expression, and so on...



ExamplesOfPowerSeries

Author: Mrose

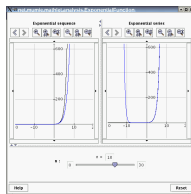
This applet visualizes examples of power series. The user may switch between the examples and may also change the n of the power series by using a slider.



ExponentialAndPower

Author: Mrose

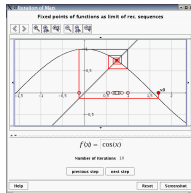
This applet allows to compare the growth of exponential function and power functions. The user may alter the exponent of the power by using a slider.



ExponentialFunction

Author: Mrose

This applet allows to compare the exponential series with the exponential sequence. The user may alter the n for both the exponential sequence and the exponential series by using a slider.



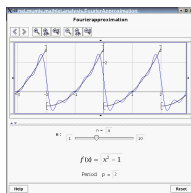
FixedPointPlotter

Author: Paehler

This applet shows the construction of a recursive sequence by visualising the expression as a function. The user may alter the sequence expression and increase or decrease the number of iterations.

Parameters:

function: An arbitrary expression of x



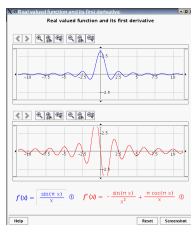
FourierApproximation

Author: Mrose

This applet approximates a given periodic function by the n -th partial sum of its Fourier series. The user may alter the function expression, the value for n and the period of the function.

Parameters:

function: An arbitrary expression of x



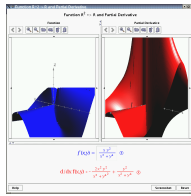
FunctionAndDerivativeOverR

Author: Paehler

This applet computes for a given function the derivative and draws it. The user may alter the function expression by editing its text field.

Parameters:

function: An arbitrary expression of x



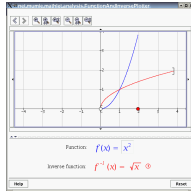
FunctionAndDerivativeOverR2

Author: Paehler

This applet visualizes a function over \mathbb{R}^2 and its partial derivative. The user may alter the function expression by editing its text field.

Parameters:

function: An arbitrary expression of x and y



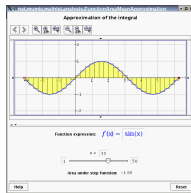
FunctionAndInversePlotter

Author: Paehler

This applet shows for a given function and domain the inverse function. The user may alter the function expression by editing its text field, additionally he may alter the domain interval by dragging its right border with the mouse.

Parameters:

function: An arbitrary expression of x



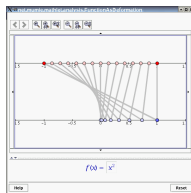
FunctionAreaMeanApproximation

Author: Mrose

This applet approximates the area under the graph of a function by step functions. The user may either edit the function expression, alter its domain or change the number of steps used in the approximation.

Parameters:

function: An arbitrary expression of x



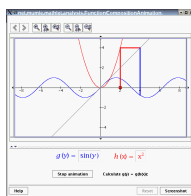
FunctionAsDeformation

Author: Paehler

This applet demonstrates the mapping aspect of a function by using the 'function as deformation' metaphor. The user may alter the function expression. and move the interval to be 'deformed' by dragging its boundaries with the mouse.

Parameters:

function: An arbitrary expression of x



FunctionCompositionAnimation

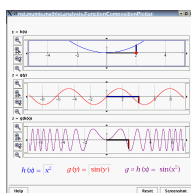
Author: Paehler

This applet constructs the composition of functions in an animation. The user may edit each of the functions to be composed.

Parameters:

gFunction: An arbitrary expression of x

hFunction: An arbitrary expression of x



FunctionCompositionPlotter

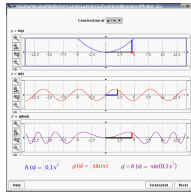
Author: Paehler

This applet constructs the composition of functions. The user may edit each of the functions to be composed.

Parameters:

gFunction: An arbitrary expression of x

hFunction: An arbitrary expression of x



FunctionOperationPlotter

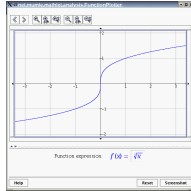
Author: Paehler

This applet constructs operations (sum, product, quotient, composition) of functions. The user may edit each of the operand functions.

Parameters:

gFunction: An arbitrary expression of x

hFunction: An arbitrary expression of x



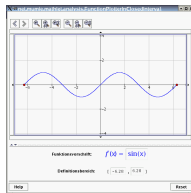
FunctionPlotter

Author: Paehler

This applet plots various functions. The user may edit the function by entering an arbitrary expression of x.

Parameters:

function: An arbitrary expression of x



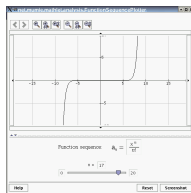
FunctionPlotterInClosedInterval

Author: Paehler

This applet plots functions defined within a closed interval. The user may edit the function by entering an arbitrary expression of x, additionally he may change the domain interval by dragging its boundaries with the mouse.

Parameters:

function: An arbitrary expression of x



FunctionSequencePlotter

Author: Paehler

This applet visualises function sequences. The user may either edit the function sequence expression or alter the n for which the function sequence is displayed.

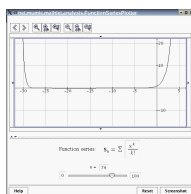
Parameters:

sequence: An arbitrary expression of n and x

leftBound: Left bound of the slider

rightBound: Right bound of the slider

initialValue: Initial value of the slider



FunctionSeriesPlotter

Author: Paehler

This applet visualises real valued function series. The user may either edit the function series expression or alter the n up to which the function series is displayed.

Parameters:

series: An arbitrary expression of x and k

leftBound: Left bound of the slider

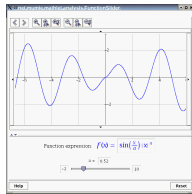
rightBound: Right bound of the slider

initialValue: Initial value of the slider

FunctionSlider

Author: Paehler

This applet visualises a parameterised function the parameter of which can be altered by using a slider for each parameter. The user may edit the function by entering an arbitrary expression of x and the parameter(s). Up to three parameters can be declared by the author.



Parameters:

function: An arbitrary expression of x

functionEditable: 'true' or 'false'

paramName[2,3]: Label of the first [second, third] parameter, default is 'a'

paramAllowOnlyIntegers[2,3]: 'true' or 'false'

paramInitialValue[2,3]: Initial value of the first [second, third] parameter slider, default is '1'

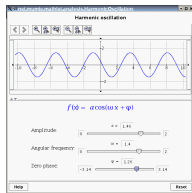
paramLeftBound[2,3]: Left bound of the first [second, third] parameter slider, default is '-2'

paramRightBound[2,3]: Right bound of the first [second, third] parameter slider, default is '2'

HarmonicOscillation

Author: Mrose

This applet visualizes a harmonic oscillation whose parameters may be altered by sliders. The user may alter each of the parameters by using a slider.



Parameters:

amplitudeParameterName: label for the amplitude, default is 'a'

angularFrequencyParameterName: label for the angular frequency, default is 'omega'

zeroPhaseParameterName: label for the zero phase, default is 'phi'

ampLeftBound: left bound of amplitude slider, default is 0

ampRightBound: right bound of amplitude slider, default is 3

ampInitValue: initial value of amplitude slider, default is 1

afLeftBound: left bound of angular frequency slider, default is 0

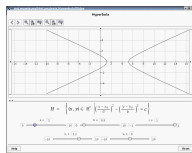
afRightBound: right bound of angular frequency slider, default is 3

afInitValue: initial value of angular frequency slider, default is 1

phaseLeftBound: left bound of zero phase slider, default is -3.14

phaseRightBound: right bound of zero phase slider, default is 3.14

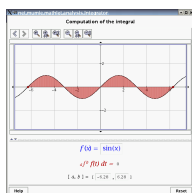
phaseInitValue: initial value of zero phase slider, default is 0



HyperbolaSlider

Author: Liu

This applet shows an hyperbola of which the user may alter the parameters by using sliders.



