Application for an FWF-Grant

\textbf{ISAC} —

a Hierarchy of Problemtypes for Applied Mathematics

Project Leader
Mag.rer.nat Dr.techn Walther A. Neuper

Technische Universität Graz
Institut für Software Technologie
Inffeldgasse 16b, A-8010 Graz, Austria
Phone: (+43) - 316 - 873 - 5711
Fax: (+43) - 316 - 873 - 5706
neuper@ist.tugraz.at

January 29, 2003
Contents

1 Scientific aspects 3
   1.1 Status of research and related work 3
   1.2 Scientific innovations addressed by ISAC 8
   1.3 Impact on the scientific progress 12
   1.4 Methodology 12
   1.5 Work plan and timetable 14
   1.6 Cooperations 20

2 Human resources 22

3 Further implications by ISAC 23
   3.1 Implications for other scientific disciplines 23
   3.2 Implications beyond science 23
1 Scientific aspects

ISAC is an acronym for ‘use ISA belle knowledge for Computations in applied mathematics’. The theorem prover Isabelle [Pau89] has a large body of mathematics formalized 1. This knowledge is being used and extended by other kinds of knowledge which are the concern of the proposal at hand. In the given context ‘applied mathematics’ means, that some knowledge is already given and being applied in order to solve problems in engineering and science, in particular for educational purposes.

1.1 Status of research and related work

Combining concepts and technologies of several disciplines is the concern of very active R&D efforts which evolved during the recent years. The issue is to develop integrated tools for all aspects of doing mathematics: tools for inventing – formalizing – exploring – proving – managing – applying – publishing mathematics. (The development of integrated tools for all phases in the software development cycle is another line of efforts, gradually joining computer mathematics).

These R&D efforts raised many workshops at related international conferences, and even more independent events, for instance ‘Frontiers of Combining Systems’2, ‘Calculemus Workshop’3 and, MKM20001 (First International Workshop on Mathematical Knowledge Management), recently launched in Austria4.

The reason for efforts combining systems is, apparently, that the concepts and technologies necessary for the development of integrated math-tools are emerging from different disciplines, and their integration is still at the beginning; relevant for ISAC are the following.

Computer Algebra systems are indispensable tools in scientific and engineering work for more than a decade. The calculational power of these

1http://isabelle.in.tum.de/library/HOL/index.html
2http://www.mrg.dist.unige.it/conferences/frocos2002/
3http://www.mathweb.org/calculemus/
4http://www.risc.uni-linz.ac.at/institute/conferences/MKM2001/
systems contrasts with their weakness in logic. The algebra systems even lack a type system providing efficient means for the most elementary checks; important exceptions are Axiom [Dav92] and Aldor [PT00] already employing type systems.

As logic is the fundament of mathematics, several efforts try to overcome the weakness in logic by different ways: Some combine algebra systems with theorem provers (see below), e.g. [AGSM] or [DGKM]; the Theorema project [TVM98] covers the algebra system Mathematica [Wol96] with a logical framework of its own. Another kind of combination has been done by [DS97].

A problematic feature (problematic at least w.r.t. the aims of ISAC) of algebra systems is, that these systems hide their steps towards a result, thus prohibiting the opportunity to check and to control them – and the opportunity to learn from them (as being tried in ISAC).

Theorem proving is quite the opposite to computer algebra w.r.t. two points: (1) computer theorem provers implement a reliably mechanized logical basis, however, they are weak in computation. Thus this weakness also has been tackled, but the other way round by combining them with algebra systems, e.g. [BHC95] or [HT98].

The combination of the strengths of both concepts, of theorem proving and computer algebra within one system, is one of the advantages of Theorema [Buc96b]. Another example is the IMPS prover [FGT93], which is the basis of a very general approach for combining proving and calculating (together with other math activities) started recently [Far01].

(2) Interactive theorem provers work in stepwise interaction with the user, where each step is justified by some theorem. The ingenious steps have to be supplied by the user, i.e. the user is in control on how the result is being achieved. This interaction, however, needs an expert user (this can also be seen with ’Proof General’ [Asp00], a very interesting approach to a generic user interface adaptable to various provers).

