# FrameIT:An User Interface for Applying Mathematical Theories to Real World Problems

**Daniel Rachev**

*Computer Science*
*Jacobs University Bremen*
*Campus Ring 1*
*28759 Bremen*
*Germany*

*Type: Guided Research Proposal*
*Date: October 15, 2012*
*Supervisor: Prof. Dr. M. Kohlhase*

---

## Abstract

This thesis aims to outline the FrameIT method as a tool of managing mathematical knowledge and presenting it to the user. Using MMT, a scalable, foundation independent mathematical knowledge management system we provide a graphical interactive application that utilizes background theories to express the various existing solutions to mathematical problems. The interface aims to provide an easier way for people without mathematical background to grasp various concepts, stored in MMT as theories. Furthermore, we give the users the opportunity to apply these concepts into real world problems and show the numerous ways these problems can be approached.

# Contents

# 1 Introduction

Many aspects of our everyday work require knowledge and understanding of the underlying mathematical concepts and theories. This knowledge in most cases can be acquired in one of two ways - studying technical documents or via personal instruction. Personal instruction while in many cases can be the most effective way of learning, is very inefficient and can often be inconvenient, expensive and most importantly very hard to scale. Learning from technical documents, on the other hand, can be very scalable because of the abundance and availability of learning materials. However, there are a number of problems associated with these documents. The level of detail required for that medium increases the complexity of the concepts explained and makes them harder to grasp for the reader. Another problem is the fact that these documents are highly theoretical and difficult to apply to practical situations. Even tough they sometimes present applications and practice problems these scenarios are often static and do not provide the opportunity to understand the mappings between the learning objects and the particular situation. Furthermore the style and the structure of a large portion of the mathematical literature in many cases struggles to hold the interest of the reader. Since people have affinities for different ways of acquiring information the relatively rigid structure makes it hard for technical documents to appeal to everyone.

Overall, the readers issues can be divided into two parts: problems with motivation and problems with learning and application. The **FrameIT**[12] method was developed to deal with the second type of issues. It generates mappings between application scenarios and graph - structured representation of a set of theories. The term "frame" has been used in Communication Research [16] and is understood as *"scaffolding of concepts that influence the understanding of a situation"* In this thesis we extend the technical realization of frames [6] and apply them to various learning situations and the respective theories stored in a mathematical knowledge management system MMT[14].

Furthermore, we handle the motivational problems by providing a User Interface that delivers information to the user in another way. Presenting mathematical theories through direct application in real world problems addresses the problem of complexity and structure of technical documents more precisely described in [4]. The User Interface will also aim towards providing the user with the chance to continuously apply their knowledge surpassing the initial experience threshold needed to make the problem solving process implicit.

The thesis is organized as follows: In section 2, we show previous attempts of a similar project - ActiveMath [11]. In section 3 we provide descriptions of MMT and FrameIT and discuss a simple framing example. The example is continued from theoretical perspective in section 4 where we explain the main mechanisms that take place in the framing process. In section 5 we discuss the implementation of the User Interface, the design choices and basic functionalities.

## 2 Related Work

One of the systems in the past that tried to deal with similar problems is ActiveMath [10]. ActiveMath is an Intelligent tutoring system(ITS) that provides a web-based adaptive learning environment for mathematics. ITSs aim to improve long-distance learning by providing artificially generated, immediate and specific feedback to students. ActiveMath tries to deal with six major aspects: a) problem solving b) rule-based systems c) knowledge representation d) user modeling e) adaptive systems and adaptive hyper-media f) diagnosis The system does not force the student along a predefined solution path. Instead it is only able to verify the users final answer is correct according to the requirements for the current subgoal. This allow the user to take advantage of external systems that can support him or her and let the learner focus only on specific parts of the problem and delegate routine tasks. In the demo however, it seemed that, in questions that represent larger clusters of theoretical information and really benefited from different versions of multiple choice design, luck was a possible factor in the solution of the problem. In our case it will be possible to circumvent this problem by providing multiple problem-solution pairs for a single problem and apply them depending on the users actions. An important part of ActiveMath is the course generator that by using the rule- based system Jess [3] suggests different content and actions to the user. This course generator can be adapted to various properties for example: technical equipment, students curriculum, language, field of study and educational needs and preferences. Active Math also implements a User Modeling component that stores the history and results for every user. In the history it keeps reading times, success rate for exercises and IDs of the contents of the read pages viewed. Using this information and an AI algorithm that measures the students level in knowledge, comprehension and application and gives appropriate advise. Realizing such components takes immense amount of resources and therefore is out of the scope of my project but will be noted as possibilities for future work. For knowledge representation Active map uses an older version of OMDoc. Using a semantic rather than syntactic representation of the data allows the organization of mathematical documents by creating divisions such as theorem, proof, and definition. For this thesis I will use the newest version of OMDoc formed by the Module system for Mathematical Theories(MMT) [14] and TNTBase - a storage system for XML documents. Since MMT is highly modular, foundation independent and TNTBase is scalable and MMT compliant it will provide a great environment to set up the theories and theory morphisms of the FrameIT method.

