A Language for Building Web Interfaces to Mathematical Software

Rachel Sun

Supervisors:
Professor Wolfgang Schreiner and Professor Elena Kartashova

Department:
Research Institute For Symbolic Computation (RISC) & Institute For Analysis

SUPPORTED BY FÖRDERUNG DER WISSENSCHAFTLICHEN FORSCHUNG (FWF)

Outline

Motivation

Why is a Language for Building Web Interfaces to Mathematical Software facility needed?

I Will Explain... !

Motivation

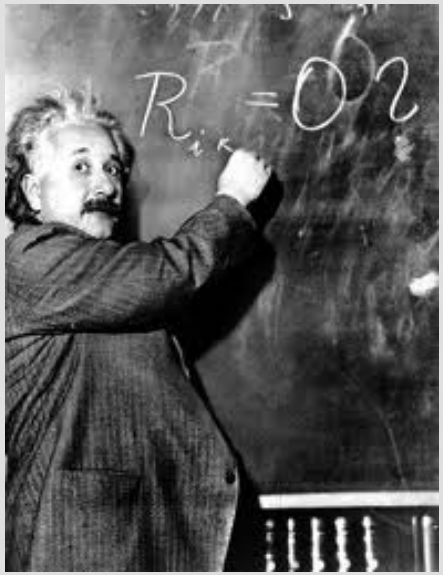
Just Imagine as a **mathematician**:

You come up with an algorithmic solution to a problem

- Solution can be implemented in any language
- Solution can be written using existing mathematical software

And you want to publish it on the internet:

- Making it accessible to a broader audience (as a web application) without rewriting the solution



All you need is to tell a generic framework:

How you like it to be displayed?
How to perform the computation?

And a web application is automatically generated!

Won't it be nice?

Motivation

Just Imagine as a student or a researcher:

You do not have a copy of
a particular mathematical software
on the local system...

You go online and there you find
a bunch of web-based applications

- that solve some kind of problem domains
- without having to install any software
- available anytime, anywhere

Won't it be nice?

A More Clearer Illustration

What You HAVE:

- ✓ Mathematical solution to a particular domain
- ✓ - written in any language
- ✓ - using existing software

What You DON'T HAVE:

- ✓ Specific knowledge how to write a web application

What You WANT:

- ✓ Publish it to the Internet
- ✓ Share your knowledge to a broader audience

What You DON'T WANT:

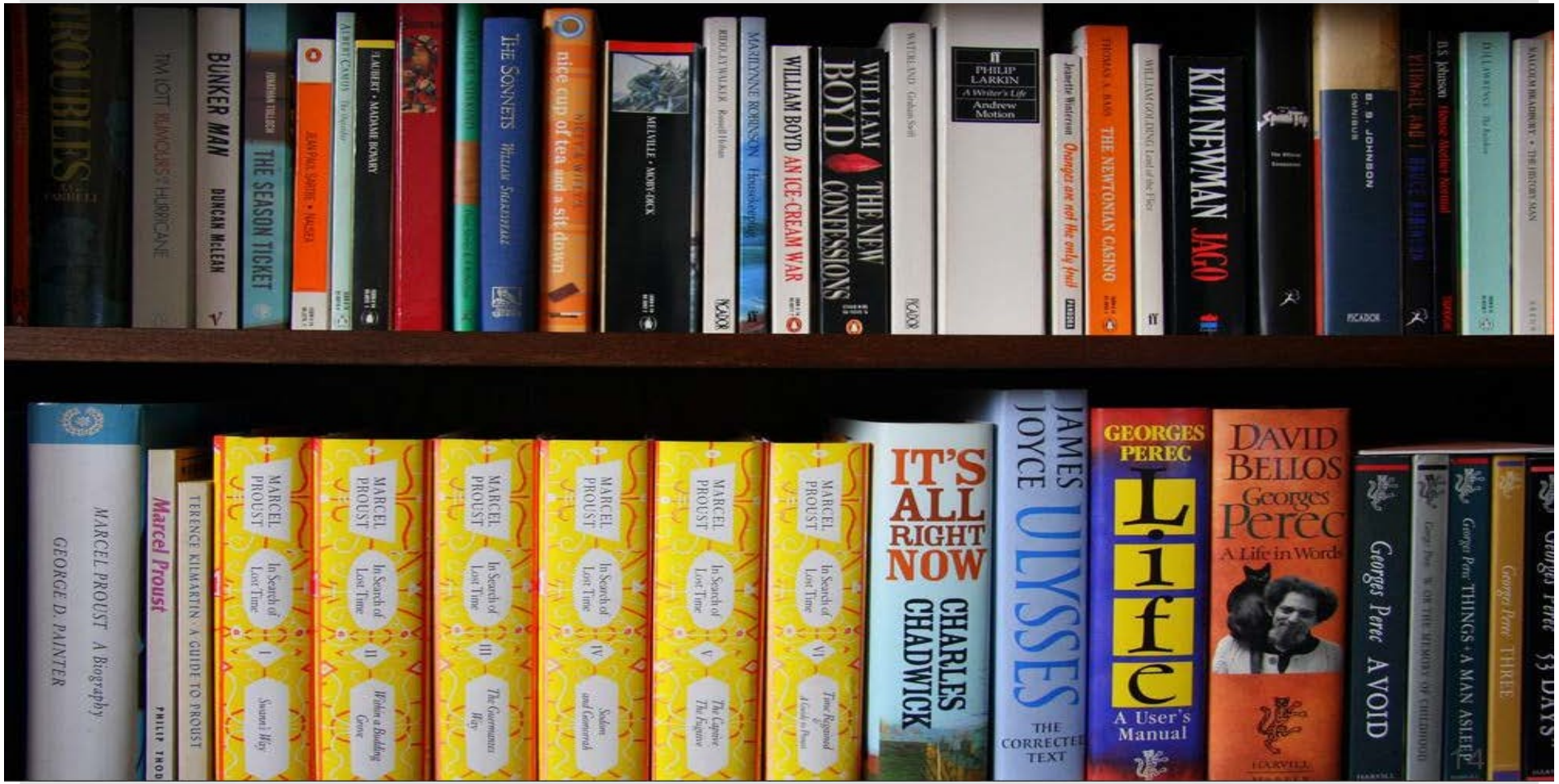
- ✓ To rewrite solution to adapt to web Technologies

