Adessowiki - Collaborative platform for writing executable papers

Rubens C. Machado\textsuperscript{a}, Leticia Rittner\textsuperscript{b}, Roberto A. Lotufo\textsuperscript{b}

\textsuperscript{a}Renato Archer Information Technology Center (CTI)
\textsuperscript{b}School of Electrical and Computer Engineering, State University of Campinas - UNICAMP

Abstract

Adessowiki is a collaborative platform for scientific programming and document writing. It is a wiki environment that carries simultaneously documentation, programming code and results of its execution without any software configuration such as compilers, libraries and special tools at the client side. This combination of a collaborative wiki environment, central server and execution of code at rendering time enables the use of Adessowiki as an executable paper platform, since it fulfills the need to disseminate, validate, and archive research data.

Keywords: executable paper, collaborative systems, wiki, web 2.0

1. Introduction

Frequently, when reading a tutorial, a text book, an on-line course or even a paper that describe computer science tools and algorithms, it is not possible to know exactly how equations, figures, graphs were generated. Even when the processing steps are described, it is almost impossible to reproduce it. The lack of information on used parameters, different software and libraries versions, distinct system configurations, are some of the reasons that derail the reproduction of demonstrations and experiments.

A few systems were developed, not addressing all these issues, but at least with the intention of keeping together source code and documentation, such as literate programming \cite{1} and reproducible research \cite{2}. Most of the developed systems in fact split documentation and source code in order to generate the document and to compile the code. This segregation does not guarantee the synchronism of documentation and code, meaning that you could end up with different versions of code and documentation.

The proposed system, called Adessowiki \cite{3}, combine literate programming, collaborative development and Web 2.0 \cite{4} easiness. It addresses several important issues regarding its use as an executable paper platform, accommodating the need to disseminate, validate, and archive research data, as well as a method to allow this data, in some way or form, to be validated, citable, tractable, and executable.

2. Adessowiki as an executable paper platform

The executable paper platform proposed in this document relies heavily on the modern Internet and its clouds of processing for the creation, validation and dissemination of scientific knowledge. All the resources needed to reproduce a particular scientific result, like programs and input data files, are kept on some internet server and accessible...
for execution through the web. The authors, reviewers and readers use nothing but web browsers to accomplish their tasks. This platform is called Adessowiki and is presented in the following.

Adessowiki is a distributed system, based on the technologies and concepts of Web 2.0, designed to facilitate the collaborative creation of scientific content. It is a kind of collaborative literate programming [1] tool powered by the modern world-wide-web. What makes Adessowiki special in relation to other wikis and Web applications is the ability to include portions of code that are executed when rendering the wiki documents, in such a way that the results of the computation may become part of the rendered documents, in the form of figures, texts and tables. These code fragments are executed in a centralized place, with pre-configured hardware and software. Another advantage of this arrangement is that the user only needs a web client and a web connection to develop and test his application.

The Adessowiki includes resources to facilitate the writing of books and papers. Text in Adessowiki can be created in a collaborative way, as is natural to happen in a wiki. Adessowiki has tools to save the documents in Latex, RTF and PDF format enabling simple integration with the required book or journal editor format.

The most important issues adressed by Adessowiki are:

- **Executability**
  Equations, tables and graphs are generated by programming code embedded in the Adessowiki pages. By changing parameters in the code or via HTML form widgets defined in the code, results can be checked and manipulated by reviewers and readers.

  Each article in the Adessowiki can be viewed in at least two formats: a source text written in a special markup language and a corresponding HTML page (other forms, like the PDF version, may be generated by using other transformations tools). This HTML page is generated from the source document and includes the result of the execution of code fragments contained in the source document. This constitutes the basic interaction mode -- modify the source and regenerate the HTML to observe the changes, as in any wiki. The difference here is that figures, tables, graphs and text can be generated by executing the author code, possibly based on parameters and inputs that can be varied to see the effects on the resulting document.

  The system also has a second form of interactivity: the code snippets embedded in an article may have input parameters that are obtained through HTML widgets, generated when the page is rendered. In this way, the generated view of the document can be modified without editing the source document, providing another useful interaction mode. This interaction mode can be extended to address the monitoring and control of long-running code fragments, enabling the experimenting with control systems, robotic systems and so on. The next release of the Adessowiki will embody these capabilities.

- **Short and long-term compatibility**
  The code fragments are executed in a centralized place, with pre-configured hardware and software, making the system independent of the user’s operating system and architecture. This centralized character of the Adessowiki facilitates the evolution of the system. When an incompatible upgrade has to be made, the old execution profile is kept. This way the executable papers are always rendered properly.