# 3 Preliminaries

## 3.1 MMT

MMT (module system for mathematical theories) [7] is a formal module system for mathematical knowledge. The MMT language is designed to be applicable to a large collection of declarative formal base languages and all MMT notions are fully abstract in the choice of the base language.

MMT represents logical knowledge on three levels :module level, symbol level and object level [9]. On a modular level the primitive modular concepts MMT uses are **theories** and **theory morphisms**.

On the symbol level, the system uses named **symbol** declarations that might have a type or a definition to form a simple declarative language. One of the goals of this language is to include all declarative languages in its definition and it does that using the Curry-Howard correspondence [15, 2] to represent **axioms** and **theorems** as constants and proofs as terms. An axiom in MMT is a constant symbol that stores the axioms proposition in its type. On the other hand, theorems are defined constants whose proof is the symbols definiens.

At the object level MMT uses the grammar of of OpenMath [1] to represent theoretical objects and still remain foundation independent. The semantics of objects is verified by comparing different objects. In order to be precise in these comparisons we have to commit to a specific foundation.

The system adopts the "*little theories approach*"[5] where separate theories represent separate contexts. Theory morphisms represent structural relationships between these contexts and are used by theories to pass various definitions and theorems. Different theories can be related by inheritance. The inheritance in MMT is realized by different structures such as **include** which simply imports the inherited theory. In order to represent translations between two theories MMT uses explicit theory morphisms called **views**. This structure results in **theory graphs** where nodes represent theories and views are represented by edges.

*Example* 1. On Figure 1 we can see an example of a theory graph. The graph depicts the theories of groups, monoids, integers and natural numbers as nodes and the relationships between them. Monoid is a set, S that together with a binary operation " $\times$" satisfies the following axioms:

- **Closure**: $\forall a, b \in S : a \times b \in S$
- **Identity element**: $\exists i \in S : \forall a \in S : i \times a = a \times i = a$
- **Associativity**: $\forall a, b, c \in S : (a \times b) \times c = a \times (b \times c)$

A group is a monoid with one extra property **Inverse element**: $\forall a \in S \exists b \in S : a \times b = b \times a = i$

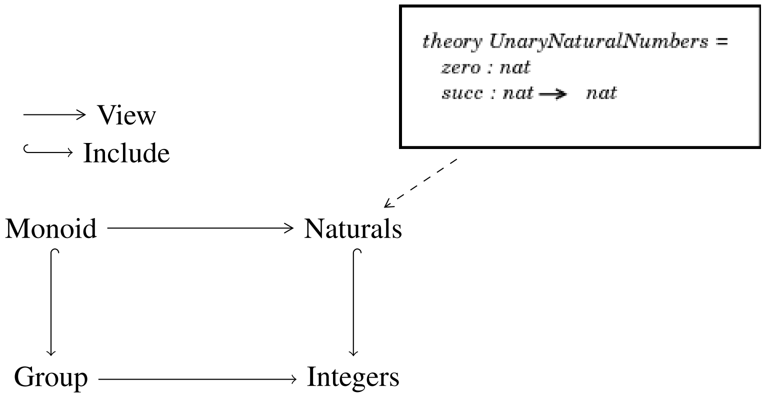


Figure 1: MMT Theory Graph Example and example syntax

In the theory graph on Figure 1 we show the inclusion between monoid and group that signifies the import of the three axioms of the binary operation. We can also see the views between the theories for monoid and natural numbers that can be achieved by mapping S to the set of Unary natural numbers and the binary operation to addition. Analogically there exists an inclusion between natural numbers and integers as well as a view between groups and integers.