Goals And Problem Definition

Goals And Problem Definition

- **Goal:** Design and implementation of a framework to create a dynamic web-based mathematical application that solves a class of mathematical problems.
- A generic framework which is independent of a particular mathematical domain problem
- All the mathematically computations can be performed in background by existing mathematical software
- Mathematical programmers needs to provide a workflow and an interface description, and necessary programs to the framework
- The actual web application will be generated from these information and the service is deployed on the server

Literature Review And Background



Current Approaches In Mathematical Web Application:

- Service Oriented Mathematical Systems
- Mathematics on the Web
- E-Learning Environments
- Virtual Laboratories
- Web Technologies

Service Oriented Mathematical Systems

- Aims to accelerate rapid application development by service reuse and discovery.

Example Applications:

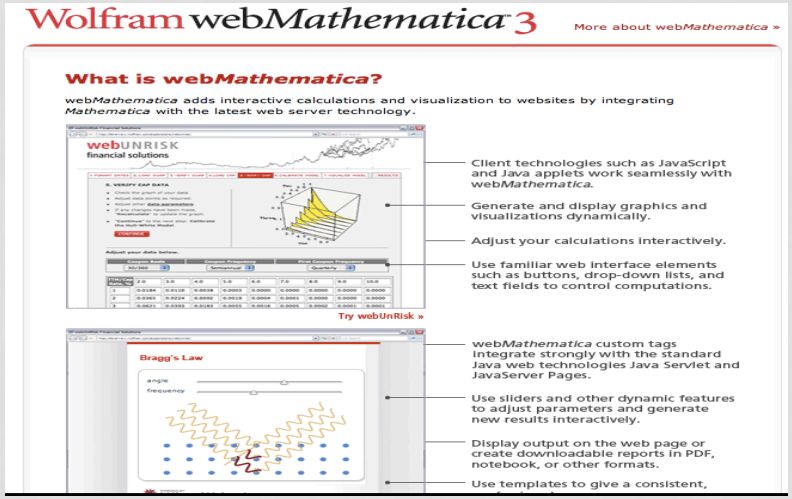
- **Netlib**
  - A general-purpose repository comprising a collection of free software, libraries for mathematics computing via user interfaces.
- **NetSolve**
  - A remote procedure call (RPC) based middleware system that integrates software resources distributed across a network.
- **Monet**
  - A framework to discover mathematical web services dynamically based on published descriptions.
- **MathBroker**
  - A software framework for brokering mathematical services.
- **MathWeb**
  - Provides an infrastructure for linking mathematical services using a general distribution architecture.

Mathematics On The Web

Wolfram Research

Mathematica

- Interactive scientific computation software that offers the combination of numeric, algebraic, geometric and logic algorithms together with web services.



webMathematica

- A service that merges the computational power of Mathematica with the web.
- Intended to provide specialized computations over the web.

Mathematics On The Web

Wolfram Research

Wolfram Alpha

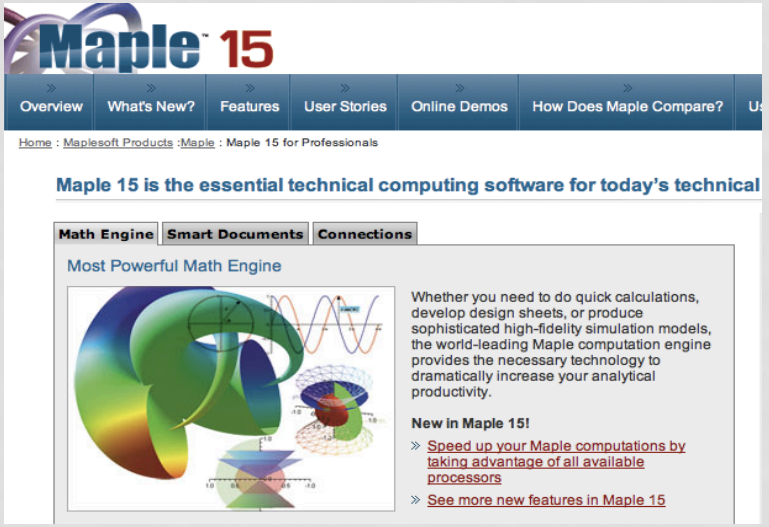
- An artificial intelligence based online answer engine with Mathematica running in the background.