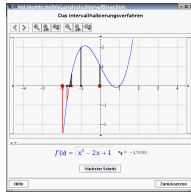
Integrator

Author: Mrose

This applet computes and visualizes the integral of a function over an interval. The user may alter the expression of the integrand and edit the boundaries of integration.

Parameters:

function: An arbitrary expression of x



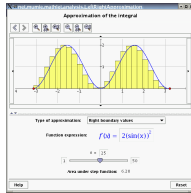
IntervalBisection

Author: Paehler

An applet for visualising the Interval Bisection method of approximating a zero of a function. The user may enter an arbitrary function expression and increase the number of iterations.

Parameters:

function: An arbitrary expression of x



LeftRightApproximation

Author: Mrose

This applet approximates the surface under the graph of a function by step functions. The user may switch between left- or right-approximations, edit the function expression and alter the number of steps using a slider.

Parameters:

function: An arbitrary expression of x

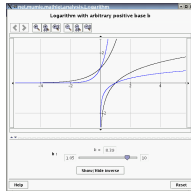
leftBound: Left bound of the interval, default is -6.28

rightBound: Right bound of the interval, default is 6.28

nInitialValue: Initial value of n, default is '10'

nLeftBound: Left bound of n slider, default is '1'

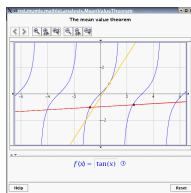
nRightBound: Right bound of n slider, default is '50'



Logarithm

Author: Mrose

This applet allows to compare the logarithm of arbitrary base with the natural logarithm. The user may alter the base of the logarithm by using a slider.



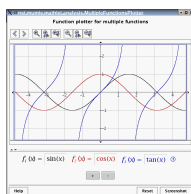
MeanValueTheorem

Author: Mrose

This applet visualizes the mean value theorem for continuous functions: For each secant, a tangent of equal slope can be found between the two secant points. The user may edit the function expression or alter the secant and tangent points.

Parameters:

function: An arbitrary expression of x



MultipleFunctionsPlotter

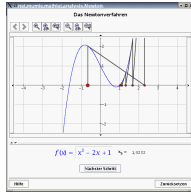
Author: Paehler

This applet shows multiple functions. A user may edit the function expressions and add and remove further functions.

Parameters:

function1: An arbitrary expression of x

function2: An arbitrary expression of x



Newton

Author: Liu, Paehler

An applet for visualising the Newton method of approximating a zero of a function. The user may edit the function expression and increase the number of iterations.

Parameters:

function: An arbitrary expression of x

OscillationSuperposition

Author: Mrose

This applet visualizes the superposition of two harmonic oscillations. The user may alter each of the parameters by using a slider.

Parameters:

amplitudeParameter1Name: label for the amplitude of the 1st oscillation, default is 'a'

angularFrequency1ParameterName: label for the ang. freq. of the first oscillation, default is 'omega'

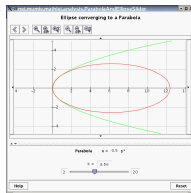
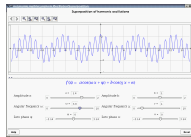
zeroPhaseParameter1Name: label for the zero phase of the 1st oscillation, default is 'phi'

amplitudeParameter2Name: label for the amplitude of the 2nd oscillation, default is 'n'

angularFrequency2ParameterName: label for the ang. freq. of the 2nd oscillation, default is 'gamma'

zeroPhaseParameter2Name: label for the zero phase of the 2nd oscillation, default is 'alpha'

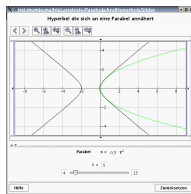
see HarmonicOscillation for other parameters



ParabolaAndEllipseSlider

Author: Liu

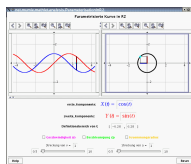
This applet visualises the approximation of a parabola by an ellipse. The user may alter the ellipse parameter using a slider.



ParabolaAndHyperbolaSlider

Author: Liu

This applet visualises the approximation of a parabola by a hyperbola. The user may alter the hyperbola parameter using a slider.



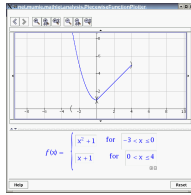
ParameterisationInR2

Author: manya

This applet shows the construction of a parameterized curve in \mathbb{R}^2 . The user may edit each of the coordinate expressions and choose between the display of some kinematic observables.

PiecewiseFunctionPlotter

Author: Paehler



This applet plots various piecewise defined functions. The user may alter any of the function's expressions or domain intervals.

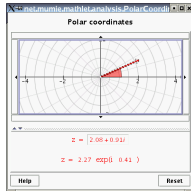
Parameters:

function1: The first function expression of the piecewise function

interval1: The interval for the first function expression

function2: The second function expression of the piecewise function

interval2: The interval for the first function expression, and so on...

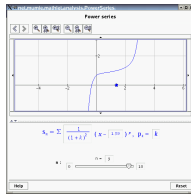
**PolarCoordinates**

Author: Mrose

This applet visualizes complex numbers in polar coordinates. The user may alter the number by dragging the vector with the mouse or by editing the value in a textfield.

PowerSeries

Author: Mrose



This applet visualizes the n-th partial sum of a power series. The user may edit the expression of the power series and alter the value of n for which the series is displayed.

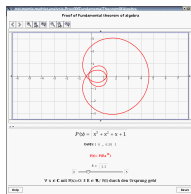
Parameters:

coefficient: An arbitrary expression of k

leftBound: Left bound of the slider, default is 0

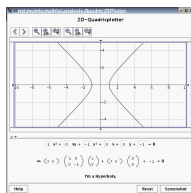
rightBound: Right bound of the slider, default is 10

initialValue: Initial value of the slider, default is 0

**ProofOfFundamentalTheoremOfAlgebra**

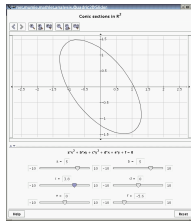
Author: manya

This applet shows a proof of fundamental theorem of algebra. The user may edit the polynomial to be used and alter the radius of the circle to be 'deformed' by the polynomial by using a slider.

**Quadric2DPlotter**

Author: Liu

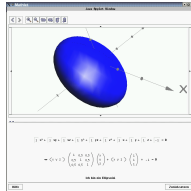
This applet shows a 2D quadric of which the user may alter the parameters.



Quadric2DSlider

Author: Liu

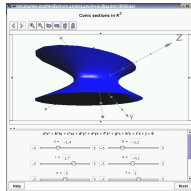
This applet shows a 2D quadric of which the user may alter the parameters by using sliders.



Quadric3DPlotter

Author: Mrose

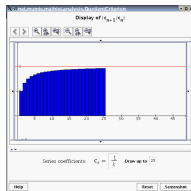
Plots a quadric in three dimensional space which can be determined by the user by entering the quadric equation. The applet determines the type of the quadric and plots the corresponding surface in \mathbb{R}^3 .



Quadric3DSlider

Author: Mrose

This applet visualizes a quadric in \mathbb{R}^3 . The user may alter its parameters by using sliders.



QuotientCriterion

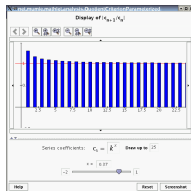
Author: Paehler

This applet visualises the quotient criterion for a real valued series. The user may edit the series expression and specify the k up to which the quotient is displayed.

