On the Emergence of TP-based Educational Math Assistants

Walther Neuper
wneuper@ist.tugraz.at
Institute for Software Technology
Graz University of Technology
A-8010
Austria

Abstract

Presently Computer Algebra Systems, Dynamic Geometry Systems and Spreadsheets cover most of e-learning in high-school mathematics and as well are used for education in formal parts of science. Recently and largely unnoticed in public, the academic discipline of interactive and automated Theorem Proving (TP) has become of major importance for mathematics and computer science.

This paper considers the promises of TP technology for education in science, technology, engineering and mathematics at the full range of levels from high-school to university.

Starting from prototypes of TP-based educational mathematics systems, conceptual foundations are considered: TP-based software which implements reasoning as an essential part of mathematical thinking technology. Educational objectives for the development of TP-based systems are discussed and concluded with some predictions on possible impact of TP-based educational mathematics assistants.

The final conclusion suggests to announce the emergence of a new, TP-based generation of educational mathematics software.

1 Introduction

"Recently and largely unnoticed in public, applications in science and technology drove the development of automated and interactive theorem proving (TP) technologies, which have become of major importance for mathematics and computer science. Although based on expressive logical foundations and implemented in a highly trustable way, and although used in some scenarios roughly similar to mathematical tutoring systems, their potential for a wide-spread education technology is unexplored: beyond proving theorems, TPs [1] can manage formal content, check its logical consistency, and verify given problem solutions".

[1] This paper uses the abbreviation "TP" for two different things without danger of confusion: the respective software products and the respective underlying concepts and technologies.

The above text introduces a project proposal, abbreviated "Deducation" (deduction ∩ education) in 2012, which first time joined experts from Europe's leading TPs, Coq [8][2] and Isabelle [22][3]. Writing this proposal led to the first collection of publications [28] on the topic "TP Components for Educational Software". On the way of TP towards educational practice there is also a first success: the project "E-Math" [4] will make TP-based educational software available at high-schools following these points:

- Build a web based environment for studying mathematics.
- Provide electronic course material to the web based environment with "structured derivations" method.
- Train teachers in structured derivations as a method of teaching mathematics.
- Carry out pilot studies of "structured derivations" as a teaching method.

The first pilot courses start in autumn 2012 in seventeen schools in four Baltic cities.

Besides first practical success broadly based theoretical work has been done. One line of such work is being pursued at CADGME, the conference on "Computer Algebra and Dynamic Geometry Systems in Mathematics Education" in a specific working group on the topic "TP Components in Educational Software (ConvMath [20], now eduTPS)" [5] [6] [7] , so it seems time to give a survey on this topic.

First the question arises, what is this TP and where does it come from. There is an overwhelming palette of educational software, most of it is based on well established standardised components, on Computer Algebra Systems (CAS), Dynamic Geometry Systems (DGS) and Spreadsheets — so, what does existing software *not* cover?

Furthermore, the theory of mathematics education has been extended with theories covering the introduction of Information and Communication Technologies (ICT) into education. A highly elaborated body of knowledge reflects a wealth of aspects, cognitive, socio-cultural, collaborative and social aspects. Based on these theories *"new software have been developed with potential impact on all phases of education and on informal contexts of education"* — a citation from p.2 of the 17th ICMI Study on "Mathematics Education and Technology — Rethinking the Terrain" [14]; however, this thorough accountancy of the state-of-the-art does not mention TP.

Above the importance of TP for mathematics and computer science has been mentioned (which is justified by significant contribution of TP within "Formal Methods" supporting engineers to master ever increasing complexity of technical systems) — but this does not imply that TP needs to be introduced to education, in particular not at high-school. So, what do TP's features promise for mathematics education? And if the contribution is really fundamental to mathematics, does it carry

[2] http://coq.inria.fr/
[3] http://isabelle.in.tum.de/
[4] http://emath.utu.fi/
[5] http://www.risc.jku.at/conferences/cadgme2009/WG/ConvMathAssist.pdf
[6] http://home.pf.jcu.cz/čadgme2010/annotations/ConMAs-10.pdf
[7] http://sites.dmi.rs/events/ 2012/CADGME2012/mformats.html (eduTPS).

over to STEM [8] education? What are the advantages for education and what impact can be expected for the science of STEM education?

The paper will tackle these questions as follows: §2 recalls some preliminaries: existing prototypes of TP-based systems in §2.1, general adoption of software in STEM education in §2.3 and conceptional foundations in "reasoning" — as foundations of both, mathematics *and* TP, are discussed in §2.3. The main section §3 investigates TP's features promising for education (§3.1), shows that the wide coverage of TP technology allows for "software models of math" (§3.2) and is an integrative concept for math software (§3.3). Finally some predictions are tried on impact to be expected for the practice and research in STEM education in §4.1 and §4.2 respectively, as well as on educational administration (§4.3), which in §5 conclude with the announcement of a new generation of TP-based educational math assistants.

2 Mathematical Software and STEM Education

Starting from some notes on some prototypes of TP-based educational math assistants in §2.1 a wide perspective is opened: A perspective on all mathematical software which became relevant for education in §2.2, because TP systems are distinct from CAS [9] and other kinds of software, as CAS are distinct from DGS [10] — and nevertheless, TP will be considered as integrative basis for both, CAS and DGS. Another perspective recalls the fact that "reasoning" is a conceptual base of both, of TP technology and of mathematical thinking.

2.1 Prototypes of TP-based Educational Math Assistants

Any description of TP-based systems influential in education would require lots of space and would, nevertheless, never be complete. The selection below is rather arbitrary and focuses features typical for TP and relevant for further discussion.

GCLC is a tool for visualizing and teaching geometry [15] [11], and for producing mathematical illustrations. GCLC provides easy-to-use support for many geometrical constructions, isometric transformations, conics, parametric curves, flow control,etc. The basic idea behind GCLC is that constructions are formal procedures, rather than drawings. Thus, in GCLC, producing mathematical illustrations is based on "describing figures" (in a custom geometry construction language) rather than of "drawing figures". This language is integrated with automated TP. So GCLC can *prove* (by logics, not by numerical computation) that the intersection O_1 of $\overline{CB}$ and $\overline{AC}$ is the same point as the intersection of $\overline{CB}$ and $\overline{AC}$, see the line at the bottom of the left window in Figure 1 on p.113:

Socos is a programming environment [3, 4] [12] where correctness proofs are developed hand-in-hand with the program. The loop invariants are formulated before the code. Once the invariant

[8] "**STEM**" is **S**cience, **T**echnology, **E**ngineering and **M**athematics.
[9] "CAS" abbreviates Computer Algebra Systems, like Mathematica, Maple, Maxima.
[10] "DGS" abbreviates Dynamic Geometry Systems like GeoGebra, Cabri, Cinderella.
[11] Available at http://poincare.matf.bg.ac.rs/ janicic/gclc/
[12] Available at http://www.imped.fi/socos