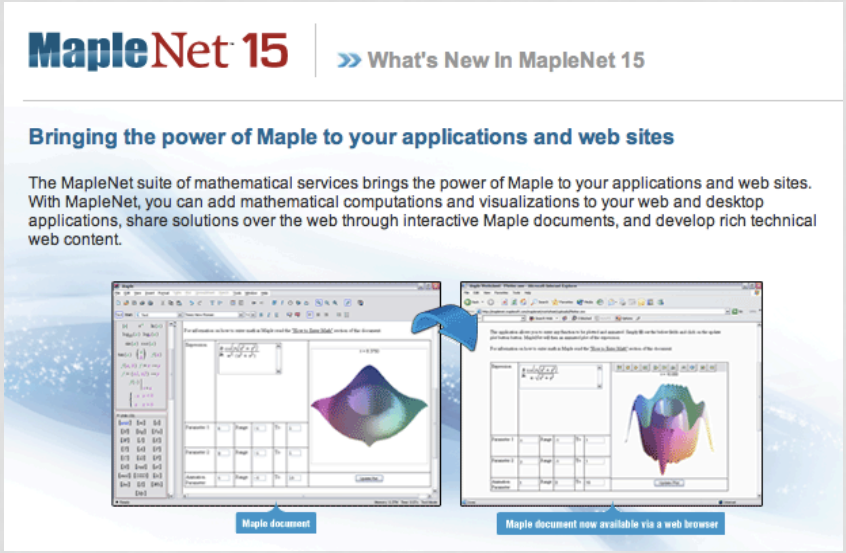
Mathematics On The Web

Maplesoft

Maple

- Like Mathmatica, Maple manipulates information in a symbolic or an algebraic manner.



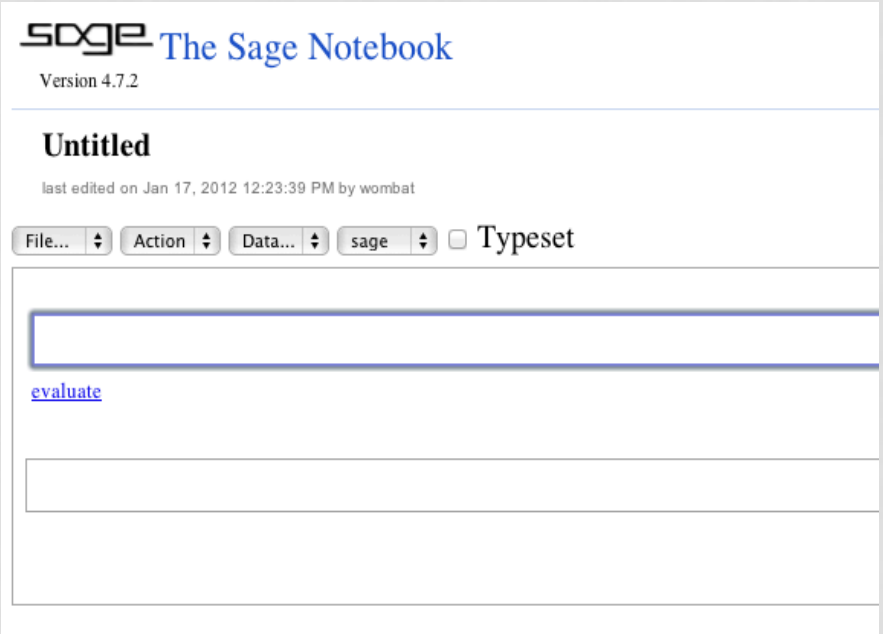


MapleNet

- With MapleNet Mathematical computations and visualizations are integrated into web and desktop applications.

