Developing an Inclusive Approach for Representing Mathematical Formulas

Natalie Karl BSc

M A S T E R A R B E I T

eingereicht am

Fachhochschul-Masterstudiengang

Software Engineering

in Hagenberg

im September 2016

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Hagenberg, September 8, 2016

Natalie Karl BSc

Contents

Acknowledgement

I would like to thank the people of Institute “Integriert Studieren” at JKU Linz for their support during the time of creation of this thesis, especially during the development of the project.

Thank you Dipl. Ing. Bernhard Stöger for not only letting us work in your office but also for giving me an insight in how visually impaired people perceive the world especially when working on a computer and mathematical problems. This thesis was enriched by you in many forms.

A special thank you goes to a.Univ.-Prof. Dr. Klaus Miesenberger for consulting us and giving us the chance to visit the ICCHP, which took place at Linz in 2016.

Last but not least, I want to thank Dr. Walther Neuper for his support and cooperation during the creation of this thesis. Dr. Neuper introduced me to all ISAC related infrastructure and the respective codebase. I am very grateful for everything he taught me about ISAC, Isabelle and the given hints on my written thesis.

Abstract

There are about 3,000 legally blind people in a population of 8.4 million people in Austria. The educational system is not fully adapted to this relatively small number of visually impaired people. Especially mathematics is an important subject for our daily life but represent one of the bigger challenges for visually impaired people. Sighted people often use spatial information and symbols, which cannot be displayed on assistive technology software and hardware such as screen reader and Braille displays. Unfortunately there is no general solution in literature or existing prototypes at this point. As there have already been many prototypes in this area, the existing open source project *ISAC* was extended. *ISAC* comprises a mathematical authoring tool, which should be changed to provide an inclusive way (visually impaired together with sighted people) of working with the system.

Formulas are an essential part to describe mathematics knowledge. They are the smallest non-atomic unit which enable you to perform a calculation and check it for correctness. Visually impaired people cannot access two dimensional data, such as often used in mathematics, that is why the tree representation of a formula was developed within this thesis. A tree is a familiar navigable structure and is used in the context of this thesis to display a formula as a tree of terms. During the first trials of the term-tree we relied on screen reader default and Braille output and considered the speech output confusing and too wordy. A newly designed auditive model which used earcons (different pitches of sound and repetition of sounds) for playing the node structure in a formula represented as a tree was introduced.

In the last section of this thesis an evaluation was conducted with a visually impaired mathematician and a sighted mathematician.

The master thesis showed that a formula represented as a term-tree is useful in finding answers to questions about structure and content of a complex formula in connection with a screen reader and Braille display, whereas the newly developed auditive model turned out to be less helpful.

Kurzfassung

Es gibt circa 3.000 gesetzlich blinde Menschen bei einer Einwohnerzahl von 8,4 Millionen Menschen in Österreich. Das Schulsystem ist an diese kleine Zahl von blinden Menschen nicht optimal angepasst. Speziell die Mathematik ist ein Bereich, welcher wichtig für den Alltag ist aber eine der größeren Hürden für sehbehinderte Menschen darstellt. Sehende Menschen verwenden oft örtlich verteilte Informationen und Symbole, welche ein Problem für assistierende Geräte für Blinde wie z. B. Screenreader und Braille Displays darstellen. Unglücklicherweise gibt es keine Allzwecklösung in der Literatur und bestehenden Projekten zum jetzigen Zeitpunkt.

In der Mathematik sind Formeln ein essentielles Element mit dem man einen sehr großen Teil des Mathematikwissen beschreiben kann. Versteht und kombiniert man dieses kleinst zusammenhängende Element der Mathematik beliebig oft, so kann man Rechenschritte durchführen und auf Richtigkeit überprüfen. Im Rahmen dieser Masterarbeit wurde ein Lösungsansatz entwickelt, welcher eine neue Formeldarstellung im Baumformat ermöglicht. Bei dieser Darstellungsform waren wir von der Standardausgabe eines Screenreaders abhängig, welche jedoch zu verwirrend und lang war. Die Nachteile der Sprachausgabe des Screenreaders konnten wir bereits bei der Entwicklung des Term-Baumes feststellen, deswegen entwarfen wir ein neues Tonmodel, das aus Earcons (verschiedene Tonhöhen und deren Wiederholung) bestand. Im neuen Tonmodel wurde jedem Knoten im Baum sein eigener Ton zugewiesen.

Im Rahmen von Tests mit sehbehinderten und sehenden Mathematikern hat sich gezeigt, dass eine Formeldarstellung als Term-Baum nützlich ist um mit Hilfe von Screenreader und Braille Display Antworten auf Fragen zu Struktur und Inhalt einer komplexen Formel zu finden. Das neu entwickelte Tonmodel für die Term-Baum Darstellung wiederum hat sich als eher nicht hilfreich herausgestellt.

Chapter 1

Introduction

1.1 Background/Motivation

Mathematics is essential for everyone. Whether it is used when shopping, in school or in working situations - mathematics is omnipresent. However different levels of mathematical difficulty need different strategies and approaches in order to solve mathematical problems. Comprehension is key to starting on a solving process. Once the whole problem is understood there are many ways to get a solution. Certain levels of comprehension are required to start an academic career and furthermore to achieve an academic degree. Especially for visually impaired people it is hard to have an academic career because mathematics is largely inaccessible and therefore incomprehensible.

1.1.1 Inclusion as a Role-Model

Concerning math education of visually impaired people inclusion should be preferred. In contrary to integration (integrated groups of pupils, separate classes, proprietary methods of teaching, ...), inclusion strives for equal opportunity in education. The goal is a mutual way of learning and teaching, which includes everyone [1]. As a positive outcome, inclusion leads to an increased motivation and results in a good working atmosphere. Solely the method of access to learning materials and the method of producing input on them, can vary when you compare sighted and visually impaired people's classes. Nowadays the computer is a frequently used working tool, which sighted and visually impaired people use for studying as well as working. This is why the computer is the ideal kind of platform to establish a potential cooperation between sighted and visually impaired people. At the time of

writing there is no programme which completely fulfills the need for an inclusive way of working with math in education [42].

1.2 Difficulties in Mathematics

Mathematics has its own kind of language. The main building blocks of mathematics are numbers, operators, functions and structural information. Especially in structural information there is a lot of hidden semantics, for example the special location of numbers (exponents, limits, ...) and operators (fraction, positive or negative numbers, ...), which give additional clarity in the process of solving a mathematical problem. Additionally there are rules, which define arity, associativity, commutativity and distributivity towards numbers and variables. A certain order of rules contributes to the solving process of a mathematical problem. Sometimes a mathematical problem is easier to solve, if you decompose it in order to apply a rule on a smaller problem of the formula. The decomposed parts of a calculation will be solved (with the help of side calculations, sketches, etc.) and the interim results are passed on into the original solving process. With the rising complexity of mathematical problems, more and more aids are created to find a solution.

The described approach is taught in math classes at compulsory school and secondary schools. A solving process in mathematics is not always as linear as it is needed for visually impaired people to be compatible with their assisting technologies such as screen reader and Braille display.

According to Stöger and Miesenberger [42] and Archambault [3] the Reading, the Navigation and the Manipulation are the three main categories of problems which can occur when visually impaired people handle mathematics.

1.2.1 The Reading

The Reading concerns written mathematics for example the readability of a mathematical exercise or calculation. Calculations and steps of calculation have to be designed well readable and editable. Regular sighted people usually work with two or multidimensional representations, whereas visually impaired people work best with linearised and well guided structured content. Visually impaired people have problems with two or multidimensional dimensional representations of formulas [5, 18, 42].

Which means that there is a need for an integrated assistive tool for for providing a readable and comprehensible representation of a mathematical

example. A visually impaired person has tools which linearizes multidimensional problems [18].

1.2.2 Navigation

The navigation mechanism has to consist of a suitable structure for a mathematical problem (typically a formula) on which it must ensure a navigation without reaching an impasse. A visually impaired person usually uses navigation keys (comparable to arrow keys) on his Braille display when working on a computer. The combination of screen reader and Braille display enables to go completely through a (known or unknown) structure. Trees as navigational structures are familiar to visually impaired people (e. g. in websites on the Internet). There are problems with very deep tree structures, because it is very difficult to survey all the content. To solve this issue, additional tried and trusted effective guidance is needed. For example a sort of history which lists the last steps [37] could be used. Visually aiding constructs like the drawing of strategies such as the expansion of a product of sums, encircling, striking out are difficult to mimic for visually impaired people [3].

Spatially divided mathematical expressions, such as nested fractions are hardly solvable due to accessibility issues with screen reader and Braille displays [4]. Additionally there are constructs which require the content to be located below each other or has indentations.

1.2.3 Manipulation

Actively *doing* mathematics is indispensable for learning; just reading is not enough, students need to do their own trials and need to encounter errors in order to become confident and actually master mathematics. The most challenging representations of mathematical expressions for assistive technology of visually impaired people are two dimensional or multidimensional representations [5].

Working environments and aiding tools are problematic if they contain context switches during the manipulation of a calculation. A regular sighted mathematician works with side calculations and jumps into different formulas and rules. The focus is an important tool in an accessible programme. The developer has to have full control of the focus because screen reader and Braille display rely on it and it can only center on one thing. What the focus cannot capture, the visually impaired person has to keep in mind. The limited powers of recall have to be extended by technology.

1.3 The Key Point: Comprehension of Terms

Even if a big part of calculations can be displayed in a linear way, the mathematical content can be too large to fit into Braille display or to long to be readable by some screen readers. Moreover the information about mathematical semantics are lost because of the linear display, this is a problem when thinking about hearing formulas via screen reader. Therefore the mathematical content has to be adjusted to fit into means of assistive technology. A calculation can be decomposed into its terms. A formula consists of one or more terms. When two or more terms meet, there is a need to determine their precedence. This normally happens through the evaluation of the semantics in structured information and/or with the help of brackets.

The following calculation for example looks very complex to sighted and visually impaired people.

$$1 + \frac{2 \cdot x \cdot (y+3) + \frac{(y^2+6 \cdot y+9)}{y+3}}{4 \cdot z \cdot (y+3)} + 5$$

It consists of an additions, multiplications and most important a double fraction. When sighted people start solving this calculation, they may first look for similar terms in the counter and in the denominator of the main fraction. The given calculation can be simplified in order to ease the solving process. Therefore the sub-term $y + 3$ is cancelled from the counter and denominator. The result could look like this:

$$6 + \frac{2 \cdot x + 1}{4 \cdot z}$$

For visually impaired people there are many challenges in performing the first presented calculation. In a linear form it would be very large and nested:

$$1 + ((2 \cdot x \cdot (y+3) + ((y^2 + 6 \cdot y + 9)/(y+3)))/(4 \cdot z(y+3))) + 5$$

First the visually impaired people have to get an overview, second they have to know if they are located in a fraction or just got into another fraction and third they have to remember what terms they stepped through before. This is why especially the second fraction, simplifying and the associativity (addition of 1 and 5) pose problems.

A solution to the problems could be found with the help of terms because a term takes only a small space on the keyboard and is easy to comprehend.

When designing an interactive math editor for visually impaired people you have to keep some issues in mind. Common input and output possibilities

for visually impaired people have limits which you have to keep in mind. The technology for screen reader (output tool) and Braille display have problems with certain structures, long text lines and calculations. For historic reasons Braille displays usually have up to 80 usable characters per line. If more than these 80 characters are necessary, a new line will be used, which is inconvenient when dealing with calculations. Moreover structural information can be lost when a content is linearized because of acoustic or Braille display [18]. Visually impaired people are able to create a mental map of spatially divided structures but with every additional line the needed power of recall increases. It is important to develop an inclusive approach (visually impaired and sighted people can work together) to enable each participant to think outside of the box and gain additional knowledge on how to perform a calculation in a new way.

1.4 Visually Impaired People and Society

In Germany and Austria blindness means that a person is capable of only seeing 2% of a healthy sighted person. The main reasons for blindness are genetical diseases or degenerative diseases of the retina, lens or the nerve of the eye (see glaucoma, cataract, ...). According to a survey of Statistik Austria [49] there are approximately 318,000 visually impaired (of different degrees) people in Austria. These 318,000 people mean a share of 4% of the entire population in Austria. For these 4% it is possible to classify blindness more precisely into low, medium, high and full grade visual impairment. A full grade visual impairment have about 0.9% of the visually impaired (see figure 1.1), which are about 3,000 people in Austria.

Visually Impaired People and Mathematics – this specific combination is the focus of this thesis. This is a topic which is addressed in the IIS, the "Institute Integriert Studieren" at JKU, Johannes Kepler University Linz too. IIS is the world-wide only academic institute dedicated to assist students with special needs in their studies. The experts at IIS confirmed: there is only a very small number of visually impaired mathematicians in Austria.

One of them is Dipl.-Ing. Bernhard Stöger (see [50]) and this thesis had the great advantage to enjoy his support. His research focus is software support for learning mathematics, he also contributed experience from earlier projects [4], [5]. His contribution comprised requirements engineering, general advice and finally evaluation of the practical part of the thesis.

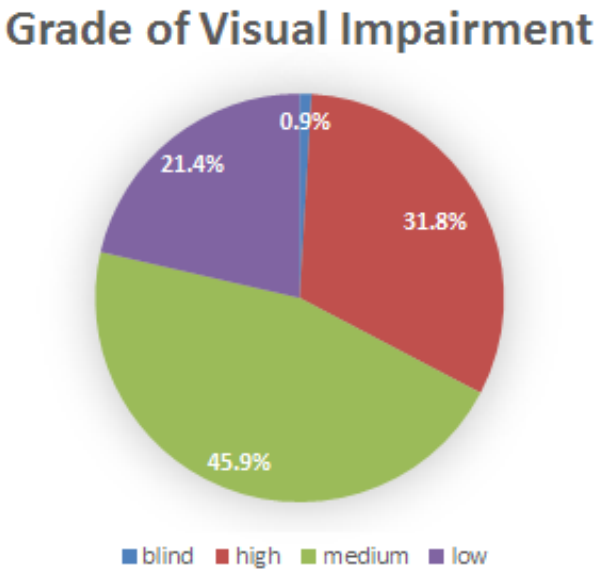


Figure 1.1: Pie chart showing the grades of people with visual impairment in Austria.

1.5 Goals of this Thesis

Since there is no established solution and only few programmes which rarely leave prototype state [42], it is reasonable to extend an existing project.

1.5.1 Extending *ISAC*

In this thesis the open source project *ISAC* is going to be extended. The *ISAC* project [68] started about 15 years ago at the TU Graz. *ISAC* is an acronym which means ISAbelle for Calculations in applied mathematics [68]. Isabelle [32] is a state of the art theorem prover developed at the TU Munich. A theorem prover can check mathematical content on its correctness by comparing the given mathematical content with the existing mathematical knowledge (axioms, formal logic, ...). *ISAC*'s development is coupled with Isabelle's development, because *ISAC* relies on complete mathematics knowledge and therefore has an interface to Isabelle. *ISAC* is still a prototype, which has been developed along students' projects in cooperation with respective supervisors with specific knowledge at different universities.