This also means, that the ingenious steps cannot be done automatically, and thus there is no possibility for learning by watching the system at work (a possibility, however, for the calculational steps of an algebra system, which
can be done automatically).

**Formal methods** as a discipline of software engineering [FJL97] developed a wealth of interesting models in various application domains, e.g. on railways [BGH+97] or on safety guards (where for the latter the application software could be *automatically* generated from the specification [Dro00]). Integration of specifying and proving (or model checking) is natural, not yet the integration with major parts of computer mathematics.

A problematic (w.r.t the aims of ISAC) point is, that there are not yet abstractions of specifications to some notion of *type* of specification (problem) — recently the issue of re-usability of specifications has been raised in informal discussions [Bjo01], apparently a first step in the direction of abstracting several specifications or parts of them to some kind of type which can be reused in other applications.

This kind of abstraction, however, is indispensable for the specification of mathematic problems: problem-types are common sense in mathematics, for instance the 'types of equations'.

**WWW-standards:** Although the World Wide Web is considered the *medium* of our technology-oriented civilization, it is surprisingly weak in handling math. Standard browsers just become capable of the mere graphical representation of math symbols, but handling math structures (e.g. proofs on various levels of detail) is beyond their scope, not to speak of inter-operability of math services.

All these issues are addressed by the MoWGLI-project [ABG+02]. The goal of the project is to overcome these limitations, passing from a machine-readable to a machine-understandable representation of the information, and developing the technological infrastructure for its exploitation. MoWGLI builds on previous standards for the management and publishing of mathematical documents (MathML [Mat01], OpenMath [Ope99], OMDoc [Koh01]), integrating them with different XML technologies (XSLT [W3C99b], RDF [W3C99a], etc). 5

5Cited from MoWGLIs home-page http://www.mowgli.cs.unibo.it
Integrated tools — educational systems, both are closely related within the actual developments: All projects pursuing integrated tools, and also the most relevant mentioned below, explicitly discuss their impact on education. As ZS4C is planned as an educational tool (in the actual phase, with the aim to proceed towards, or at least contribute to, usage in engineering later on), we include relevant projects on educational math software here.

The TH∀OREM∀ - project [TVM98] ’aims at extending current computer algebra systems by facilities for supporting mathematical proving. The present early-prototype version of the Theorema software system is implemented in Mathematica 3.0. The system consists of a general higher-order predicate logic prover [BJ97] and a collection of special provers [Buc96a] [BMT97] [BV97] [Mat97] that call each other depending on the particular proof situations [TVM98]. The individual provers imitate the proof style of human mathematicians and produce human-readable proofs in natural language presented in nested cells [Buc97]. The special provers are intimately connected with the functors that build up the various mathematical domains [Tom97].

Theorema is to be considered a part of a great vision (e.g. [Buc01a], [Buc00a]) on the future of mathematics and math education; it already brings to the point several claims of the QED-manifesto [Ano94]. The presently available system [Buc01b] is dedicated to a very elaborated way of generating knowledge within incremental ’exploration rounds’ [Buc00b]. In this context, user-guidance does not make sense.

IMPS is an ’interactive mathematical proof system’ [FGT93] which provides for concepts and mechanisms for a ’formal framework for managing mathematics (FFMM)’ [Far01]. Doing mathematics is seen as ’creating, exploring and connecting mathematical models’.

The envisaged framework is intended to support the whole process of doing mathematics and the handling of the knowledge produced by the process. In particular, the ’derivation graph’ and the operations on it [Far01]

\footnote{Cited from the Theorema Home-page at http://www.theorema.com}
lay a flexible foundation for interactive construction and re-construction for variants.

There is also a proposal for the development of an interactive mathematics laboratory for math education [Far00] in America. The realization of both, the framework for mathematicians and the system for education, however, has not yet started.

And it is not clear from the references, whether the systems envisaged are appropriate for support in solving problems of engineering and science, i.e. appropriate for applying mathematics.

**ActiveMath** ‘is a web-based learning environment that dynamically generates interactive mathematical courses adapted to the student’s goals, preferences, capabilities and knowledge’ [LMU]. For this purpose it employs a well designed user model and techniques from artificial intelligence, e.g. an expert system processing pedagogical rules.