Parameters:

series: An arbitrary expression of k

drawTo: The value up to which the criterion is drawn



QuotientCriterionParameterized

Author: Paehler

This applet visualises the quotient criterion for a parameterised real valued series. The user may edit the series expression and specify the k up to which the quotient is displayed, additionally he may alter the parameter by using a slider.

Parameters:

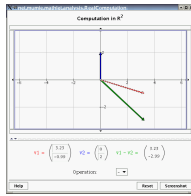
series: An arbitrary expression of k

drawTo: The value up to which the criterion is drawn

paramLeftBound: The left bound of the parameter on the slider

paramRightBound: The left bound of the parameter on the slider

paramInitialValue: The initial value of the parameter on the slider



RealComputation

Author: Mrose

This applet visualizes addition and subtraction in \mathbb{R}^2 . The user may alter the vectors by dragging them with the mouse or by editing their coordinates.

RecursiveSequencePlotter

Author: Paehler

This applet visualises recursive sequences. The user may alter the number of sequence members to be displayed and the starting member a_0 by using sliders.

Parameters:

sequence: An arbitrary expression of n

leftBound: Left bound of the slider

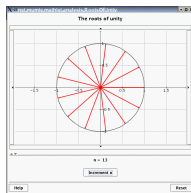
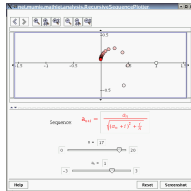
rightBound: Right bound of the slider

initialValue: Initial value of the slider

a0leftBound: Left bound of the a_0 slider

a0rightBound: Right bound of the a_0 slider

a0initialValue: Initial value of a_0



RootsOfUnity

Author: Mrose

This applet visualizes the n complex solutions of the n -th root of 1. The user may increase or decrease the n .

SequencePlotter

Author: Paehler

This applet visualises real and complex valued sequences. The user may edit the sequence expression and alter the number of sequence members to be displayed by using a slider.

Parameters:

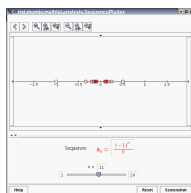
numberClass: 'complex' for complex valued sequences or 'double' for real valued sequences

sequence: An arbitrary expression of n

leftBound: Left bound of the slider

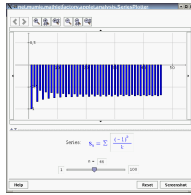
rightBound: Right bound of the slider

initialValue: Initial value of the slider



SeriesPlotter

Author: Paehler



This applet visualises a real valued series. The user may edit the series expression and alter the number of series members to be displayed by using a slider.

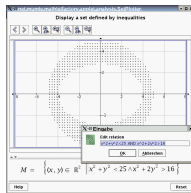
Parameters:

series: An arbitrary expression of k

leftBound: Left bound of the slider

rightBound: Right bound of the slider

initialValue: Initial value of the slider

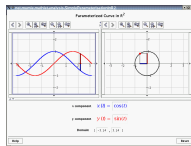
**SetPlotter**

Author: Paehler

This applet visualises a subset of the \mathbb{R}^2 that is defined by a relation. The user may edit the set by entering an arbitrary relation.

Parameters:

relation: The relation of x and y defining the set

**SimpleParameterisationInR2**

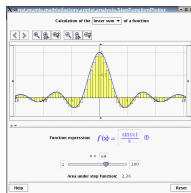
Author: Paehler

This applet shows the construction of a parameterized curve in \mathbb{R}^2 . The user may edit each of the coordinate expressions and the boundaries of the domain interval.

Parameters:

xFunction: An arbitrary expression of t

yFunction: An arbitrary expression of t

**StepFunctionPlotter**

Author: Paehler

This applet draws a riemann sum for a given function. The user may either edit the function expression, alter its domain or change the number of steps used in the approximation.

Parameters:

function: An arbitrary expression of x

leftBound: Left bound of the interval, default is -6.28

rightBound: Right bound of the interval, default is 6.28

nInitialValue: Initial value of n , default is '10'

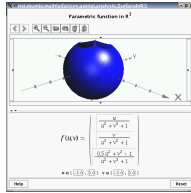
nLeftBound: Left bound of n slider, default is '1'

nRightBound: Right bound of n slider, default is '50'

SurfaceInR3

Author: Paehler

This applet draws an arbitrary user defined parameterized surface in \mathbb{R}^3 . The user may edit each of the coordinate expressions and the boundaries of the domain intervals.



Parameters:

xFunction: An arbitrary expression of u and v

yFunction: An arbitrary expression of u and v

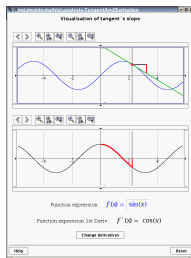
zFunction: An arbitrary expression of u and v

Param1LeftBound: Left bound of the parameter u

Param1RightBound: Right bound of the parameter u

Param2LeftBound: Left bound of the parameter v

Param2RightBound: Right bound of the parameter v



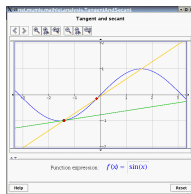
TangentAndDerivative

Author: Schimanowski

An applet for visualising the construction of the derivative by a moving tangent for a differentiable function. The user may edit the function expression and move the tangent by dragging it with the mouse.

Parameters:

function: An arbitrary expression of x



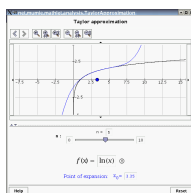
TangentAndSecant

Author: Schimanowski

An applet for visualising the tangent and the secant of two points for a differentiable function. The user may edit the function expression and move the tangent and secant by dragging its points with the mouse.

Parameters:

function: An arbitrary expression of x



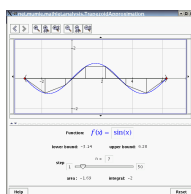
TaylorApproximation

Author: Mrose

This applet approximates a given function in a point of expansion by its Taylor polynomial of n-th degree. The user may edit the function expression or set the value of n to be displayed, additionally he may move the point of expansion by dragging it with the mouse.

Parameters:

function: An arbitrary expression of x



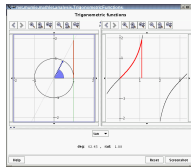
TrapezoidApproximation

Author: Liu

An Applet that shows the approximation of the integral by using trapezoids. The user may either edit the function expression, alter its domain or change the number of steps used in the approximation.

Parameters:

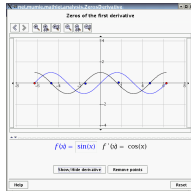
function: An arbitrary expression of x



Trigonometric Functions

Author: Mrose

This applet visualizes the definition of the trigonometric functions in the unit circle. The user may choose between different functions and move the point to be constructed on the circle.



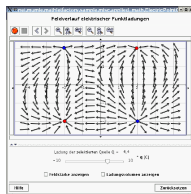
ZerosDerivative

Author: Mrose

Exercise for students to find the zeros of the derivative by analyzing the graph of a function. The user may edit the function expressions and add points for the guessed zeros of the derivative.

Parameters:

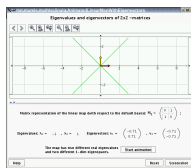
function: An arbitrary expression of x



ElectricPointCharge

Author: gronau

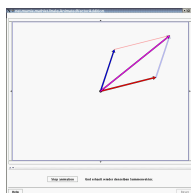
Visualizes the electric field of a custom point charge distribution. The user may add and move an arbitrary number of point charges with the mouse, additionally he may alter their charge value.



AnimatedLinearMapWithEigenvectors

Author: Mrose

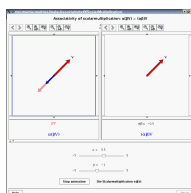
Shows the behavior of a map given by a matrix specified by the user and the associated eigenspaces. Additionally a vector and its image is displayed, which can be altered by the user.