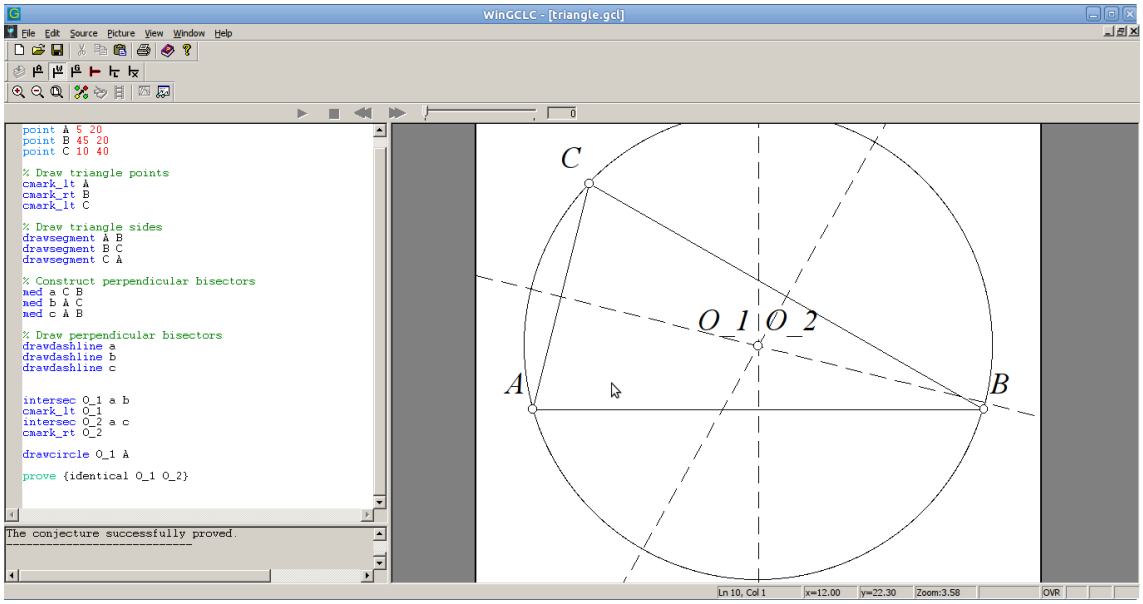


Figure 1: GCLC can *prove* {*identical O_1 O_2*} (bottom left)

structure is defined, the code is added in small increments while continuously being verified to be consistent with the invariants. The programmer draws the invariant diagram in a graphical IDE (implemented as a plug-in to Eclipse) and the tool generates the verification conditions and checks them automatically. Figure 2 shows one (of the infinitely many possible) diagrams, where the nodes, are labeled by predicates (invariants), and the arcs are labeled by statements — while all these formulae and their logical inter-dependencies are checked by use of a TP:

Structured Derivations is the format for interactive calculations [2] checked by TP in the E-Math project mentioned in §1. The front-end connects to a server via Internet, allowing student solutions to be checked on request. All errors found in the proposed solution will immediately be reported back to the student (and possibly to the teacher). The main challenge here is ensuring that the feedback is timely and presented in a comprehensible format, so that interaction with the system in an explorative way continuously motivates the student to improve his or her reasoning skills. Figure 3 on p.115 shows, how close the format is to traditional work with paper and pencil:

Isac is an experimental system [13] based on the TP Isabelle [22] and aiming at STEM education. The focus of the experiments is "next-step-guidance" [21]: if the learner gets stuck during interactive construction of a problem solution, the system gives various kinds of help at any step: one or several rules to be applied at the current step and/or the formula of the step, probably with gaps to be filled, displays underlying specifications, algorithms or theories on request — the latter is shown in the right window in Figure 4 on p.116:

[13] Available at http://www.ist.tugraz.at/isac/

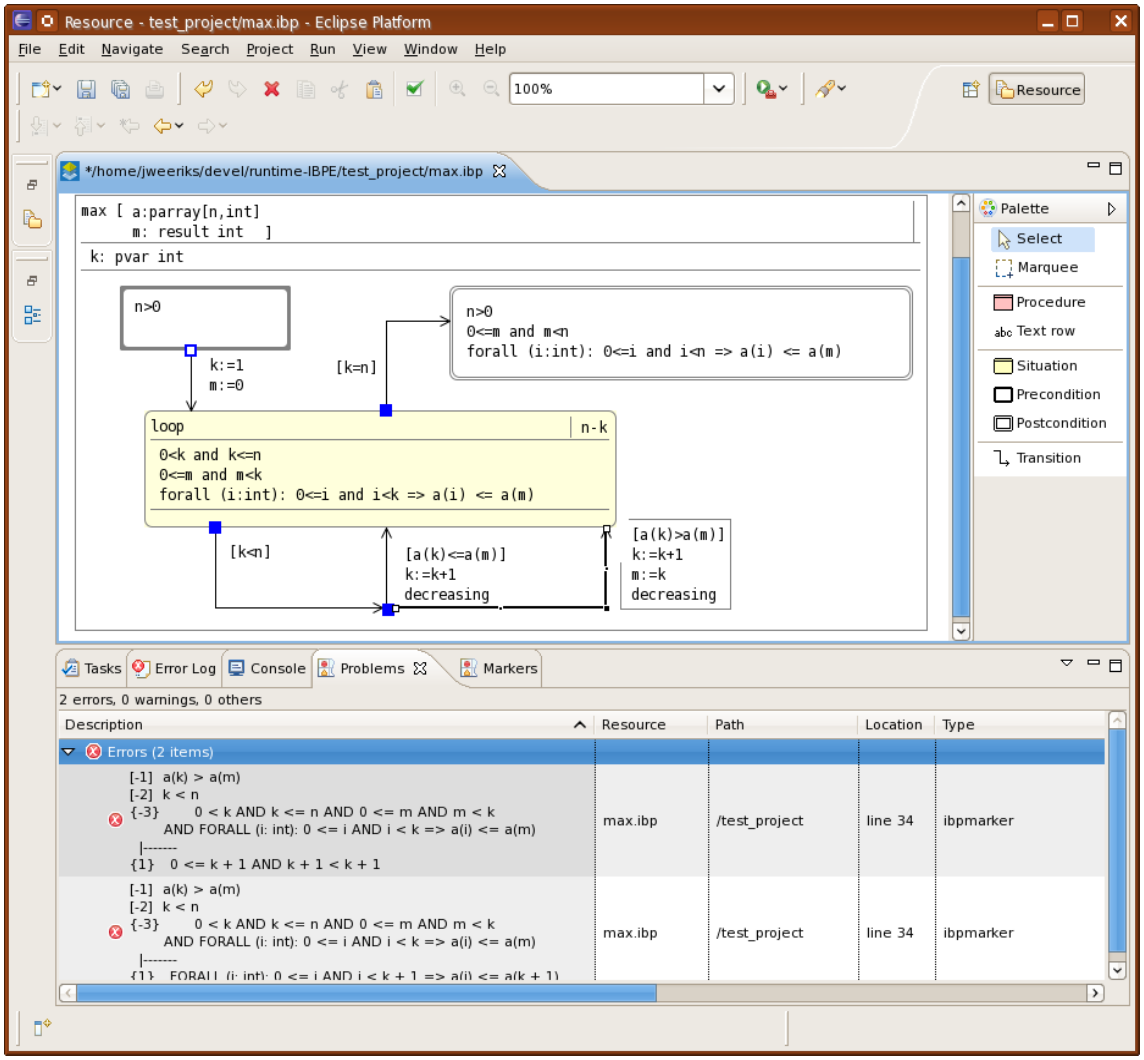


Figure 2: Socos checks $\forall(i : int) : 0 \leq i \wedge i < n \Rightarrow a(i) \leq a(m) \longrightarrow ok$ etc.