The *ISAC* prototype includes software for an authoring-system, a tutoring-

system and a mathematics-kernel. These components are described as follows [68]:

ISAC's mathematics-kernel has an underlying concept named "Lucas Interpretation" [28], which comprises a stepwise mathematical processing approach based on Computer Theorem Proving (sequence of steps is evaluated during interpreting) and Automated Theorem Proving (checks if knowledge is derivable). The interpreter implemented in ISAC processes certain sections of a calculation and then passes performing a calculation to the user. This interpreter takes the existing mathematical theorems which can be exported from Isabelle and checks the calculation step with the help of combining deduction and computation. The mathematical part of ISAC is later called the MathEngine.

ISAC's authoring-system is made for the administration of pupils and inserting examples into ISAC. The examples can be accessed via the included example browser (see figure 1.2), from which you can insert and load mathematical examples. Like the mathematical examples you can find in a math school book, these examples can have pictures and text, whereby there has to be at least one interactive calculation (link in text). With this calculation a Worksheet can be opened. Additionally there is the method (applicable rules) and the problem browser (mathematical conditions) in which the example can be described in detail.

ISAC's tutoring-system's centrepiece is the Worksheet, which contains a formula editor. A Worksheet is opened on example or blank. In the figure 1.3 we see a Worksheet in which the calculation $Diff(x^2 + sin(3 * x^4), x)$ was performed. An important fact of the existing ISAC Worksheet is the dynamic calculation sequence, because if a step is altered the ISAC programme reacts and computes if the changes are correct or not. ISAC gives freedom of performing a calculation on your own style. If the steps are not correct according to the mathematical knowledge of ISAC, a red icon will appear next to your typed step immediately.

With the Worksheet the supportive buttons "NEXT" and "AUTO" are given. The buttons enable the user to get help in an active calculation by showing the next step in the solving process ("NEXT") or by performing the whole calculation ("AUTO"). In the figure 1.3 the applied tactics such as (`Check_Postcond ["derivative_of","function"]`) of certain steps in the calculation are shown. The closed folder symbol stands for hidden intermediate steps, which can be provided by ISAC when needed.

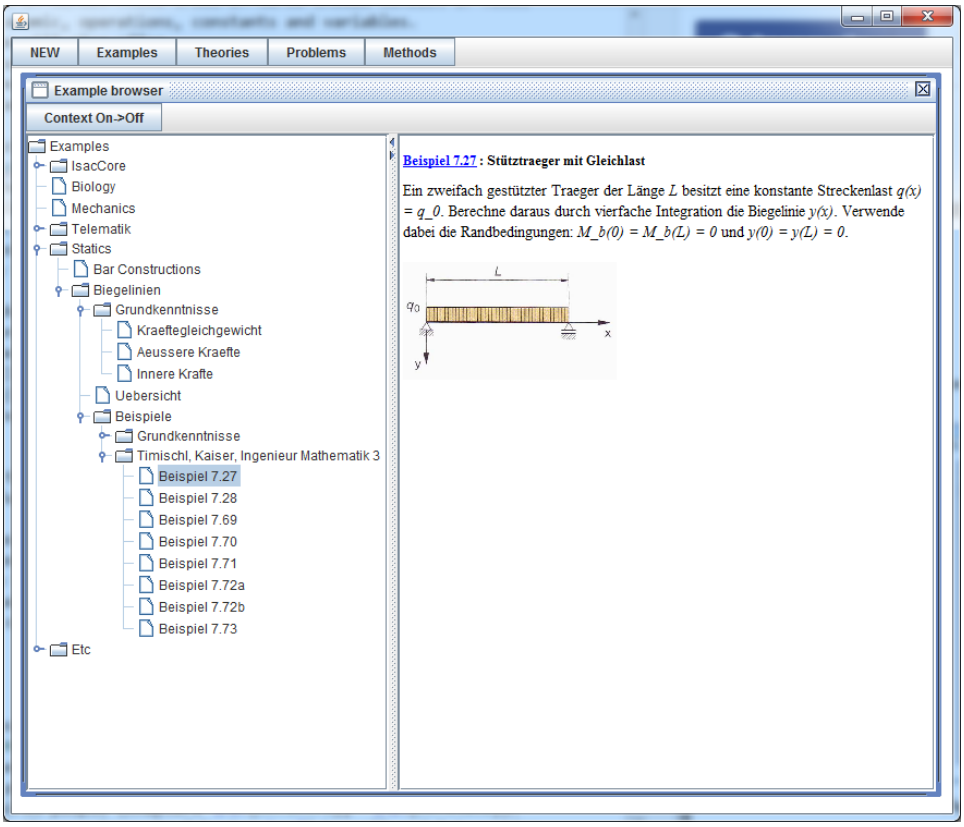


Figure 1.2: Image showing the example browser of *ISAC* and a selected example on an uniformly distributed load.

1.5.2 Introducing Inclusion into *ISAC*

Until now visually impaired people had not been considered in *ISAC*'s user group, this is why it should be extended by accessibility mechanisms. The desired accessible outcome in *ISAC* should look like this:

A generally readable math exercise is loaded into *ISAC*'s example browser by a teacher. The existing editor enables predefined examples and the creation of new formulas. These examples are opened into a worksheet, where the calculation can be edited. In the worksheet it should be possible to work together (visually impaired people and regular sighted people) with the future extensions of *ISAC*. This is why a mechanism is needed to switch between representations of the programme for visually impaired and regular sighted people.

The newly developed tree representation enables the decomposition existing formulas from the math worksheet into a tree of terms. Those trees of terms

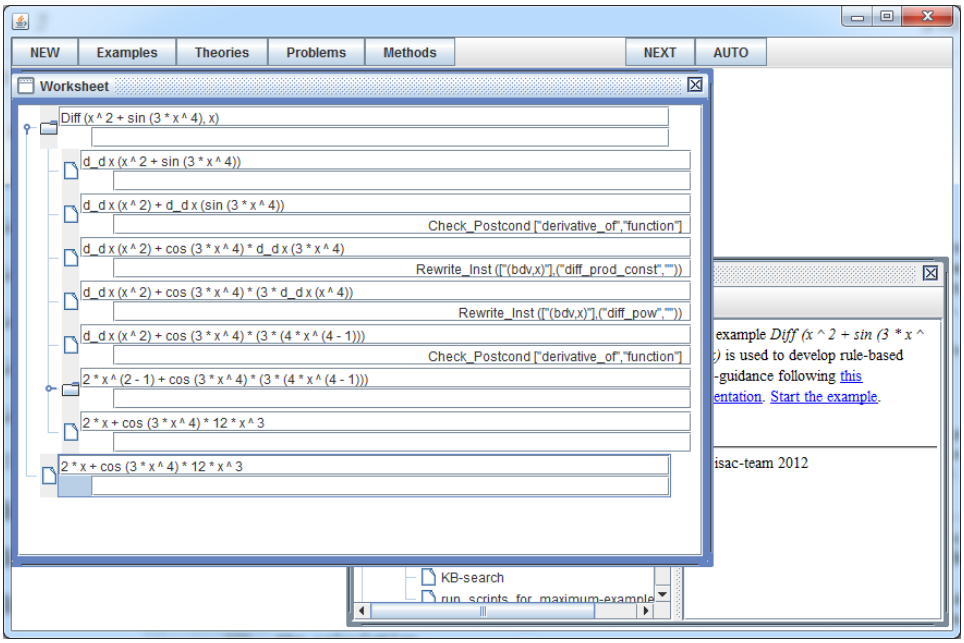


Figure 1.3: Image showing a Worksheet of the example "rule-based dialog" in *ISAC*.

should consist of nodes which can be terms and, most atomic, operations, constants and variables. However the new representation will also offer a possibility to see the calculations as a linear string representation. In the context of this work there is going to be an unconventional approach to represent mathematical formulas with the help of prefix trees of terms.

A similar attempt with restriction to the Navigation in MathML based formulas has been developed by [19] with MathGenie. Moreover an appropriate mechanism for the navigation in a calculation should be established to enable visually impaired as well as regular sighted people good support in learning math. A small set of test examples (which are already part of *ISAC*) should be tested to find a good solution to support people in learning math. One of the key features will be a mutual way to learn math.

At the end the thesis there should be a concept on how to improve the accessibility in *ISAC*. Therefore it is necessary to examine the existing project. Since the graphical user interface (GUI) is extended, design patterns like the Model View Controller Pattern [38] should be applied. Moreover with a pattern the separation of data and structure are divided and certain components could be extended with less effort.

Summarized one can say, that the development of an interactive (reading, navigating and manipulating functions) mathematical formula representa-

tion has to consider the limits of input and output tools (screen reader, Braille display) of a visually impaired person. The extension by supportive functions of formula representation should live within these limitations. Supportive functions are most important when doing math, to create a mutual base for visually and blind people to work together equally within the same mathematical context.

1.6 Structure of the Thesis

This thesis is divided into seven chapters. The first two chapters should give a picture of the problem and current solutions when speaking about mathematics and visually impaired people (chapters 1, 2). The further chapters deal with the requirements (chapter 3) and integration of a prototype which tries to solve the mathematical representation problems in *ISAC* (chapters 4, 5). The requirement phase and the implementation phase are not in a chronological order, because it is much easier to find requirements with a prototype for the visually impaired people. The iterative process consisted of requirements specification according to literature and visually impaired people, implementing a prototype, reevaluation and increase the existing requirements. Finally a evaluation of the prototype is done by visually impaired and normal sighted people. The results of the evaluation can be found in chapter 6. The last chapter 7 describes the lessons learned and possible future improvements.

Chapter 2

State of the Art in Assistive Technologies

In this chapter the assistive technologies for visually impaired people are examined and therefore respective issues and connected scientific works, particularly dedicated to mathematics, are surveyed. There have already been projects which target visual impairment and mathematics. Most of the applications which have been going along with these projects never left prototype stage. However a lot of good ideas for assistive tools have been created, which are described in the following sections.

2.1 Technical Prerequisites for Screen Reader

The development of a new feature for *ISAC* is guided by its target group, which are the visually impaired *and* sighted people. *ISAC* currently runs and is developed on a computer with the operating system Linux Ubuntu 14.04. Statistics show shows that Windows is the most commonly used operating system with Linux far in the background [62]. This statistics applies to average desktop user, without information if there were visually impaired people included. However if we look at another statistic in figure 2.1 we see that the most commonly used screen reader software is JAWS [59]. JAWS is a closed source screen reader which is usable with Windows only. ZoomText [71] is a tool for magnifying and reading screen text, whose user are not the main target group (low to high grade visually impaired) of this thesis. The next popular screen readers are Window-Eyes [69] and NVDA [65] which are like the previous named screen reader software products limited to Windows too. NVDA additionally has the advantage of being open source, which

gives many programmers the chance to adjust their application and NVDA (Python) to their needs. Also NVDA doesn't cost money except for custom languages which can be bought (in comparison JAWS has a base price of $1,095.00). The price factor is essential especially to schools. This means that even poor people in developing countries could afford education for visually impaired people in their computer lab.

Which of the following is your primary desktop/laptop screen reader?

Screen Reader	# of Respondents	% of Respondents
JAWS	743	30.2%
ZoomText	545	22.2%
Window-Eyes	508	20.7%
NVDA	360	14.6%
VoiceOver	188	7.6%
System Access or System Access To Go	36	1.5%
ChromeVox	8	0.3%
Other	72	2.9%

Figure 2.1: Pie chart and table showing the most popular screen reader of people with visual impairment [48].

2.2 Assistive Technologies Concerning Mathematics

The development of assistive software technologies comprises visual and tactile support for visually impaired people. To be fully accessible, a content has to be readable, navigable and manipulable via screen reader and/or Braille display. This even gets more complex when we think about mathematical expressions, which are non trivial information for the existing tools of a visually impaired person.

2.2.1 Certified Standards in Assistive Technologies

To maintain the importance of a newly developed technology it is crucial to keep up and work with already existing certified standards. Assistive technologies use standards to make life easier for blind people. Visually impaired persons, who work with computers often come to their limits. Especially when talking about mathematics, which is a quite complex field of education, it is difficult for the blind people to access certain mathematical content. This issue can be divided into three major problem classes, which are the reading, the navigation and the manipulation of mathematics. For every problem class there are solutions ranging from certified standards to early prototype status.

2.2.2 The Reading Comprehension

Visually impaired people like regular sighted people use their computer to work or study in their everyday life. Normally a Braille display and a screen reader programme are used by visually impaired people to control and navigate on the computer. The following examples are not mentioned to be extraordinarily adapted to speech output, because speech can get very monotonous when used standalone (without Braille display) for reading [42]. Moreover the preference of speech or Braille output in mathematical context is heavily depending on the people's habit and daily routine of using assistive technologies [39]. Speech can fail when the content is interpreted incorrectly, but Braille is able to show the expression in a way which is more similar to the sighted person's vision [41].

A Braille display is a piece of hardware, which acts as an extension to a standard keyboard. It contains a line of dynamic piezo elements which are used for transmitting tactile information. Additionally a Braille display has navigation keys. There are many producer of Braille displays which use different drivers, so it is complex to find a uniform way to make your application suitable for every device.

A screen reader is a complex piece of software which has to include a speech synthesizer, a text to Braille converter and sometimes middleware programmes to fully access content on desktop and the web. Moreover the screen reader decides on what and in which order to speak about the content on "screen". Not only has it to read the text but also guide through navigable structures, list possible actions and describe which actions are happening (i.e. in NVDA [65] a progress bar is represented by an increasing pitch of sound).

Reading is crucial in understanding what to do and what is asked in a text.

Unfortunately screen reader software and Braille displays have trouble in processing and putting out some specially formatted or structured texts, like they are commonly used when expressing mathematics. A mutual foundation for programmes is important to stay up-to-date and to provide an interface to a broader audience.

Following formats of data are accessible [42]:

Braille

Standard Braille is a globally used format of text for visually impaired people. Whether printed on tactile paper or on Braille display, Braille is helpful when reading text and even in graphs. Printed paper enables two dimensional representation of information, but Braille displays often don't have the possibility of displaying this many elements on a standard 40 to 80 Braille display.

Additionally there is not only one standard in Braille. Depending on the country you live in Braille can have a different layout (up to 50 different layouts [42]). Additionally mathematics comprises a particular Braille layout in some countries for example Nemeth Code, which can even take up to 8 elements instead of the standard 6 elements for special formatting or characters (as they are commonly used in mathematics).

An approach by Martos et al. [25] was to make Nemeth Code language independent. UMCL (universal maths conversion library) [6] tried to convert MathML to various Braille formats. This is done with the use of InftyReader [44] which is able to read mathematical content from scientific paper (in pdf format) and images. The resulting files can be converted into other formats such as (LaTeX, MathML, Braille ,...).