The resulting features are user-adapted content, sequencing of the material, user-adapted suggestions for learning, support of active and explorative learning, support of teachers by information about their students, etc.

Due to a general knowledge representation, OMDoc [Koh01], course material is reusable for different courses.

ActiveMath integrates several mathematical services [LMP+], for instance the proof planner ΩMEGA [BCF+97] and the algebra system Maple, others are planned. However, there seem to be no logical framework sufficient for consistent integration of proving and solving.

**MathXpert** is a commercially available educational algebra system [Bee84] with carefully designed logical foundation [Bee85]. The calculations are done in steps which can be controlled by the user, and the system guides the user towards a solution.

In the applicants opinion this is the only system which really supports the current mainstream of teaching mathematics in higher education, which is drill and practice of algorithms to a great extent.

However, MathXpert does not provide means to specify problems, and thus does not support solving problems by breaking them down to subprob-
lems.

And secondly, its knowledge is hard-coded like in other computer algebra systems; thus this knowledge cannot be inspected, and cannot be adapted or extended by a course designer or a teacher.

Summarizing the status of research and development we see: concepts and techniques are available from several disciplines to construct mathematics systems. But all systems discussed above are missing some features justified as indispensable in the subsequent section. (And the references reviewed do not present concrete plans aiming at these features, as well.)

1.2 Scientific innovations addressed by $\mathcal{ISAC}$

At the intersection of computer mathematics and software technology scientific development goes towards clarification of formal languages and the respective interrelations based on formal logic, and towards mechanic interpretation of increasingly abstract languages.

For instance, the theorem prover Isabelle clearly separates three language layers: (1) the implementation language (SML [MTHM97]), (2) the object language (math formulas; $\mathcal{ISAC}$ uses Isabelle/HOL, high order logic) and (3) the deductive knowledge (theories, which even define the logic, the formulas are to be interpreted with).

$\mathcal{ISAC}$ adds another language layer of application-oriented knowledge which does not yet exist in math systems: types of problems, based on the grounds of 'formal methods' [FJL97] in software technology. These 'problem-types' (e.g. types of equations) $\mathcal{ISAC}$ can be 'matched' interactively with a formalized 'problem' (with specified input items, output items, pre- and post-condition). $\mathcal{ISAC}$ arranges the problem-types in a hierarchy which can be mechanically searched for automated problem refinement [Neu01a].

Here is an example for the textual description of a problem:

*Given a circle with radius $r$, inscribe a rectangle with length $u$ and width $v$. Determine $u$ and $v$ such that the rectangles area $A$ is a maximum (see Fig.1).*
And this is the complete formal specification of the problem, capturing pre-condition (where) and post-condition (with) [DH96]:

\[
\begin{align*}
given : & \quad r \\
where : & \quad 0 \leq r \\
find : & \quad \text{Maximum } A \\
\quad \text{AdditionalValues } [u, v] \\
with : & \quad A = 2uv - u^2 \land (\frac{u}{2})^2 + (\frac{v}{2})^2 = r^2 \land \\
& \quad \forall A' \ u' \ v'. \ A' = 2u'v' - (u')^2 \land (\frac{u'}{2})^2 + (\frac{v'}{2})^2 = r^2 \implies A' \leq A \\
relate : & \quad [A = 2uv - u^2, (\frac{u}{2})^2 + (\frac{v}{2})^2 = r^2]
\end{align*}
\]

The boxes above mark the items meant for input by the user.

In order to achieve the advantages (see 1.3) of this language layer, ZS4C has to address the following issues (explained w.r.t. the example above):

1. The system has to handle variants of specifications (the user might specify the relations \(u^2 = r \sin \alpha, v^2 = r \cos \alpha\) as well).
2. The problem needs to be arranged in a hierarchy together with other 'problems of calculus' such that the user finds it by systematic interactive search and such that automated problem refinement works.
3. Identify the 'standard problems' this problem should be broken down (derivative, equation solving, . . .) in the solving process.
4. Handle input items (in \given above) required for subproblems, but not for the above root-problem (e.g. the bound variable for the derivative, \(u, v\) or \(\alpha\)).