2.2 Adoption of Mathematical Software in Education

Since software has been used in mathematics education in a serious way, such software has been either adopted directly from engineering domains or has been developed further from products previously established in science and engineering (the major exception, DGS, will be mentioned). So, when looking at the future of educational mathematics software, a survey on mathematically biased software seems appropriate.

The survey in Fig.2.2 on p.117 has three columns: the left one shows three academic disciplines, the middle one shows the domains, software tools have been created for, and the right column shows the products shaped out by standardisation over the years (the spatial distances between the items try to express close/loose relations):

In the left column the discipline of (computer) *mathematics* is the source of the software-tools in the middle column, the disciplines of *applied sciences* in academia and industry (introduced as

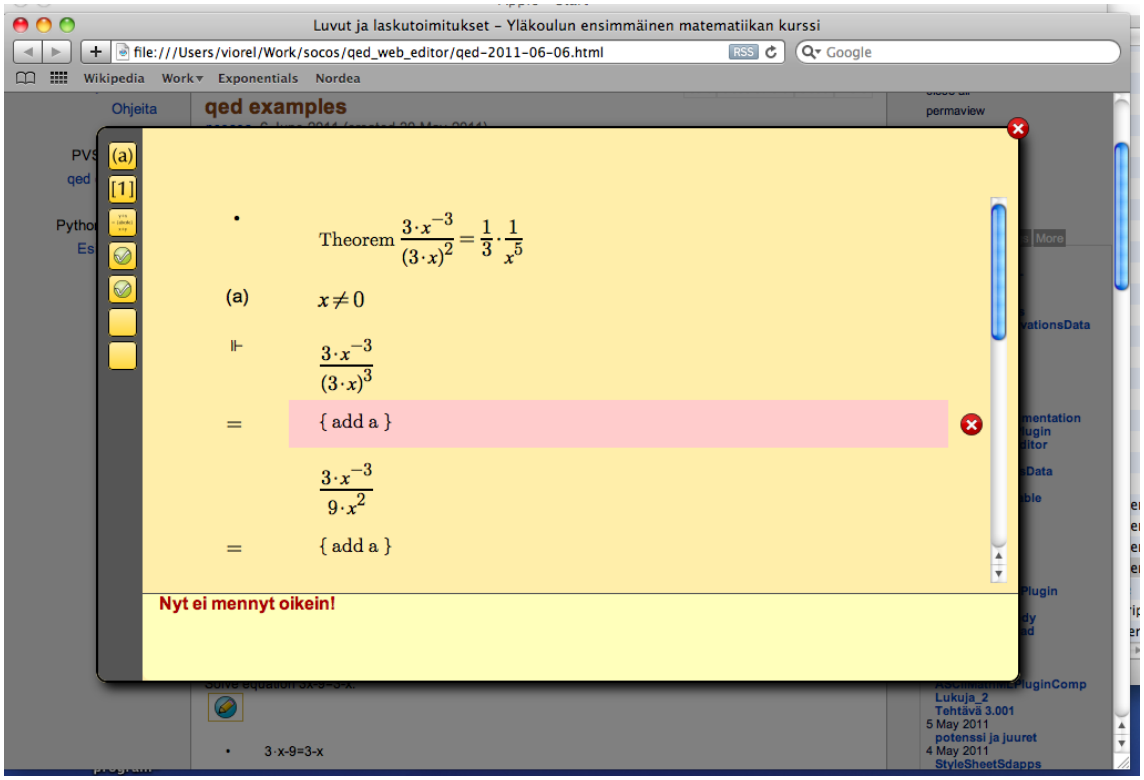


Figure 3: Structured derivation editor with erroneous step marked.

"STEM" on p.112) drove and still drive further development, usually leading to some kind of standardised products listed on the right. Remarkably, development of Computer Algebra (Macsyma 1968) started at the same time as development of Theorem Proving (Automath 1967); however, request from academia and industry drove development to mature CAS very soon, while TP only recently reaches relevance in industry.

Relevant for *"math. education"* and development of respective software are windows of opportunity at the transition of software from academia to industry: At this transition Spreadsheets and CAS have been adapted to educational needs (DGS are the major exception [14], developed as educational tools from scratch); now this kind of window can be exploited for TP as well, now the respective systems are open source and interested in attracting future scientists and engineers. In a few years the successors of Socos, Coq or Isabelle might have adapted to the needs of industrial practice such that requests from the side of education will be not as well accepted as presently.

"TPS" above could stand for TP-systems; however, it is marked by *"?!?"* due to the authors opinion, that the primary kinds of software used in high-schools will be TP-based systems where TP technology works in the background as done in the four examples presented in §2.1 above and discussed in detail in §3.3 below. However, TP-systems (academic systems downgraded for education, probably in several levels, for instance [40]) are expected to provide smooth transitions to academic mathematics in the higher grades of high-school.

[14] Exceptions in formula-based mathematics are MathXpert [5], T-Algebra [31] and others.