AnimatedVectorAddition

Author: manya

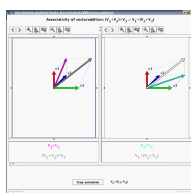
This applet shows in animation the addition of two vectors.



AssociativityOfScalarMultiplication

Author: manya

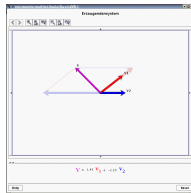
This applet shows the associativity of scalar multiplication of vector spaces



AssociativityOfVectoraddition

Author: manya

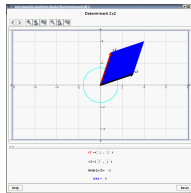
This applet shows the associativity of vector addition of vector spaces



BasisOfR2

Author: manya

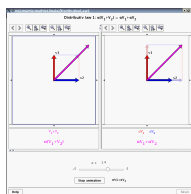
This applet shows a generated system of \mathbb{R}^2 .



DeterminantInR2

Author: manya

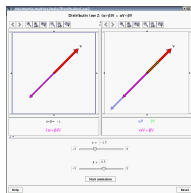
This applet shows the geometric interpretation of 2x2 matrix



DistributiveLaw1

Author: manya

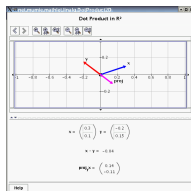
This applet shows the (first) distributive law of vector spaces.



DistributiveLaw2

Author: manya

This applet shows the (second) distributive law of vector spaces



DotProduct2D

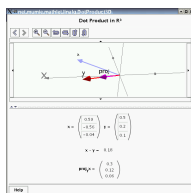
Author: Erkoc

Visualizes the scalar product of two vectors u and v in \mathbb{R}^2 . The user may alter the vectors by dragging them with the mouse or editing their coordinates.

Parameters:

uCoords: Coordinates of vector u as '(x,y)' with double values x and y

vCoords: Coordinates of vector v as '(x,y)' with double values x and y



DotProduct3D

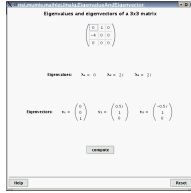
Author: Erkoc

Visualizes the scalar product of two vectors u and v in \mathbb{R}^3 . The user may alter the vectors by dragging them with the mouse or editing their coordinates.

Parameters:

uCoords: Coordinates of vector u as '(x,y,z)' with double values x,y and z

vCoords: Coordinates of vector v as '(x,y,z)' with double values x,y and z



EigenvalueAndEigenvector

Author: Liu

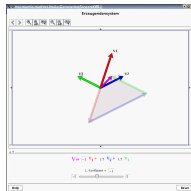
This applet displays for an arbitrary 3x3 matrix the eigenvalues and eigenvectors.

Parameters:

r1Coords: Coordinates of the matrix' first row vector as '(x,y,z)' with double values x,y and z

r2Coords: Coordinates of the matrix' second row vector as '(x,y,z)' with double values x,y and z

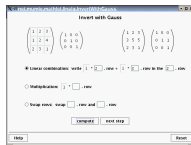
r3Coords: Coordinates of the matrix' third row vector as '(x,y,z)' with double values x,y and z



GeneratingSystemOfR2

Author: manya

This applet shows a generating system of \mathbb{R}^2 with three vectors



InvertWithGauss

Author: Liu

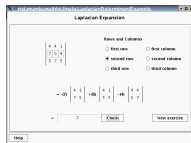
This applet lets a user interactively invert a matrix with the gauss algorithm. The user may enter an arbitrary matrix and perform steps of linear row manipulation.

Parameters:

r1Coords: Coordinates of the matrix' first row vector as '(x,y,z)' with double values x,y and z

r2Coords: Coordinates of the matrix' second row vector as '(x,y,z)' with double values x,y and z

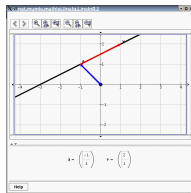
r3Coords: Coordinates of the matrix' third row vector as '(x,y,z)' with double values x,y and z



LaplacianDeterminantExample

Author: Erkoç

This applet offers exercises for the computation of the determinant using the laplacian expansion rule. The user may choose an expansion and test his computed results.



LineInR2

Author: Paehler

Visualizes a line in \mathbb{R}^2 and its parametric definition. The user may alter the line by dragging its vectors with the mouse or editing their coordinates.

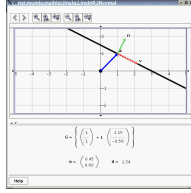
Parameters:

aCoords: Coordinates of origin vector as '(x,y)' with double values x and y

vCoords: Coordinates of direction vector as '(x,y)' with double values x and y

LineInR2Normal

Author: Paehler



Visualizes a line in \mathbb{R}^2 , its parametric definition and a normalized normal vector. The user may alter the line by dragging its vectors with the mouse or editing their coordinates.

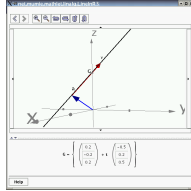
Parameters:

aCoords: Coordinates of origin vector as '(x,y)' with double values x and y

vCoords: Coordinates of direction vector as '(x,y)' with double values x and y

LineInR3

Author: Paehler



Visualizes a line in \mathbb{R}^3 and its parametric definition. The user may alter the line by dragging its vectors with the mouse or editing their coordinates.

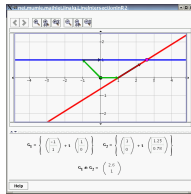
Parameters:

aCoords: Coordinates of origin vector as '(x,y,z)' with double values x,y and z

vCoords: Coordinates of direction vector as '(x,y,z)' with double values x,y and z

LineIntersectionInR2

Author: Paehler



Visualizes two lines and their intersection in \mathbb{R}^2 . The user may alter the lines by dragging its vectors with the mouse or editing their coordinates.

Parameters:

a1Coords: Coordinates of first line's origin vector as '(x,y)' with double values x and y

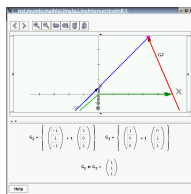
v1Coords: Coordinates of first line's direction vector as '(x,y)' with double values x and y

a2Coords: Coordinates of second line's origin vector as '(x,y)' with double values x and y

v2Coords: Coordinates of second line's direction vector as '(x,y)' with double values x and y

LineIntersectionInR3

Author: Paehler



Visualizes two lines and their intersection in \mathbb{R}^3 . The user may alter the lines by dragging its vectors with the mouse or editing their coordinates.

Parameters:

a1Coords: Coordinates of first line's origin vector as '(x,y,z)' with double values x, y and z

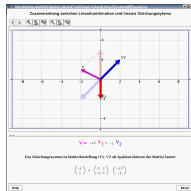
v1Coords: Coordinates of first line's direction vector as '(x,y,z)' with double values x, y and z

a2Coords: Coordinates of second line's origin vector as '(x,y,z)' with double values x, y and z

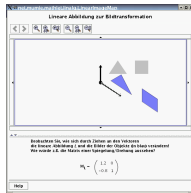
v2Coords: Coordinates of second line's direction vector as '(x,y,z)' with double values x, y and z

LinearCombinationAndLinearSystemOfEquation

Author: manya



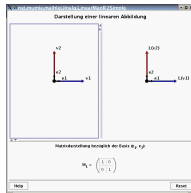
This applet shows the relationship between the linear combination of vectors and a system of linear equations.



LinearImageMap

Author: gronau

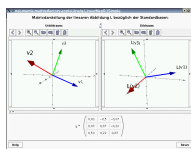
Demonstrates the usage of linear maps for image transformations.