Upon completion of the formal specification (as in the example above) mechanical solving can start. The description of the solving process concerns \emph{algorithmic knowledge}. For this knowledge ZS4C uses a preliminary language [Neu01b] to be formally underpinned in a subsequent phase.\footnote{ZS4C employs a logic of computation in the line of Gries [Gri81], but does not follow Gries into natural deduction. ZS4C’s proof format looks ‘more natural’, and the ‘structured calculational proofs’ introduced by [BW99] might provide the missing link to the sound basis of natural deduction.}
Re-engineering the basic functions of algebra systems is one of the main activities [Neu02c] within the phase covered by the proposal at hand: hard-coded knowledge has to be extracted into separated language layers (separated from the interpretation language) which is (1) expressed close to traditional math notation (see the example above, Isabelle/HOL) (2) mechanically interpretable (for matching and automated refinement [Neu01a] within \( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C} \) and which is (3) stored in a standard format (under development by MoWGLI \(^8\) – see 1.4).

The basic functions of algebra systems are: equation solving and simplification within the elementary domains. Equation solving will employ \( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C}\)'s mechanism for automated refinement with particular good prospects: in algebra systems \(^9\) the equation is matched with some pattern (a ‘problem type’ in terms of \( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C} \)), eventually transformed into some normalform, matched again (this may go on several times) and transformed until the desired disjunction with the bound variable(s) made explicit (or some other predicate on the solvability) has been calculated.

Simplification is frequently done by rewriting. For rewriting only theorems proven in Isabelle are used. Each rewrite is a 'step' in the calculation. In \( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C} \) stepwise calculation enables the user to modify each step, either by input of the theorem, or by input of a formula (where \( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C} \) constructs a sequence of rewrites from the actual formula to the input formula). Rewriting is intuitively comprehensible (like matching of problems), thus simplification should be presented as stepwise rewriting to the naive user, even if the normalform cannot be obtained by rewriting ([Kar02] shows how this can be done by 'reverse rewriting' for rationals).

A challenge is to handle user-input in absence of terminating and confluent simplifiers, another one is how to suppress the many uninteresting rewrites in AC-rewriting.

\( \mathcal{I}\mathcal{S}\mathcal{A}\mathcal{C} \) is a maximally transparent system as a consequence of the two previous points: it is transparent w.r.t. the underlying knowledge, and the knowledge-interpreter works transparently as well (with rewriting and match-

\(^8\)http://www.mowgli.cs.unibo.it
\(^9\)Remarkably, we could not find any detailed literature on these technicalities of solvers!
ing as the atomic steps, which the user can interactively guide and modify).

This is the point, why ISAC can be used for education\textsuperscript{10}: such a system allows the users to ask ‘what is going on’ in a calculation and go into the details down to the atomic steps of the current algorithm, and down to the underlying definitions, say $t$ divides $n$ or even the definitions of $+$ or $=$ or $\land$.

The system thus answers the ‘what’ question mechanically w.r.t. a particular problem, and if the users follow the interrelations within the highly structured knowledge base they even get answers to ‘how comes’, ‘what for’ and ‘why’ questions. In general such questions can \textit{not} be foreseen and prepared by an author in advance to the actual learning situations, thus the system has to answer them dynamically.

\textbf{The hypotheses} underlying the R&D phase covered by the proposal at hand, and to be proven within this phase are:

\textit{?!} The knowledge hard-coded in algebra systems can be extracted into separated language layers (of theories, problem-types, algorithms) which are human readable.

\textit{?!} The mechanic interpretation of the (human-readable) knowledge can be implemented efficiently enough to serve interaction for educational purposes.

\textit{?!} The knowledge covering the basic functionality of algebra systems (i.e. simplifying and equation solving within the range of high-school math) can be implemented within this phase (i.e. with justifiable effort).

\textit{?!} This knowledge is a sufficient base for adding (the majority of ?) problems of applied math in engineering.

\textit{?!} ISAC is innovative enough to be a prosperous testbed for other innovative projects, MoWGLI and Dinopolis (see p.13 below).

\textsuperscript{10}Nevertheless, the actual phase of ISAC’s development is too early to involve experts of didactics or learning theory: The construction of a transparent math engine is the concern of computer math and software technology, and not a didactic concern.
1.3 Impact on the scientific progress

While primarily geared towards education, ISAC is a part of the presently vivid development in the intersection of computer math and software technology, and thus several of ISAC’s aspects are valuable on their own within the actual scientific progress:

- **Improved interactive control** in the specification phase, which is important in problem areas without general solutions (e.g. partial differential equations) — and which is an occasion for learning.

