Adessowiki
On-line Collaborative Scientific Programming Platform

Roberto A. Lotufo
University of Campinas
(UNICAMP)
Campinas, Brazil
lotufo@unicamp.br

Rubens C. Machado
Renato Archer Information
Technology Center (CTI)
Campinas, Brazil
rubens.machado@cti.gov.br

André Körbes
University of Campinas
(UNICAMP)
Campinas, Brazil
korbes@dca.fee.unicamp.br

Rafael G. Ramos
University of Campinas
(UNICAMP)
Campinas, Brazil
rafaelgramos@gmail.com

ABSTRACT

Adessowiki (http://www.adessowiki.org) is a collaborative environment for development, documentation, teaching and knowledge repository of scientific computing algorithms. The system is composed of a collection of collaborative web pages in the form of a wiki. The articles of this wiki can embed programming code that will be executed on the server when the page is rendered, incorporating the results as figures, texts and tables on the document. The execution of code at the server allows hardware and software centralization and access through a web browser. This combination of a collaborative wiki environment, central server and execution of code at rendering time enables a host of possible applications like, for example: a teaching environment, where students submit their reports and exercises on Adessowiki without needing to install special software; authoring of texts, papers and scientific computing books, where figures are generated in a reproducible way by programs written by the authors; comparison of solutions and benchmarking of algorithms given that all the programs are executed under the same configuration; creation of an encyclopedia of algorithms and executable source code. Adessowiki is an 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.

Categories and Subject Descriptors

D.2.6 [**Computers and Education**]: Programming Environments; K.3.1 [**Computers and Education**]: Computer Uses in Education

WikiSym 09', October 25-27, 2009, Orlando, Florida, U.S.A.

Keywords

collaborative programming, software engineering, wiki

1. INTRODUCTION

This paper presents the Adessowiki environment available at http://www.adessowiki.org. This environment has the main objective of the collaborative development of numerical programming algorithms along with its article or scientific and technical report. The system is composed of a collection of editable web pages forming a wiki. The special part of Adessowiki is the possibility of its pages to embed source code that is executed when rendering the page and incorporating the figures, texts and tables generated to the document. The portions of code are executed on the server, allowing centralization of hardware and software configuration, and also freeing the user to develop its applications only using a web client, under a PC, PDA or even browser-enabled cellphones.

The combination of a collaborative wiki environment, central server with software and data and execution of source code at render time enables an unnumbered possibilities of applications. This project is focused on the image processing area, as it is the speciality of the research team, however it may be used on nearly any area of scientific computing and engineering. Some of the anticipated cases of use are: teaching environment, where students submit their reports and exercises on Adessowiki without needing to install special software; writing of texts, papers and collaborative scientific computing books, where figures are generated in a reproducible way by programs written by the authors; collaborative research environment, allowing multiple researchers of a group to exchange scientific results along with the algorithms implementation; comparison of solutions and benchmarking of algorithms given that all the programs are executed under the same configuration; creation of an encyclopedia of algorithms and executable source code solving problems with the same data set, easing the task of comparisons between them. The main point of this project is the incorporation of the ability to execute code on the server as a part of the render of a wiki document. The concept is still new and certainly other applications will arise from this environment.