We have also demonstrated the syntactical declaration of the Unary Natural Numbers theory in order to illustrate a sample MMT theory declaration.

All declarations within MMT can be identified by a globally unique URI. Theories in the system can be referenced by the URI of the theory graphs they belong to and their name. Analogically we can reference symbols knowing their name and the URI of the containing theory and assignment declarations by symbol name and containing view.

Another feature important feature MMT provides for our purposes is the ability to evaluate expressions. This is achieved by the *Universal Machine for Biform Theory Graphs*[8] through combining two semantic aware assistant systems : proof assistants that convert semantics to logic focus on deduction and computer algebra systems that implement semantics in a programming language and focus on computation.

## 3.2 FrameIT

This thesis is based on the FrameIT method [12]. Its main purpose is to integrate the knowledge representation approach and knowledge management systems and it accomplishes that with the **Learning Object Graphs** (LOG) approach.[12] This approach generalizes mathematical knowledge to a set of theories and theory morphisms. Theory morphisms represent the different relations learning objects can have. For example there are inclusions that represent an inheritance relation between two theories and are useful for providing a modular and structured representation of the information. Views on the other hand represent a relation between learning objects that do not inherit from each other and can help us view one of them as the other.

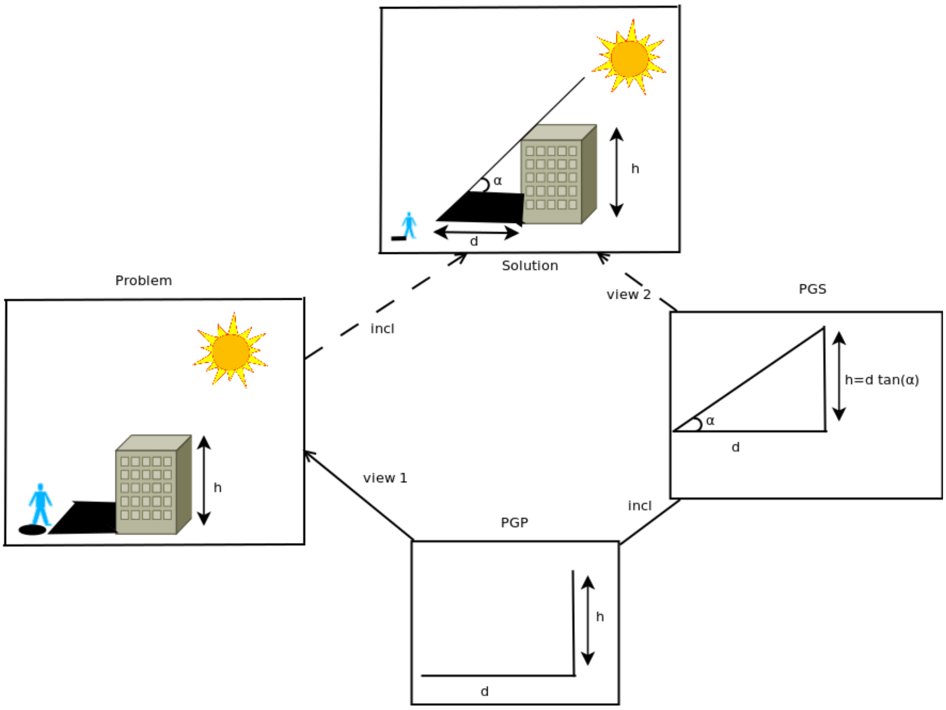


Figure 2: A simple example that demonstrates the LOG approach

*Example* 2. Figure 2 represents a simple real world problem that can apply the theory of trigonometry. The problem statement is: How can you find out the height of the building using only a protractor and a tape measure. The less objects you use the more points you get. The first solution that comes to mind is to measure the length of the shadow of the building and using the protractor to catch the angle at which the sunlight is falling on the edge of the shadow. Then the height h is simply $h = d \times tan(\alpha)$. The system though has to apply a few more steps to arrive at the solution. First we have to determine which objects are crucial for the situation and create a minimalistic problem statement. Then by providing our measurements we can arrive at the solution. The problem statement and solution together with the theory morphism (of inclusion type) between them form the Problem Solution Pair. In order to solve a problem the only thing we have to do is choose a Problem Solution Pair. This process is expressed by view 1 on the diagram. Views can be formally denoted as p : $\phi$ which means view p with a symbol mapping $\phi$. These views are also called framing morphisms. An important implication to note is the fact that we can apply more than one Problem Solution Pair to a single problem. In