LaTeX

LaTeX is a format which is frequently used when writing scientific papers or very long documents, but it can be an opportunity to make mathematical content available to visually impaired people. For blind people the reading comprehension of mathematics in documents is made easier by certified standards such as LaTeX [61] and/or MathML [53]. LaTeX source provides linearization of calculations and other textual contents and thus it is easy to process by the commonly used screen reader and Braille display technologies and therefore made accessible for visually people [13]. However Stöger and Miesenberger [42] have stated that LaTeX is not the optimal solution to the problem of reading mathematics, because it is still a source code with a lot

of text formatting overhead and the text's nested structures can take a lot of space on the Braille display.

MathML

MathML is a state of the art standard technology, which is used frequently by other projects [44][6][63]. In general MathML [53] enables to express mathematics as XML based tags and elements that are accessible via screen reader and Braille display. There are two types of MathML: Content and Presentation MathML. The Content type has information about the semantics and supports editing of a mathematical expression whereas the Presentation type is used for rendering and displaying graphically [66]. In the web-browser Internet Explorer 11 you have to install the MathPlayer [64] plug-in, to fully access MathML nested in HTML pages (Equivalent in Chrome is ChromeVox [51] and in Safari it is VoiceOver [67]). Firefox needs the Gecko [56] engine to be able to make MathML accessible to NVDA.

Overall the advantage of Presentation MathML is the possibility of rendering the output graphically and a good auditive output for visually impaired people. According to the central idea of inclusion the MathML data format is suited for inclusive applications.

Web Content

HTML – HTML [52] is an ongoing standard in the area of web technology. HTML as well as MathML have a DOM (document object model) based document structure, which means that their elements are built up like a tree structure. In HTML web pages other technologies such as JavaScript, MathML and many more can be embedded.

When a visually impaired person accesses a web page, the static content gets loaded into the screen reader first. Dynamic content can be a problem with assistive technologies because the loaded static content is updated infrequently. That is why developers in web have to pay extra attention to providing an accessible interface of their website [17]. Often a screen reader tries to interpret dynamic or complex content and fails to recall it correctly, that is why the WAI-ARIA [54] web standard has been created. In this way the programmer can convey more meta information into a website which is read uninterpreted by a screen reader. Additionally it enhances the interactivity and the platform independence.

In the web there exists a variety of web accessibility standards (WAI [47], WAI-ARIA [54], WCAG [55], ...), which can or cannot be considered when creating a website.

MathJax – MathJax [63] is a JavaScript library which represents LaTeX and MathML based mathematical content and makes this content interactive to screen reader and Braille displays. MathJax is open source and therefore used by variety of other Web browser plugin-based formula editors like OERpub editor, FireMath, CKeditor, and many more.

2.2.3 The Navigation

Not all the information which we read is linear and understandable on its own, thus we need to navigate through a bulk of information in order to fully understand what we read. Some pieces of information have to be collected in more than one place for example when we think about mathematical expressions such as double- fractions. People with regular sight can perceive more data with looking at something once. Additionally they can change context easily [2].

In the following possible structures for text based content are analysed and assessed by their possibility of navigation and related work is presented:

Lines of Text

The simplest (but sometimes insufficient) structure to read and work with are lines. When a text must be read carefully or when writing something – linearity is needed.

A long linear text line can be wrapped as a block, which is read rather randomly than line by line by regular sighted people.

Information in single line will not be able to give an impression of spatially distributed content (such as fractions in mathematics). Additionally lines are easy to navigate in but when it comes to search some context, which is not directly expressed by word, it is hardly possible to find information in a short time. Spatially distributed structures (like in tables, graphs, trees, ...) are able to show more information with a certain used semantic and syntax. Often it is not necessary to read every line in detail and therefore it is more comfortable for sighted people to work with spatially distributed content.

In contrary visually impaired people heavily rely on linear information. Assistive technologies such as screen reader and standard Braille displays (40 or 80 elements in a line) process underlying information as linear speech or linear tactile information. Multi-line Braille displays exist, but are not without fault. Text blocks can be displayed, but the tactile information can

only be read with the help of one finger, which needs training when changing lines on the Braille display [34]. All the underlying information which is not linear must be optimized for screen reader and Braille displays and therefore made linear. Particularly in the field of mathematics it is common to use multi dimensional representations of mathematical expressions [3]. Some mathematical expressions like calculations can be linearised but the linear representation takes more space on the Braille display and structural information is going to be lost [18], [3], [4]. For example: A linearly displayed double fraction can require a large amount of space and sometimes there can be mix ups with the operator precedence of terms related to wrong interpretations of screen reader speech output.

However to a visually impaired person linearity is fundamental in reading and hearing, so the content has to be made accessible for them.

Tables

For regular sighted people tables are used to rapidly compare a lot of information in a dense location. The overhead of additional words to describe a fact are removed. Small tables already require complex eye movement and context changes, which are not available to visually impaired people. It is difficult for visually impaired people to extract information from tables [16].

A big field of use for tables is the web, where tables are a special content and have to be dealt with properly by the designer in order to gain accessibility for screen reader and Braille display [43]. Goble, Harper, and Stevens [15] analyzed structures in the web for accessibility and they found a lack of navigational information with tables for visually impaired people. There also can be problems with formatting of the columns and the separation by the table frames has to be replaced by another construction or textual explanations [43].

Graphs

Graphs (discrete mathematics) are structures which holds elements that contain information and interconnected elements. The connection between the elements can give additional information too. Organigrams and other diagrams like class diagrams (e.g. UML) are examples for graphs.

For regular sighted people graphs can give a lot of information but also can get very complex and confusing very fast. A navigation is easy (follow the connections) but a query can get harder the larger (increased amount of elements and connections) a graph is.

Másilko and Pecl [26] examined the usability for graphs displayed as tables of elements and respective connections in an approach to make mathematics more accessible. This shows that visually impaired people must be provided with an replacement structure or a description text if they want to work with graphs [43]. Amongst other things graphs can be a part of many requirement specification processes and are therefore really important to visually impaired and regular sighted people [9].

DiGVis [45] is a software and hardware (matrix of pins) for guiding through, creating and reading directed graphs. According to Syal, Chatterji, and Sardana [45] the most difficult thing for visually impaired people is to visualize the structure of the graph and find connections and relations in it.

Trees

A tree is similar to a graph – it also comprises elements (nodes) that hold information and connections between these elements. In general connections in a tree are limited to give information about parent-children-relationship of a node. A navigation in a tree may start at a root node and may stop at any node that has the desired information in it.

In general as a regular sighted person a navigation in a tree structure is easy but with large trees searching for a single information becomes more difficult.

Scientific work on using a tree structure for the ease of navigation for visually impaired people had been done by Baker, Milne, and Ladner [7] and Smith et al. [37]. A tree is used in both examples for making coding easier for visually impaired people. In contrast to a tree's rather visual attributes such as indentation (when reaching a child node/branch) and syntax highlighting which are usable for regular sighted people, visually impaired people can navigate through a tree by using keys and acoustic output [7]. Visually impaired people have to build the tree structure in their memory, so they can keep the context in which they are located in. So it is important to reduce the needed brain power by introducing some features to convey structural information such as level and line of the current tree node.

Visually impaired people need an additional structure to navigate in, in order to fully capture a mathematical expression. Therefore trees have been used in projects connected to mathematics before in the MathGenie [19], Lambda [12] and in the MaWEn [5] project. Archambault et al. [5] mentioned a possible long instruction and trial phase for the navigation and the functions (collapse and expand of blocks) in a tree, as a concern but the benefit for a visually impaired person who used MaWEn outweighed this first phase (according to the following paper written by Archambault et al. [4]).

Speech Used for Navigational Purpose – Speech or sound features (earcons) can help when navigating. For instance work has been done with audio cues (earcons and spearcons [46]). Earcons can be understood as an auditive picture which resembles an event. While spearcons are spoken icons, which means they include words, earcons are designed as simple pitches of sounds [46]. Some guidelines on these audio cues are their shortness, they should propagate important information first and they should support hearing just a few words and immediately be able to skip to the next line [46], [7]. Audio feedback in form of speech and earcons had already been used successfully by Cohen et al. [9] to display a graph to a visually impaired person [8].

2.2.4 The Manipulation

Doing math comprises reading, navigating and manipulating content. This is a rather complex task to tackle considering all possible problems which can occur when designing such an application. Especially the context changes and the increased needed concentration pose the biggest issues [42]. At the moment there is no fully satisfying solution that left the prototype stage.

However some projects seem to be very promising such as:

Lambda

The Lambda Project [12] is a closed source project, which takes linear input and produces Braille, speech, two dimensional and linear output. It uses Aster [35]. Aster (Audio System For Technical Readings) provides a way of reading documents with complex structures such as mathematical texts. It has its own Braille code for special characters such as “Open compound fraction” and therefore needs at least an instruction course to be usable for visually impaired people.

GeoGebra

GeoGebra [57] is an educational mathematics open software that has focus on calculations for mathematical graphics. With GeoGebra a visually impaired person has the chance to draw, read and manipulate geometric figures. However this application is not crucial for this thesis which focuses on formula editors.

WIRIS Editor

The WIRIS editor [70] is a web application that allows doing mathematics which is compatible with any modern browser. Because of the usage of HTML4 and JavaScript a big range of devices (including mobile) can be reached by WIRIS. In this formula editor MathML and LaTeX format are supported too.

Chapter 3

Requirements of Visually Impaired People

This chapter addresses user requirements for an application which is going to have a small target group at the moment. The more important is the specification of requirements based on projects done in the past, researches (literature) and the direct contact with visually impaired people.

In a second requirement specification stage Stöger [41] helped us to come up with meaningful requirements based on an already developed prototype. The prototype was needed to be able to discuss issues and usability for a second improved approach. Without a prototype it is hardly possible to bring a visually impaired person into specifying requirements because the application is not feasible and therefore cannot be adapted meaningfully. The two requirement stages are not described separately in this thesis.

3.1 General User Requirements

The topic of this thesis addresses mathematics education supported by software, which in turn supports inclusive learning as discussed in the introduction (chapter 1). Inclusion assumes, that there is software used by both, by sighted students as well as by visually impaired students – however, there is no software with sufficiently wide-spread usage, neither at universities nor at high-schools.

This thesis assumes wide-spread availability of educational math software like *ISAC* in the future since recent development has shown that "making mathematics a first class citizen" [40] remains to be an important concern.

3.1.1 User Requirements for MAWEN

The expectation, that *ISAC* or similar software will be available wide-spread in the near future, is motivated by the software's features. These features presumably support visually impaired students as well as sighted students, as shown below.

These requirements will be part of the *ISAC* documentation after completion of the thesis. "MAWEN" is the abbreviation introduced by Archambault et al. [5] and re-used for the project, which aims at an accessible *ISAC* and has been started together with this thesis.

UR 1 ***MAWEN covers all phases in math problem solving,*** i.e. the phase of modelling (translation of a textural or figural problem statement into a formal specification), the phase of specification (relating formulas to existing knowledge about theorems, problems and methods) and the phase of step-wise constructing a solution of the problem [30].

This requirement is special, because in the field of mathematics and visually impaired people, there are applications which mainly specialize in one feature (see [5]). However *ISAC* should comprise a big field of mathematics, which is important for wide-spread usage in education.

UR 2 ***MAWEN checks user-input reliably and generously.*** This requirement is important for learning: each reasonable input of a student should be accepted by the system. *ISAC* is built upon the computer theorem prover Isabelle [14]. Such provers cooperate internally with a couple of automated provers [33]. This technology is the most powerful for checking user-input with respect to a logical context at the state of the art.

UR 3 ***MAWEN can suggest a next step*** if the students gets stuck in a calculation. This ability of *ISAC* results from Lucas-Interpretation [29], which extends Isabelle's power in deduction with computational power. The interpreter acts similar to a debugger: each step (i.e. a tactic in the respective program) is handled like a "break point" where control is handed over to a dialogue component. This component guarantees maximal freedom for interrupting a calculation, for instance exploiting the UR.4 below.

UR 4 ***MAWEN is a transparent model of mathematics.*** This requirement is accomplished by the fact, that Isabelle defines mathematics knowledge from basic axioms (of natural deduction [36], which are implemented in one single file, the so-called "trusted kernel"). All these definitions, theorems and respective proofs are implemented in a human readable format i.e. in the traditional notation of predicate logic.

3.1.2 Specifics for Visually Impaired People

Here user requirements are restricted to the most crucial parts for visually impaired students, to comprehension of formulas.

UR 5 ***Assistive components can be configured*** in a way that they are available as personalised settings at the login screen as well as during a session.
Such components are screen readers and respective settings, specific formula representations, etc.

UR 6 ***Mathematical objects can be referenced unambiguously,*** which is indispensable for inclusion: independent from respective settings from UR.5 each object can be identified unambiguously by two different students at two different computers working on the same calculation so they can compare and discuss variants of this calculation efficiently.
Such objects are a calculation, a formula or a theorem in the calculation, a knowledge item in the example-browser, theory-browser, problem-browser or in the method-browser.

UR 7 ***Sub-terms of a formula can be accessed individually*** in a way that comprehension of the formula is supported (which requires multiple access to sub-terms of the formula and random access to the structure of the formula). Thus random access is superior to sequential access as in string representation or in speech representation.

UR 8 ***Sub-terms in formulas can be referenced unambiguously*** on one computer, which is indispensable for inclusion: focusing at a certain formula, a visually impaired student and a sighted student can cooperatively discuss certain sub-terms with respect to application of theorems, etc. For that purpose sub-terms can be selected such, that both students can read this sub-term separately.
Note the emphasis of UR.6 to *one* computer.

UR 9 ***One formula can be shown in different representations.*** A visually impaired student wants to switch from string representation to tree representation (including audio information UR.12), as soon as the string gets too long for immediate comprehension. And as soon as a formula is comprehended according to UR.7, switching back to string representation might be required.

UR 10 ***Formula representations can be combined for inclusion.*** Given a pair of students, a visually impaired student (VIP) and a sighted student (SIP), collaborating at *one computer*: each of them requires the representation appropriate for him or her. The following combinations are foreseen:

1. SIP has graphical representation, VIP has string representation ...
2. SIP has graphical representation, VIP has tree representation with optional audio information ...
3. SIP and VIP have string representation ...

... of one and the same formula under collaborative consideration.

UR 11 ***Structure of trees can be surveyed by audio information.*** The survey comprises number of elements at the current state of expanded or collapsed branches and the depth of branching.
This requirement addresses various tree-representation found in ISAC's objects (compare UR.6): calculation, formula or theorem in the calculation, knowledge item in the example-browser, theory-browser, problem-browser or in the method-browser.

UR 12 ***Structure of formulas can be surveyed by audio information.*** "Structure" addresses the following information:

- the number of elements in the formula
- the depth of operator nesting of the sub-terms
- the location of a sub-term under focus within the whole formula.

3.2 Technology-Related Requirements

The requirements of visually impaired students learning mathematics are restricted to requirements on the most crucial details, on reading, navigating and manipulating formulas. Mainly the Reading (see section 1.2.1) is focused because one of the bigger problems concerning the readability which should be overcome, is the two dimensional representation of mathematical content.