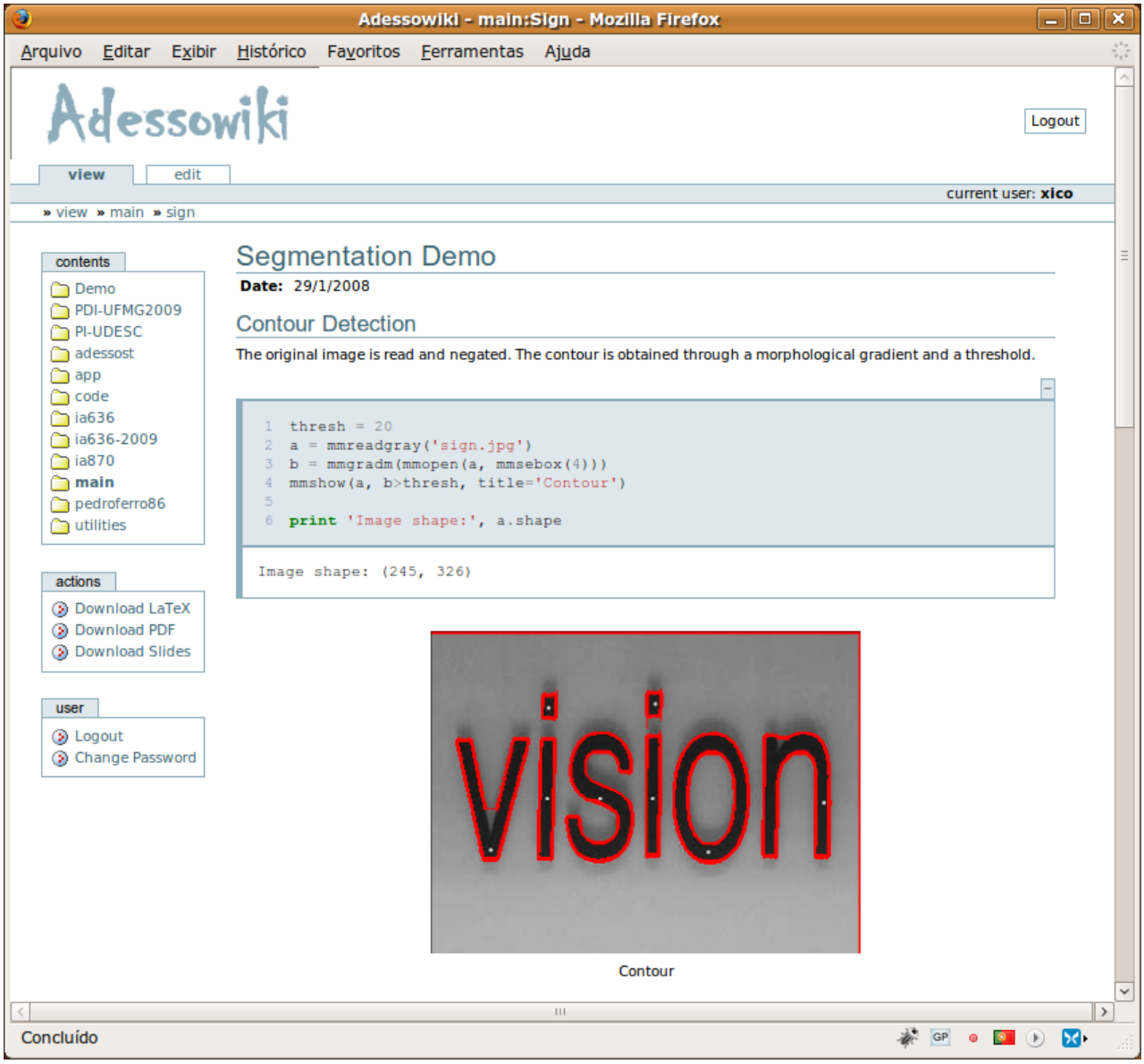

Figure 1: The Adessowiki window

Some similar initiatives of this concept are: literate programming [1, 2] and reproducible research [3]. Adessowiki takes together with these concepts many aspects of collaborative development and Web 2.0 [4] easiness. There is no proposal like this in the literature that the authors are aware, even though wikis have already been studied for the development of software, aiding programmers to write and debug code under a collaborative environment [5, 6]. There is a believe that these concepts may have a great acceptance by the users as well as several similar environments may be developed.

2. ADESSOWIKI OVERVIEW

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] powered by the modern world-wide web. Fig. 1 shows the general appearance of a typical Adessowiki page.

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.