Mathematics On The Web

Sage

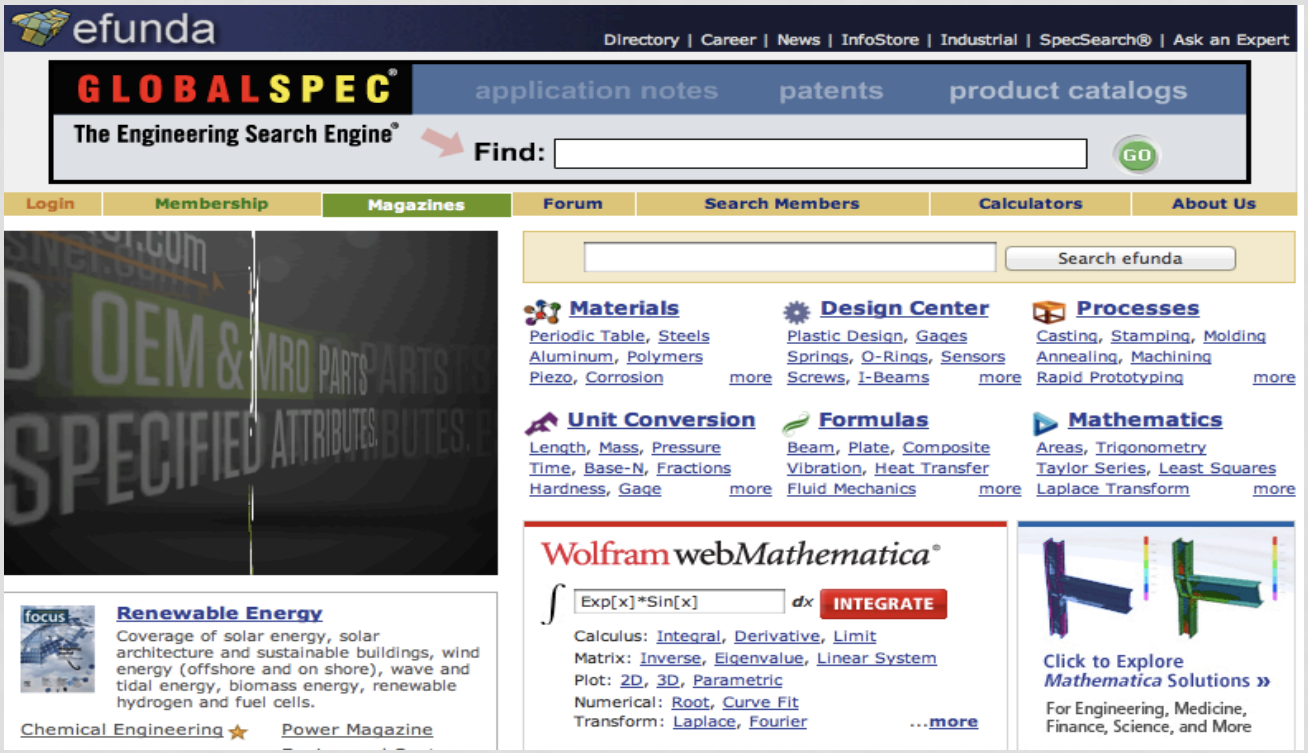


- A free web-based Mathematical software system.
- Aims to create an open source alternative to Magma, Maple, Mathematica, and MATLAB.
- Uses Python programming language to support procedural, functional and object-oriented constructs.

E-Learning Environments

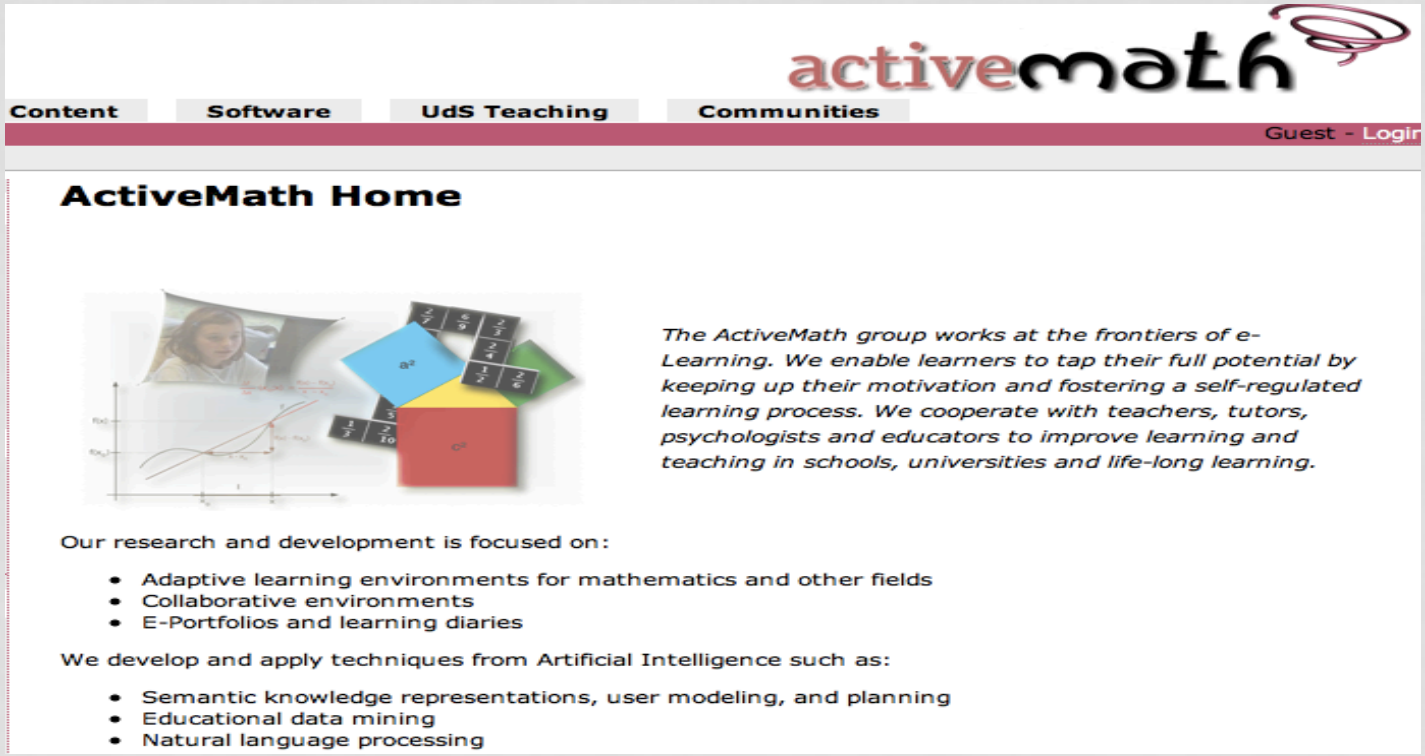
E-Learning Environments

- eFunda (engineering fundamentals):
  - Offers a convenient internet resource for online engineering computation.



E-Learning Environments

- ActiveMath:
  - An artificial intelligence online collaborative learning environment for mathematics and other fields.