UR 13 ***A formula is mathematically correct.*** When working on a formula it is important, that it is performable with valid mathematical knowledge. Therefore an application is needed which provides a mechanism to provide the user with valid formulas. A valid formula has a correct syntax, operations, numbers, ...

UR 14 ***An existing formula editor needs to become inclusive.*** A state of the art mathematical authoring system with an existing formula editor is open source and needs an extension to be accessible by VIP and SIP. Additionally it is meaningful to extend an existing project, rather than starting a new one from scratch.

UR 15 ***Different representations are provided to the user.*** Inclusion can only be enabled, if a proper representation of a formula is provided. A proper representation for a VIP is an accessible interface, which is adapted to the needs of a VIP. Whereas a SIP will need another representation in order to feel fully supported and to feel comfortable when working with formulas. Different representations need different navigation, that is why matching navigable details have to be clarified too.

UR 16 ***Formulas can be divided into terms.*** To get a tree representation of a formula, there has to be a mechanism for creating terms of a formula. Terms are building blocks for formulas and can be inspected closer until they are easier to understand as a mathematical unit.

UR 17 ***A tree is a navigable structure for VIP and SIP.*** As mentioned in section 2.2.3 a tree is a good alternative to linear representation of strings. To make it navigable for VIP and SIP the optimal combination of input possibilities (keyboard, mouse, speech, ...) has to be found. Moreover the there must always be an option to get to the parent, child or sibling node.

UR 18 ***Focus and context mechanisms are offered.*** Tree representations usually consist of nodes. These nodes could be used to show information on a selected node which is interesting with more details and currently unnecessary information should be hidden/minimized from the user. A focus and context [24] solution with terms should be included into a tree representation. This could be a possible solution to comprehend nested sub-terms, which are not easy to comprehend when seen as a large formula.

UR 19 ***Default orientation functions in a tree are insufficient.*** This requirement was created in the second requirement specification phase, after observing a VIP using a tree. The screen reader output in our prototype didn't help the VIP because it was very wordy and therefore disturbing (more details in section 4.2.1). An orientation mechanism has to be offered to support the usage of Braille displays while giving information on the structure of the formulas current representation.

Chapter 4

Concept of an Interactive Math System

This chapter is about the existing concepts and the concepts needed in ISAC.

4.1 Review of ISAC's Architecture

In the wiki-page of ISAC [68] you can find following definition:

> ISAC is an educational mathematics assistant, a single-stepping system for applied mathematics based on the computer theorem prover Isabelle, ISAC is an "interactive, transparent and complete model of mathematics" which aims at learning by trial and error in the same way as supported by good chess playing software.

Prototyping has been organised along students' projects in cooperation with respective supervisors with specific knowledge at different universities. 30 student projects contributed to the code until now. The initial architecture, in particular the interface of Isabelle to ISAC's mathematics engine, remained stable (and is therefore out of scope for this thesis).

The present focus of development is in Linz, in particular Johannes Kepler University and University of Applied Sciences Hagenberg. Currently the project is continued by this master thesis at the JKU Linz at the Institute for Integriert Studieren (IIS). As mentioned in the previous chapters - mathematics is a big problem for visually impaired people, therefore ISAC should be extended to support visually impaired people in doing math.

4.1.1 General View on ISAC's Architecture

ISAC is a distributed system, it consists of four parts, running in different JVMs and connected by Java RMI: Bridge, KE-Store, ObjectManager and the WindowApplication, see Fig.4.1 below. On the other side there is the `MathEngine`, which is a ISAC specific component to access the existing theorem prover Isabelle [14].

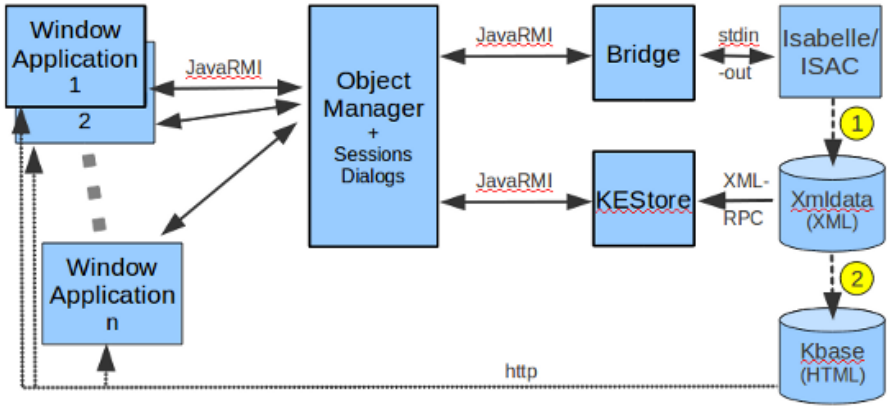


Figure 4.1: Survey on ISAC's components [68]

Bridge – This component does the work, which is needed to send and receive data of Isabelle/Isac (ISAC's mathematics engine). Additionally it also writes a protocol of the sent and received data to a console window.

KE-Store – Predefined interactive mathematical examples can be loaded with this component. Together with specific parts of the mathematics knowledge the examples (XML format) are exported from Isabelle/Isac. Represented by HTML, the representation in the `ExampleBrowser` is generated by Java code from XML in the `KE-Store`. The examples can be selected and started at run time in the `WindowApplication`.

ObjectManager – This component of ISAC generates and administrates sessions which are used e.g. when identifying a `Worksheet`'s current user and dialogues. Sessions get very important when ISAC is used in a class room situation.

WindowApplication – Everything which concerns the graphical user interface can be found in the `WindowApplication`. This means that the `WindowApplication` has to be changed or extended in order to build an inclusive formula representation which is appropriate for visually

impaired people. Therefore this is the most important component for this thesis.

4.1.2 Interaction in Stepwise Problem Solving

ISAC's distinguishing feature is stepwise problem solving in applied mathematics: a student should interact with the system in a way similar to a human tutor in a private lesson, or as described in *ISAC*: the interaction in *ISAC* within a calculation on a `Worksheet` (formula editor) should be similar to the interaction with chess software. Interaction in this case means, that the mathematical examples should not only be shown perfectly calculated but should support the user in learning math stepwise. A certain dialogue has to be established to enable this kind of interaction.

Consequently *ISAC*'s dialogue component is the result of a costly development process tackled and described in several theses ([31],[22],[20],[21],[10]) and published in some papers (e.g. [23],[11]).

Here we collect the design principles and provide a short overview at implementation by identifying *ISAC*'s respective code `in this font`:

1. ***ISAC*'s dialogues are seen between partners on an equal basis**, between the student and the system [31]: both can do a step in a calculation, both can accept a step of the partner or not (while the system checks correctness and gives feedback, and the student just overrides a step proposed by the system), and both can decide to become active (e.g do the next step). This principle is reflected by interactions (defined in `EUIElement`, which is used in both directions, from the student to the system and vice versa (the student represented by the `Worksheet`, the system by the `WorksheetDialog`).
2. **The dialogue component consists of several dialogs**: Doing mathematics comprises very different tasks which require different software components: specification of problems involves searching the mathematics knowledge by use of several browsers (`TheoryBrowser`, `ProblemBrowser`, `MethodBrowser`), constructing a problem solution stepwise uses a `Worksheet` as mentioned above. All the dialogues are managed by the `DialogGuide`.
3. **Dialogues contain `UserActions`**: Krempler and Neuper [23] give a preview to plans, how dialogues can look like in *ISAC*. Presently *ISAC*'s `UserModel` is only a stub (not yet implemented but part of the architecture), and so are the dialog-identifier, which makes *ISAC* only ready for presentations.
4. **The `WorksheetDialog` controls all `UserActions`** (as do the other dialogues) because there might be a dialog-mode "written exam", which

just disables all the assistive mechanisms of *ISAC* and leaves the whole system as it is. The main purpose of *ISAC* is independent and exploratory learning.

5. **The `WorksheetDialog` is a listener** to both sides, to the student and to the `MathEngine`. An asynchronous communication mechanism is established between users (`Worksheet`) and the system (`WorksheetDialog`). Asynchronous communication with the `MathEngine` reflects the fact, that the latter needs some time for checking user input for validity.

Although the `WorksheetDialog` controls *all* interactions with the user on the `Worksheet`, there is an additional connection between `Worksheet` and *ISAC*'s back-end (`MathEngine`). The separated handling of the subject matter, mathematics is modelled by a separate connection with *ISAC*'s back-end, which is established by a `ICalcIterator`. This interface allows to iterate over an existing calculation from the beginning to the end and to fetch respective elements from the calculation.

The connections relevant for interactions addressed in this thesis are shown in Fig.4.2

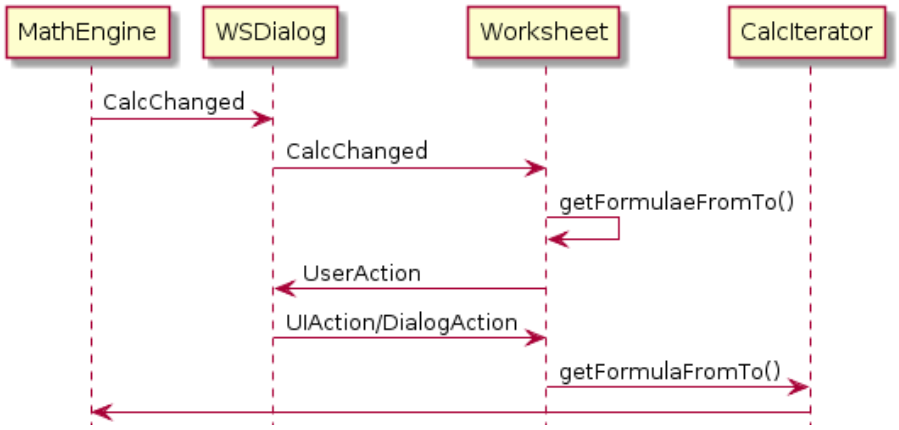


Figure 4.2: Events and methods which are needed to show a calculation on a Worksheet.

4.2 Visual and Auditive Representation of Terms

MAWEN (mentioned in section 3.1.1) defines the overall planned features of *ISAC* becoming usable to visually impaired and regular sighted people at

the same time. As we saw ISAC is a a very complex programme which took a while to be comprehended for the purpose of extension, that is why this thesis's contribution to ISAC is limited to introducing an inclusive representation of formulas and the evaluation what should be improved in the GUI in order to be accessible.

By the end of this thesis, the formula representation will display a single formula in a way it is readable to a visually impaired person and a regular sighted person. These formulas are going to be passed by the active `Worksheet` via a context-menu entry to the respective newly designed representation window.

4.2.1 Concept of a Formula in Tree Representation

Visually impaired people have problems with reading and estimating the size of a formula, that is why Archambault et al. [5] came up with the idea of turning the formulas into trees. Especially two-dimensional representation of formulas (like fractions) are inaccessible and therefore unreadable to visually impaired people. An approach of switching between representations for visually impaired or sighted people is meaningful, because then they can work together with their preferred representation of the formula. This inclusive way of working together should be extended by the possibility of setting the focus for the counterpart. A Tree should be understood by both, visually impaired and sighted people because we all are confronted with trees when using a desktop computer or the web browser. These structures are commonly used for navigational purposes such as used in File System Explorer and websites.

Tree structures are present in the ISAC project for example in the main formula editor. You have a root which is the main mathematical problem you want to solve and the solving steps with certain sub-problems (branches). The cells which are the representation of the tree nodes contain editable strings of formulas. These formulas can get too long for the Braille display and therefore should be adapted in a new tree representation.

Concept of Formula Decomposition

To get formulas into tree structures a meaningful decomposition has to take place. As introduced earlier terms can be understood as building blocks of formulas. The smallest term is a constant or variable, which should be displayed in the tree representation as a leaf. Operators are like nodes that must have at least one to two leaves in its simplest form. Branches can also contain nodes for children. To be able to navigate in this tree, keys for

movement have to be introduced, because the mouse is not accessible for visually impaired people but still kept for inclusive scenarios.

The concept of focus and context is defined by providing a mechanism to show more information in selected and less detail in unfocused areas [24]. Focus and context is realised by using the expand (more information) and collapse (less information) function a tree representation offers. A formula can be decomposed by using the expand function of a node. This means, that the selected formula will be divided on its main operator into the operator (now a branch) and its arguments which will be two proper subsequent nodes. The tree is only fully expanded, if every branch is expanded and every node will contain only one mathematical unit (a single number, constant or operator).

When fully collapsed the term-tree representation of a node looks quite the same as the formula in the formula editor of *ISAC* (immediate source of data), which means it is in linear representation. The first step can only be to expand and therefore get three subsequent lines.

An example how the final tree representation should look is the formula

$$1 + \frac{2 \cdot x \cdot (y + 3)}{4 \cdot z \cdot (y + 3)} + 5$$

which looks like this tree below, when decomposed but not fully expanded:

```
01    +
02      +
03        1
04        /
05          *
06            2 * x
09            y + 3
12          *
13            4 * z
16            y + 3
19      5
```

A formula displayed as a tree has the advantage to hide parenthesis, which can get quite confusing when reading a long and/or nested formula. Parenthesis are added the more the term-tree representation of a formula is collapsed. A fully expanded term-tree has no parenthesis in it because the arity should be clear to the user by looking at the sub-tree (nested) structure. Additionally the tree design was chosen because it is easier (for a regular sighted person) to think of the terms as a composed linear formula when designed as right hand-side and with subsequent nodes.

The Spoken Structure

The structure of the audio rendering is build of the depth, content, state of node (collapsed, expanded, selected) and remaining sibling nodes. One of the possible speech outputs of NVDA for a certain node in a tree could be: *level 3, y+2, collapsed, selected, node 1 of 2*. The output of the implemented nodes varies with the order of the pieces of structure. Sometimes the depth information is said at the last place. The reason is that screen reader software tries to interpret the meaning and to optimize the output sentences for the visually impaired, which leads to different spoken information.

4.2.2 Concept of an Auditive Tree Representation

After the second requirement specification phase, it came out that the wordy output of screen reader (when orientating in the term-tree) made the accessible navigation in the tree representation a lot harder to understand than it was thought to be like. Rapidly it was clear that neither the Braille display (no common standard interface for variety of hardware) nor the screen reader software (proper open source project) could be adapted at this point of the thesis. So the new idea of the sound model of the tree representation was born. For this approach the screen reader output is limited to generating information for the Braille display (as mentioned in section 2.2.2 screen reader software is responsible for producing Braille output too) and not producing speech output. It should be supportive for Braille display users who find the speech output of screen reader irritating. Based on the concept of earcons, we came up with a model that gives a quick survey of the represented formula in a tree representation. Therefore different pitches of sound, length of sounds and repetition of sound were instruments, that could easily be differed from each other. We assumed that a visually impaired person has a good comprehension when it comes to auditive hints. After a few combinations were tried (e.g. pitch of sound corresponds to line, every node has its fixed sound, ...), the following solution has been chosen:

- pitch of sound corresponds to depth of nodes
- a click (short sound of a matching pitch) is played for every node
- repetition of sound corresponds to collapsed nodes under a branch
- a long click marks the location in a tree representation
  Alternatives: At the time of implementation it was not clear, which of these two alternatives are more helpful: The special sound signalling the focus in the tree is given
  1. *before* the series of clicks denoting the size of the respective subterm

2. *after* the series of clicks denoting the size of the respective subterm

- the interval between the clicks that denote each node is as small as possible to be perceivable
- the repetition interval (collapsed children) is as small as possible to be perceivable

These settings are changeable in an ISAC initialisation file.