LinearMapR2Simple

Author: i.a. manya

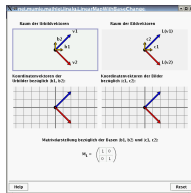
This applet shows a linear map in \mathbb{R}^2 . The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapR3Simple

Author: Paehler

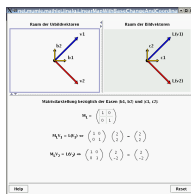
This applet shows a linear map in \mathbb{R}^3 . The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithBaseChange

Author: vossbeck, manya

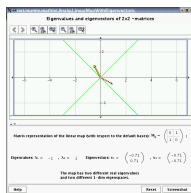
This applet visualizes a linear map on \mathbb{R}^2 with respect to different bases in domain and range. The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithBaseChangeAndCoordinateTuples

Author: vossbeck, manya

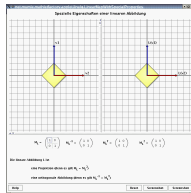
This applet visualizes a linear map on \mathbb{R}^2 with respect to different bases in domain and range. Furthermore it shows the equations of their appropriate coordinate representations. The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithEigenvectors

Author: Schimanowski

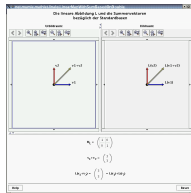
Shows the behavior of a map given by a matrix specified by the user and the associated eigenspaces. Additionally a vector and its image is displayed, which can be altered by the user.



LinearMapWithSpecialProperties

Author: vossbeck, manya

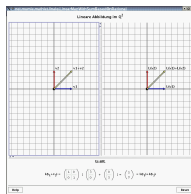
This applet shows specific properties of a linear map. The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithSumBasedOnDouble

Author: i.a. manya

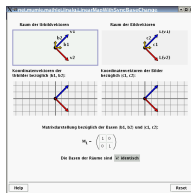
This applet shows a linear map in \mathbb{R}^2 . The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithSumBasedOnRational

Author: i.a. manya

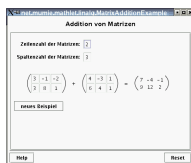
This applet shows a linear map in \mathbb{Q}^2 . The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



LinearMapWithSyncBaseChange

Author: vossbeck, manya

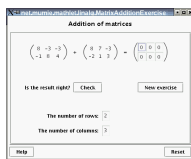
This applet visualizes a linear map on \mathbb{R}^2 with respect to different (or same) bases in domain and range. The user may alter the map by either editing its matrix entries or by dragging the domain or range vectors with the mouse.



MatrixAdditionExample

Author: klich

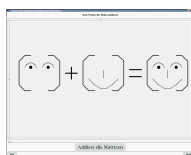
Shows examples of the addition of two matrices and computes their result.



MatrixAdditionExercise

Author: klich

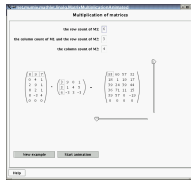
Applet to exercise the addition of two matrices.



MatrixAdditionFace

Author: nobody

Visualizes the addition of two matrices as a superposition of two images.



MatrixMultiplicationAnimated

Author: klich

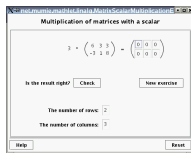
This applet demonstrates the multiplication of two matrices.



MatrixMultiplicationExercise

Author: klich

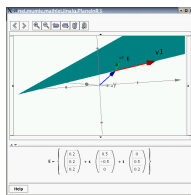
Applet to exercise the multiplication of two matrices.



MatrixScalarMultiplicationExercise

Author: klich

Applet to exercise the multiplication of a matrix with a scalar.



PlaneInR3

Author: Paehler

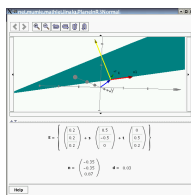
Visualizes a plane in \mathbb{R}^3 and its parametric definition. The user may alter the plane by dragging its vectors with the mouse or editing their coordinates.

Parameters:

a1Coords: Coordinates of plane's origin vector as '(x,y,z)' with double values x, y and z

v1Coords: Coordinates of plane's first direction vector as '(x,y,z)' with double values x, y and z

v2Coords: Coordinates of plane's second direction vector as '(x,y,z)' with double values x, y and z



PlaneInR3Normal

Author: Paehler

Visualizes a plane and its normal in \mathbb{R}^3 and its parametric definition. The user may alter the plane by dragging its vectors with the mouse or editing their coordinates.

Parameters:

a1Coords: Coordinates of plane's origin vector as '(x,y,z)' with double values x, y and z

v1Coords: Coordinates of plane's first direction vector as '(x,y,z)' with double values x, y and z

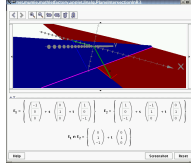
v2Coords: Coordinates of plane's second direction vector as '(x,y,z)' with double values x, y and z

PlaneIntersectionInR3

Author: Paehler

Visualizes two planes and their intersection in \mathbb{R}^3 . The user may alter the planes by dragging its vectors with the mouse or editing their coordinates.

Parameters:



a1Coords: Coordinates of first plane's origin vector as '(x,y,z)' with double values x, y and z

v1Coords: Coordinates of first plane's first direction vector as '(x,y,z)' with double values x, y and z

v2Coords: Coordinates of first plane's second direction vector as '(x,y,z)' with double values x, y and z

a2Coords: Coordinates of second plane's origin vector as '(x,y,z)' with double values x, y and z

u1Coords: Coordinates of second plane's first direction vector as '(x,y,z)' with double values x, y and z

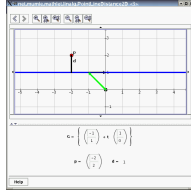
u2Coords: Coordinates of second plane's second direction vector as '(x,y,z)' with double values x, y and z

PointLineDistance2D

Author: Erkoç

Visualizes the distance between a point and a line in \mathbb{R}^2 . The user may alter the line by dragging its vectors with the mouse or editing their coordinates, the point can also be dragged.

Parameters:



rCoords: Coordinates of origin vector as '(x,y)' with double values x and y

lCoords: Coordinates of direction vector as '(x,y)' with double values x and y

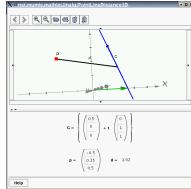
pCoords: Coordinates of point as '(x,y)' with double values x and y

PointLineDistance3D

Author: Erkoç

Visualizes the distance between a point and a line in \mathbb{R}^3 . The user may alter the line by dragging its vectors with the mouse or editing their coordinates, the point can also be dragged.

Parameters:



rCoords: Coordinates of origin vector as '(x,y,z)' with double values x,y and z

lCoords: Coordinates of direction vector as '(x,y,z)' with double values x,y and z

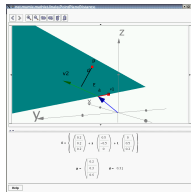
pCoords: Coordinates of point as '(x,y,z)' with double values x,y and z

PointPlaneDistance

Author: Erkoç

Visualizes the distance between a point and a plane in \mathbb{R}^3 . The user may alter the plane by dragging its vectors with the mouse or editing their coordinates, the point can also be dragged.

Parameters:

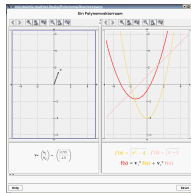


aCoords: Coordinates of plane's origin vector as '(x,y,z)' with double values x, y and z

v1Coords: Coordinates of plane's first direction vector as '(x,y,z)' with double values x, y and z

v2Coords: Coordinates of plane's second direction vector as '(x,y,z)' with double values x, y and z

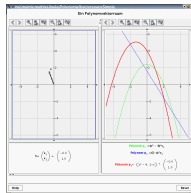
pCoords: Coordinates of point as '(x,y,z)' with double values x,y and z



PolynomialVectorspace

Author: paladini, manya

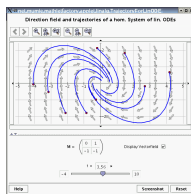
This applet visualises the vector space of two polynomials.