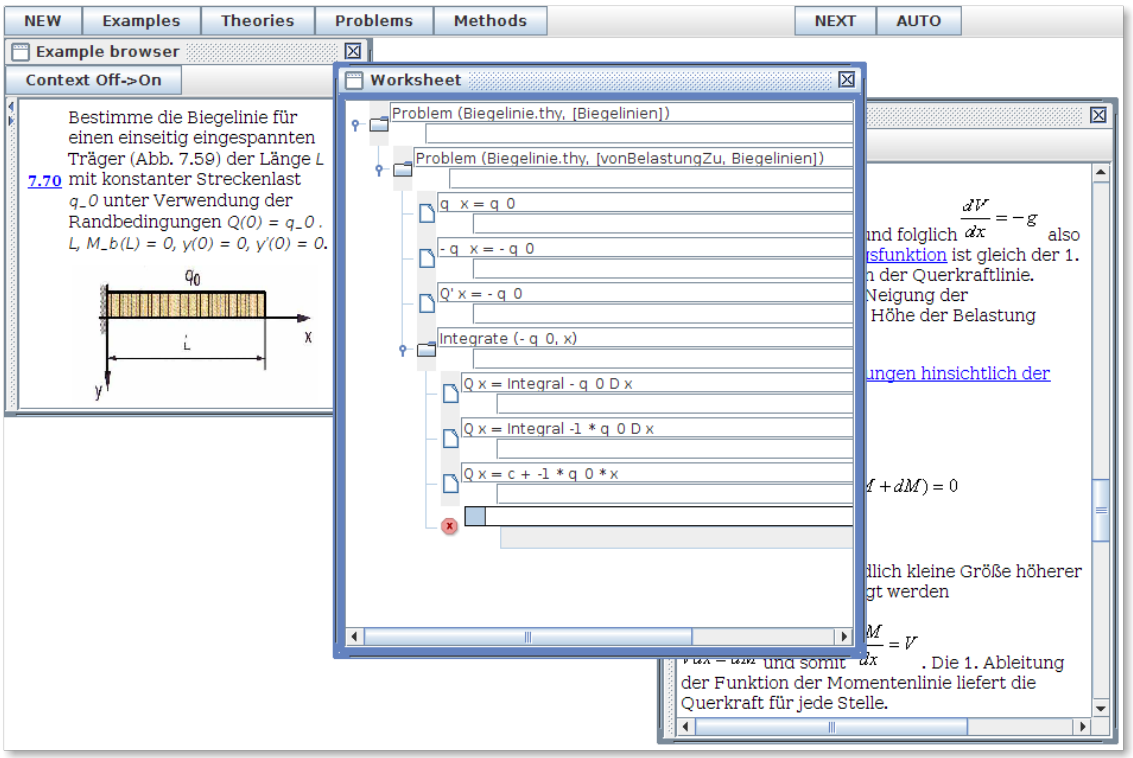


Figure 4: ISAC checks input $\int -q_0\ dx = c - q_0 \cdot x$ in the Worksheet.

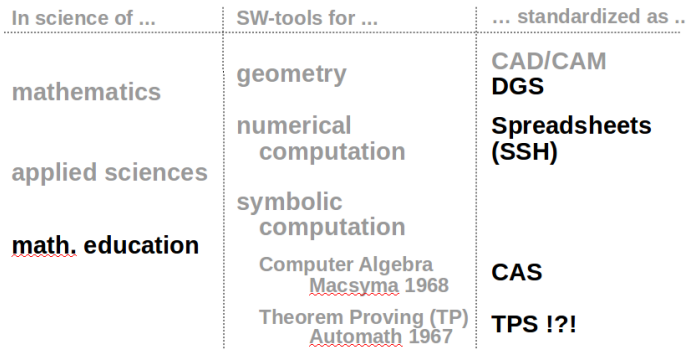
2.3 "Reasoning" in TP technology and in Mathematical Thinking

Rigorous reasoning is fundamental to mathematical thinking (such that an approved publication like [14] not even lists it in the Subject Index, apparently due to ubiquitous occurrence).

However, mathematisation (see below) of reasoning succeeded only in the last century due to the work of logicians like Russel and Whitehead, Gödel, Church and Tarski. And mechanisation of logics in TP succeeded only in the last decades: The first "proof" of the Four-Colour-Problem by Appel and Haken was not accepted as a proof for good reasons: Appel and Haken could not prove that their program did not contain errors in 1976. Now, in 2005, Georges Gonthier used the TP Coq to create a surveyable proof [9]. In 2003 another proof of a difficult problem, Kepler's conjecture, was judged 99% correct by twelve referees after four years of work — so the flyspeck project [15] has been set up to produce a formal proof of the Kepler Conjecture. Since all that is very new, there still goes the saying, that a serious mathematician uses a computer only for LaTeX, for mail and for retrieving references.

As rigorous mechanical proving is still controversial in academic mathematics, it is even more in mathematics education. The 19th ICMI Study [18] states *"re-examination of the role of proof in the curriculum and of its relation to other forms of explanation, illustration and justification (including dynamic graphic software) has already produced several theoretical frameworks, giving rise to many discussions and even heated debates."*

[15] http://code.google.com/p/flyspeck/



This paper circumvents such debates [16] by approaching mathematics education by the way of mathematical software, asking: How can TP technology support development of mathematical thinking? Taking PISA's most recent competence model for mathematics [24], the above question can be approached by a more concrete one: Which capabilities underlying mathematical thinking can be supported by TP in principle? PISA's "seven fundamental capabilities" can be supported by TP-technology as follows:

1. *Communication: ... perceiving the existence of some challenge and recognizing a problem situation ...* is not specific to TP, but general to ICT (with presentation tools, mail, information retrieval).

2. *Mathematising: ... transforming a problem defined in the real world to a strictly mathematical form ...* is a prerequisite for TPS support such that even libraries of formal specifications need to be prepared; specification is indispensable, but might be hidden from learners not specifically interested.

3. *Representation: ... selecting, interpreting and using a variety of representations to capture a situation ...* is well implemented in DGS showing geometric and analytic representation of objects if requested; TP technology can add logical representation, ready for investigation in case of particular interest.

4. *Reasoning and argument: ... logically rooted thought processes that check a justification that is given ...* have a strong counterpart in TP technology where each step of a problem solution must have a mechanical justification; these justifications are human readable and can be augmented by multi-media explanations.

5. *Devising strategies for solving problems: ... critical control processes that solve problems ...* involve inquiry-based processes which efficiently can be supported by software: "next-step-guidance" (see §3.1) allows to quickly try alternative specifications and algorithms and to accurately compare them.

6. *Using symbolic, formal and technical language and operations: ... within a mathematical context ...* is considered a major obstacle already in using CAS; however, TP-based software has

[16] §3.2 will come back to "reasoning" with respect to optional, dynamically linked representations.

formal specifications and algorithms in the background, which can be used for additional help (last not least by "next-step-guidance").

7. *Tools: ... being able to make use of various tools that may assist math activity ...* while the comprehensive assistance expected for TP-based tools suggests to rethink, if separation of this point from the other six points is still appropriate.

So, all capabilities relevant for doing mathematics are related to some support by TP-based technology in the enumeration above — but caution is called for methodological reasons: PISAs "seven fundamental capabilities" represent branches of a tree, which is firmly rooted in the grounds of science of mathematics education; the seven branches cover mathematical capabilities in full completeness at the well settled state-of-the-art in the science of mathematics education.

Above, specific kinds of TP support are attached to the tree's branches — to the authors best knowledge, this is the first attempt to give a survey on TP features from the point of view of STEM education. So this assemblage of TP features is necessarily ad hoc (like hanging strange Christmas tree balls into the branches of a well-rooted Christmas tree; and in particular, the conclusion "that TP technology would support everything viewed as essential for math education" is invalid). This might be particularly painful to TP experts, who know their science well rooted in mathematics and who would like to see some of their well-elaborated structures in this assemblage. Structuring this new field will, in the author's opinion, require joint efforts for some years.