The sound model extends the term-tree representation, that is why they are used together in the latter scenarios. However the developed sound model is also thought of as giving an impression of other tree representations in ISAC (`Worksheet`, menus, browser, ...).

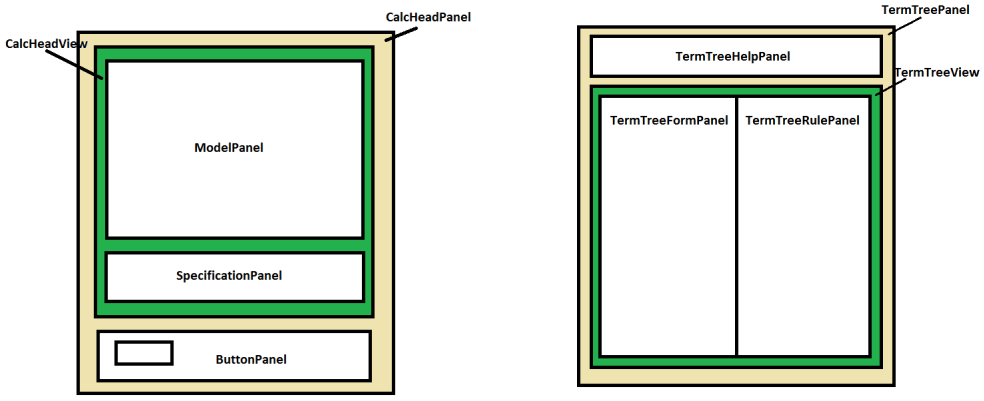
4.3 Integrating a New Representation

ISAC already has an representation, but the new representation should be able to decompose a selected formula into terms.

The term-tree representation can be invoked by selecting the context-menu entry on a specific formula. To help visually impaired people in learning math, the term-tree should be navigable. Eventually it can be easier for a visually impaired person to read a tree structure than reading a long linear string representation of a formula.

4.3.1 Communication with the Existing ISAC Structure

The new representation has to communicate with the `WorksheetDialog` in order to get formulas from the `MathEngine`. The existing code is dedicated to structures for event-based communication. When designing these communication patterns, you have to keep the principles in section 4.1.2 in mind. Events in ISAC always have a context (place where to find the item which lies beneath) and an element (possible interaction with the item). An example of events are the events, that are going to be fired when editing an existing formula. In detail we examine the methods `doUIAction` in (Worksheet) and the `notifyUserAction` in (WorksheetDialog) with the example of clicking an existing formula (not the `CalcHead`) and editing it. The `notifyUserAction` method is used for the communication in the direction of the `MathEngine` (back-end) and the `doUIAction` is used for repainting and informing the front-end of changes.



(a) CalcHeadPanel **(b)** TermTreePanel

Figure 4.3: Modular design of the existing CalcHeadPanel and the respectively designed TermTreePanel.

To be able to open our tree representation, we need an entry in the context-menu of the selected formula on the `Worksheet`. Therefore we design our list entry matching the other existing list elements in the context-menu such as "open calc-head panel". A calc-head is a description of a problem or sub-problem in a sequence of calculations. It can only be shown when this formula which represents a problem or sub-problem is selected to be opened a context-menu. We reuse the same internal frame to draw our tree representation, because the calc-head and the tree representation are not meant to be shown at the same time. Moreover the design of the tree representation frame is designed similar to the calc-head (see figure 4.3). The single panels in the frame are planned to be optional displayed when needed.

The `TermTreeHelpPanel` is planned to contain help functions (such as buttons, lists, histories, ...) you can compare it to the `ButtonPanel` in `TermTreeHelpPanel`. Like the `TermTreeRulePanel` the `TermTreeHelpPanel` should be optionally displayed depending on the connected context in the `Worksheet` (is the rule or the active formula selected for term-tree) and the user's need for guidance (is he a beginner or a professional) the shown context in the `Worksheet`. The `TermTreeHelpPanel` should never display more information than needed by the pupil to challenge him in doing math.

Like the `CalcHeadView`, the `TermTreeView` is only used as a component container for other panels (in `CalcHeadPanel`: `ModelPanel` and `SpecificationPanel`), therefore it is kept for similarity in the design of the term-tree. The `TermTreeFormPanel`, which shows the term-tree of a formula, should be an inevitable part of the representation, which is always shown

just like the ModelPanel, which displays the input parameters and conditions for the given mathematical problem. Next to these main panels the TermTreeRulePanel which is planned to optionally show an applicable rule in term-tree representation next to the active formula in term-tree representation of a formula. In the CalcHeadPanel context the SpecificationPanel exists to be able to define and show what kind of problem or method is currently worked with.

In this thesis we focus on the content of the TermTreeFormPanel, which is going to be focused first, when we selected the tree representation in the context-menu.

To get data into the JTree in the TermTreeFormPanel a new event or at least an unique combination of events have to be introduced. As mentioned in requirements section 1 there are more phases of math problem solving (modelling, specification and solving) of which the newly developed approach will take place in the solving phase (UI_SOLVE). To make use of the existing event-based communication structure, we needed to choose a unique identifier in the code, UI_SOLVE_TERM_TREE. Additionally the UI_CONTEXT_ONEELEMENT identifier is used, which is connected to a selected formula in the Worksheet. These two unique identifiers (UI_SOLVE_TERM_TREE and UI_CONTEXT_ONEELEMENT) are used to build a JInternalFrame, which contains the selected formula in a tree representation.

4.3.2 Data Formats in ISAC

When calculating, the data is retrieved by the MathEngine. The MathEngine only knows the line of calculation it gets from the Worksheet and sends back the applicable rule or the next formula in form of a Scala term (custom XML format used in Isabelle). However a Scala term is raw unusable to perform user interactions on or to be readable by a user. So a conversion algorithm was needed. The raw data as it can be found in a Scala term, has by far the most semantics and information about a selected calculation, this is why a matching algorithm was needed for the raw Scala term to become a more representative format like MathML (presentation format) or string (Java). The requirement on the output format is the need for uniform type of data that works with all representations and is accessible by assistive technologies and therefore we decided to use MathML. MathML is a common standard (details see [53]) and should be suitable for displaying a formula for sighted and visually impaired people. In addition the Scala term in string format can be usable at least for console printing and tests. Backwards compatibility must be guaranteed if changes occur in a formula.

Chapter 5

Implementation of Term Representation

5.1 Extending ISAC

The *ISAC* project is an open project which is in development since 2002 at the University of Technology in Graz. In course of time a lot of different people extended the project by diverse components (mathematics examples, graphical user interface, event management, ...), which makes the code very difficult to understand in certain cases. With the help of current developers and the *ISAC* documentation (mainly in form of scientific papers) it was possible to find a meaningful location to extend the existing *ISAC* code. Finding the extension point was difficult because *ISAC* consists of more than three Java RMI sections, which have diverse tasks in communicating with the `MathEngine` and (multiple) Desktop clients. Additionally the debugging and testing was easier when it was clear on which components to work. The components were defined by their location of the `main` method. For our extension the `WindowApplication` of *ISAC* was important because it was the main goal to extend the graphical user interface by an accessible representation of formulas.

5.1.1 Necessities/Used Technology

Windows is the preferable operating system for screen reader technologies (see section 2.1), this is why we decided to work with a Windows computer. To be able to extend *ISAC* it was necessary to install at least the following software components:

- Java 8
- Isabelle [14]
- ISAC [68]
- Eclipse for Scala 4.1.1

For developing in concerns of Accessibility you additionally need:

- NVDA [65]
- Java Access Bridge
- Braille display driver (here for HumanWare Brailliant BI 40)

Accessibility in Java

Java's policy on Accessibility can be described in rules such as [58]:

1. Always set a *name/accessibleName* for a component.
2. Make movement logical – with tabs or key down it should be possible to explore the whole user interface.
3. The focus has to be set logically for example for listeners. The focus is the control for screen reader output.
4. Insert some tooltips for better comprehension of what is in front of you.

Java Applications need the Java Access Bridge to be readable for screen reader and Braille displays on Windows. Programmatically Java Swing offers an `Accessible` Interface which is implemented by the default `JComponents`. There is a range of different implementations of the Accessibility interface such as `AccessibleAction` which is needed for example for the creation of an accessible `JButton`. An `Accessible` Object is provided when a screen reader wants to access data of a accessible `JComponent`.

The focus in a Java Swing application controls what is going to be put out on Braille displays and screen readers. A component has to be visible, enabled and focused on to be readable. Moreover the navigation must enable the visually impaired user to explore content without coming to a dead end. Normally this is made by listening on certain navigation keys and moving the focus logically through the developed Java Swing components. If a certain limit is reached (no navigation in a certain direction is possible) then it is advised to inform the user about the limit i.e. a certain sound (earcon see section 2.2.3 for explanation) is played.

Initial Problems

The ISAC project had not been installed under Windows before. The change to Windows caused problems because of differing file paths (different orientated slash as a path separator), Java permissions (policy file), missing environment variables and run configurations. When the change to Windows happened, an Isabelle (2014 to 2015) upgrade was published and was needed because of existing errors in the ISAC initialising process. This upgrade depended on Java 8, another software component ISAC had not yet considered to require at that moment. The last mentioned components made problems on both, Windows and Linux for a while.

5.1.2 Setup a Test Environment

The goals for a test environment were:

1. Isolate the test from the surrounding ISAC system as much as possible. In particular, formulas handled in the test, should not come from ISAC's mathematics-engine, the test should neither be concerned with ISAC's multi-user session management, nor with networking in the distributed system, etc.
2. Use ISAC's mechanisms for interactions on formulas. This goal is indispensable for later integration of the specific formula editors: ISAC handles formulas using Java Swing, which uses event-based communication integrated in dialog guidance.

A look at ISAC's architecture (see chapter 4) shows, that formulas are displayed and input to a `Worksheet` – so should the test (without changing the *Worksheet*). And then the problem for a test setup is, that a `Worksheet` is bi-directionally connected with a `WorksheetDialog` – so we cannot receive events from the *Worksheet* in our test, without changing code in the `Worksheet`.

The standard solution for that problem is to mock the `WorksheetDialog` so that it mimics the behaviour expected by the `Worksheet` and drops all other connections to the surrounding system. A mock is a class which implements all necessary content from the original class to bypass the existing system with manipulated data but contains the original behavior we need. Thus the communication between our test (called `TestWorksheetForMawen`) and the `Worksheet` (we do *not* want to change) is as shown in Fig.5.2.

A closer look at [20] reveals, that we need code from the `WSDialogManager` in order to start a worksheet, see Fig.5.1. The same code is needed in the mock,

MockWorksheetDialog. `TestWorksheetForMawen` gets all the code required for the Java Swing GUI (adapted from `WindowApplication`).

We need to put our example into the `CalcTree` which is shown and editable in the `Worksheet`. The `Worksheet` takes interactions from the User and the `WorksheetDialog`. The `WorksheetDialog` informs the `Worksheet` about changes (produced by the `MathEngine`) in the calculations. These changes are passed on by a CalcEvent/Message which is received in the calcChanged Method (and in the calcResponse Method). When the event is received in the `Worksheet`, the changed formula is directly fetched from the `MathEngine`. This process uses Remote Method Invocation (RMI), which we do want to ignore for our tests. Therefore we pass the lines of the example as Iterators directly into the CalcChanged Event which is passed on to the `Worksheet`. The flowchart Fig.4.2 shows the eventchain.

After the code was implemented in the test environment, the reintegration was done without problems, because the test environment was designed as close as possible to the production code. Changes in the `WorksheetDialog` and the `Worksheet` were adapted into the existing *ISAC* code. The classes, which are needed to build the `TermTreePanel`, `SoundUtils` and the key and mouse listeners were simply moved into the new package (`isac.gui.mawen.termtree`).

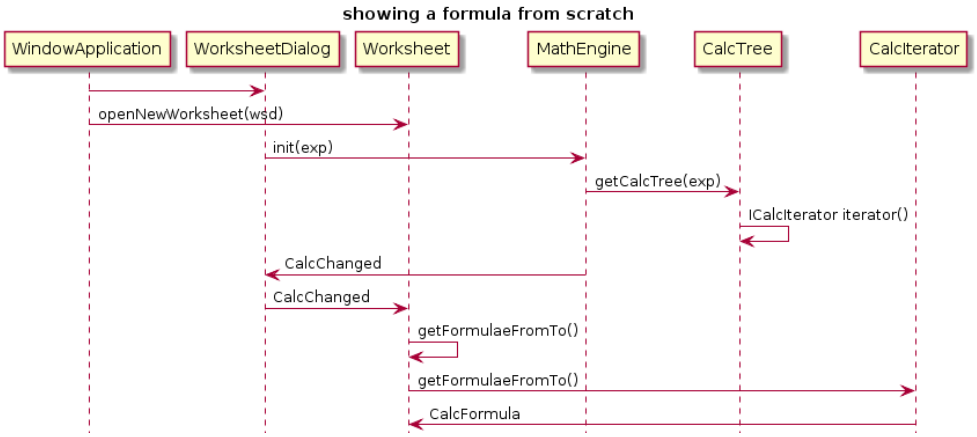


Figure 5.1: Interaction diagram for "Open a Worksheet with Formula"

5.2 Term Representation in Java and Scala

This thesis mainly addresses the Reading (see section 1.2.1) problem. One of the bigger problems concerning the readability is the two dimensional

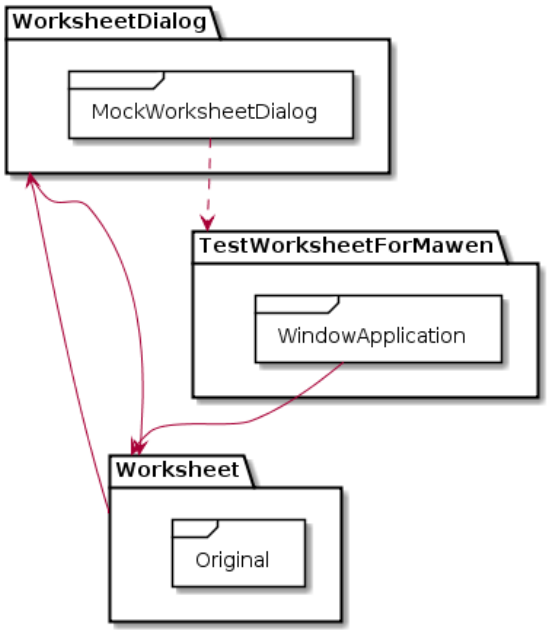


Figure 5.2: Relations between the Worksheet, WindowApplication and the WorksheetDialog in the test setting.

representation of mathematical expressions. That is why we came up with the idea of decomposing a linear formula into a tree of terms. This tree of terms should be accessible via screen reader and Braille display. Therefore the `MathEngine` had to be extended to return not only a string but also the Scala term it receives by Isabelle. The Scala term includes more semantics about a formula than a string. Additionally the existing `CalcFormula` had to be adapted to hold a term, next to a formula and its position in the tree. The attempt to save this data in MathML presentation format was successful but there were problems when it came to displaying the MathML data. A self developed parser and a renderer would have been necessary to represent the data from a MathML format, because screen reader would not be able to access content directly in MathML format. An example of the imagined MathML usage would have been embedded MathML in JavaScript (rendering representation), where MathJax [63] provides the screen reader with interactive information. The final version of the nodes had a string and term object as node-content, whereas the string was displayed.