- **Extensibility** of the underlying knowledge allows to implement any topic of (applied) math, including new basic definitions, axioms, notations etc.

- **Separation of concerns**: what is one chunk of code within algebra systems is separated into data and interpreter, computing and proving (pre-/post-condition!) etc. – a prerequisite for formal verification in the future.

- **Standardization** in the representation of math knowledge – in a line with the impact of the MoWGLI-project[ABG+02], whose technologies ISAC applies as one of the first demonstration projects.

- **Demonstration** of the usefulness of Dinopolis [Sch02] as a middleware for the distributed components of ISAC (see 1.4).

1.4 Methodology

**Adopt existing concepts, tools and standards**, which is obvious at the high state of the art. ISAC builds upon the following components:

- **SML** as implementation language with prospects to re-establish it as a meta-language for ISAC in the future.

- **Isabelle** as logical framework, and as open source product exploited w.r.t. knowledge management (use_thy), matching, parsing, etc.
- MoWGLI-standards delivered just in time to give ISAC the chance to act as a demonstration project. The hierarchy of problem-types, the algorithmic knowledge (the deductive knowledge is left to native Isabelle efforts) and the examples will be presented by standard formats; thus, for manipulating and displaying all these data standard tools will be available for immediate use or for adaption.

- Dinopolis [Sch02] is a middle-ware under construction at TU Graz during 2003/04 (the first release of the interface descriptions will come early in 2003); Dinopolis generalizes and thus simplifies the management of ISAC’s distributed objects (interpreter, knowledge, front-end) w.r.t. the access-rights of different kinds of users (students, teachers, admins) 11.

**Build upon own preliminary work** already performed at the Institute for Software Technology, TU Graz:

**ISACs knowledge interpreter** is developed to such an extent that the implementation of the knowledge can start. Representing more than 2 man-years of work, it consists of a problem handler performing matching and refinement of problems, the method interpreter, interpreting the algorithmic knowledge, the rewriter, using theorems proven in Isabelle, and the proof-state-handler building the proof-tree.

Some of the features have been demonstrated by a prototype [Fin00].

**Ongoing work at TU Graz** within the ISAC-project is, due to the delay of funding, restricted to four diploma students: two on mathematics ([Kar02] taught ISAC 'calculate rationals' and demonstrates 'reverse rewriting', and [Lan03] shows the principal functionality of automated problem refinement on elementary equations), and two on software technology doing the architectural- and software-design [GKN02a, GKN02b] 12. This work already takes the interfaces to MoWGLI and Dinopolis into account.

---

11 [www.dinopolis.com](http://www.dinopolis.com)
12 The respective documents are updated on [www.ist.tugraz.at/projects/isac/status.html](http://www.ist.tugraz.at/projects/isac/status.html) on a regular base
The S.E.A.L. project\(^\text{13}\) has been conducted under the title 'computer assisted, paradigm oriented learning of programming methods' [Aus00] and was funded by Jubiläumsfonds grant No.7004. Within this project a software component for animated operations on formulas has been developed. The proposal at hand plans the application of this feature to animation of matching.

**Integrate diploma theses:** Due to the high modularization in the design of ISAC [GKN02a] the work can be partitioned into subtasks, many of which are appropriate for diploma theses (i.e. balanced in work on the literature, design and implementation). The current phase of development does not envisage 'waterproof' software. However, the software quality should be higher than one can expect from students work. In order to ensure the necessary quality, and for integration of the sub-projects, two full-time jobs (see ??) are solicited.

**Prepare cooperation with didactics:** The work planned within the proposal at hand is the prerequisite for the involvement of experts in didactics and learning theory. This involvement is essential for an educational software like ISAC. Thus cooperation with ACDCA - Austrian Center for Didactics of Computer Algebra\(^\text{14}\) will start already within the current phase. Some preparations done from the ISAC-side are [Neu01c, Neu02b, Neu02a].