- **Validation**
  Code and documentation are kept together within the paper, and the data are stored as attachments. And because code is executed in order to generate the HTML page, validation is redundant. What the reviewers see is what the code generated, together with the attached data. Additionally, one of main difficulties in areas where the performance of the solution must be compared, Adessowiki is the ideal solution as the experiments reported in the papers can be executed using the same hardware and same data.

- **Copyright/licensing**
  The data used to validate the ideas, algorithms and code presented in the paper are available to researchers in source code or binary code form. Many authors believe that their source code should be available with a specific copyright notice. Adessowiki has a link at the bottom of each page to the copyright notice. If the author do not want to release his source code, it may be possible to upload his program as a library in binary form. In this way the reviewer and users in general can reproduce the results of the paper but are not allowed to view the source code.
• Systems
The code fragments contained in the Adessowiki articles are executed through a controlled execution environment implemented by special server processes called “execution sandboxes”. These servers are distributed and communicates through an advanced messaging protocol. This architecture lead to the implementation of specialized sandboxes that makes special hardware/software available for use in the executable paper code snippets, like, for example, sandboxes with special hardware accelerator machines. Adessowiki currently supports CUDA programming language for GPU - Graphics Processing Unit cards.

• Size
Input data files are treated as attachments in the Adessowiki and kept in modern internet media servers, designed to work with very large data files.

• Provenance
Adessowiki is based on the present state-of-the-art for distributed systems hardware and software. Thanks to the built-in access control system and the version control of each page, actions taken on the executable papers are fully registered and, therefore, trackable.

• Other issues
As Adessowiki is a web application, it is relatively easy to incorporate plagiarism tools in such a way that the texts and code can be readily compared to other contribution in Adessowiki but also other available in other database or the Internet. The critical point of allowing users to incorporate executable code is the requirement of a protected way to execute users code. Adessowiki already has this mechanism as the user code runs in a virtual machine and the programming languages have commands that are not allowed to be executed.

3. Adessowiki overview
The Adessowiki is a distributed system designed to support the collaborative creation of scientific content. It is a kind of literate programming [1] system based on the technologies of the modern web. A typical page of the Adessowiki containing text, code and an image generated by the code is shown in Fig.1.

Fig.2 shows the computational organization of the Adessowiki. The system is composed of two separated web servers: the wiki server, that serve information from a database, and the media server, which serve files from the local filesystem. The main components of the Adessowiki are the wiki itself, the wiki markup language, reST, and the execution module, XSandbox, responsible for the execution of code fragments in a controlled environment. The sandbox executes in the same host computer as the media server, in such a way that the media created by the code execution are made visible in the web by the server. These components are described in the following sections.

3.1. The Wiki
Besides its software development features, Adessowiki may be characterized as a modern and fully-featured wiki. Like other wikis, Adessowiki is composed by a collection of hypertext pages without any structure except those created by the links. Adessowiki is formed by a set of spaces (or sections), that ultimately are individual wikis that share administrative information, like users, groups, etc. Corporative wikis in general use these spaces to ease the management of multiple wikis. On Adessowiki permissions for reading and writing for users and groups are adjusted for each space. The users and groups database, however, is global.

Adessowiki has a built-in access control system, with permissions for users and groups specified by space. Registering of new users is done through an invitation system. An invitation is a encrypted text that specifies user permissions. It is created by an Adessowiki administrator and sent to the users. The registration system validates the user through its e-mail before activating its account.

Every Adessowiki page is under version control. Changes made to the wiki pages are stored on a history of revisions such that any previous version of any page can be recovered. This feature incorporates implicitly concepts of source code version control like the Subversion (http://subversion.apache.org/) and CVS systems.

The wiki implementation is based on the Model-View-Controller, MVC, pattern and uses an Object-Relational Mapping, ORM, to work with its database. The implementation language is Python and the supporting web framework is the Django.
3.2. The Markup Language

The markup language used to create Adessowiki pages is called **reST** and is based on *reStructuredText*, a plain text markup language widely used in the Python community. Fig. 3 shows the document processing flow.

i) The user creates a **reST** document using the Adessowiki web interface.

ii) The document is parsed to create an XML tree. In this step the Execution Sandbox is used to execute the Python code fragments embedded in the page. The resources created by this code execution, images for example, are incorporated in the XML document.

iii) The XML document is transformed to create an HTML page that will be displayed by the Adessowiki web interface.

iv) The same XML document can be used for the generation of other kinds of representations like, e.g. LaTeX, RTF and PDF documents.
The most important features of the Adessowiki markup language are the ones related to code execution, creation of Python modules and building of C/C++ Python extensions.