Fig. 2 shows the computational organization of Adessowiki. The system is composed of two separated web servers: the *wiki server*, that serve information from a database and

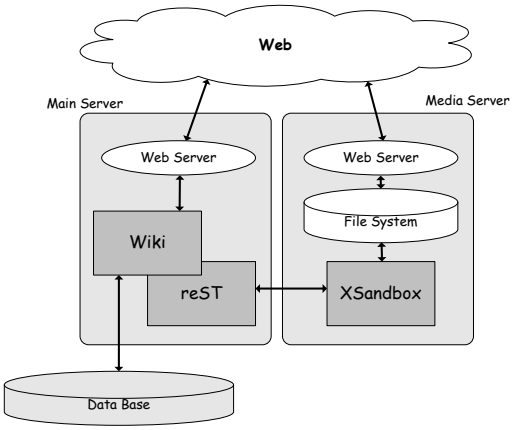

Figure 2: System organization.

never access the local filesystem, 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 execution are made visible in the web by the server. These components are described in the following sections.

2.1 The Wiki

Besides its software development features, Adessowiki may be characterized as a modern and fully-featured wiki [7]. Like other wikis, Adessowiki is composed by a collection of hypertext pages without any structure except those created by the links. This is one of the main points of a wiki: in contrast to most content managing systems, a wiki has no implicit hierarchical structure.

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.

In Fig. 1 the *spaces* that are accessible to the current user (identified on the upper-right portion of the window) are listed in the first left menu (*contents*). This provides shortcuts for quick access to the pages on these sections. Other menus may appear on the left of the page, depending on the user permissions. The tabs on the upper-left also reflects the permissions of the current user and provides functions for editing the page, attaching files, and view the page history.

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 and CVS systems.

Adessowiki has a built-in access control system, with permissions for users and groups. As exposed previously, permissions are specified by space, whereas users and groups

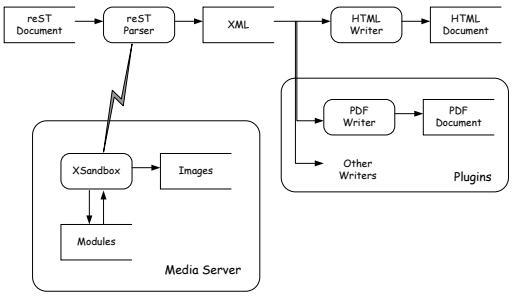

Figure 3: Document processing.

themselves are global information accessible to all the spaces. 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.

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 [8].

2.2 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. 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.

2.2.1 Execution of Python Code

A source document example is presented below:

```
=================
Segmentation Demo
=================

:Date: 29/1/2008

Contour Detection
=================

The original image is read and negated.
```

```
The contour is obtained through a
morphological gradient and a threshold.

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

    thresh = 20
    a = mmreadgray('sign.jpg')
    b = mmgradm(mmopen(a, mmsebox(2)))
    mmshow(a, b>thresh, title='Contour')

    print 'Image shape:', a.shape
```

This simple example is the source code of the page shown in Fig. 1. 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 if 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.

2.2.2 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.

In order to implement the mechanism for the creation of reusable modules, the syntax for code inclusion is extended to indicate the kind of code:

```
.. code:: python module

    def function01():
        return 'function01 return value'
```

If the above code fragment is contained in the page *page01* that belongs to section *space01*, then the function can be called from another page using, for example, the following code fragment:

```
.. code:: python

    from space01.page01 import function01

    v = function01()
    ...
```

2.2.3 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. An example follows:

```
.. code:: cpp module

    #include <sys/sysinfo.h>

    int sysinfo()
    {
        struct sysinfo info;
        sysinfo(&info);
        return (int)info.loads[0];
    }

.. code:: cpp header

    int sysinfo();
```

If the above code is from page *info* in section *myspace*, we can use it like this:

```
.. code:: python

    from myspace import info

    print 'System one-minute average load is',
    print info.sysinfo()
```

The public area of the Adessowiki site has many examples of pages with Python and C/C++ code.

2.3 The Execution Sandbox

The code fragments embedded in a wiki page are executed through the Execution Sandbox, **XSandbox**: 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.

3. USE CASES

The integrated collaborative environment containing documentation, programming code and execution results creates several possibilities of new applications. We will describe some of the possible applications and use cases where Adessowiki can be used.