### 1.5 Work plan and timetable

The work shall proceed in two groups of activities, which can be done in parallel: R&D for the contents of the knowledge-base and R&D for the tools accessing the knowledge base.

**R&D for the contents of the problem hierarchy.** The contents shall cover the 'basic functions of algebra systems' (see 1.2); this is a laborious task which is divided into several sub-projects to be done within diploma theses.

\(^{13}\)S.E.A.L. is an acronym for *Surf the Web and Learn.*

\(^{14}\)http://www.acdca.ac.at
These theses require preparatory and finishing work (the labels Cont\textsubscript{N} refer to Tab.1 on p.20):

**Cont1: select methods of symbolic computation.** One part of this task requires to study well-documented concepts from the literature to be used for the theses. A special problem is the implementation of floating-point numbers within the logical framework of Isabelle/HOL; a logically consistent implementation requires cooperation with Isabelle’s developer team (task Isab below).

**Cont2: combine the parts of knowledge.** For instance the integration of the simplifiers (e.g. for rationals, for radicals) is a non-trivial combination problem [BN98]. This task includes implementing the major and laborious parts not covered by the sub-projects The\textit{C}1 - The\textit{C}8 below and preparing \textit{Appl} (see p.16 below).

**Coordinate the sub-projects** is a task covering the duration of the theses; thus this task is not mentioned explicitly in Tab.1. In particular, this task includes supervision of the practical part of the diploma-theses, and the related part in the written theses.

**Cont3: extend the knowledge interpreter** due to extensions of new and not foreseen functionality of the system. Several functions require speed-up for the multi-user system.

**Cont4: integrate the code of the sub-projects** into the knowledge base. This task includes to make the code perfect, which generally cannot be done (and should not be a goal) in diploma theses.

**Cont5: documentation** will be done, which collects and restructures the documentation of the sub-projects, and which leads to a reference manual on the knowledge. A suite of demo examples for the WWW-presentation will be compiled as well.
Appl: An engineering application, i.e. special knowledge for an advanced course should be based on the general knowledge provided within this phase. The selection of the topic for this application will be done in cooperation with a lecturer at a polytechnic university. The design of the respective knowledge will require special domain knowledge contributed by the lecturer.

Topics for diploma theses (TheC1-8 in Tab.1 on p.20). Particular reasons for including diploma theses into the projects tasks are (1) the special education of the mathematics students at TU Graz and (2) the expectation, that some of these students will be available for later extensions of the knowledge base.

TheC1: factorization in multivariate polynomial rings, plus application for equation solving.

TheC2: symbolic computation on radicals is indispensable for most applications. The implementation includes an efficient canonical simplifier, and a novel application of 'reverse rewriting'.

TheC3: term-orders for AC-rewriting provide for normal forms in associative and commutative domains (i.e. all the domains of elementary mathematics); the normal forms, in turn, are prerequisites for matching formulas with patterns in problem types. A tool-set of (recursive path) orders is required for this purpose and various others.

TheC4: algebraic and transcendental equations. This topic requires the design of the hierarchy (integrated into the existing hierarchy of polynomial and rational equations), i.e. appropriate normal forms, patterns and predicates for the preconditions; it includes the implementation of the respective problem types together with methods solving them.

15The students of ‘Techno-Mathematik’ are well-trained in SML, the special program language of $\text{ISAC}^{15}$’s knowledge interpreter and the ‘glue-code’ for the object language of problem types.
TheC5: floating point numbers will be introduced to Isabelle as a new datatype; this topic includes the numeric computation features, and addresses the approximative nature of floating point numbers (when they represent reals) following hints given by [Har97].

TheC6: symbolic computation on complex numbers will be treated as operating in algebraic extension fields, implemented in several special cases. How results from the above sub-projects can be used herein, will be clarified in Cont1.

TheC7: derivative and integral, preferably Riemann Integral. This sub-project should follow [Bee95] and build upon experimental Isabelle theories on nonstandard analysis.

TheC8: problem-types for calculus provide for a knowledge widely used application domain in engineering. This knowledge will comprise the stuff taught at (technical) high-schools, and if possible, the application selected in Appl..