3.2.1. Execution of Python Code

A source document example is presented below. This simple example is the source text of the page shown in Fig.1.

```
.. code:: python
   :show_code : yes
   :show_output : yes
   :show_images : yes

   thresh = 20
   a = mmreadgray('sign.jpg')
   b = mmgrad(mmopen(a, mmsebox(2)))
   mmshow(a, b, thresh, title='Contour')
   print 'Image shape:', a.shape
```

Note the markup used to specify the document and section title and the special markup used to embed a code fragment. The code markup contents is made of a set of parameters (the names between colons) and a code fragment that will be executed by the execution sandbox. The parameters control how the generated artifacts are included in the HTML page.

The code enclosed in this example reads an image, process it and shows the resulting image. The function `mmshow` is overridden by the sandbox execution environment so that the image is captured and made available through the Adessowiki media server. If the parameter `show_images` is true, the image URL is included in the generated page. The string generated by the `print` statement is intercepted and inserted in the rendered page if `show_output` is true.
3.2.2. Creation of Python Modules

Another important feature of the Adessowiki is the creation of reusable Python components that can be used in Adessowiki pages through the normal Python import system. These components are normal Python packages and modules named after the space/page where they are defined.

3.2.3. Building C/C++ Python Extensions

For image processing, in particular, and scientific programming, in general, the utilization of C/C++ programs to code performance critical tasks is very important. The reusable component architecture of the Adessowiki allows the inclusion of C/C++ code. The code is wrapped with additional code to built an interface for use from Python code. The process is automatic and the user only specify the code fragments containing the C/C++ source and the prototypes of the functions to be exposed to Python.

3.3. The Execution Sandbox

The code fragments embedded in a wiki page are executed through the Execution Sandbox: a controlled environment for evaluation of Python code. The sandbox is implemented as a separated process that shares the same filesystem as a media server. The execution environment enforces safety restrictions for the code that will be executed. The code cannot: (i) write to the filesystem (reading from some places is allowed, so that uploaded files are available); (ii) open sockets or access another host directly; (iii) spawn processes or threads; (iv) make other system calls, such as signals.

In order to be resilient to bugs eventually present in the user code to be executed, it is important for the server to be fork-based (as opposed to threaded servers). The use case is as follows. When the wiki starts the parsing of a page, it establishes a connection to the sandbox. The sandbox forks a process to provide the execution service. The code fragments are executed in this environment (so that the state is kept between fragments). When the parsing finishes, the wiki closes the connection and the forked process terminates. The images and other resources created by the execution are stored in the local filesystem, that is shared with the media web server.

The modules created by the sandbox are also kept in the filesystem as Python packages. These packages are registered in the Python path and can be imported by the pages that have access to the current namespace.
4. Use cases

The integrated collaborative environment containing documentation, programming code and execution results creates several other possibilities of new applications. This section describes some of the possible applications and use cases of Adessowiki.

4.1. Computer Science Courses

During a course, Adessowiki can assume two very distinct functions: as a wiki, it allows the students to register course information (students notes, homework and exercises; score and teacher comments on homework and developed codes); as a programming environment, it serves as a platform for laboratory and programming activities requested by the teacher, since code and its results can be inserted and the output of the program can be seen in the wiki pages.

4.2. Software Development

On Adessowiki, the code can be developed in collaborative way, with the advantage that code, documentation and execution results can live on the same page. This feature facilitates comprehension and avoids the lack of synchronism between documentation and code, so common in practice. It also dismisses the set of special configuration for executing the code and the software configuration for compiling, linking and loading is defined at the server and automatically invoked during the page rendering of the document.

4.3. Application Development

One of the difficulties in working with numeric computing is its multidisciplinary aspects, as the applications can vary but some class of algorithms and solutions are applicable for many areas. Adessowiki can help in the development of such applications as the client can upload the data and insert the explanation about the problem. Simultaneously, the developers can have access to the data and propose solutions which can be viewed at the pages of Adessowiki. As everything is stored in the server, collaboration among the clients and the developers are easily established. The client can compare results from many proposed solutions and the developers can improve their solutions based on the other previous solutions. Once the application is developed, it can be used immediately as the programming code is stored in Adessowiki without the need of further installations or configurations.