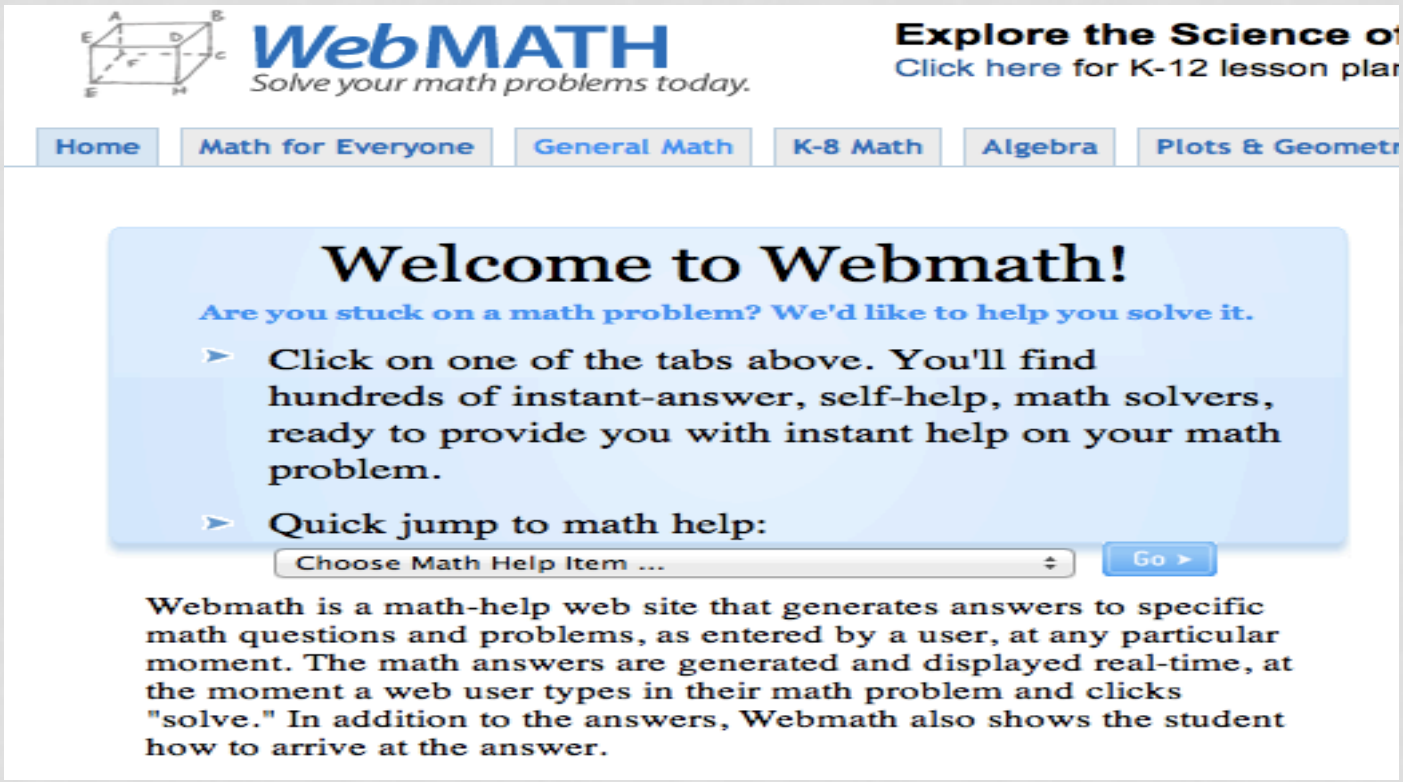


E-Learning Environments

- MERLOT:
  - A free educational resource that comprises a collection of "learning objects" on the Web that refers users to useful Internet sites.

E-Learning Environments

- WebMath:
  - A free online mathematical tutoring assistance site to help students with their mathematics proficiencies.



Virtual Laboratories

- **Virtual Computing Lab (VCL)**
  (Peng Li, Lee W. Toderick, and Philip J. Lunsford, 2009)
  - An open source remote learning environment.
  - Goals is to provide remote access scheduling and image management system to the students at East Carolina University (ECU).
- **Virtual Math Teams (VMT)**
  (Baba KofiWeusijana, Jimmy Xiantong Ou, Gerry Stahl, and StephenWeimar)
  - Offers service for synchronous collaboration learning and group cognition within a rich integrated Internet-based environment.
- **Distributed Control Lab (DCL)**
  (Andreas Rasche, Frank Feinbube, Peter Tröger, Bernhard Rabe, and Andreas Polze, 2005)
  - Designed for teaching and research demonstration.
  - Provides access to remote and real-time control experiments.
- **The Virtual Networked Laboratory (VNL)**
  (Lee J. Leitner and John W. Cane, 2005)
  - Replicates key aspects of traditional hands-on laboratory environment by providing a simulating remote laboratory experience.
- **Virtual environment for learning networking (Velnet)**
  (Bruce Kneale, Ain Y De Horta, and Ilona Box, 2004)
  - Enables laboratory experiments to be performed over network connections on a single desktop host computer.

Summary of The Current Approaches

- Need specific knowledge for the service oriented programing and web application.
- Cannot use arbitrary programming languages for the solution, such as in Sage, Mathematica, Maple etc.
- Specific software or hardware requirements on the client like in some of the virtual laboratories solutions.
- Tailored for one specific problem domain like in the virtual laboratories solutions.

Web Technologies

Web Technologies	Description
HTML	• Used to describe WWW content. • Not so strict concerning the structure of an HTML document.
XHML	• Define HTML as an XML. • Documents must be marked up correctly and 'well-formed'. • Gives 'semantic' to a given tag. • Separates the presentation from structure.
XML	• A universal data representation markup language. • Aimed to encode semantics into web documents. • Facilitates information interchange and integration from heterogeneous systems
XML Schema (XSD)	• Define the possible structure and contents of an XML format. • A validating parser can check whether an XML instance document conforms to an XSD schema or a set of schemas • Check the correctness of node hierarchy and data-types in the XML

Web Technologies

Web Technologies	Description
PHP	• Language for creating interactive web pages. • Can be embedded into HTML. • Deployed on most web servers. • Executes scripts directly on the web server. • Supports many databases, such as MySQL, Oracle, Sybase, Solid and etc.
JavaScript	• Client-side object-oriented scripting language for web pages. • Designed to add functionalities and interactivity to HTML Pages. • Embedded directly into HTML pages. • Script is run by the web browser. • Used to validate forms and communicate with the server.