R&D for tools accessing the problem hierarchy. Due to the adoption of standards and tools of MoWGLI and services of Dinopolis this part does not involve original research, but intimate contact with the respective R&D teams. Such a contact cannot be done by diploma students (at least not without strict supervision), for this a highly qualified full-time developer is required. Also some educational effort for the new technologies has to be taken into account.

Tool1: establish contacts to MoWGLI will require a visit at University of Bologna (Italy), Department of Computer Science, and provisionally at Institut National de Recherche en Informatique et Automatique (INRIA) Rocquencourt. Also the finalization of $\exists^4$Cs rôle as a demonstration project for MoWGLI has to be done.
**Tool2:** knowledge transfer to the ISAC-team as a follow-up activity to Tool1.

**Tool3:** implement the tools not covered by the sub-projects TheT1-TheT6. These comprise auxiliary and system functions not demanding enough for a thesis, but essentially for the usability of the system. There will be indispensable authoring tools to be implemented, and a long list of desirable functionality to be implemented if time is left: display dynamic relations between the 3 axes, dynamic display of the hierarchy 'is subproblem' (this hierarchy is different from the one discussed so far!), statistics on the problem types (frequency of a problem type $S$ being a subproblem while solving type $T$), animation of matching, etc.

**Tool4:** integrate the code of the sub-projects. This task is analogous to Cont4, as well as the task of coordinating the sub-projects - and both of them done in cooperation with the project leader.

**Tool5:** documentation and installation-routines. In addition to the documentation accompanying each (sub)task, a user-manual will be compiled by this task, and a reference manual together with Cont5. Installation routines for the many system components are required, too.

**Topics for diploma theses** ("TheTN" in Tab.1 on p.20) are the following. They build on the extensibility and customizability of MoWGLI and Dinopolis.

**TheT1:** XML - SML transformation for ISACs knowledge (theories, problem-types, algorithms) and output (proof-tree) from SML data-structures into the 'MoWGLI standard-format' for math knowledge and vice versa. Low-level DTDs for these data (with attention to the dependency-graphs).

**TheT2:** data transformation services (by XSLT-stylesheets) from the low-level format to the 'MoWGLI standard-format' for math content,
and from this further to representation format to be displayed on standard browsers.

**TheT3: standards for math-data and services** from MoWGLI need to be adopted and adapted: Editors for problem-types, algorithms (manually add meta-data), examples (layout) and the API for searching math content (indexing, interlinking).

**TheT4: MathML rendering and input engine.** This task will isolate the respective code from Pcoq, the open source GUI of the theorem prove COQ; indispensable for input formulas at ISAC's worksheet.

**TheT5: admintools for Dinopolis services,** i.e. tools for setting and editing Dinopolis parameters for the communication between multi-user front-end and central knowledge interpreter, for the access rights of students, knowledge authors and course admins.

**TheT6: Dinopolis wrapper for knowledge interpreter** implementing the SML-core as a Dinopolis object with the desired properties (persistence, adaption to the workload by user-requests, separate processes for math-authors, etc.)
**Timetable**  The phases of R&D and evaluation follow the timetable in Tab.1 on p.20.

<table>
<thead>
<tr>
<th></th>
<th>1st year</th>
<th></th>
<th>2nd year</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Cont1</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Cont2</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Cont3</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Cont4</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Cont5</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheC1,2</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheC3,4</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheC5,6</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheC7,8</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Isab</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Appl</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Tool1</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Tool2</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Tool3</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Tool4</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>Tool5</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheT1,2</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheT3,4</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>TheT5,6</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>=</td>
</tr>
</tbody>
</table>

The duration of the diploma theses is estimated with one year, actually it may vary within the usual range.

**1.6 Cooperations**

with Austrian institutions are the following
Institute of Algebra and Computational Mathematics, Vienna University of Technology. This is the only institute (in Austria, besides RISC Linz) issued at computer math.Ao.Univ.Prof.Dr.Günther Karigl will supervise, if available, some of the diploma students working on *TheC*∗. He gives the lectures on 'mathematics, logics and system theory' and 'mathematics for computer scientists' (among others) at the institute.

IICM - Institute for Information Processing and Computer Supported New Media, Graz University of Technology. The institute is carrying out research in areas related to different hypermedia-based educational, communicational and cooperational aspects. Klaus Schmaranz, leader of the Dinopolis-project, already contributed massively to *ISAC*’s software architecture.