However, the author is convinced, that the above ad hoc relations provide starting points useful for mutual approaches between education and TP. And probably, the structure suggested above might serve future field-test on TP-based mathematics assistants in educational practice.

Staying optimitstic that way, we see the above points also outlining phases and iterations in problem solving, and conclude: almost the full range of mathematical problem solving can be supported by software. With respect to these possibilities the statement *"marginalization of technology [...] points, in part, to a failure to theorize adequately the complexity of supporting learners to develop a fluent and effective relationship with technology in the classroom"*([13] cited from [14]) suggests the comment: mastering such complexity cannot be accomplished by technology limited to (numerical and symbolic) computation (by CAS) and drawing (by DGS) — TP technology adds logic and reasoning as a feature fundamental to mathematical thinking.

The subsequent section will show details of how TP-technology can support a wide range of mathematical problem solving.

3 TP's Features Promising for Education

Combining the two facts, that (1) reasoning is fundamental to mathematical thinking and (2) TP technology implements reasoning and logics, the wide coverage of TP support for doing mathematics does not come as a surprise anymore. The following features are a preliminary account collected from the four TP-based prototypes already described, now restructured alongside educational points of view.

3.1 Technological Features: Flexibility, Transparency, Guidance

The various features of many TP-based systems have been distilled into three groups by discussions in the three CADGME working groups and during proposal writing for Deducation. So the three groups are a collection from existing systems' features, and not some abstract requirements capture. The description of the features will closely refer to the four TP-based prototypes introduced in the above §2.1.

Flexibility and reliability in checking user input is the main contribution of TP to educational software: given a logical context from a formal specification, an input formula establishes a proof situation — the formula must be (automatically) derivable from the context. Such automation allows to support stepwise problem solving in software as shown in Figure 3 on p.115 for steps in a calculation; such support is possible for all kinds of steps.

Also DGS begin to include TP methods [19]; for instance, GeoGebra [17] already proceeds from algebraic methods to TP methods [27] which come close to human readable proofs. In order to work, TP requires formal specification of the problem to be solved; this is specifically interesting for geometry, because it involves elementary clarifications, for instance what a triangle is, see Figure 3.1. In order to *prove* $\{$*identical* O_1 $O_2\}$ from Figure 1 on p.113, TP requires the "non-degeneracy condi-

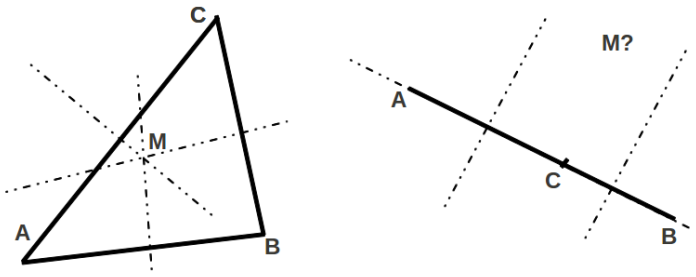


Figure 5: When proving properties of the circum center M: What is a triangle?

tions" that the points A, B and C are not collinear and $A \neq B \neq C$. Such specifications are easily comprehensible [18].

And if a student experiences the invariant M by dragging the points A, B, C, she or he might be interested not only in a dynamically linked algebraic representation of the geometric elements, but also in a dynamically linked logical representation shown on request. Questions like "What happens with a proof in a degenerate case?" link geometric intuition with formal mathematics in a novel and worthwhile way.

Transparency of the systems is prepared by TPs like Coq and Isabelle following the "LCF-paradigm": in these TPs all knowledge is mechanically deduced from the basic axioms and definitions of math-

[17] http://www.geogebra.org/

[18] Building also constructions on Hilbert's or Tarski's axiom systems poses problems not yet solved for automated proofs.

ematics — following Russell's principle of "honest toil" ([32] p.71). And this knowledge is represented in a format which is not only interpreted mechanically by the TP, but which can also be read by humans.

The reader is invited to look-up some of the knowledge mechanised by Isabelle [19]; the theories *Groups, Rings, Fields* involve only few technicalities, also the definitions of numbers in *Int, Rat, RealDef, Complex*. So, for instance, a learner facing a term like $2 \cdot \pi$ can be mechanically guided to the definition of multiplication, to the proof of commutativity of multiplication (proved for Cauchy Sequences), etc. The issue is not any more to create that knowledge, but to filter off overwhelming details. The upcoming graphical user-interface Isabelle/jEdit [38] already allows to delve into underlying knowledge just by klicking tooltips.

The scope of the knowledge already exceeds the range required for high-school mathematics as well as for STEM education (see MacLaurin and Taylor Series, HOL_Multivariate_Analysis, etc), but the kind of knowledge is not all what is needed for TP-based systems; for instance, given this knowledge one can easily prove [20] ...

$$
\begin{array}{l}
lemma \\
\quad fixes\ x :: "'a :: \{idom, number\_ring\}" \\
\quad shows\ "x^2 * y = x^2\ \&\ x * y^2 = y^2 \longleftrightarrow x = 1\ \&\ y = 1 \mid x = 0\ \&\ y = 0" \\
\quad by\ algebra
\end{array}
$$

... but one has to find the solution first: the knowledge widely lacks algorithmic parts. Incorporating algorithmic knowledge from CAS into the logically reliable framework of TP is concern of ongoing R&D.

Next-step-guidance is the most notable adaption of TP technology to educational needs. Given the above mentioned feature of checking input allows to automatically generate feedback, when a formula cannot be derived from the context — then a learner might get stuck without knowing how to proceed towards a solution of a problem. In this case the system needs to "know the next step", and this feature can be accomplished by Lucas-Interpretation [21] combining deduction with computation.

The application of Lucas-Interpretation is connected with incorporating algorithmic knowledge into TP as mentioned; the CAS algorithms, programmed in a TP-based programming language [10], become verified and also accessible to step-by-step investigation.

Exploitation of next-step-guidance for adaptive user guidance in a TP-based educational math assistant are under construction [7].

3.2 Interactive, Complete and Transparent Models of Maths

TP's features identified so far relevant for education suggest a new way of thinking about the role of computers, infeasible without TP.

"In the old way of thinking, computers were seen as human tutors and evoked a vision of the teacherless learning environment. New avenues for using technology take advantage of, rather than

[19] http://isabelle.in.tum.de/dist/library/HOL/

[20] http://isabelle.in.tum.de/dist/library/HOL/Groebner_Basis.html

marginalize, teacher, task, and classroom cultures" [25]. Confirming the teachers' role another vision can be added: the computer as a *model of mathematics* itself. Now, what are the features of such software models of mathematics ?