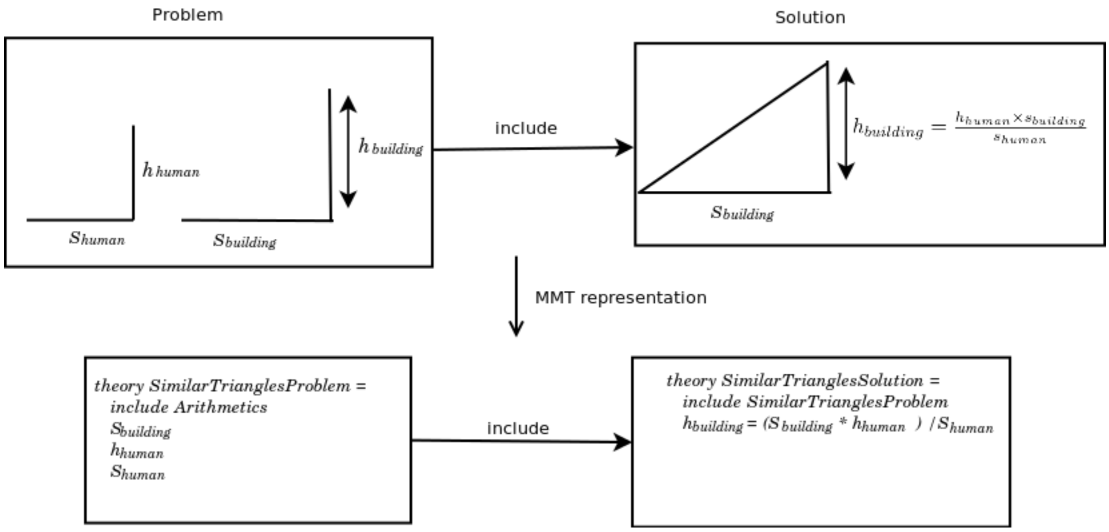


Figure 3: MMT Problem Solution pair Example

our case we can use only the measure tape and measure the length of the shadow of the building as well as the height of the protagonist and the length of his shadow. Then we will have two similar triangles and can find the height of the building to be: $h_{building} = \frac{h_{human} \times s_{building}}{s_{human}}$ where s denotes the shadow length of the epecified object.

# 4 Realizing Framing using MMT

For this thesis, we developed a technical realization of the FrameIT method based on MMT. We represent FrameIT problems and solutions as MMT theories and FrameIT morphisms as as MMT morphisms. Specifically we represent the inputs in a problem as well as the expected result as MMT symbols. A solution is then represented by adding a definition to a result based on the inputs (Figure 3). For instantiating the problem statement with concrete values for the inputs we use MMT views (Figure 4).

Lets take a look at the example explained on Figure 2 from the MMT perspective. As we can see on Figure 3 we begin the theory that represents the problem statement by including a predefined theory containing a set of arithmetic operations needed to perform basic computations. Since there is no "initial basis" we can rely on the notions of numbers and arithmetic operations predefined in the *Arithmetics* theory. This theory utilizes the Universal Machine for Biform Theory Graphs [8] which represents MMT theories as Scala [13] classes in order to provide functionality to the defined arithmetic operations. Afterwards we define a set of symbols that represent objects we can acquire from the given problem. In the Solution the-