Web Technologies

Web Technologies	Description
Ajax	• Used primarily for developing highly interactive web applications. • Offers a rich desktop-like user experience. • Allows asynchronous access of information to the server.
GWT	• Makes it easier to develop interactive, AJAX web applications. • Java is the only language used in the web development program. • Compiles the Java source code directly into HTML and JavaScript code. • Run in all major web browsers. • No single line of HTML or JavaScript code is necessary.
Pyjamas	• Is a port of GWT. • Compiles the Python source code directly into HTML and JavaScript code.

Tools

Tools

- ✓ Eclipse Indigo 3.7
- ✓ Pydev
- ✓ Python 2.7
- ✓ XML Schema
- ✓ XML
- ✓ lxml Toolkit
- ✓ Pyjamas 0.7
- ✓ Apache Web Server 2.2.20
- ✓ JSON-RPC 2.0

Tools

Why use Pyjamas over GWT?

- We do not want Mathematicians to provide files (handler functions) written in Java.
  - Python is a lightweight and an intuitive language.
- To have Python script ability in GWT, a Python interpreter is needed to be implemented in JavaScript to run in the web browser.
- Here comes the Pyjamas - a port of GWT:
  - If there is already a compiler which translates python into JavaScript, why do we need an extra step?
  - It will hence as well translate the Python handler functions to JavaScript.
  - No extra interpreter needed.

Tools and Programming Knowledge For Mathematician

What you ONLY NEED :

- ✓ Basic knowledge in XML and Python
- ✓ Any text editor (for programming and XML editing)

What You DON'T NEED:

- ✓ Eclipse Indigo 3.7
- ✓ XML Schema
- ✓ lxml Toolkit
- ✓ Pyjamas 0.7
- ✓ Apache Web Server
- ✓ JSON-RPC 2.0

Approach

Files To Write And Provide

Two XML Files

- GUI Definitions
- Mathematical Server System-Calls Definitions

Two Python Scripts

- Client-side Handler
- Server-side Handler

Mathematical Computation Files

- Arbitrary scripts, libraries, programs etc.
- Used to perform actual mathematical computation