PolynomialVectorspaceSimple

Author: paladini, manya

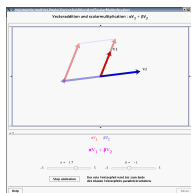
This applet visualises the vector space of two polynomials.



TrajectoryForLinODE

Author: Paehler

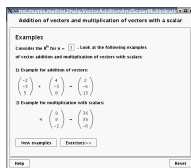
This applet shows for a linear homogeneous system of ODEs the direction field and allows the user to construct trajectories for certain starting points. He may also alter the ODE system by editing its matrix entries and trace the trajectories by using a slider or animation.



VectorAdditionAndScalarMultiplication

Author: manya

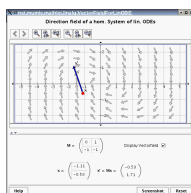
This applet shows the scalar multiplication and the addition of two vectors



VectorAdditionAndScalarMultiplicationExercise

Author: klich

An applet to demonstrate and practice vector addition and multiplication with scalars.



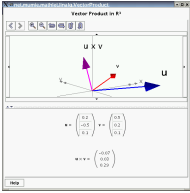
VectorFieldForLinODE

Author: Paehler

This applet shows the direction field of a linear homogeneous system of ODEs. The user may alter the ODE system by editing its matrix entries and drag the point, for which the vector is displayed with the mouse.

VectorProduct

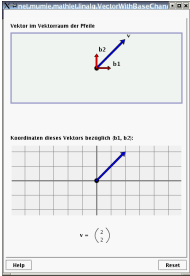
Author: Erkoc



Visualizes the vector product of two vectors u and v in \mathbb{R}^3 .The user may alter the vectors by dragging them with the mouse or editing their coordinates.

Parameters:

- uCoords: Coordinates of vector u as '(x,y,z)' with double values x,y and z
- vCoords: Coordinates of vector v as '(x,y,z)' with double values x,y and z



VectorWithBaseChange

Author: klich

Visualizes a vector on the one hand in a vectorspace of arrows and on the other hand in a coordinates space respective to a chosen basis.

Appendix E

Installation CD

The CD included with this thesis was developed for allowing offline use of mathlets and testlets. It contains a complete server image of <http://mumie.iram.rwth-aachen.de>. It can be easily installed by running the script `setup.bat` on an MS Windows platform or by invoking `setup.sh` on a Unix platform as `root`.

Bibliography

- [ADL04] Advanced Distributed Learning. Sharable Content Object Reference Model (SCORM) 2004 2nd Edition. 2004.
<http://www.adlnet.org>
- [AM98] J. R. Anderson, M. P. Matessa. The rational analysis of categorization and the ACT-R architecture. In: M. Oaksford & N. Chater (Eds.) Rational models of cognition, pp. 197-217. Oxford: Oxford University Press. 1998.
- [AM04] ActiveMath homepage. 2004.
<http://www.activemath.org>.
- [ASU86] A: Aho, R. Sethi, J. Ullman. Compilers - Principles, Techniques and Tools. Addison Wesley. 1996.
- [Bau02] C. Bauer, A. Frink, R. Kreckel.
Introduction to the GiNaC Framework for Symbolic Computation within the C++ Programming Language.
Journal of Symbolic Computation Volume 33, Number 1, 2002.
- [Bl56] B.S. Bloom (Ed.) Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain. New York. 1956.
- [Blu98] A. Blumstengel. Entwicklung hypermedialer Lernsysteme. Wissenschaftlicher Verlag Berlin. 1998.
http://dsor.upb.de/de/forschung/publikationen/blumstengel-diss/main_index_titel.html
- [BmBF04] Bundesministerium für Bildung und Forschung. Neue Medien in der Bildung – Hochschulen. Kursbuch eLearning 2004 – Produkte aus dem Förderprogramm. 2004.
<http://www.bmbf.de/pub/nmb-kursbuch.pdf>.
- [Bo88] B.W. Boehm. A spiral model of software development and enhancement. IEEE Computer, vol.21, No. 5, May 1988, pp 61-72.
- [Br66] J.S. Bruner. Towards a Theory of Instruction. Harvard University Press. 1966.
- [Br56] J.S. Bruner. A cognitive theory of personality: You are your constructs. Contemporary Psychology, 1, 355-357. 1956.
- [Bu96] F. Buschmann et al. Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. Addison-Wesley 1996.

- [Ch01] J.R. Chiles. *Inviting Disaster: Lessons from the Edge of Technology* HarperBusiness. 2001.
- [Co03] H. Comon et al. *Tree Automata Techniques and Applications*. Preprint, 2003.
<http://www.grappa.univ-lille3.fr/tata>
- [CP91] J. M. Clark, A. Paivio. Dual coding theory and education. *Educational Psychology Review*, 3(3), 1991.
- [Cr02] C. Crawford. *The Art of Interactive Design*. No Starch Press. 2002.
- [Do02] H. Donker *Didaktisches Interaktions-und Informationsdesign*.
<http://www.dissertation.de/PDF/hd519.pdf>
- [Do00] M. Dobes. *Mathe-Online: Evaluation aus medienkritischer und unterrichtspraktischer Sicht*, TELL & CALL April 2000.
http://www.mathe-online.at/literatur/dobes_tellcall.pdf
- [EC00] *Elektronische Kreide: Eine Java-Multimedia-Tafel für den Präsenz- und Fernunterricht*, R. Rojas, L. Knipping, W. Raffel, G. Friedland in Technical Report B-00-17, FU Berlin, Institut für Informatik, Oktober 2000.
<http://kazan.inf.fu-berlin.de/echalk/docs/report001031.pdf>
- [EC04] *Electronic Chalkboard homepage*. 2004.
<http://www.e-chalk.de>
- [eC04] *eCase – Mobile Unit for Lecture Recording homepage*. 2004.
<http://www-i7.informatik.rwth-aachen.de/d/projects/uli/ecase.html>
- [Em99] F. Embacher. *Acceptance of a Maths Online Project*. Paper prepared for the 10th International Conference of the Society for Information Technology & Teacher Education (SITE). San Antonio, Texas, February 28 – March 4, 1999. Association for the Advancement of Computing in Education (AACE), Charlottesville, 1999, p. 955.
<http://www.ap.univie.ac.at/users/fe/MERLIN.MPI/site99.doc>
- [Em04] *Emilea-stat homepage*. 2004.
<http://www.emilea.de>
- [FI04] *FreeImages homepage*. <http://www.freeimages.co.uk/>
- [Fl03] *Florida Today*. *Columbia Lost: NASA culture played down risks*. Mar 30, 2003.
<http://www.floridatoday.com/columbia/columbiastory2A47849A.htm>
- [Gr03] D. de Gruijter, L. van der Kamp. *Statistical Test Theory For Education And Psychology*. 2003.
<http://icloniis.iclon.leidenuniv.nl/gruijter/>
- [Ga00] S. Garibaldi. *Bloom's taxonomy in mathematics*. 2000.
<http://www.mathcs.emory.edu/~skip/prop/blooms.html>
- [GS02] H. Gumm, M. Sommer, W. Hesse, B. Seeger. *Einführung in die Informatik*. 2002.