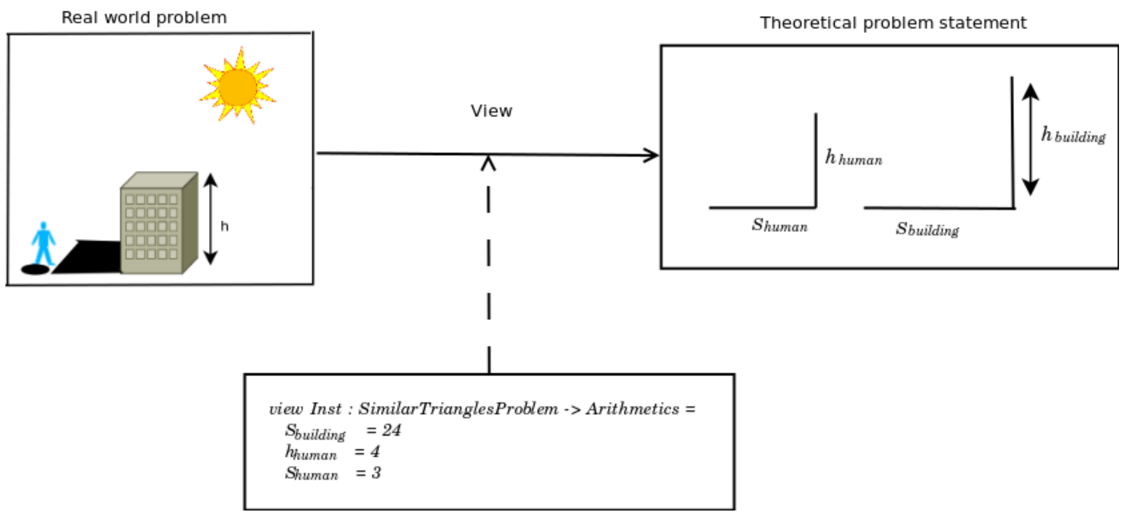


Figure 4: MMT View Example

ory we define the unknown value we are trying to compute and provide the needed arithmetic expression.

The values needed for the solution are provided by the user in order to create view 1 from Figure 2. The view, demonstrated on Figure 4, creates a mapping between the real world problem and the theoretical one we described in MMT.

After the view is generated and submitted to MMT we have all the information needed in order to compute a pushout and calculate the solution of the problem.

The real strength of our approach does not lie in the simple computation of the problem. We are applying the knowledge stored in MMT in various situations and in many cases we can apply different concepts to achieve a result. This gives us the opportunity to prepare a larger set of Problem Solution Pairs and allow multiple solutions to be applied to a single problem Figure 5. In the next chapter we will show how we use that to create a user interface that combines the concepts we laid out so far.

# 5 Implementation

On the MMT side I implemented a theory graph starting with the Problem Solution Pair on Figure 3. I also added a *Simple Trigonometry* and a *Law of Cosine* problem solution pairs and theories regarding follow up questions e.g. “What is the volume of the building?”

When I completed the sample theoretical base I created a User Interface that lets us put our knowledge management system to use. After careful consideration I