3.1 Image Processing Courses

Wikis have already attracted the attention of some educators to its potential for teaching purposes [9]. The same kind of experience have already attracted educators previously with the Khoros system and other teaching environments [10, 11]. A wiki can be used in many ways during a course: students can save their summaries and notes, it

can be used for deposit their homework and exercises. Intraclass collaboration is encouraged as the homeworks can be consulted by everyone and as everything is logged with time and text comparison between editions, plagiarism is discouraged. The collaborative nature of the wiki creates a teamwork environment where learning with the teacher and with the colleagues is the goal.

In the case of Adessowiki, the possibilities are even greater, because not just text, but also executable code and its results can be inserted in the documentation. Laboratory activities of Image Processing courses, or other Computer Science courses, can use Adessowiki as a platform, and programming activities requested by teacher can be developed by the students on the wiki. The output of the program can be seen in the wiki pages. Adessowiki is also a programming environment where there is no need to configure any software as everything is executed in the server. Unlike automatic code submission, used in some courses, which are difficult to configure and difficult to maintain, the programing code in the Adessowiki is stored together with its execution results, enabling easy verification if the program works correctly or not. It will be possible, in the future, to expand the system to allow automatic code and automatic plagiarism verification. Scores and teacher comments can be inserted the same page, next to the code. In Adessowiki the lectures materials prepared in the wiki, can be saved in presentation slides or as a document format. Additionally the materials and programing code solutions can be used in the next course enabling eventual improvements of already known solutions.

3.2 Software Development

Adessowiki can be used to support collaborative software development. Actually, wikis are already been used to support documentation and Software Engineering processes development [5]. On Adessowiki, the code can also be developed in collaborative way, with the advantage that code, documentation and execution results can live on the same page, facilitating comprehension avoiding the lack of synchronism between documentation and code so common in practice and avoiding 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.

3.3 Collaborative Scientific Writing

One of the central philosophies of wiki is that text is created in a collaborative way, so the collaborative writing of books is natural to happen in a wiki. The Adessowiki includes additional resources to facilitate the writing of books where embedded programming code can generate the figures, tables and graphics. We can envision that this will be the future of electronic books where text, data, programming code and results can be accessed in an integrated environment ready for experimentation on parameter changing and code inspection increasing the speed for knowledge learning. Adessowiki has tools to save the documents in Latex, RTF and PDF format enabling simple integration with the required book editor format. The chapter 8 of [12] was developed using Adessowiki.

3.4 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 solutions that can be proposed 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.

As an example we can mention a typical program for counting cells used in a pathology lab. Normally the lab requires image processing tools that must be installed, configured and maintained. Not all such labs have the expertise and resources to keep this kind of software in a running condition. If this automatic counting cell solution is in Adessowiki, most of these problems would be solved.

3.5 Algorithms Benchmarking

An important feature about Adessowiki is that the embedded programming code is executed in a centralized and specific hardware/software environment. This 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. The evaluation of algorithms is a particular problem that the image processing scientific community faces for a long time. Images are rarely made available as reference data base for evaluation of the algorithms that, in addition, are usually complex and hard to test. Although we can see an increasingly practice of some authors to publish their source code and their data, they are not always in a readily format where the user can experiment immediately as with Adessowiki.

3.6 Algorithm Repository

Maybe one of the most useful application of a wiki is for knowledge repository, being Wikipedia the best example where in a very short period of time could gather so many structured information in different languages covering articles about the most diverse themes. The Adessowiki, being capable of supporting programming code, 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. The page could also contain performance comparisons with other algorithms solving the same problem.

3.7 Reproducible Experiments

One of the basic principles of experimental science is that experiments are reproducible, that is, given the same conditions and followed the same steps, the same results will be obtained. This principle is one of the basis of the scientific methodology. In numeric and scientific programming, it is necessary to guarantee that not only the code be the same, but also the input data and other factors that may influence

compilation and execution be the same.

The main differential point of this project is the incorporation of the ability to execute on the server the code that is part of article on the wiki document. Some similar initiatives of this concept are: literate programming [1, 2] and reproducible research [3]. The Adessowiki adds to this concept many aspects of collaborative development and the Web 2.0 features.