- [He93] R. Hersch (Ed.). Visual and Technical Aspects of Type. Cambridge University Press. 1993.
- [He02] W. Hesse. Evolutionary object oriented software development and project management. 2002.
<http://www.mathematik.uni-marburg.de/~hesse/papers/EOS.pdf>
- [He98] Wolfgang Hesse. Baustein-orientiert statt phasen-zentriert: Neue Entwicklungsmethoden erfordern neuartige Vorgehensmodelle. 1998.
http://www.mathematik.uni-marburg.de/~hesse/papers.html#Hes_98a
- [Hm01] HartMath Java Computer Algebra Tool homepage. 2001.
<http://www.hartmath.com>
- [HU79] J.E. Hopcroft and J.D. Ullman. Introduction to Automata Theory, Languages and Computation. Addison Wesley. 1979.
- [Il03] Ilias opensource homepage. 2003.
<http://www.ilias.de>
- [IM04] In2Math homepage. 2004. <http://www.in2math.de>
- [I2M04] itex2mml Translator homepage. 2004.
<http://pear.math.pitt.edu/mathzilla/itex2mml.html>
- [ISC03] International Study Center. TIMSS 2003 Technical Report. 2003.
<http://isc.bc.edu/timss2003.html>
- [Ja04] Javadoc Tool Documentation 1.3. 2004.
<http://java.sun.com/j2se/1.3/docs/tooldocs/javadoc/>
- [1] [Je03] S. Jeschke, R. Keil-Slawik, R. Seiler, C. Thomsen, W. Veen. Multiverse - Mathematics in Collaborative Virtual Knowledge Spaces (EU-Hearing). August 2003.
http://www.math.tu-berlin.de/~sabina/Talks/EU-Hearing_2003.ppt
- [Je04] S. Jeschke. Mathematik in Virtuellen Wissensräumen – IuK-Strukturen und IT-Technologien in Lehre und Forschung. 2004.
http://edocs.tu-berlin.de/diss/2004/jeschke_sabina.htm
- [JO03] Journal of Online Mathematics and its Applications. Mathlet Definition. 2003.
<http://www.mathdl.org/criteria.html#mathlet>
- [JSK04] S. Jeschke, M. Kohlhasse, R. Seiler: eLearning-, eTeaching- & eResearch-Technologien – Chancen und Potentiale für die Mathematik, DMV-Mitteilungen. No. 78, June 2004.
- [JV04] JavaView homepage. 2004.
<http://www.javaview.de>
- [JCM01] Java Components for Mathematics homepage. Hobart and William Smith Colleges. 2001.
<http://math.hws.edu/javamath/>

- [Ke98] C. Kettenbach. Analyse des V-Modells als Entwicklungsstandard für IT-Systeme des Bundes – ein Konzept zur inkrementellen Softwareentwicklung. Diploma thesis. 1998.
<http://www.v-modell.iabg.de/Diplom/Kettenbach.pdf>
- [Kl02] M. Klesse. Evaluation von "mathe online" für den Übergang Schule/Hochschule, Seminararbeit am Institut für Wirtschaftsinformatik, Universität St. Gallen. 2002.
- [Kn03] G. Knolmayer, C. Montandon. Eignung multimedialer Lernobjekte zur Erreichung der in Blooms Taxonomie unterschiedenen Lernziele. Internationale Tagung Wirtschaftsinformatik. 2003.
<http://www.ie.iwi.unibe.ch/services/konferenzen/wi2003/resource/Dresden%20submitted%202003-05-16.pdf>
- [Kn82] D. Knuth. Documented T_EX source code. 1982.
<http://www.ctan.org/tex-archive/systems/knuth/tex/tex.web>.
- [Lo03] V. Lowndes, S. Berry. In-Depth Learning in Mathematics Courses. MSOR Connections Aug 2003 Vol 3 No 3.
<http://ltsn.mathstore.ac.uk/newsletter/aug2003/indepthlearning.pdf>.
- [Ma99] R. Martin. Failure Case Studies in Civil Engineering Education. 1999.
http://www.eng.uab.edu/cee/reu_nsf99/Report.PDF
- [Ma04] Umfrageergebnisse zur Unterrichtsreihe "Newton- und Intervall-Halbierungsverfahren"
<http://www.mathletfactory.de/application/bischof/statistik.html>
- [ML01] B. McLaughlin. Java and XML. O'Reilly. ²2001.
- [MIT03] MIT Open Courseware homepage. 2003.
<http://ocw.mit.edu>
- [MJ91] W. Jank, Hilbert Meyer. Didaktische Modelle. Cornelsen Scriptor. ³1991.
- [Mo02] Le Massachusetts Institute of Technology choisit la gratuité sur le Web. In: Le Monde October 21, 2002.
<http://www.lemonde.fr/article/0,5987,3416-295113-,00.html>
- [MO91] V. Midoro, G. Olimpo et al. Multimedia Navigable Systems and Artificial Intelligence. In: Lewis, Otsuki (eds). Advanced Research on Computers in Education. Proceedings of the IFIP TC3 International Conference on Advanced Research on Computers in Education. North Holland 1991.
- [Mo93] K. Morisse. Datenstrukturen und Speicherverwaltung in MuPAD. In: mathPAD Vol. 3 No. 2. 1993.
<http://www.mupad.de/mathpad.shtml>
- [MO03] maths online homepage. 2003.
<http://www.univie.ac.at/future.media/moe>

- [MSWWF99] Ministerium für Schule und Weiterbildung, Wissenschaft und Forschung des Landes Nordrhein-Westfalen. Richtlinien und Lehrpläne für die Sekundarstufe II – Gymnasium/Gesamtschule in Nordrhein-Westfalen. 1999.
https://www.ritterbach-interaktiv.de/verlag/schulwelt/lp_online.asp
- [Mu95] H. Muckenfuß. Lernen in sinnstiftenden Kontexten. Berlin: Cornelsen. 1995.
- [Mu04] Mumie Projekt homepage. 2003.
<http://www.mumie.net>
- [Mu01] J. Muzio, T. Heins, R. Mundell. Experiences with Reusable eLearning Objects: From Theory to Practice. 2001.
http://www.cedarlearning.com/CL/elo/eLearningObjects_sml.pdf
- [Mv04] Multiverse homepage. 2004.
<http://www.math.tu-berlin.de/multiverse/>
- [MT01] Mathematical Java homepage. Division of Mathematics & Computer Science of Emporia State University. 2001
<http://mathcsjava.emporia.edu/WebContent/default.htm>
- [MV98] H. R. Maturana, F. J. Varela The Tree of Knowledge. Boston, MA: Shambhala, 1998.
- [Ni97] J. Nielsen. Changes in Web Usability Since 1994.
<http://www.useit.com/alertbox/9712a.html> 1997.
- [NY01] Auditing Classes at M.I.T., on the Web and Free. In: New York Times April 4, 2001.
<http://education.mit.edu/tep/11125/opencourse>
- [Ob98] P. Oberhuemer. mathe online. Beitrag zum 8. internationalen Symposium zur Didaktik der Mathematik Universität Klagenfurt. 1998.
http://www.mathe-online.at/literatur/symp_klu98.doc
- [OM04] OpenMath homepage. 2004.
<http://www.openmath.org>
- [Os00] G. Ossimitz. Gesamtbericht zur Evaluation von mathe-online. Report for the European Social Fonds (ESF) ADAPT project. 2000.
<http://www.mathe-online.at/zweiterbw/dokumente/endber.PDF>
- [OECD99] Organisation for Economic Co-operation and Development (OECD). Measuring Student Knowledge and Skills: A New Framework for Assessment. 1999.
<http://www.pisa.oecd.org/Publicatn/Assess2000.htm>
- [OECD03] Organisation for Economic Co-operation and Development (OECD). The PISA 2003 Assessment Framework. 2003.
http://www.pisa.oecd.org/Docs/Download/PISA2003Frameworks_final.pdf
- [P4L04] Pics4Learning – copyright friendly images for education. homepage. 2004
<http://www.pics4learning.com/>