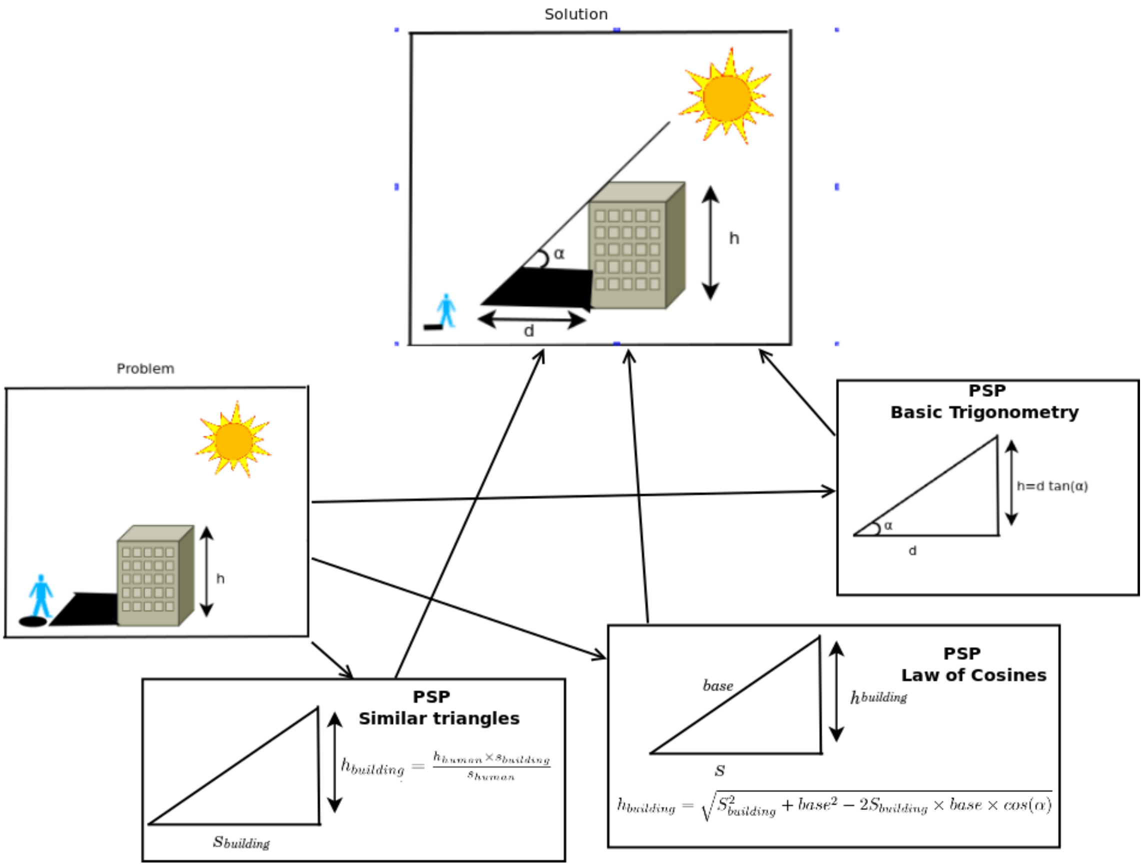


Figure 5: MMT Multiple Problem Solution Pairs Example

decided that it would be most appropriate for our purposes to develop a browser based application using JavaScript and jQuery. An important reason for this decision was the usability of the interface. Browser based applications tend to be very intuitive and most users are well accustomed to the browser environment. This choice also makes our software platform independent which makes it both more accessible and easier to test and develop. Another factor I took into consideration is the ubiquity of web based applications. This prevents any problems that might arise with distribution and updates. Finally Ajax and jQuery bring to the table very high level of simplicity.

On Figure 6 we can see a screenshot from the current version of the interface. On the left we can see the *"user playground"*. There he can investigate the problem and measure various parameters using the ruler and protractor tools provided by the buttons at the top. On the right we can see the problem statement and an accordion menu containing the different paths the user can take to approach the problem. Every path represents a Problem Solution Pair stored on the MMT side. After the user selects an approach and performs the measurements required he can submit his or her findings in order to **frame the problem**. Then a view that maps the needed information to the respective Problem Solution Pair will be created and sent to the MMT server via an Ajax request. Afterwards the servers computes the result and the interface presents it to the user. The Interface also allows the user to provide his or her own scenario with the *choose scene* button and apply the provided mathematical theories on a real world problem.

## 6 Conclusion and Future Work

We have presented an application of the FrameIT method to that provides an alternative means of knowledge gathering. Using MMT we managed to set up a background knowledge base consisting of multiple mathematical theories. Then we applied the FrameIT method using a browser based user interface that allows the users to provide real world scenarios where they can later apply various mathematical concepts and solve the problem at hand.

Our current implementation has many limitations. At this point of development the user interface supports only a single type of scenarios - height measurement. Future work will focus on expanding the theoretical knowledge base in order to have problem solution pairs applicable to a larger variety of situations. With this knowledge base we will be able to provide a set of example problems and create various challenges. These challenges can then be presented to the users and both test and further improve their mathematical skills.