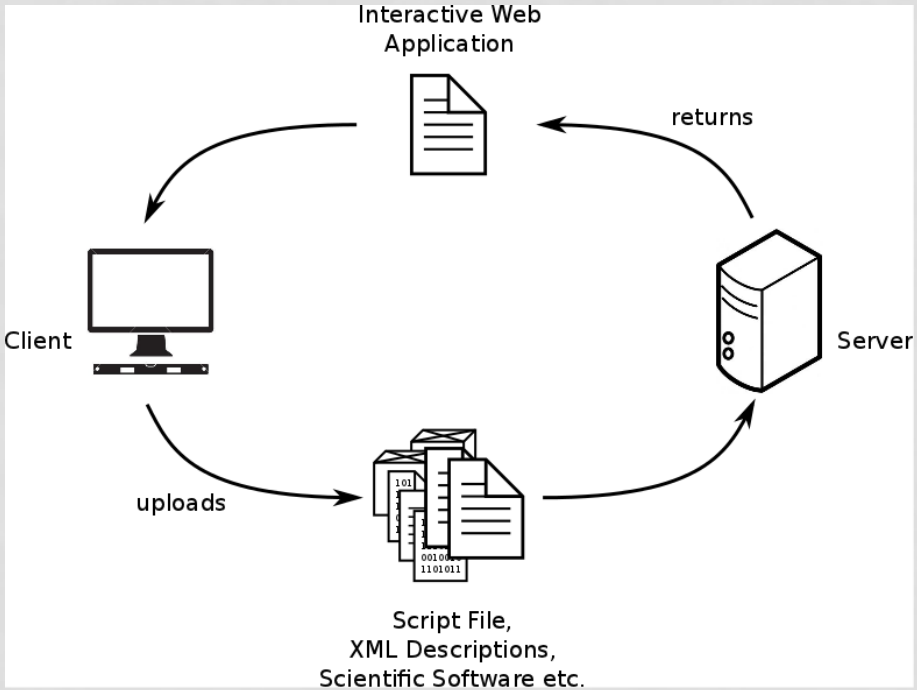
A Simple System Workflow

1. Client on his host computer uploads the necessary files for the framework to the server.

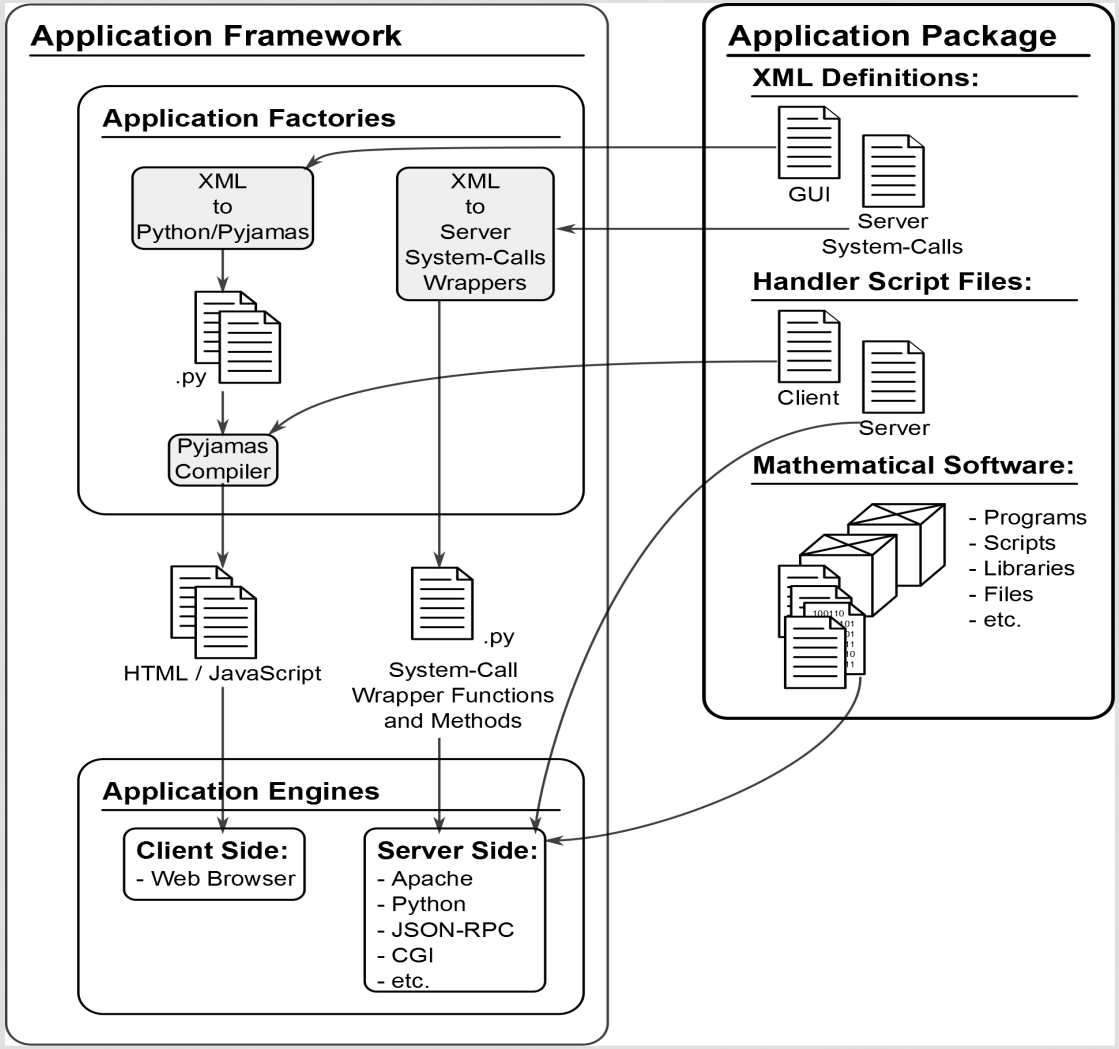
2. The server generates based on the uploaded files the interactive web application.

3. It then either returns a link pointing to the web application or an error message to the client.

4. Web users will now be able to use the software for performing computation over a web-browser.



A More Insight Workflow



XML File: GUI Definitions

- Define how the static web interface should look like.
- Define what methods are available in a remote service (JSON-RPC) and where to find them.
- Widgets that are currently supported:
  - Panel: Horizontal, Vertical
  - Widget: Button, CheckBox, RadioButton, ListBox, TextBox, TextArea, Label, HTML and Image.

XML File:
GUI Definitions Example

```
<GUI name= "MyFirstWebApplication" >

        <ServerCallBack name= "server" url=http://localhost/toyservice/cgi-bin/calculator.py
                        methods= "add, sub" >
        </ServerCallBack>

        <VerticalPanel>
                <HorizontalPanel>
                        <Label text="Input 1 : "/>
                        <TextBox name="input0"/>
                        <Label text="Input 2 : "/>
                        <TextBox name="input1"/>
                </HorizontalPanel>
                <HorizontalPanel >
                        <RadioButton name="rb0" label="Add" group="group1" checked="True" />
                        <RadioButton name="rb1" label="Sub" group="group1" checked="False"/>
                        <Button label="Calculate" listener="calc_callback"/>
                <HorizontalPanel>
                        <Label text="Output : "/>
                        <TextBox name="output0" />
                        <TextBox name="output1" />
                </HorizontalPanel>
        </VerticalPanel>
</GUI>
```

JSON Proxy object for handling JSON-RPC asynchronous communication to the server.

There can be multiple JSON Proxy objects.

Location of the Service.

Layouts of GUI.

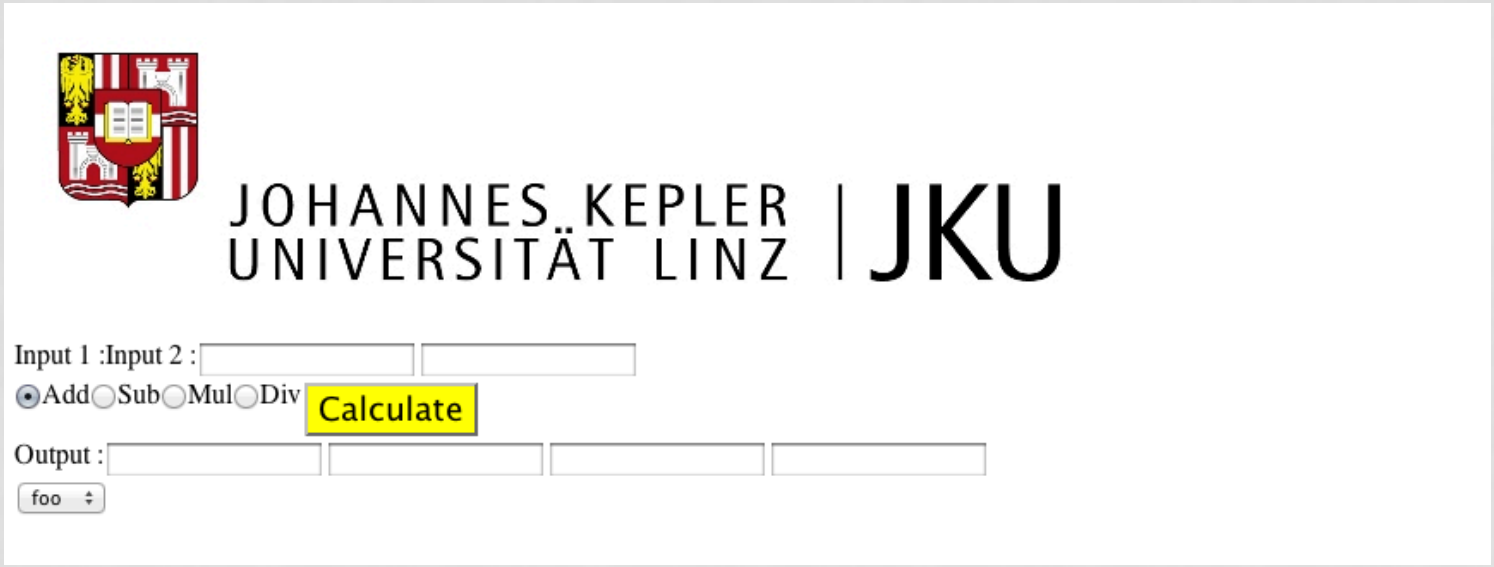
Widget with a "name" attribute can be accessed from within the client handler function.