ACDCA - Austrian Center for Didactics of Computer Algebra is engaged in serving exchange at the interfaces between science, software production and educational practice, and in internationally representing Austrian science and expertise in the field. A meeting working out the details of the cooperation will be held in February 2002.

International cooperations concern European leading research, which *ISAC* takes advantage from.

The ‘Theorem Proving Group’ at Informatik IV - Software and Systems Engineering, Munich University of Technology. This group is one of the two sites developing Isabelle, the interactive theorem prover *ISAC* is based on. We gave a presentation of *ISAC* at the , and Tobias Nipkow, promised help by his group (task *Isab*).

MoWGLI - IST-programme project IST-2001-33562: The details of the cooperation (w.r.t. serving as a testbed for the standards and

---

16citation from http://www.acdca.ac.at: Der Verein will einen Beitrag zur Lösung der Schnittstellenproblematik zwischen Wissenschaft, Softwareproduktion und Unterrichtspraxis im Bereich der Computeralgebra-Systeme leisten und die österreichische Forschung und Erfahrung auf diesem Gebiet international vertreten.
tools, eventually contributing by *TheT4*) will be discussed in a meeting at MKM (Mathematical Knowledge Management, February 16th - 18th, 2003) and the subsequent MoWGLI-meeting at Bologna.

2 Human resources

Walther Neuper, the applicant, is proposed to be the project leader, too. He has been a teacher in various types of schools, was one of the pioneers introducing computers into Austrian high-schools, and he was a consultant in two UNESCO projects (Educational software development, Educational Assistance Programme for Croatia and Slovenia, the latter in a leading position). In a second series of studies in computer math (Bruno Buchberger) and software technology (Peter Lucas) he designed _MSAC_ and developed the knowledge interpreter based on Isabelle.

On the impact on the career, as requested by FWF: Due to the (informal) position in between two very disjoint realms (academic research and high-school education) nothing can be indicated. This position has already been advantageous for the interdisciplinary work (computer math – software technology) and will hopefully continue to do so in extending to didactics and, above all, practical education in math classes.

Andreas Griesmayer is proposed for the post as a research assistant. He has been a successful tutor in the institutes software construction exercises, and thus is excellent in SML. In parallel to his academic education he has already gained experience in a commercial hyper-media development project, thus he is excellent in XML and HTTP, too. He already contributed substantially to the development if _MSAC_ by cooperating in the software-design [GKN02a, GKN02b, GKN02c].

The impact of the proposed work (tasks Tool1 - Tool5) on his career will be (1) his diploma thesis, and (2) leading edge expertise in XML-representation of math knowledge.
3 Further implications by ISAC

3.1 Implications for other scientific disciplines

ISAC is designed for educational use (at least in the first go), thus the implications for the following disciplines are those strived for, primarily.

**Didactics of mathematics** will find a major part of applied math (e.g. high-school math) in a machine readable and searchable representation in the near future. This will allow to tackle research questions (e.g. which problem-types are required most for solving a particular group of problems, which steps frequently cause errors at a particular curriculum, etc.) in a completely new way.

**Learning theory,** in particular learning theory biased towards constructivism [Cop92], will find data at a high level of abstraction and structure within the logfile of ISAC’s dialog module. Mathematics is the only subject, where software in principle is a model of the subject (i.e. math) itself (see [Neu]).

3.2 Implications beyond science

Developments in the realm of education can hardly be foreseen, in particular if they are related to technology.

We expect advantages of a 'transparent software' in the practice of math education (the applicant taught so-called notebook-classes and observed precarious consequences of permanently using inadequate math software). ISAC will be a tool for learning applied mathematics, where the student can passively watch the knowledge-interpreter stepwise solving a problem, where students can go into the details of the steps and view the underlying knowledge at their own pace any time, and where active students get their own steps checked by the system, as well.

eLearning will adopt interactive software with user-guidance like ISAC; in particular universities of technology (Fachhochschulen) announced actual demand already.
Last not least we can expect, that ISAC as an authoring system will redraw the boarder-lines between (computer) mathematics and didactics (of math), between authors of text books, course designers and teachers — opening opportunities to establish new cooperations.
References