5.2.1 Trees in Java Swing

A `JTree` is a component in the Java Swing package, which allows to represent data as a tree. Moreover a `JTree` has a `TreeModel`, that consists of nodes, which can (branches: e.g. Fruits in fig. 5.3) or cannot (leaves: e.g. Apple in fig. 5.3) be expanded when manipulating a `JTree`. To differentiate the `JTree` from the `TreeModel` in the code, is rather complicated. If a node is selected, it can either be seen as a `TreePath` which is an array of nodes or as a row number (can change if nodes collapse/expand) in the `JTree`. Whereas a `TreePath` is the unique representation of the node in the `JTree` (for example the `TreePath` of Banana in fig 5.3 is Root/Fruits/Banana) and the row number is just a row which the node is currently in (in our example Banana is in row 8). An expand function is included in the `JTree` and not in `TreeNode`. If a node should be re-read by the screen reader, an easy solution is to reassign the node itself. Other methods (such as using the `TreeModel` for refreshing the content in the tree) lead to problems with the node structure. The tree concept in Java Swing is held very generic by returning objects of type `Object` by default methods. A list of children is returned as an `Enumeration` of `Object` which leads to the necessities of iterators and a frequent use of type casts.

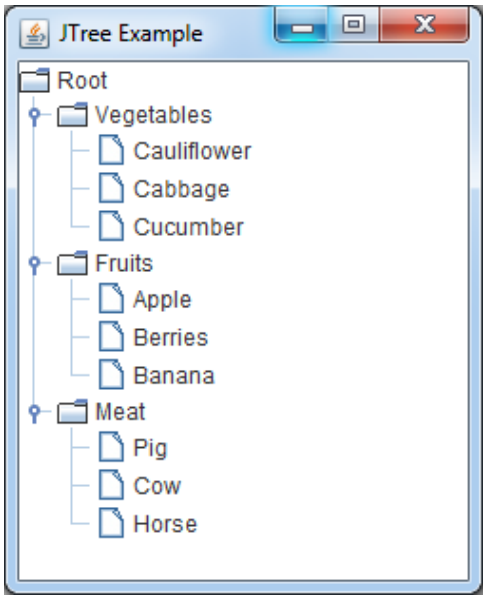


Figure 5.3: Image showing the default example of a JTree in Java Swing.

5.3 Trees for Special Needs

We thought about a tree structure which is only visible and accessible for assistive technology and a synchronised view for a regular sighted person which can set focus too. However it is not possible to simultaneously work on one computer together on two different representations (maybe one invisible), because this would mean there are two focuses. So the inclusive mechanism is cut down to sequential actions on a tree, where the blind as well as the regular sighted person can set the focus for the other person.

The newly developed tree representation enables the user to collapse and expand a node (only with branches) for better navigation and comprehension. Typically the user of the term-tree representation is visually impaired and uses the term-tree to explore a formula in different levels of decomposition.

5.3.1 Representation of Sub-Trees

When a formula is chosen in a `Worksheet` to be represented as a tree, it first appears as a simple string in one line. This representation differs from the representation of a formula in a `Worksheet` only by the visual icon next to the string and its ability to be decomposed into a binary tree of terms. Term-Trees in *ISAC* consist of nodes with at least an operator as a branch and numbers and variables as its smallest blocks. A valid term for example would be e.g. 1 + 4. In *ISAC* a term has brackets to determine its precedence when a formula contains more than one term. For visually impaired people brackets and nested structures pose a problem when facing mathematics. This problem is solved in small by introducing the decomposition of a formula into terms. The terms are displayed in different depths, which results in non-verbal information about the arity of terms (see figure 5.4). With a certain key, a node (if it is a branch) can be expanded into its main operator and the two arguments, which are shown in three subsequent lines on the screen. A simple tree example would be: 1 + 2 becoming + 1 2. To return this step, a collapse at the newly shown operator has to be performed. We chose an abstract syntax tree (prefix) over a concrete syntax (infix) because from a mathematical and programmatic point of view it is more logical to have a function (operators, sine, ...) with their arguments following (numbers, variables, ...). Moreover infix is easier to comprehend when displayed as a (right side view) tree.

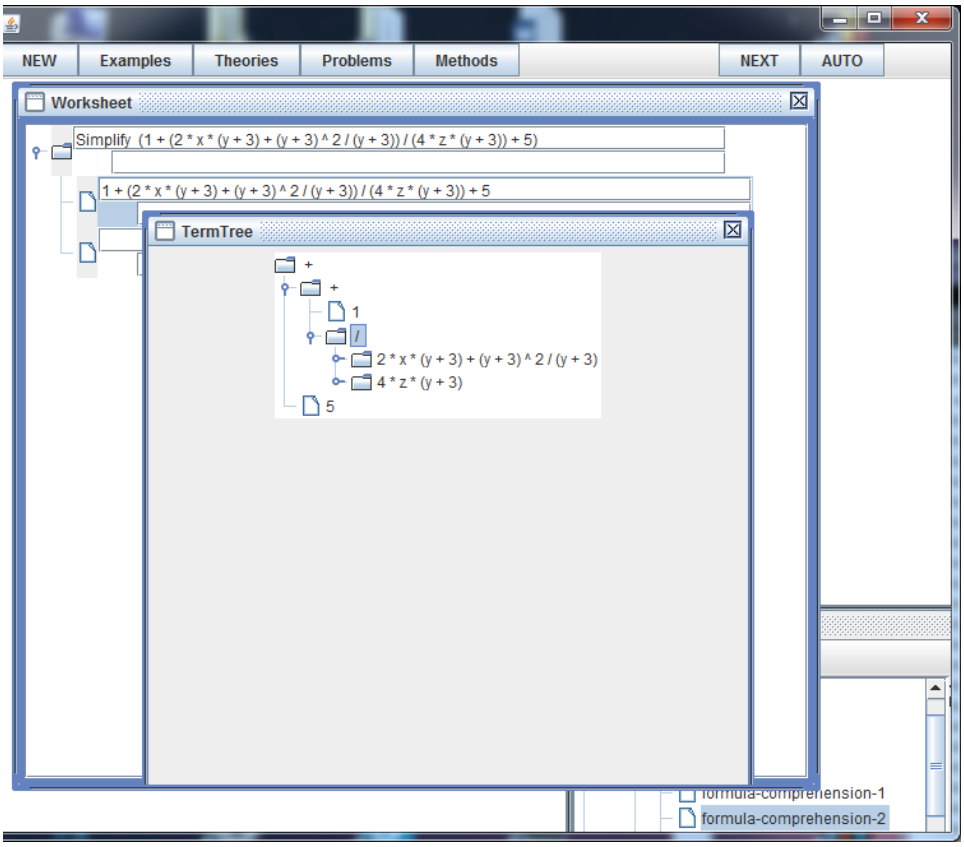


Figure 5.4: Image showing a formula in a Worksheet and in the term-tree view.

5.3.2 Auditive Representation of Trees

A new auditive model was designed because of the unsatisfying speech output of NVDA (for details see section 4.2.1). The auditive model of the tree representation of a formula is implemented in a `SoundUtils` class, which is able to produce sounds (earcons) and repeat them. A play-method in this class corresponds to an auditive model (i.e. the higher the line number of a node is, the higher is the pitch of sound). In detail the methods include an recursive algorithm to visit every node and play a chosen sound (sequence) directly to the user, when pressing the assigned key on the numeric keypad.

Used Keys in Term-Tree

Key	Function as used in the tree in the context of this thesis
NUMPAD8	UP – makes navigation in upward direction in tree possible
NUMPAD5	DOWN – makes navigation in downward direction in tree possible
NUMPAD6	EXPAND – expands branches, thus exposes the operator's sub-tree
NUMPAD4	COLLAPSE – collapses branches, thus shrinks a sub-tree to a string
NUMPAD9	AUDIO(1) – plays sound model with subsequent location marker
NUMPAD7	AUDIO(2) – plays sound model with preceding location marker

The numeric keypad keys are used because of their similarity to the arrow keys and their spatial attributes concerning additional usable keys.

Chapter 6

Evaluation

The following chapter consists of three different scenarios. Each scenario comprises different test settings and discussions, which are tested on the practical part of this thesis.

6.1 Setup of Scenarios for Evaluation

The test scenarios cover different aspects of comprehension, navigation and orientation in a term composed tree.

6.1.1 Preliminaries

The setup of the scenarios for evaluation is constrained by the following side-conditions:

Test persons were an important aspect in the evaluation of the practical part of the thesis: the implementation part of the thesis should be tested by visually impaired people – these cannot be seriously mimicked by regular sighted people. In Austria there are about 3.000 visually impaired people (legally blind), but only one person in the immediate vicinity of the JKU Linz has sufficient mathematics knowledge (see section 1.4) to be addressed by *ISAC* and the respective evaluation of the extension done in the run of this thesis.

This person is the legally blind Dipl-Ing. Stöger Bernhard [50]. The other test person is sighted mathematician Dr. Walther Neuper.

Methodology and Evaluator appear already determined by the above: the very little number of test persons induces qualitative testing. The tests are distributed over scenarios, which are separated with respect to observable variables.

Prior to the evaluation there was no introduction of the visually impaired test person. So evaluation includes usability of the implementation done in this thesis.

The evaluator is the author of this thesis. Furthermore a.Univ.-Prof. Dr. Klaus Miesenberger [60], Vice-head of IIS, attends the evaluation as an observer.

Technicalities were limited by facts introduced in chapter 2: assistive technologies are better adapted to Windows than to Linux or to Mac OS. Trying and comparing all three was out of scope of the thesis, so evaluation has been restricted to Windows. On the test machine the following components were installed:

- The *ISAC*-prototype
- Java 8
- Java Access Bridge
- HumanWare Brailliant BI 40 Braille Display
- NVDA, NonVisual Desktop Access, with these settings (due to preferences of test person):
  - Voice settings
    - Voice rate: 23
    - Punctuation/symbol level: all
    - Language: German

For further details see section 5.1.1.

6.1.2 The Scenarios for Evaluation

All evaluation within this thesis is captured within five scenarios; the scenarios are presented to the test person(s) in the sequence as given below, beginning with SCEN.1 and ending with SCEN.5.

The first scenario SCEN.1 serves to investigate the readability and navigability of the newly implemented tree-representation. SCEN. 2, 3, 4 target the readability (in form of getting a fast overview of a formula and orientating in a large formula) and the usability of the newly designed sound model of

the tree-representation. The existing *ISAC* formula editor is surveyed under the aspects of inclusion in the last SCEN. 5.

SCEN 1 ***Formula Comprehension***

Description: A large formula is presented to the visually impaired test person who should comprehend the formula. The formula is estimated incomprehensible in linear representation. The scenario combines assistive technology speech and Braille output by screen reader software NVDA, with the access to a tree of terms and sub-terms as implemented during this thesis.

Comprehension is evaluated with respect to a known difficulty in formulas (details see section 1.3), a double fraction. Comprehension is checked by the asking questions for cancellation.

Two formulas of different complexity are prepared for presentation (the reader may note, that the test person does *not* have the advantage of *seeing* the structure directly in the two-dimensional representation as below):

$$1 + \frac{2 \cdot x \cdot (y+3) + \frac{(y+3)^2}{y+3}}{4 \cdot z \cdot (y+3)} + 5$$

The formulas with less complexity is:

$$6 + \frac{2 \cdot (x+1) \cdot (y+3)}{4 \cdot z \cdot (y+3)}$$

Sequence of Actions: The evaluator has chosen an example prepared in *ISAC*. The example is located in the example-browser's hierarchy as *IsacCore > Tests > Mawen > formula-comprehension-1*. At the beginning of the test, the example is opened in an *ISAC* `Worksheet` and the focus is set to the formula.

1. The scenario starts with the simplest formula.
2. The evaluator opens the context menu of the formula and selects "Tree representation". The formula appears in the respective window, still collapsed (single line string representation as it can be found in the `Worksheet`).
3. The evaluator gives the following information:
   - You now have a representation of the formula which allows to access parts of it (sub-terms).
   - For that purpose you get a tree, where the operators $(+, -, /, \dots)$ are the branching points: $a + b$ can be expanded to three lines, where the $+$ is followed by a and b on subsequent lines.

- Each sub-term can be expanded by EXPAND key (see section 5.3.2) and collapsed by the COLLAPSE key.
- Then each of the lines opened can be reached by the DOWN and UP keys.
- Any line can be collapsed and expanded (in case there are sub-terms)

4. Then the evaluator asks: "Do you see whether the formula comprises a sum of terms or a double fraction?"
5. "Do you see whether the fraction can be reduced?"
6. If no answer is given then the next question is: "Do you see whether $y + 3$ can be cancelled?"
7. Independent from the answers the more complex formula (example *formula-comprehension-2*) is checked by the questions from Pt.4 to Pt.6
8. Finally the evaluator asks the test person: "What opinions, suggestions, criticisms do you have now at the end of this scenario?"

SCEN 2 ***Understandability of Audio Information***

Description: This scenario addresses the question, if the audio information according to section 5.3.2 can be understood by a visually impaired person without any additional information. The audio information is intended to give a survey on the formula (displayed as an abstract tree of terms).

Three different formulas are represented one after the other:

```
1+(2+(3+(4+5)))
+
  1
  +
    2
    +
      3
      +
        4
        5
```

```
(((1+2)+3)+4)+5
+
  +
    +
      +
        1
        2
      3
    4
  5
```

```
(1+2+3)*(4+5+6)
*
  +
    1
    +
      2
      3
  +
    4
    +
      5
      6
```

Sequence of Actions: Again, *ISAC* `Worksheets` are opened with the focus set to the respective formula from above in the example "formula-audio-1", "formula-audio-2" and "formula-audio-3".

1. The scenario starts with the first formula.

2. The evaluator opens the context menu of the formula and selects "Tree representation". The formula appears in the respective window, still collapsed.
3. The evaluator expands the tree completely and sets the focus to the tree's root.
4. The evaluator instructs about the meaning of what is presented: "You get auditive information on a tree; this tree represents a formula. The exact content of the formula is not important in this scenario. Important is: each click denotes a node in the tree and the pitch denotes the indentation in the tree."
5. The evaluator activates the AUDIO(1) key as often as required for the test person to answer the following questions sufficiently:
6. Question 1: "Can you tell the number of nodes in the tree?" (for sighted people: number of lines in the tree).
7. Question 2: "Can you tell the maximal depth in the tree?"
8. After the answers the steps are repeated with the next formula.
9. Finally the evaluator asks the test person: "What opinions, suggestions, criticisms do you have now at the end of this scenario?"