4.4. Algorithms Repository and Benchmarking

By using the Adessowiki, it is possible to build a kind of wikipedia of algorithms, containing solutions to classical problems. For each algorithm there would be a page containing: a description of the problem, a description of the algorithm, their proof for correctness and other text of theoretical content; the corresponding code, and examples of input and output. And because in the Adessowiki the embedded programming code is executed in a centralized and specific hardware/software environment, it allows accurate comparisons of the performance and correctness of different scientific algorithms. As the data is also stored in the server, it is possible to compare both the speed efficiency and the overall efficiency in solving the problem.

5. Experiences and Results

The Adessowiki system is being used as an educational environment and as a collaborative tool for writing articles and other scientific texts since its first release in 2008. Due to the image processing tools currently available in the system, the experience of teaching, learning and collaborative document writing is specially interesting for that domain.
5.1. Collaborative writing

The Adessowiki has been used as a collaborative tool for writing scientific papers, book chapters and thesis. As any common wiki, it is an excellent tool for collaborative writing. Additionally, after finishing the collaboration, the Adessowiki server is able to post-process the markup text and generate the final document in the style required by the conference or by the publisher, thanks due to reStructuredText markup specification. During the post-processing, the server system can generate PDF, postscript or LaTeX source files, even applying stylesheets (such as LaTeX stylesheets). It is not what happens in common wikis, where the markup text must be taken to external tools for post-processing.

Examples of scientific paper written in Adessowiki can be found in http://www.adessowiki.org/ [5, 6, 7]. During the writing process, all the authors in collaboration were able to see and modify the text and also the image processing code in the same wiki page. A dissertation that compared several watershed algorithms was also written using Adessowiki [8].

Some image processing book chapters [9, 10] were also prepared using Adessowiki. In addition to text content, such chapters included a large set of image processing examples. Image processing codes were embedded in the text file and the figures of each chapter were generated by the codes, as opposed to the usual way of producing such examples, where the image processing code that generates the images for the example and the text comments for the example are written in separate files. In all chapters the authors were not worried about presentation issues, since the stylesheets in post-processing were able to generate the pages of each book chapter in the style required by the editors.

5.2. Computer science courses

Besides collaborative document writing, Adessowiki was used as an environment for graduate courses, comprising not only general information about the course (agenda, grades, bibliography), but also solutions to proposed problems edited as wiki pages by the students. Solution pages contained not only source code, but also images resulting from code execution and documentation.

Some of the teaching experience can be found in http://www.adessowiki.org/ (links beginning with “courses”). The first teaching experience occurred in 2008 at Unicamp, as an image processing course for undergraduate students. During the first semester of 2009 three lecturers used Adessowiki in parallel for three graduate and one undergraduate course. The methodology applied for each course was the same as that applied for the image processing course lectured at Unicamp the year before: periodical problems and open solutions. Although the professors proposed different problems to their students, the students from one institution could reuse and comment solutions given by students from other institutions, allowing students to explore possible solutions for their own problem by exploring solutions given by other students for other problems.

In the year of 2010, not only the number of courses conducted using Adessowiki has grown, but also their variety. Courses such as robotic, digital signal processing and microcomputing laboratory has made use of the Adessowiki environment, if not in its totality, at least as an additional resource. Specially significant was a course of Image processing using GPU. The course uses a specialized execution sandbox to enable the implementation and evaluation of image processing GPU-based parallel algorithms.

6. Conclusions

This paper presented the main concepts of a new collaborative environment, suitable for the generation of executable papers. It allows the documentation, coding and executable results to be stored in the same document and to be accessed by a web browser. The environment, called Adessowiki is experimentally available since 2008 at http://www.adessowiki.org. This environment is a mixture of collaborative document writing, numeric software development, teaching platform, collection of algorithms with source code, among other applications. We have presented the main benefits of putting together three main concepts: collaborative wiki environment; programming code with its executable results; centralized server to store document, code, data and executable results. In summary, it makes possible for researchers and reviewers to manipulate, execute, check, and validate the data and code presented by a paper.

Experiments of Adessowiki as educational environment showed that the wiki model facilitates not only the instructors tasks, such as the writing of class notes and exercises or the assessment of students performance, but also the
students tasks, since it synchronizes between the student implementation and the outputs (images, graphics, tables, equations and diagrams) automatically generated. Further advantages for the students are: possibility of dialogue between the participants through the inclusion of comments; ability to see solutions of colleagues or toolboxes with visible reusable source code.

The use of Adessowiki as a writing environment accelerated the process of producing scientific texts, due to its easiness for collaboration, image generation and text formatting. We envision that this will be the electronic book for scientific programming where the user can read the paper, but can readily access the programming code, change parameters and data, and see the new results using a simple web browser.

References