- [Pe85] H. Petroski. From Slide Rule to Computer. In: To Engineer Is Human. St. Martins Press, New York, NY. 1985.
- [Py04] Pyramit homepage. 2004.
<http://www.pyramit.de/>
- [Ve78] F. Vester. Denken, Lernen, Vergessen. dtv. 1978.
- [Wa67] P. Watzlawick, J. Beavin, D. Jackson. The Pragmatics of Human Communication. Norton, New York. 1967.
- [Wa91] Maple Language Reference Manual. Waterloo Maple Publishing. 1991.
- [WM97] R. Wilhelm, D. Maurer. Übersetzerbau. Springer. 1997.
- [Wo91] S. Wolfram. The Mathematica Book. Cambridge. 1991.
- [Wo02] L. Wood, G. Smith, P. Petocz, A. Reid. Correlation between student performance in linear algebra and categories of a taxonomy. in: Proceedings of 2nd international conference on the teaching of mathematics at the undergraduate level. Greece 2002.
<http://www.math.uoc.gr/~ictm2/Proceedings/pap338.pdf>
- [Sch02] R. Schulmeister. Grundlagen hypermedialer Lernsysteme. Oldenbourg Wissenschaftsverlag. ³2002.
- [Sa02] S. Si Alhir. Understanding the Unified Process. Published in: Methods & Tools (March 2002). <http://home.earthlink.net/~salhir#understandingtheup>
- [Sch03] I. Schwank. Einführung in funktionales und prädikatives Denken. Zentralblatt für Didaktik der Mathematik 3/2003.
- [Si97] D. Siegel. Creating killer websites. Indianapolis: Hayden Books. 1997.
- [Sun04] Sun Java Tutorial. Defining and Using Applet Parameters. 2004.
<http://java.sun.com/docs/books/tutorial/applet/appletonly/param.html>
- [Te04] Mumie TestletFactory homepage. <http://mumie.iram.rwth-aachen.de/testletfactory>
- [Th99] F. Thissen. Lerntheorien und ihre Umsetzung in multimedialen Lernprogrammen – Analyse und Bewertung. In: BIBB Multimedia Guide Berufsbildung. Berlin 1999
<http://www.frank-thissen.de/lernen.pdf>
- [Th04] L. Thorlacius. Aesthetics and function in web design. Published in: Proceedings of European CADE (Computers in Art and Design Education) 2004.
http://http://asp.cbs.dk/cade2004/proceedings/fullpapers/14.thorlacius_final.fullpaper.pdf
- [TTH04] TTH – The T_EXto HTML Translator homepage. 2004.
<http://hutchinson.belmont.ma.us/tth/>
- [TUM01] Integration in der Ingenieuranalysis homepage. 2002. <http://www-hm.mathematik.tu-muenchen.de/integration/branch.htm>

- [UL04] ULearn(Math) – Unified Learning in Mathematics. Projektantrag zum Förderschwerpunkt “Neue Medien in der Bildung” – Förderlinie “eLearning-Transfer” eingereicht beim Bundesministerium für Bildung und Forschung (BmBF). 2004.
- [W3C04] W3C Math Home. 2004. <http://www.w3.org/Math/>
- [Wi59] E. Wigner. The Unreasonable Effectiveness of Mathematics in the Natural Sciences. Communications on pure and applied mathematics. Vol. XIII 001-14 (1960).
- [Wi81] E. Wittmann. Grundfragen des Mathematikunterrichts. Braunschweig, Wiesbaden.
⁶1981.
- [Y01] J. Yanik. A Math Toolkit for Java Developers. Journal of Online Mathematics Vol.1, No. 2, 2001.
<http://www.joma.org/vol1-2/articles/yanik/index.html>
- [Ya99] M. Yacci. Interactivity Demystified. A Structural Definition for Distance Education and Intelligent CBT. In: Performance Improvement Quarterly, Vol 12, No. 3, 1999.
<http://www.it.rit.edu/~may/interactiv8.pdf>
- [Ze95] F. Zech. Grundkurs Mathematikdidaktik. Weinheim, Basel ⁹1995
- [Ze03] Die verschenkten Kronjuwelen. In: Die Zeit 28/03. 2003.
http://www.zeit.de/2003/28/C-Open_Courseware

Danksagung

Die Entstehung dieser Arbeit ist durch das Zutun vieler Leute überhaupt erst möglich geworden. Ich bedanke mich bei Ulrik Schroeder und Volker Enß für die Übernahme der Betreuung der Arbeit und für den großen Freiraum, den sie mir bei ihrer Entwicklung und Ausgestaltung gelassen haben. Den Mitarbeitern des Instituts für Reine und Angewandte Mathematik der RWTH Aachen danke ich für ihre Unterstützung des Mumie-Projektes und die Bereitschaft, einzelne Applets zu begutachten und zu testen; ebenso bedanke ich mich bei den Mumie-Teams der TU Berlin und TU München für die gute Zusammenarbeit.

Unter den zahlreichen Lehrern, die mir beim Schuleinsatz der Applets behilflich waren möchte ich besonders Markus Bischof hervorheben, der sich viel Zeit genommen hat, um eine Unterrichtsreihe mit Mathlets zu entwickeln und dabei mit vielen wertvollen Rückmeldungen die Entwicklung des Systems vorangebracht hat.

Meinen Kollegen Olaf Post und Fernando Lledó Macau danke ich für die alltäglichen Einladungen zum Kaffee und für die Durchsicht des Manuskriptes.

Meinen Brüdern Jan und Moritz verdanke ich Hinweise auf sprachliche Fein- und Grobheiten. Zum Schluss und vor allem möchte ich meinen Eltern danken, die mir meine Ausbildung ermöglicht und mich in jeder Lebenssituation unterstützt haben.

Lebenslauf

| | |
|------------------|--|
| 03. 04. 1972 | geboren in Köln; Eltern: Dr. jur. Hans H. Paehler, Katrin Paehler geb. Perseke |
| 1991 | Abitur am Konrad Adenauer Gymnasium Meckenheim |
| 1991 - 1992 | Zivildienst an der Malteser Rettungswache Rheinbach; Ausbildung zum Rettungssanitäter |
| 1992 - 1994 | Studium der Mathematik und Physik an der Rheinischen Friedrich-Wilhelms-Universität Bonn |
| 1994 - 1999 | Fortsetzung des Studiums an der Westfälischen Wilhelms-Universität Münster (Zwischenprüfungen 1994 und 1996); Aufnahme des Studiums der Informatik |
| 1995 - 1998 | Gründung von netserve oHG/trisinus GmbH; Leitung des Bereichs WWW-Entwicklung |
| 1996 - 1998 | Studentische Hilfskraft im Fachbereich Mathematik und Informatik im Bereich Systemadministration |
| 1998 - 1999 | Studentische Hilfskraft am Institut für Geoinformatik in den Bereichen Systemadministration und Software-Entwicklung |
| 1999/2000 | 1. Staatsexamen Sek II/I in Mathematik und Physik; Staatsexamensarbeit: 'Visualisierung von Funktionen einer komplexen Veränderlichen insbesondere im Zusammenhang mit der Riemannschen ζ -Funktion' |
| 2001 - 2002 | Referendariat an der Ursulinenschule Köln; Leitung der WWW-AG, Entwicklung eines Curriculums und einer webgestützten Lehrerfortbildung zur Computergraphik |
| Mai - Juli 2002 | Wissenschaftliche Hilfskraft am Institut für Geometrie und Praktische Mathematik (IGPM) der RWTH Aachen, Betreuung des Mathematischen Praktikums und Weiterentwicklung der Graphikbibliothek IGL |
| seit August 2002 | Wissenschaftlicher Mitarbeiter am Institut für Reine und Angewandte Mathematik (IRAM) der RWTH Aachen im Projekt Multimediale Mathematik-Ausbildung für Ingenieure (Mumie) |