SCEN 3 ***Orientation in Large Formulas***

Description: This scenario continues the previous SCEN.2 and emphasises the tree interpreted as a formula. The questions if audio can help comprehending the structure of the formula is researched in this scenario.

The third formula from SCEN.2 is represented again:

```
(1+2+3)*(4+5+6)
*
  +
    1
    +
      2
      3
  +
    4
    +
      5
      6
```

Sequence of Actions: An *ISAC* `Worksheet` is opened with the example "formula-audio-3" and the focus set on the respective formula.

1. The evaluator opens the context menu of the formula and selects "Tree representation". The formula appears in the respective window, still collapsed.

2. The evaluator instructs about the AUDIO(1) key.
3. The test person experiments with expanding and collapsing sub-terms and activates the AUDIO(1) key as often as required to answer the following questions:
4. Question: "Can you tell, how the audio information is related to what you read on the Braille display?"
5. The AUDIO(2) key is introduced if the function of AUDIO(1) is understood by the test person.
6. Finally the evaluator asks the test person: "What opinions, suggestions, criticisms do you have now at the end of this scenario?"

SCEN 4 ***Formula Comprehension with Audio Information*** :

Description: This scenario is similar to SCEN.1 and offers additional audio information. The question is, if the audio information helps to comprehend the content of formulas.

The formulas are different from the previous ones, but comparable in complexity of structure:

$$6 + \frac{(1 + 3 \cdot b) \cdot (2 + a)}{4 \cdot c \cdot (2 + a)}$$

$$5 + \frac{\frac{4+4 \cdot a+a^2}{2+a} + 3 \cdot b \cdot (2 + a)}{4 \cdot c \cdot (2 + a)} + 1$$

Sequence of Actions: *ISAC* `Worksheet`s are opened with the focus set to the respective formula from above in the example "audio-comprehend-1" and "audio-comprehend-3".

1. The evaluator instructs the test person how to change the focus from one formula to the other and how to open and close the window with the tree representation.
2. The test person is suggested to start with the first formula and open the window for the respective tree representation. The formula appears in the window, still collapsed (an thus represented as the same string as in the `Worksheet`).
3. Then the evaluator asks: "Do you see whether the formula comprises a sum of terms or a double fraction?"
4. "Do you see whether the fraction can be reduced?"
5. If no answer is given then the next question is: "Do you see whether $y + 3$ can be cancelled?"
6. Independent from the answers, the more complex formulas is checked by the questions from Pt.3 to Pt.4.

7. Finally the evaluator asks the test person: "What opinions, suggestions, criticisms do you have now at the end of this scenario?"

SCEN 5 ***Inclusive Discussion of a Calculation***

Description: A visually impaired test person and a regular sighted person discuss a sequence of formulas with respect to simplification of fractions on an *ISAC* `Worksheet` (whereas the sighted person already knows how to use *ISAC*). The sequence is a calculation in which formulas can be simplified stepwise.

For navigational purpose the mouse is used by the sighted person to move from one formula within the calculation to another — while the visually impaired person uses the Braille display and the keyboard. The two test persons discuss steps of simplification, in particular cancellation. In *ISAC* sequences in a calculation are determined by the `MathEngine`, that is why we could not predefine this example in detail for the evaluation. Given the same example, the sequence in the *ISAC* version is similar in difficulty as in the following sequence:

$$1 + \frac{2 \cdot x \cdot (y+3) + \frac{y^2+6 \cdot y+9}{y+3}}{4 \cdot z \cdot (y+3)} + 5 \tag{6.1}$$

$$1 + \frac{2 \cdot x \cdot (y+3) + \frac{(y+3)^2}{y+3}}{4 \cdot z \cdot (y+3)} + 5 \tag{6.2}$$

$$1 + \frac{2 \cdot x \cdot (y+3) + (y+3)}{4 \cdot z \cdot (y+3)} + 5 \tag{6.3}$$

$$1 + \frac{(2 \cdot x + 1) \cdot (y+3)}{4 \cdot z \cdot (y+3)} + 5 \tag{6.4}$$

$$1 + \frac{2 \cdot x + 1}{4 \cdot z} + 5 \tag{6.5}$$

$$6 + \frac{2 \cdot x + 1}{4 \cdot z} \tag{6.6}$$

In contrast to step 6.6 in the sequence, *ISAC*s result looks like this:

$$\frac{1 + 2 \cdot x + 24 \cdot z}{4 \cdot z}$$

Sequence of Actions: The evaluator has chosen the example "formula-comprehension-3". This shows the above formula 6.1 on the `Worksheet`. The first formula is in the focus.

1. The sighted person starts discussion with the question "Can the fraction be simplified beyond the result at the bottom?"
2. The discussion between the two test persons proceeds dealing with questions like:
   - "In which step does y leave the sequence?"
   - The visually impaired person has to find certain number.
   - The sighted person shows a certain sequence.
   - The sighted person shows some features of *ISAC*.
   - etc.
3. Finally the evaluator asks the test person: "What opinions, suggestions, criticisms do you have now at the end of this scenario?"

6.2 Results of Scenarios for Evaluation

This chapter comprises qualitative testing on a developed practical part of this thesis, that is why the analysis won't rely on statistics or tables of data but uses discussion as an instrument of describing the evaluation process and the ensuing results. Every scenario is analysed due to comments by the involved persons and observations the evaluator makes.

Results SCEN.1: Formula Comprehension

On the first example of the first scenario the visually impaired person did not use the developed tree representation of terms. According to this fact, the test person was fast in giving a correct answer for the first question but missed the second question of cancellation by responding with number (2) and not term related $(y + 3)$ cancellation. Due to his prior routine in reading maths in linear representation the test person was faster using the Braille display. The test person tried to use linear representation only on the second example too. After a while trying to comprehend the formula, he was given the suggestion to use the tree mechanism. With the help of the tree representation the test person figured out the structure of the formula rather quickly and could give correct answers.

During the evaluation of this scenario, it became clear that the tree representation is useful for reading and comprehending if you have a rather nested expression which can only be isolated with a high amount of concentration. The visually impaired person stated, that in relation navigating mechanisms there is more feedback needed, if there was no possibility to expand/collapse on certain nodes. Additionally the visually impaired needed a key for stepping out of nested sub-terms, which was previously assumed to be covered

by the existing keys on the tree (primarily UP and DOWN in this case). The key assignment was assessed as unsymmetrical and improvable.

The speech output of NVDA was not used at all. This could be due to the fact, that visually impaired people who have early access (financially problematic) to Braille displays are more adapted to Braille output and other visually impaired people rather use screen reader output when working [39]. Also the test person stated, that Braille is much more comparable to the sighted people's optical perception, e.g. because of random access in reading.

Overall it was easy for the test person to learn how the tree representation works. The visually impaired person liked the newly designed collapse and expand possibility. In connection with a formula, a mind map could be created and that made it more practically in use than the linear representation.

Results SCEN.2: Understandability of Audio Information

On the first example of the first scenario the visually impaired person answered correctly in terms of amount and depth of nodes. After few replays (1 to max. 3) of the sound model of the tree representation, the visually impaired person gave answers to the question on the amount and depth of the nodes in the second and third example. The amount of nodes in the tree had always been answered correctly and the depth was scored with a maximum deviation of two. The clicks were assessed to be too fast and extra difficult to comprehend with trees not looking like the first in this scenario (linear). Moreover most difficult example in this scenario was claimed to be "formula-audio-3" and was rated "only guessable" when trying to find out the depth of the tree. Despite this fact, the test person knew the exact amount of nodes and the perceived depth was at an acceptable level.

We came to positive results looking at the given answers in the course of this scenario. However the advantages in getting a quick overview (readability) of the sound model as a standalone solution was doubted, because the test person saw no reason to know the depth of a formula on its own. The scenario was left with a rather negative attitude towards using a sound model to get a glimpse at the structure of a formula represented as a tree.

Results SCEN.3: Orientation in Large Formulas

The trials of the visually impaired person using Braille display and the developed sound model of the tree representation showed that it is not usable without any further instructions than the information given in SCEN. 2. The semantics, which included the longer click as a marker for the focused location in the tree and faster clicks in a row for hidden nodes in a collapsed

branch, were not identified. An orientation in the tree was hardly possible for the visually impaired person.

We used the last example formula of SCEN. 2 in this scenario. Unfortunately it was already rated as difficult in the previous scenario 2. When we created the sound model we didn't consider that Mathematics and sound perception in combination could be a fundamental problem. As a clarification Miesenberger [27] was referring us to cognitive issues when both, Mathematics and sound, inputs are received at the same time.

Results SCEN.4: Formula Comprehension with Audio Information

This scenario was decided to be skipped during the evaluation process of SCEN. 3. It was interpreted as not solvable after the clarification in SCEN. 3 and because it is considered more challenging than the previous scenarios.

Results SCEN.5: Inclusive Discussion of a Calculation

The visually impaired person is not able to navigate through the formula editor on his own, because container information instead of the formula content was shown in the Braille display. The sequence's formulas were shown to the visually impaired test person by clicking the formula (edit mode). Text markings in the formula made it difficult for the visually impaired person to be able to read the formula.

*ISAC*s formula editor is still inaccessible. After this thesis the possibility for accessibility is given only for single formulas and their representation. This was shown for example by dead-ends in navigation in the formula editor and inaccessible browser windows (containing information on problems, theories, ...) etc. .

Chapter 7

Summary

There are about 3,000 legally blind people in Austria. Only few of them will be able to pursue an academic career, not only because of their preferences but also because teaching and learning is not always made fully accessible. Especially mathematics is an important subject for our daily life but represent one of the bigger challenges for visually impaired people. Sighted people often use spatial information and symbols, which cannot be displayed on assistive technology software and hardware such as screen reader and Braille displays. Unfortunately there is no general solution in literature or existing prototypes at this point. As there have already been many prototypes in this area, we wanted to extend the existing open source project ISAC, which is interested in providing an inclusive way (visually impaired together with sighted people) of working with their system.

That is why we came up with showing a formula in tree representation. A tree is a familiar navigable structure and is used in the context of this thesis to display a formula as a tree of terms. Moreover the tree provides collapse and expand functions to be able to isolate difficult parts of a formula, which can easier be understood represented as a shorter term. We relied on screen reader default output and were not fully satisfied with the speech output, that was delivered during the first trials of the term-tree. As a consequence an auditive model was developed to replace the speech output and solely work with the Braille display. The auditive model used earcons (different pitches of sound and repetition of sounds) for playing the node structure in a formula represented as a tree.

In the last section of this thesis an evaluation was conducted with a visually impaired mathematician and a sighted mathematician. This evaluation showed that the term-tree representation is useful in finding answers to ques-

tions about structure and content of a complex formula in connection with a screen reader and Braille display. In the second scenario we removed the speech output and replaced it by the developed auditive model, which was not as intuitive in use as the term-tree with speech. Doing math and hearing the earcons were not possible to be processed at the same time. Unfortunately the test person was not able to take advantage of the auditive model for formulas represented as a term-tree.

7.1 Conclusion

The key point of this thesis was positively received: problems with two dimensional representations were made accessible to visually impaired people. The term-tree and its features were well received. Especially the collapse and expand function and the decomposition of a formula into terms were liked by our test person. In contrast the auditive model was proven unintuitive to handle and was not usable in connection with doing math. Probably a tactile feedback of the tree's structure (e.g. with indentations on Braille display) could pose a bigger benefit to visually impaired people [27]. Another alternative would be to use spearcons in connection with hot keys as an opportunity for orientation [27]. Anyways in all alternatives a short and merely nested formula will probably stay more comprehensible to a visually impaired people in one dimensional (string) representation.

The last scenario in the evaluation process showed that *ISAC* needs more accessibility introduced into its graphical user interface, because the newly integrated features (tree representation and auditive model) are not sufficient to be fully accessible. As a first step it is most important to make the formula editor in the `Worksheet` accessible.

7.2 Future Work

The *ISAC*'s formula editor must increase accessibility. Although the formula editor mainly consists of a tree representation of calculation sequences, the nodes consist of a custom class which include a cell made out of two components. Screen reader and Braille display have a problem with custom cells and cannot read the content until their content is marked as editable (cursor symbol). Perhaps this problem is solved solely by introducing an `AccessibleContext` to this custom class. In order to get more accessibility into the *ISAC* graphical interfaces the framework JavaFX should be considered, because it introduces new meaningful concepts like better support for assistive devices and technologies such as screen reader

software and it is integrable in existing Java Swing applications. If the accessibility is given the next step should be to implement an inclusive opportunity to work with each other on separate computers. This could be done by using the existing RMI which *ISAC* already has used before in class room test situations. Additionally concepts for synchronous manipulation are needed to enable an unproblematic synchronous view of the formula editor on more than one computer at the same time. The interface of the formula editor needs different representation of formulas for the inclusive scenario, in which the implemented code during this thesis can be used as an option to show a linear string representation of a formula as a term-tree for better readability and comprehensibility.

An interesting contribution of Prof. Miesenberger [27] after the evaluation was the question of orientation of the tree for sighted students in case of an inclusive scenario: a pedagogically relevant activity could be to construct trees for both, for sighted and for visually impaired students. Such construction may be more intuitive with an actual tree than with the traditional graphical representation of formulas. When working on a tree, the orientation is an additional design issue: should the tree grow up from the root at bottom (the biological view), from the root at top down branch by branch (the view of computer science) or from left to right (like in our tree representation – the file-browser view). Moreover a concept of blocks of terms as represented in the work of Sorge et al. [40] can be combined with place markers such as presented by [39] would be an interesting match. Visually impaired people would be offered the chance to have or set a meaningful chance to switch context.

References

Literature

[1] Hinz Andreas. *Zeitschrift für Heilpädagogik 53 (9)* (2002), (cit. on p. 1).

[2] Patricia Apkarian-Stielau and Jack M Loomis. "A comparison of tactile and blurred visual form perception". *Perception & Psychophysics* 18.5 (1975), pp. 362–368 (cit. on p. 16).

[3] Dominique Archambault. "Non visual access to mathematical contents: State of the art and prospective". In: *Proceedings of the WEIMS Conference*. 2009, pp. 43–52 (cit. on pp. 2, 3, 17).

[4] Dominique Archambault et al. "A Software Model to Support Collaborative Mathematical Work Between Braille and Sighted Users". In: *Proceedings of the 9th Interonal ACM SIGACCESS Conference on Computers and Accessibility*. Assets '07. Tempe, Arizona, USA: ACM, 2007, pp. 115–122. URL: http://doi.acm.org/10.1145/1296843.1296864 (cit. on pp. 3, 5, 17, 18).

[5] Dominique Archambault et al. "Mathematical working environments for the Blind: what is needed now?" *Interaction et usages des modalités non visuelles, accessibilité des contenus complexes* (2007), p. 36 (cit. on pp. 2, 3, 5, 18, 22, 30).