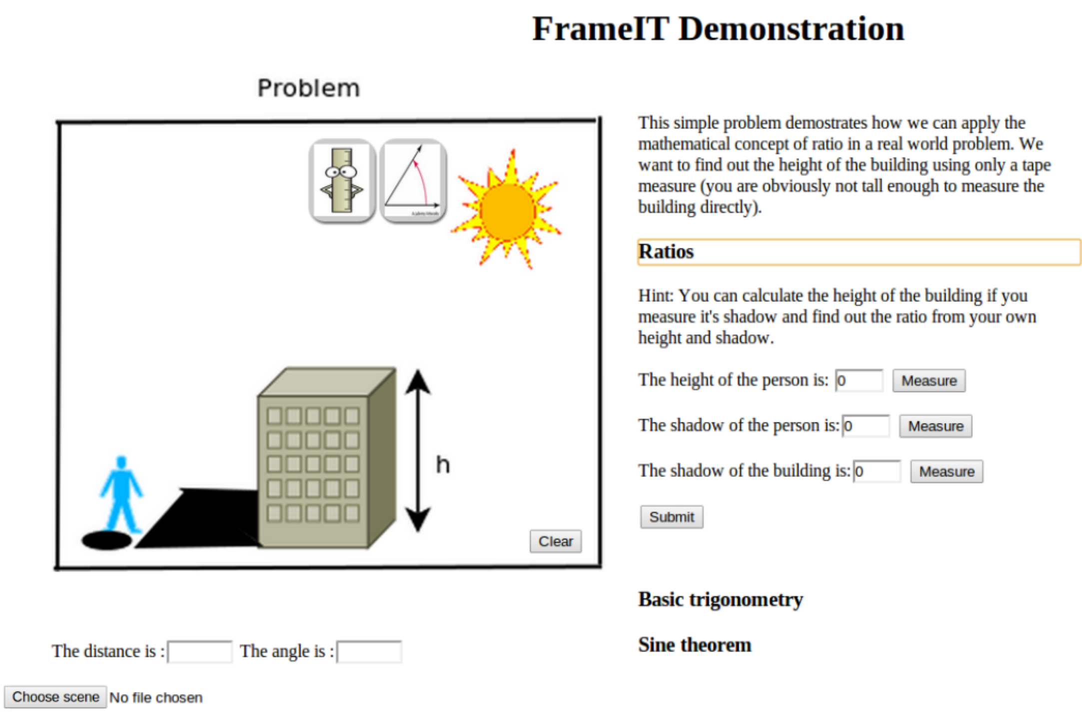


Figure 6: UI screenshot

# References


[1] Stephen Buswell, Olga Caprotti, David P. Carlisle, Michael C. Dewar, Marc Gaëtano, and Michael Kohlhase. The Open Math standard, version 2.0. Technical report, The OpenMath Society, 2004.

[2] H. B. Curry and R. Feys. *Combinatory Logic, Volume 1*. North Holland, 1958.

[3] E. Friedman-Hill. *Jess, the Java Expert System Shell*, 2000. `http://herzberg.ca.sandia.gov/jess/`.

[4] Mary Hegarty. *Components of Spatial Intelligence*, volume 52. 2010.

[5] D. Kapur, editor. *Proceedings of the 11$^{th}$ Conference on Automated Deduction*, number 607 in LNCS, Saratoga Springs, NY, USA, 1992. Springer Verlag.

[6] Andrea Kohlhase and Michael Kohlhase. Spreadsheet interaction with frames: Exploring a mathematical practice. In Jacques Carette, Lucas Dixon, Claudio Sacerdoti Coen, and Stephen M. Watt, editors, *MKM/Calculemus*

*Proceedings*, number 5625 in LNAI, pages 341–356. Springer Verlag, July 2009.

[7] Michael Kohlhase and Mihnea Iancu. Searching the space of mathematical knowledge. In Petr Sojka and Michael Kohlhase, editors, *DML and MIR 2012*. Masaryk University, Brno, 2012. in press.

[8] Michael Kohlhase, Felix Mance, and Florian Rabe. A universal machine for biform theory graphs. 2013.

[9] Michael Kohlhase, Florian Rabe, and Vyacheslav Zholudev. Towards MKM in the large: Modular representation and scalable software architecture. In Serge Autexier, Jacques Calmet, David Delahaye, Patrick D. F. Ion, Laurence Rideau, Renaud Rioboo, and Alan P. Sexton, editors, *Intelligent Computer Mathematics*, number 6167 in LNAI. Springer Verlag, 2010.

[10] E. Melis and J. Siekmann. Activemath: An intelligent tutoring system for mathematics. In *Seventh International Conference 'Artificial Intelligence and Soft Computing' (ICAISC)*, number 3070 in LNAI, pages 91–101. Springer-Verlag, 2004.

[11] Erica Melis, Eric Andres, George Goguadze, Paul Libbrecht, Martin Pollet, and Carsten Ullrich. Activemath: System description. In Johanna D. Moore, Carol Redfield, and W. Lewis Johnson, editors, *Artificial Intelligence in Education*, pages 580–582, Amsterdam, 2001. IOS Press.

[12] MichaelKohlhase. Frameit: Active examples for technical documents.

[13] M. Odersky, L. Spoon, and B. Venners. *Programming is Scala*. 2007.

[14] Florian Rabe and Michael Kohlhase. A scalable module system. Manuscript, submitted to Information & Computation, 2013.

[15] Jonathan P. Seldin and J. Roger Hindley, editors. *To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus and Formalism*. Academic Press, 1980.

[16] David A. Snow, E. Burke Rochford, Steven K. Worden, and Robert D. Benford. Frame alignment processes, micromobilization, and movement participation. *American Sociological Review*, 51(4):464–481, 1986.