Name of the client handler function. The event is fired when the corresponding button is clicked.

XML File:

GUI Definitions Example

Generated Web Application Graphical User Interface (GUI):



Web user will not be blocked waiting for the result during the asynchronous JSON-RPC communication.

Python Script File: Client-Side Handler

- Handles all the events fired in the webpage.
- Use the name of the widget defined in the XML GUI file to get and set values.
- All the method calls upon a widget are wrapped inside a wrapper class.
- If there is any changes in the Pyjamas API in the future:
  - Only the wrapper classes need to adjust to the changes.
  - No changes on the method calls in the client handler file - method names stay the same.

Python Script File:
Client-Side Handler Example

```
def event_Handler( gui ):
    a = int( gui.input0.text )
    b = int( gui.input1.text )

# Client-side Computation

    if gui.rb0.checked:
        gui.output0.text = a + b

# Server-side Computation

    elif  gui.rb1.checked:
        gui.output1.text = gui.server.sub( a, b )
```

Object for manipulating the GUI.

Gets the values of text boxes.

Ordinary value assignment: Translated directly by Pyjamas Compiler.

Set the value of a + b instantly to output0.

Triggered by RPC: Call the server´s method with appropriate arguments.

Set the response from the server to the desired widget on arrival. (asynchronous communication)

XML Files: Mathematical Server System-Calls Definition

- Define how to call the mathematical software to perform actual computation.
- Wraps mathematical system calls into python functions:
  - Can be reused in the server-side script/handler file that defines services.
- Work still in progress. ..

XML Files: Mathematical Server System-Calls Definition Suggestion (not implemented yet)

"ComputeAdd()" will be the actual function name in the python server-side handler script.

a and b are the arguments of the function and c is the return value.

```
<Function name="c = ComputeAdd(a,b)">
    <Exec>
        <Prog>./bin/add $a $b</Prog>
        <Pipe from="stdout" to="c"<Pipe>
    </Exec>
</Function>
```

"Prog" defines
- which program (mathematical software) is used for the computation
- location of the program.
- how to place the arguments.

"Pipe" redirects the output of the program from stdout to the return value c.

The above XML code should be translated into something like this in Python:

```
def ComputeAdd(a,b):
   c = os.popen('/bin/add %s %s' % (a,b)).read()
   return c
```

Python Script File: Server-Side Handler

- Defines how to handle service requests on the server.
- Gives the mathematician all the possibilities of Python (conditions, loops etc.):
  - Provides Mathematicians with the freedom and flexibility in defining how services should be handled.
  - When to call mathematical software on the server, how many times, under which conditions etc.

Python Script File:
Server-Side Handler Example

```
class CalculatorService(object):

    @ServiceMethod
    def add(self, a, b):
            return ComputeAdd (a, b)


    @ServiceMethod
    def sub(self, a, b):
            return a - b

    @ServiceMethod
    def mul(self, a, b):
            return a * b

    @ServiceMethod
    def div(self, a, b):
            return a / b
```

Methods with @serxviceMethod decorator are exposed as services to callers.

Calling the actual mathematical software in the background using the python wrapper function.

Perform computation directly inside the method.

Current Situation

- Client-side Communication:
  - XML GUI definition
    - Designed the XML GUI definition.
    - Half-way through designing and defining XSD for validation against XML GUI definitions.
  - Python client-side handler file
    - Redesigned to become more understandable.
  - Implemented wrapper classes to become independent of Pyjamas API.
  - Implemented the XML ---> Pyjamas converter in Python.
- Server-side Communication:
  - Python server-side handler file
    - Designed and implemented the prototype.

Problems and Future Work

Problems:

- How to display graph and other data (any multimedia content) with Pyjamas over JSON-RPC.
- How should the XML Mathematical Server System-Calls Definition look like.

Future Work:

- Convert XML Mathematical Server System-Calls Definition into Python function.
- Implement additional widgets and panels that might be needed.
- Software evaluation:
  - Perform tests with different kind of combination of widgets and different backend software.
  -
- Server installation.

Preliminary Results & Timeline

A Quick Demo

Conclusion

- There is a need for facilities in mathematical web-based applications.
- No simple general purpose framework so far that fulfills this kind of requirements.
- Beneficial for mathematical community and educators due to its simplicity.

Thank you!