[6] Dominique Archambault et al. "The universal maths conversion library: an attempt to build an open software library to convert mathematical contents in various formats". In: *Proceedings of 3nd International Conference on Universal Access in Human-Computer Interaction (joint with HCI International 2005), Las Vegas, Nevada, USA*. Vol. 9. 2005 (cit. on pp. 14, 15).

[49] Statistik Austria. *Daten vom Mikrozensus im vierten Quartal 2007*. 2008 (cit. on p. 5).

[7] Catherine M Baker, Lauren R Milne, and Richard E Ladner. "Structjumper: A tool to help blind programmers navigate and understand the structure of code". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM. 2015, pp. 3043–3052 (cit. on pp. 18, 19).

[8] Robert F Cohen, Arthur Meacham, and Joelle Skaff. "Teaching graphs to visually impaired students using an active auditory interface". In: *ACM SIGCSE Bulletin*. Vol. 38. 1. ACM. 2006, pp. 279–282 (cit. on p. 19).

[9] Robert F Cohen et al. "PLUMB: displaying graphs to the blind using an active auditory interface". In: *Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility*. ACM. 2005, pp. 182–183 (cit. on pp. 18, 19).

[10] Gabriella Daróczy. "Cognitive Aspects of Designing Dialogues in Theorem-Prover Based Mathematics Assistants; Implementation of Error-Patterns Guiding Dialogues in ISAC". http://www.ist.tugraz.at/projects/isac/publ/gdaroczy-ma.pdf. MA thesis. Vienna, Austria: ME:CogSci, 2013 (cit. on p. 28).

[11] Gabriella Daróczy and Walther Neuper. "Error-Patterns within "Next-Step-Guidance" in TP-based Educational Systems". In: vol. 7. 2. Special Issue "TP-based Systems and Education". Feb. 2013, pp. 175–194. URL: https://php.radford.edu/~ejmt/ContentIndex.php#v7n2 (cit. on p. 28).

[12] Alistair DN Edwards, Heather McCartney, and Flavio Fogarolo. "Lambda:: a multimodal approach to making mathematics accessible to blind students". In: *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility*. ACM. 2006, pp. 48–54 (cit. on pp. 18, 19).

[13] Alistair DN Edwards and Robert D Stevens. "A multimodal interface for blind mathematics students". *INSERM'94* (1994), pp. 97–104 (cit. on p. 14).

[14] *Generic proof assistant "Isabelle"*. http://isabelle.in.tum.de/ (cit. on pp. 22, 27, 37).

[15] Carole Goble, Simon Harper, and Robert Stevens. "The travails of visually impaired web travellers". In: *Proceedings of the eleventh ACM on Hypertext and hypermedia*. ACM. 2000, pp. 1–10 (cit. on p. 17).

[16] Jon Gunderson and R Mendelson. "Usability of world wide web browsers by persons with visual impairments". In: *Proceedings of the RESNA Annual Conference*. 1997, pp. 330–332 (cit. on p. 17).

[17] Joshua Hailpern et al. “Web 2.0: blind to an accessible new world”. In: *Proceedings of the 18th international conference on World wide web*. ACM. 2009, pp. 821–830 (cit. on p. 15).

[18] Marion Hersh. “MATHS FOR BLIND PEOPLE: NOT REINVENTING THE WHEEL... CROOKED” (2015). presented at IMA International Conference on Barriers and Enablers to Learning Maths: Enhancing Learning and Teaching for All Learners 2015 (cit. on pp. 2, 3, 5, 17).

[19] A. I. Karshmer et al. “UMA: A System for Universal Mathematics Accessibility”. In: *Proceedings of the 6th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets ’04. Atlanta, GA, USA: ACM, 2004, pp. 55–62. URL: http://doi.acm.org/10.1145/1028630.1028642 (cit. on pp. 9, 18).

[20] M. Kienleitner. “Towards “NextStep Userguidance” in a Mechanized Math Assistant”. http://www.ist.tugraz.at/projects/isac/publ/mkienl_bakk.pdf. MA thesis. IICM, Graz University of Technology, 2012 (cit. on pp. 28, 38).

[21] Franz Kober. “Logging of High-Level Steps in a Mechanized Math Assistant”. Bakkalaureate Thesis. MA thesis. IICM, Graz University of Technology, 2012 (cit. on p. 28).

[22] Alan Krempler. “Architectural Design for Integrating an Interactive Dialogguide into a Mathematical Tutoring System”. http://www.ist.tugraz.at/projects/isac/publ/da-krempler.pdf. MA thesis. Graz, Austria: University of Technology, Institute for Softwaretechnology, Mar. 2005 (cit. on p. 28).

[23] Alan Krempler and Walther Neuper. “Formative Assessment for User Guidance in Single Stepping Systems”. In: *Interactive Computer Aided Learning, Proceedings of ICL08*. Ed. by Michael E. Aucher. http://www.ist.tugraz.at/projects/isac/publ/icl08.pdf. Villach, Austria, 2008 (cit. on p. 28).

[24] John Lamping, Ramana Rao, and Peter Pirolli. “A focus+ context technique based on hyperbolic geometry for visualizing large hierarchies”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co. 1995, pp. 401–408 (cit. on pp. 25, 31).

[25] Aineias Martos et al. “Towards the 8-Dot Nemeth Braille Code”. English. In: *Computers Helping People with Special Needs*. Ed. by et al. (Eds) Miesenberger. Vol. 8547. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 533–536. URL: http://dx.doi.org/10.1007/978-3-319-08596-8_83 (cit. on p. 14).

[26] Lukáš Másilko and Jiří Pecl. "Making Graph Theory Algorithms Accessible to Blind Students". English. In: *Computers Helping People with Special Needs*. Ed. by Klaus Miesenberger et al. Vol. 8547. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 549–556. URL: http://dx.doi.org/10.1007/978-3-319-08596-8_86 (cit. on p. 18).

[27] Klaus Miesenberger. personal communication. Information according to the evaluation meeting on (2016.08.09). Linz, Austria (cit. on pp. 54, 56, 57).

[28] Walther Neuper. "Automated generation of user guidance by combining computation and deduction". *arXiv preprint arXiv:1202.4832* (2012) (cit. on p. 7).

[29] Walther Neuper. "Lucas-Interpretation from Users' Perspective". In: *submitted to CICM*. http://www.ist.tugraz.at/projects/isac/publ/lucin-user-view.pdf. Bialystok, Poland, July 2016 (cit. on p. 22).

[30] Walther Neuper. "Rigor of TP in Educational Engineering Software". In: *submitted to CICM*. http://www.ist.tugraz.at/projects/isac/publ/tp-engin-sw.pdf. Bialystok, Poland, July 2016 (cit. on p. 22).

[31] Walther A. Neuper. "Reactive User-Guidance by an Autonomous Engine Doing High-School Math". http://www.ist.tugraz.at/projects/isac/publ/wn-diss.ps.gz. PhD thesis. Technical University, A-8010 Graz: IICM - Inst. f. Softwaretechnology, 2001 (cit. on p. 28).

[32] Tobias Nipkow, Lawrence C Paulson, and Markus Wenzel. *Isabelle/HOL: a proof assistant for higher-order logic*. Vol. 2283. Springer Science & Business Media, 2002 (cit. on p. 6).

[33] Lawrence C. Paulson and Jasmin C. Blanchette. *Three Years of Experience with Sledgehammer, a Practical Link between Automatic and Interactive Theorem Provers*. http://people.mpi-inf.mpg.de/~jblanche/iwil2010-sledgehammer.pdf. 2010 (cit. on p. 22).

[34] Daniel Pöll. personal communication. Information according to introduction of a working environment of visually impaired people by Daniel Pöll at the IIS. Linz, Austria (cit. on p. 17).

[35] TV Raman. "Audio system for technical readings". PhD thesis. Cornell University, 1994 (cit. on p. 19).

[36] John Riser. "Gentzen Gerhard. Investigations into logical deduction. English translation of 4422 by Szabo M. E.. American philosophical quarterly, vol. 1 (1964), pp. 288–306, and vol. 2 (1965), pp. 204–218.Bernays Paul. Introduction. Therein, vol. 1, p. 288." *The Journal of Symbolic Logic* 35 (01 Mar. 1970), pp. 144–145. URL: http://journals.cambridge.org/article_S0022481200092604 (cit. on p. 22).

[37] Ann C. Smith et al. “Nonvisual tool for navigating hierarchical structures.” In: *ASSETS*. Ed. by Julie A. Jacko and Andrew Sears. ACM, Feb. 13, 2006, pp. 133–139. URL: http://dblp.uni-trier.de/db/conf/assets/assets2004.html#SmithCFHAR04 (cit. on pp. 3, 18).

[38] Josh Smith. “PATTERNS-WPF Apps With The Model-View-ViewModel Design Pattern”. *MSDN magazine* (2009), p. 72 (cit. on p. 9).

[39] Neil Soiffer. “A Study of Speech Versus Braille and Large Print of Mathematical Expressions”. In: *Computers Helping People with Special Needs: 15th International Conference, ICCHP 2016, Linz, Austria, July 13-15, 2016, Proceedings, Part I*. Ed. by Klaus Miesenberger, Christian Bühler, and Petr Penaz. Cham: Springer International Publishing, 2016, pp. 59–66. URL: http://dx.doi.org/10.1007/978-3-319-41264-1_8 (cit. on pp. 13, 53, 57).

[40] V. Sorge et al. “Towards making mathematics a first class citizen in general screen readers”. In: *11th Web for All Conference*. Seoul, Korea: ACM Press, June 2014, pp. 43–52 (cit. on pp. 21, 57).

[41] Bernhard Stöger. personal communication. Information according to the meeting on (2016.07.01) with Dipl-Ing. Stöger, Bernhard after first prototype was made. Linz, Austria (cit. on pp. 13, 21).

[42] Bernhard Stöger and Klaus Miesenberger. “Accessing and Dealing with Mathematics as a Blind Individual: State of the Art and Challenges”. *Enabling Access for Persons with Visual Impairment* (2015), p. 199 (cit. on pp. 2, 6, 13, 14, 19).

[43] Jennifer Sutton. *A Guide to Making Documents Accessible to People who are Blind Or Visually Impaired*. American Council of the Blind, 2002 (cit. on pp. 17, 18).

[44] Masakazu Suzuki et al. “INFTY: an integrated OCR system for mathematical documents”. In: *Proceedings of the 2003 ACM symposium on Document engineering*. ACM. 2003, pp. 95–104 (cit. on pp. 14, 15).

[45] Poonam Syal, Shantanu Chatterji, and Harish Kumar Sardana. “DiGVis: A system for comprehension and creation of directed graphs for the visually challenged”. *Universal Access in the Information Society* 15.2 (2016), pp. 199–217. URL: http://dx.doi.org/10.1007/s10209-014-0387-7 (cit. on p. 18).

[46] Bruce N. Walker, A Nance, and Jeffrey Lindsay. “Spearcons: speech-based earcons improve navigation performance in auditory menus”. In: *In Proceedings of the International Conference on Auditory Display*. 2006, pp. 95–98 (cit. on p. 19).

Online sources

[47] *Accessibility Principles in the W3C Standard*. URL: https://www.w3.org/WAI/intro/people-use-web/principles (visited on 09/07/2016) (cit. on p. 15).

[48] *Adapted figure of statistics on disabled (including visually impaired) people*. URL: http://webaim.org/projects/screenreadersurvey6/ (visited on 09/07/2016) (cit. on p. 12).

[50] *Bernhard Stöger at JKU Linz, IIS page*. URL: http://www.jku.at/iis/content/e33874/index_html?team_view=section&emp=e33874/employee_groups_wiss33871/employees33870 (visited on 09/07/2016) (cit. on pp. 5, 45).

[51] *ChromeVox Homepage*. URL: http://www.chromevox.com/ (visited on 09/07/2016) (cit. on p. 15).

[52] *Definition of HTML in the W3C Standard*. URL: https://www.w3.org/standards/webdesign/htmlcss (visited on 09/07/2016) (cit. on p. 15).

[53] *Definition of MathML in the W3C Standard*. URL: https://www.w3.org/Math/ (visited on 09/07/2016) (cit. on pp. 14, 15, 35).

[54] *Definition of WAI-ARIA in the W3C Standard*. URL: https://www.w3.org/WAI/intro/aria (visited on 09/07/2016) (cit. on p. 15).

[55] *Definition of WCAG in the W3C Standard*. URL: https://www.w3.org/WAI/intro/wcag (visited on 09/07/2016) (cit. on p. 15).

[56] *Gecko Wiki*. Documentation on the Gecko engine. URL: https://wiki.mozilla.org/Gecko:Home_Page (visited on 09/07/2016) (cit. on p. 15).

[57] *Homepage of the GeoGebra Project*. URL: https://www.geogebra.org/ (visited on 09/07/2016) (cit. on p. 19).

[58] *Java documentation on Accessibility*. URL: https://docs.oracle.com/javase/tutorial/uiswing/misc/access.html (visited on 09/07/2016) (cit. on p. 37).

[59] *JAWS Homepage*. URL: http://www.freedomscientific.com/Products/Blindness/JAWS (visited on 09/07/2016) (cit. on p. 11).

[60] *Klaus Miesenberger at JKU Linz, IIS page*. URL: http://www.jku.at/iis/content/e33874/index_html?team_view=section&emp=e33874/employee_groups_wiss33696/employees33695 (visited on 09/07/2016) (cit. on p. 46).

[61] *LaTeX project Homepage*. URL: https://www.latex-project.org/ (visited on 09/07/2016) (cit. on p. 14).

[62] *Market share of desktop operating systems*. NetMarketShare. URL: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0 (visited on 09/01/2016) (cit. on p. 11).

[63] *MathJax Homepage*. URL: https://www.mathjax.org/ (visited on 09/07/2016) (cit. on pp. 15, 16, 40).

[64] *Mathplayer Homepage*. URL: https://www.dessci.com/en/products/mathplayer/ (visited on 09/07/2016) (cit. on p. 15).

[65] *NVDA Homepage*. URL: http://www.nvaccess.org/ (visited on 09/07/2016) (cit. on pp. 11, 13, 37).

[66] *Types of MathML*. URL: https://www.w3.org/TR/mathml-types/ (visited on 09/07/2016) (cit. on p. 15).

[67] *VoiceOver Documentation*. URL: https://www.apple.com/at/voiceover/info/guide/ (visited on 09/07/2016) (cit. on p. 15).

[68] *ISAC Wiki*. Documentation and source files of open source project *ISAC*. URL: http://www.ist.tugraz.at/isac/Main_Page (visited on 09/07/2016) (cit. on pp. 6, 7, 26, 27, 37).

[69] *Window-Eyes Homepage*. URL: http://www.window-eyes.at/ (visited on 09/07/2016) (cit. on p. 11).

[70] *WIRIS Homepage*. URL: http://www.wiris.com/en/editor (visited on 09/07/2016) (cit. on p. 20).

[71] *ZoomText Homepage*. URL: http://www.zoomtext.at/ (visited on 09/07/2016) (cit. on p. 11).