"Interactive" models are all of the four prototypes to a high degree, such it is possible to learn by just interacting with the system. The ISAC prototype goes a step further and strives for learning the same way as a good chess-program allows to learn playing chess just by trying some moves, going back and trying another strategy, probably by changing role with the system and watching how the system copes with a certain situation.

Quite analogously, TP's ability to generously and reliably check user-input, allows to model step-wise problem solving and to emulate traditional paper-and-pencil feel. Added next-step-guidance, the learner and the system can cooperate in going step by step towards a solution until the system (automatically) proves that a solution has been found (by proving the so-called post-condition of the problem's specification):

The learner might do some steps, explore several variants or might get stuck and request help; and the system checks the steps giving feedback, and adapts help to the behaviour of the learner by more or less gaps to fill in to the next step, etc.

Next-step-guidance also makes the problem with formatting input vanish: learners have <next>-step button and learn from just observing the system, see Figure 4 on p.116.

"Transparent" models allow to open up "black boxes" on the spot in a calculation and see intermediate steps. All prototypes previously described are clearly designed as "transparent models". Isabelle's feature of proof reconstruction supports such transparency; in case of rewriting, the term rewrite systems need to be modularised such that traces group rewrites into comprehensible "big" steps.

Since in a TP-based system each step has a mechanical justification related to the logical context, the learner can request justification for each step. Starting from the step related knowledge can be investigated; the reader is reminded the respective paragraph in §3.1. This feature of maths models goes significantly beyond the features of chess-programs: looking at the software mechanisms would not help learning chess — while look at these mechanisms, requested by learners due to spontaneous interest, definitely has the potential to help: the models advocated here are based on TP and thus on logics.

Also, if an input formula needs several steps for (automated) derivation from the context (which might be hidden in the first go), the learner might investigate the intermediate steps and be surprised, how many rewrites her or his "intuitively justified" step requires on the mechanical level.

"Complete" models cover the whole problem solving process, unlike CAS or DGS, which are a particular tools for certain parts within the whole process (the reader is reminded of the seven fundamental capabilities for mathematical problem solving in PISA's competence model in §2.3):

In a TP-based model work on the computer starts with a meaningful problem (see, for instance, the balcony in Figure 4 on p.116); the initial formalisation and specification might be prepared by an author (and even hidden from novices) ; the steps are supported by the "interactive" and "transparent"

model mentioned above until the specification's post-condition can be proved by the system (which is most likely hidden from most learners and done automatically behind the scenes).

The envisaged models are expected to be complete also in the sense, that they represent all steps of problem solving on the screen (in several layers, some of which might be hidden); several techniques of zooming and folding/unfolding can support changing from detail to survey. This kind of completeness of representation also makes an issue vanish, raised in [37] by the observation, that students start to develop debatable styles of protocol for their problem solutions using CAS and DGS. In any case, logical representations seem most fruitful when being optional and "dynamically linked" like the analytic representation in DGS.

The above has already been anticipated from an educational point of view: *"Feedback in intelligent tutors and CAS allow for the 'playability' of calculus and functions in the way that geometry has become 'playable' through dynamic geometry environments. The calculator can perform the microprocedures and let the student focus on the macroprocedures, which require higher-level processes"* ([11] cited by [25]) — this vision becomes much more reachable by TP-based technology now.

3.3 Integrative Concepts rather than Separate Systems

Figure 1 on p.113 shows a *prove* statement, which looks like an particular add-on to DGS. But §3.1 showed that proving has formal specification as a prerequisite — consequently such a DGS gets another conceptional foundation based on logics. And if a DGS adopts the conceptual foundations into the technological basis, then the advantages of TP become available like next-step-guidance in stepwise construction and the others.

In DGS the process of integrating TP has already started [27], and there are good reasons to expect continuation of this process.

Formula-based systems, for instance those shown in Figure 3 on p.115 and in Figure 4 on p.116 face the challenge to acquire CAS-functionality. Since CAS are themselves not based on logics, are not reliable (for instance, drop solutions of equations, are inconsistent with "multivalued functions") and are black boxes, meeting this challenge involves great efforts. These efforts cannot be accomplished by development of educational systems alone; rather development of educational systems is expected to take profit from ongoing and planned development aiming at industrial application.

Finally coming back to *"TPS !?!"* in the figure on p.117 with the considerations just above, it appears straight forward to see TP as unifying conceptual base for a process of system integration already started. For instance GeoGebra already integrated Spreadsheet, CAS and most likely will continue with TP — just following the requirements of educational practice, where teachers are just overwhelmed if forced to use separate tools for solving one problem in one lesson.

4 Expected Impact

The previous section showed the directions TP-based educational software might develop to; it also made clear the considerable amount of efforts to realise these developments. Such efforts need to be justified by clarification, what the use is for these efforts.

With regard to the deep changes in educational software's features, impact is expected to all aspects of education in the full range of STEM (science, technology and engineering), the research on education as well as educational administration; software development in general is affected, too.

4.1 Impact on Practice in STEM Education

"Certain uses of ICT give pupils access to advanced mathematical ideas, which are not currently considered in traditional curricula at the elementary and secondary school education level. These possibilities rely mainly on the potential of the ICT environments to facilitate learners in making crucial transitions towards a mathematical way of thinking. In this way, ICT can significantly alter how didactic and learning trajectories have been traditionally conceived" [33] p.219. Such ideas comprise mathematics of variation, modeling and generalisation [17, 30], or infinity and infinite processes [16, 34] — and ideas about *justification in different levels of rigor, theorem, axiom, definition, proof, formal system, formal reasoning* can be seen in the same line.

Probably curricula are right not to include such ideas in a too concrete manner: students' comprehending fundamental ideas and gaining deep insight can hardly be planned within a certain topic or a single lesson. On the other hand a *"good number of studies have shown that computer environments can play a role so that students, of very young ages, may works with sophisticated mathematical ideas"* [33] p.218, and *"ICT seem to facilitate transitional processes that have previously be reported as being highly complex for the vast majority of students, such as transitions from the particular to the general; from what is concrete to what is abstract; from intuitive perception to formal thinking; from non-mathematics to mathematical representations; etc."* [33] p.219

The key in supporting such transitional processes seems to be individualisation of learning: *"We also recognize that there may be diverse ways or paths for students to construct or develop mathematical thinking and problem solving competencies. We contend that student's use of different digital – and representational – means or technologies, or in accordance, to the facilities or potential associated with each tool"* [33] p.219.

This key has been addressed by the repeated confirmations above, that presentation of logical facts *may not* be isolated from concrete steps in concrete calculations or in the construction of concrete geometric objects — rather logical facts *shall* come in a "dynamically linked presentation", in the same way optional as is analytic representation linked with geometric objects in DGS.

