Chapter 1

List of terms used in the $\mathcal{ISAC}$-project

Active formula (Aktive Formel) is the unique formula marked on the →worksheet, where the next step of calculation will be performed. If another item on the worksheet is marked, the formula closest (.. to be specified) to the marked item is the active formula. If the calculation is finished, the result of the calculation is the active formula. Also →context position.

Browser (Browser): There are browsers for →theories, →problems, →methods and →examples.

Browser dialog (Browser-Dialog) is the part of the →dialog guide which is concerned with the access to the →KE-store.

Browser-Window (Browser-Window) is the presentation of data generated by the →browser, which also handles the httprequests generated by the browser window. (The browser-window is that what usually is called a 'browser').

Calc-head (Rechnungskopf) is the first (highly structured) element in a →calculation (i.e. a 'root-problem') and in a subproblem. It consists of a →headline, a →model and of a →specification. On the →worksheet it is represented only by the headline.

Calc-state (Rechenzustand) is given by an internal calc-tree (partially represented on the →worksheet) and a →active formula.

Calculation (Rechengang) leads from the →description of an →example via the →modeling phase, the →specification phase and the →solving phase to the result.

Calculator (Rechner) is that part of the →SML-kernel which does single steps of calculation without touching the →proofstate.

Course admin (Kurs-Administrator) is a person administering the use of $\mathcal{ISAC}$ for learning within a group of →learners.

Course designer (Kurs-Designer) edits the →example collection which can be solved by a given →math knowledge base (edited by a →mathematics author) and/or edits →explanations within the →math knowledge base.

Context (Kontext) is a relation between the →context position in a certain →calculation and a part of the →math knowledge.

Context position (Kontext Position) is a uniquely formula in a →calculation giving one side of a →context.

Decorated knowledge (Erweitertes Mathematik-Wissen) is the →mathemath knowledge(base) plus →explanations. Each element of the math knowledge can have $0 \ldots n$ explanations (usually specific for courses).

Description (Beschreibung) of an →example consists of →formulas, eventually of text and/or a figure. The →modeling phase transforms the description into a →model.

Dialog atom (Dialog-Atom) is a predefined, minimal unit of interaction between the →learner and $\mathcal{ISAC}$. These atoms are symmetrical w.r.t. the two dialog partners.

Dialog author (Dialog-Autor) an expert in learning theory who adapts and extends the →dialog guide.

Dialog guide (Dialog-Komponente) is a component of $\mathcal{ISAC}$ which consists of the →worksheet-dialog and the →browser-dialogs.

Dialog mode (Dialog-Modus) is assembled from →dialog patterns and supports certain learning strategies, e.g. exploratory learning, written examniation etc. Dialog patterns are designed and implementd by a →dialog author.

Dialog pattern (Dialog-Muster) is assembled from →dialog atoms such that it can adapt to certain situations in a dialog, e.g. if the →learner produces many errors. Dialog patterns are designed and implementd by a →dialog author.

Dialog profile (Dialog-Profil) defines certain →dialog modes for examples in the example collection for a certain course. A dialog profile is defined by a →course designer and set/reset for a specified duration during a course by the →course admin.

Example (Beispiel) is a unit to be calculated and solved separated from others. In general, they are prepared by an author in an →example collection. It consists of an explanation (analoguous to →explanation of an element of the math knowledge), a →formalization and a →specification.

Example browser (Beispiels-Browser) is an interactive representation of the →example collection within the →front-end.

Example collection (Beispielsammlung) contains →examples, each of them consisting of formulas, of a hidden →formalization and →specification, and eventuallly of text and a figure.

Example profile (Beispiels-Profil) describes the structure of an→example collection; this structure provides data for the →dialog guide.

Explanation (Erklärung) is an optional addon (text, formulas, figures, movies, links, →examples and any combination of these) to elements of the →math knowledge base.

Formalization (Formalisierung) contains the formulas in a minimal structure necessary for automated generation of a →model of an example. Together with a →specification this information is sufficient for automatically solve the example.

Formula (Formel) consists of variables, constants and functions constants (for logical, algebraic etc. operators); all these parts, however are not yet structured as a (typed) →term.

Guard (Guard) of a →method: prevents the method's script to be applied to an inappropriate problem. The guard has the same structure as a →modelpattern (and thus sometimes is called a 'guardpattern').

Headline (Problem-Kopf) represents a →calc-head on the worksheet; it either looks like *Problem (Reals, [univariate,equations])* or an Algebrasystem function like *solve($x^2 + x + 1 = 0$, x)*.

Interpreter (Interpreter) comprises the modules →math engine and the →calculator.

Isabelle is the name of one of the most successful interactive theorem provers; Isabelle provides the →theories containing the deductive part of ISAC's knowledge base.

Item (Item) of a →model, which can be an input item (in the field 'given'), a precondition (in the field 'where'), an ouput item (in the field 'find') or a relation (in the field 'relate'). 'Given', 'find'and 'relate' may be input by the user, where 'where' is supplied by the system. An item consists of the →item-description and the →item-data.

Item-data (Item-Daten) are the formulas following the →item-description.