The above *"may not"* and *"shall"* call for intimate cooperation between practice in education, software development and educational research. Already clear seems, that TP-based systems particularly promise to support open learning scenarios, inquiry-based and independent learning.

Impact on settings in informal education and on personalised ubiquitous learning paradigms can hardly be estimated. Just as an example, the TP-based system presented in Figure 3 on p.115 would allow a web-based public competition on solving equations in analogy to the competitions between Italian nobles during Renaissance. Technology-driven developments alongside "cloud computing" with TP-services called from handheld devices [39] and students doing maths homework by that way could well overtake professional expectations like in [12].

4.2 Impact on Research in STEM Education

The novel promises of TP technology, the full coverage of software support for phases in mathematical problem solving and the re-invocation of methodological foundations of mathematics suggest to rethink ontological questions about mathematical objects and processes and learning mathematics. Some of these questions might be:

Papert's Principle [26]: *"Some of the most crucial steps in mental growth are based not simply on acquiring new skills, but on acquiring new administrative ways to use what one already knows"* — is, for instance, "linked dynamic representation of logical items" an appropriate support for new ways to use knowledge about "justification in different levels of rigor, theorem, etc"?

Can acquisition of such knowledge, apparently belonging to "advanced mathematical thinking" [36], be boiled down to "webbing" and "situated abstraction" [23]? If *"the idea of webbing is meant to convey the presence of a structure that learners can draw upon and reconstruct for support – in ways that they choose as appropriate for their struggle to construct meaning for some mathematics"* (p.108) — can TP-based educational software *"designed in such a way that expressing generality can be made a central component of the computational setting and, thus, that webbing and situated abstraction are co-emergent constituents of the mathematical learning process"* (p.122)?

"The driving question of what new types of mathematical knowledge emerge as a result of access to digital technologies"[25] p.136 has been answered without regarding TP technology so far — so what are the answers when including TP into the selection of base technologies for educational software?

The above questions address long-term impact of early access to powerful mathematical ideas as discussed in [33]. *"However, research into ICT-based learning trajectories is still in its infancy. From a theoretical perspective, the idea of learning trajectories in the context of digital technologies still needs to be developed"* [33] p.220. §3 tried hard to show how automated TP-services can be more ore less be hidden in educational software; however, TP implements reasoning on a rigorous formal level — and the question, how to make the reasoning transparent to learners in which levels of learning, is really challenging. Particularly interesting seems the question, how the gap between "intuitive" high-school mathematics and "formal" academic mathematics might be bridged by TP-based software.

A further trajectory crossing over science, technology and engineering (STEM) is established by mathematics as basic thinking technology; also this trajectory is considered urgent [21] but it is not yet in the focus of present research.

Another kind of impact on research can be expected from the ability of TP-based software to produce high-level protocols of user-interaction; high-level means, that they convey interactions with notable semantics. This ability can be implemented at the level of abstraction, where input formulas are checked and accepted or rejected if they cannot be derived from the context. On the same level development of adaptive dialog guidance can exploit next-step-guidance (§3.1) which is at the very beginning [7], goes far beyond CA-based systems [35] and calls for involvement of educational research.

[21] http://www.ingenious-science.eu/

4.3 Impact on Administration of Education

Beyond the expectation, that TP-services will be available "from the cloud" on handheld devices for everyone ubiquitously, at least two further points will affect planning and administration of education.

One point is (probably central) availability of data from logging high-level interactions in stepwise problem solving on TP-based systems. For instance, this is already possible in the E-Math project mentioned in §1, and it might develop further [39].

Another point is more look-ahead and will be relevant as soon as the coverage of mathematics knowledge mechanised in TP-based systems will go towards 100%: Given that knowledge, mechanised support is feasible to manage cross-institutional courses of study: prerequisites can easily be identified when changing from one institution to the other.

5 Conclusions

TP's features, flexibility, transparency and guidance, enable design of software as interactive, complete and transparent models of mathematics. These advancements in technology are expected to have impact on STEM education, on educational research and on administration. Combining advancements in technology and expected impact suggests to conclude with: Let's announce the emergence of a new generation of TP-based educational mathematics assistants!

The domain of education is wide and multifaceted, and so is the domain of mathematics and respective software. So, if the latter focuses TP technology $\bigtriangledown$, the former focuses STEM education $\bigtriangleup$ as proposed by the European Commission [29], we can combine both in the follwing picture:

	Renewed Pedagogy for the Future
	==========
infinite variety of human thought	$\bigtriangleup$
distilled to science of math	STEM
abstracted to logics	individualise,
implemented in	meaningful math
software	inquiry-based learning
TP	embodyment, social experience
$\bigtriangledown$	effective practice in math education
==========	
Mechanised Foundation of Math	

The "announcement of a new generation" might join efforts in both domains to combine $\bigtriangledown$ and $\bigtriangleup$ to a common focus $\frac{\bigtriangledown}{\bigtriangleup}$. Such joint efforts might entail TP experts' acknowledgment that they have to offer specific services to educational software rather than simply downgrading their systems for education; and it might entail educators' acknowledgment, that mathematical software becomes more flexible and general, more adaptive and appropriate than before, if given logical foundations.

Acknowledgements

The author owes a great debt of gratitude to the members of the working groups at CADGME for the many discussions helping to identify software requirements in the practice of mathematics education.

The colleagues preparing the Deducation proposal helped to clarify facts about what is avaiable in TP already and what can be accomplished in the near future; very special thanks to this group ! Last not least I express my thanks to the referees, who helped a lot to narrow the points of view from education and from TP technology.

References

[1] *Theorem-Prover based Systems for Education (eduTPS)*. The Electronic Journal of Mathematics and Technology, 2013. to appear.

[2] R.-J. Back. Structured derivations: a unified proof style for teaching mathematics. *Formal Aspects of Computing*, 22(5):629–661, 2010.

[3] Ralph-Johan Back, Johannes Eriksson, and Linda Mannila. Teaching the construction of correct programs using invariant based programming. In *Proc. of 3rd South-East European Workshop on Formal Methods (SEEFM07)*. South-East European Research Centre (SEERC), 2007.

[4] Ralph-Johan Back, Johannes Eriksson, and Magnus Myreen. Testing and verifying invariant based programs in the SOCOS environment. In *Proc. of the International Conference on Tests And Proofs (TAP)*, volume 4454 of *Lecture Notes in Computer Science*, pages 61–78. Springer, 2007.

[5] Michael J. Beeson. Design principles of Mathpert: Software to support education in algebra and calculus. In N. Kajler, editor, *Computer-Human Interaction in Symbolic Computation*, pages 89–115. Springer, 1998.

[6] H Chick, K Stacey, and J. Vincent, editors. *The Future of the Teaching and Learning of Algebra. Proceedings of the 12th ICMI Study Conference*, Melbourne, Australia, 2001. University of Melbourne.

[7] Gabriella Daróczy and Walther Neuper. Error-patterns within "next-step-guidance" in TP-based educational systems. submitted to EJMT.

[8] Coq development team. Coq 8.3 reference manual. http://coq.inria.fr/reman, 2010. INRIA.

[9] Georges Gonthier. The four colour theorem: Engineering of a formal proof. In Deepak Kapur, editor, *ASCM*, volume 5081 of *Lecture Notes in Computer Science*, page 333. Springer, 2007.

[10] Florian Haftmann, Cezary Kaliszyk, and Walther Neuper. CTP-based programming languages ? considerations about an experimental design. *ACM Communications in Computer Algebra*, 44(1/2):27–41, 2010.

[11] K. Heid. Technology in mathematics education: tapping into visions of the future. In W.J. Masalski, editor, *Technology-Supported Mathematics Learning Environments: NCTM 67th Yearbook*. NCTM, Reston, 2005.

[12] C. Hoyles, I. Kalas, L.Trouche, L. Hivon, and R. Noss. *Connectivity and Virtual Networks for Learning*, chapter 22. Volume 13 of Hoyles and Lagrange [14], 2010.

[13] C. Hoyles and R. Noss. What can digital technologies take from and bring to research in mathematics education? In A.J. Bishop, M.A. Clements, C. Keitel, J. Kilpatrick, and F.K.S. Leung, editors, *Second International Handbook of Mathematics Education*. Kluwer, Dordrecht, 2003.

[14] Celia Hoyles and Jen-Baptist Lagrange, editors. *Mathematics Eduation and Technology — Rethinking the Terrain. The 17th ICMI Study*, volume 13 of *New ICMI Study Series*. Springer, 2010.

[15] Predrag Janičić. GCLC — a tool for constructive euclidean geometry and more than that. In *Mathematical Software – ICMS 2006*, number 4151, pages 58–73, 2006.

[16] K. Kahn, E. Sendova, A.I. Sacristán, and R. Noss. Making infinity concrete by programming never-ending processes. In *7th International Conference on Technology and Mathematics Teaching*, Bristol, UK, 2005.

[17] J. Kaput and M. Blanton. Algebraifying the elementary mathematics experience. part I: Transforming task structures. In Chick et al. [6], pages 344–351.

[18] Fou-Lai Lin, Feng-Jui Hsieh, Gila Hanna, and Michael de Villiers, editors. *Proceedings of the ICMI Study 19 conference: Proof and Proving in Mathematics Education*, volume 1,2, Taipei, Taiwan, 2009. The Department of Mathematics, National Taiwan Normal University.

[19] Filip Marić, Ivan Petrović, Danijela Petrović, and Predrag Janičić. Formalization and implementation of algebraic methods in geometry. In *THedu'11*, volume 79, pages 63–81, Wrocław, Poland, 2012. Electronic Proceedings in Theoretical Computer Science.

[20] Walther Neuper. Common grounds for modelling mathematics in educational software. *Int. Journal for Technology in Mathematics Education*, 17(3), 2010.

[21] Walther Neuper. Automated generation of user guidance by combining computation and deduction. In Quaresma and Back [28], pages 82–101.

[22] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[23] R. Noss and C. Hoyles. *Windows on Mathematical Meanings: Learning Cutures and Computers*. Kluwer, Dortrecht, 1996.

[24] OECD. PISA 2012 mathematics framework. Draft subject to possible revision after the field trial, OECD, 2, rue Andre Pascal 75775, Paris, Cedex 16, France, Nov 2010. www.oecd.org/dataoecd/8/38/46961598.pdf [Retrieved 17.Aug.2012].

[25] John Olive and Katie Makar. *Mathematical Knowledge and Practices Resulting from Access to Digital Technologies*, chapter 8. Volume 13 of Hoyles and Lagrange [14], 2010.

[26] Seymour Papert. An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1):95–123, 1996.

[27] Ivan Petrović, Zoltán Kovács, Simon Weitzhofer, Markus Hohenwarter, and Predrag Janičić. Extending geogebra with automated theorem proving by using opengeoprover. [1]. to appear.

[28] Pedro Quaresma and Ralph-Johan Back, editors. Proceedings First Workshop on *CTP Components for Educational Software (THedu'11)*, volume 79 of *Electronic Proceedings in Theoretical Computer Science*. Open Publishing Association, 2012.

[29] Michel Rocard and al. Science education NOW: A renewed pedagogy for the future of europe. Technical report, European Communities, Directorate-General for Research, http://ec.europroa.eu/research/science-society, 2007.

[30] T. Rojano and R. Sutherland. Arithmetic world — algebra world. In Chick et al. [6], pages 515 – 522.

[31] R.Prank, M.Issakova, D.Lepp, E.T onisson, and V.Vaiksaar. T-algebra - interactive learning environment for expression manipulation. In *7th International Conference on Technology in Mathematics Teaching*, volume 1, pages 26–29, Bristol, UK, July 2005.

[32] Bertrand Russell. *Introduction to Mathematical Philosophy*. George Allen and Unwin, London, 1919. http://philosophyfaculty.ucsd.edu/faculty/ctolley/texts/russell.html.

[33] A. Sacristán, N. Calder, T. Rojano, M. Santos-Trigo, A. Friedlander, and H. Meissner. *The Influence and Shaping of Digital Technologies on the Learning – and Learning Trajectories – of Mathematical Concepts*, chapter 9, pages 179 – 226. Volume 13 of Hoyles and Lagrange [14], 2010.

[34] A.I. Sacristán and R. Noss. Computational construction as a means to coordinate representations of infinity. *Int.J. of Computers for Mathematical Learning*, 13(1):47 – 70, 2008.

[35] Chris Sangwin, Claire Cazes, Arthur Lee, and Ka Lok Wong. *Micro-level Automatic assessment Supported by Digital Technologies*, chapter 10, pages 227–250. Volume 13 of Hoyles and Lagrange [14], 2010.

[36] David Tall, editor. *Advanced mathematical thinking*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.

[37] H.-G. Weigand. Fünf Thesen zum Einsatz digitaler Technologien im zukünftigen Mathematikunterricht. In *Mathematikunterricht im Kontext von Realität, Kultur und Lehrerprofessionalität*, Spektrum, pages 315–324. Springer, Wiesbaden, 2012.

[38] Makarius Wenzel. Isabelle/jEdit a prover ide within the PIDE framework. In J. Jeuring et al., editors, *Conference on Intelligent Computer Mathematics (CICM 2012)*, number 7362 in LNAI. Springer, 2012.

[39] Makarius Wenzel and Burkhart Wolff. Isabelle/pide as platform for educational tools. In Quaresma and Back [28], pages 143–153.

[40] W. Windsteiger. Theorema 2.0: A Graphical User Interface for a Mathematical Assistant System. In Cezary Kaliszyk and Christoph Lueth, editors, *Proceedings of UITP 2012 (part of CICM 2012)*, pages 1–8, 2012.