Item-description (Item-Beschreibung) is an identifier heading each →item in the fields 'given', 'find' and 'relate'. It indicates the kind of data to be input to the respective item by the users, serves typechecking of the data etc.

Item-status (Item-Status) gives feedback to each item of a →model with *one* of the following kinds of status: correct, true, false, missing, incomplete, superfluous, syntaxerror.

KE-store (KE-Basis) is the →decorated →math knowledge plus the →example collection.

Kernel (SML-Kern) comprises the →interpreter and the →knowledge base, all written in SML.

Knowledge base →mathematics knowledge base

Knowledge browser is one of the →theory brosers, →problem browser, →method browser.

Learner (Lernender) a user of *ISAC*, who uses *ISAC* for learning and exercising, i.e. who calculates →examples by use of the →math knowledge base.

Learner model (Lernprofil) is an abstraction over all interactions of a certain learner with *ISAC* during a course; this abstraction serves to adapt *ISAC*'s behaviour to the personal needs of the learner.

Match (Matchen): the →model of an example (or a subproblem) matches the →modelpattern of a problem, or not. This kind of matching is different from the matching-algorithm of symbolic computation: it checks if all →items are input, and evaluates the predicates in 'where'.

Math engine (Mathematik-Maschine) provides for all functions doing →calculations: for applying →tactics, for input →formulas, for calculating resulting formulas, for proposing the next tactic, and for doing calculations automatically; it maintains a →proofstate for each calculation.

Mathematics author (Mathematik-Autor) an expert in computer mathematics who adapts and extends the →mathematics knowledge base.

Mathematics kernel (Mathematik-Kern) replaced by →math engine; please, dont't use anymore !

Mathematics knowledge base (Mathematische Wissensbasis) is stored in three SML-datastructures, in an acyclic graph of →theories, in a hierarchy of →problems, and in a hierarchy of →methods. It is extensible by →math authors and can be both, read by →learners and interpreted by *ISAC*. Short form is math knowledge. See also →decorated math knowledge.

Method (Methode) contains a →script describing the algorithm for calculating the result, and a guard structured like a →modelpattern in order to inhibit inappropriate application of the script.

Method browser (Methoden-Browser)

Model (Modell) is a part of the →calc-head. It consists of →items (as well as the →modelpattern).

Modelpattern (Modell-Pattern) is the part of a →problem.

Modeling phase (Modellierungs-Phase) is the initial phase in problem solving. In this phase either the system automatically transforms a →formalization of an example into a →model or the user inputs the →items into the model.

Parsing (Parsen) is the process of transforming an 'plain' formula into a typed term. Parsing requires the specification of a →theory containing information about infix position of operators etc.

Problem browser (Problem-Browser)

Problem (Problem) consists of a →modelpattern and some technical elements (→methods solving this problem, rule sets for evaluating the precondition in →matching etc.)

Proofstate (Beweiszustand) replaced by →calc-state; please, don't use anymore !

Rewriting (Rewriting) transforms a formula into a new one by application of a →theorem. ISAC provides conditional as well as ordered rewriting.

Script (Skript) describes the algorithm solving a particular problem; a script contains →tactics, expressions for guiding the flow of evaluation, and eventually subproblems.

Selection-tool (Auswahls-Tool) displays the contents of either the →example collection, or the dependency graph of →theories, or the hierarchy of →problems, or the hierarchy of →methods; and it allows to select a respective item for detailed display.

SML-kernel →kernel

Solving phase (Lösungs-Phase) is the final phase in problem solving, which generates the solution from the →model and the →specification; this phase may comprise all problem solving phases for one or more subproblems.

Specification (Spezifikation) relates a →model (or a →guard) of a calc-head to the →modelpattern (or guradpattern) of the respective →problem (or →method) while determining a →theory, →a problem and a →method.

Specification phase (Spezifikations-Phase) is the second phase in problem solving, which determines the →theory, →the problem and the →method. This phase can be done automatically and hidden from the user, if the →dialog guide decides to do so. Sometimes, if it is clear from the context, this phase also comprises the →modeling phase.

Step ((Rechen-) Schritt) propagates a →calculation and involves both partners once, i.e. the →learner and the →dialog guide. A step is represented by one of the →dialog atoms.

Tactic (Taktik) is applicable or not to the current →formula within the current proofstate, and generates a new formula accordingly.

Term (Term) is an Isabelle term (simple typed lambda calculus) generated from a →formula by →parsing.

Theorem (Theorem) is a predicate proven true by →Isabelle w.r.t. certain preconditions. Theorems are applied by →rewriting.

Theory (Theorie) is the part of the →math knowledge base which defines (function) constants and axioms. Within a theory usually the related →theorems are being proven by →Isabelle and stored.

Theory Browser (Theorie-Browser)

User (Benutzer) of ISAC may be one of the following: →visitor, →learner, →math author, →dialog author, →course designer, or →course admin.

Visitor (Besucher) a user of ISAC, which occasionaly browses an ISAC-site, i.e. the →knowledge base and the →example collection.

Worksheet (Arbeitsblatt) contains the →calculation of an →example eventually leading to a result.

Worksheet dialog (Arbeitsblatt-Dialog) is the part of the →dialog guide which is concerned with the interaction between learner (see chapter **??** on p.**??**) and →math engine.