Adessowiki is another step on a sequence of projects related to developing scientific software by the researchers proponents of this. The first of them was a toolbox of mathematical morphology for MATLAB that raised the needs and motivated for the further projects.

The project that followed the toolbox, called **Adesso**, a computational environment for fast development of computer vision solutions [13], [14], developed between 1998 and 2000, that also received support from an institution that promotes scientific research, created the basis for code and documentation generation that were used to build toolboxes for platforms like MATLAB and Python.

Next, the **Adessoweb** project - under support of the same institution - created a distributed environment to operate **Adesso** and centered resources on a server, accessible through the Internet. The Web 2.0 concept, and the experience acquired with the developing of **Adessoweb** motivated this new project. Aside with the code generation systems, the research team developed several libraries and applications for image processing [16, 12, 13, 14, 15]. Together, this experience translated into choosing the Django [8] web application framework and the Python language, for its features, community and previous experiences.

4. FUTURE DEVELOPMENT

Currently Adessowiki is a platform where we can run several "Proof of Concept" experiments. There are still many issues to be solved in the near future, such as, scalability of the system to allow thousands of simultaneous users, programming version control, sandboxes with special hardware accelerator machines, user interface software development, copyright issues, among others. We believe that all these aspects will evolve as the project expands. We also believe that the system can grow with the help of the user community.

5. CONCLUSIONS

This paper presented the main concepts of a new collaborative environment for the development of numeric programming code that allows the documentation, coding and executable results stored in the same document accessed by a web browser. The environment, called Adessowiki is experimentally available at http://www.adessowiki.org. 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. This environment is a mixture of collaborative document writing, numeric software development, teaching platform, collection of algorithms with source code, among other applications. 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 see the new results using a simple web browser.

6. REFERENCES

[1] D. E. Knuth. "Literate Programming". The Computer Journal, 27(2):97-111, 1984.

[2] N. Walsh. "Literate Programming in XML". Oct. 2002. http://nwalsh.com/docs/articles/xml2002/

[3] J. B. Buckheit, D. L. Donoho. "WaveLab and Reproducible Research". Dept. of Statistics, Stanford University, Tech. Rep. 474, 1995. http://www-stat.stanford.edu/ donoho/Reports/1995/wavelab.pdf

[4] Tim O'Reilly. "What is Web 2.0". O'Reilly Media, Sep. 2005. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[5] W. Xiao, C. Y. Chi, M. Yan. "On-line Collaborative Software Development via Wiki". In *Proceedings of the 2007 International Symposium on Wikis*. ACM, San Diego, California, USA, 2007, 177-183.

[6] A. Aguiar, G. David. "WikiWiki Weaving Heterogeneous Software Artifacts". In *Proceedings of the 2005 International Symposium on Wikis*. ACM, San Diego, California, USA, 2005, 67-74.

[7] Stewart Mader. "Wikipatterns". Wiley, December 10, 2007.

[8] Adrian Holovaty, Jacob Kaplan-Moss. "The Definitive Guide to Django: Web Development Done Right". Apress. December 6, 2007.

[9] Kevin R. Parker, Joseph T. Chao. "Wiki as a Teaching Tool". Interdisciplinary Journal of Knowledge and Learning Objects, 2007. v. 3. p. 58-72.

[10] R. A. Lotufo et al. Interactive DSP Course Development/Teaching Environment. In: 1997/IEEE Intern. Conf. on Acoustics, Speech and Signal Processing, 1997, Munich. Proc. of 1997/IEEE Intern. Conf. on Acoustics, Speech and Signal Processing, 1997. v. III. p. 2249-2252.

[11] R. A. Lotufo, Ramiro Jordan, John Rasure. "Teaching Image Processing with Khoros". In: First IEEE International Conference on Image Processing, 1994, Austin, Texas. Proc. of First IEEE International Conference on Image Processing, 1994. v. S/N. p. 506-510.

[12] R. A. Lotufo, R. Audigier, A. V. Saúde, R. C. Machado. Morphological Image Processing. In: Qiang Wu; Fatima A. Merchant; Kenneth R. Castleman. (Editors). Microscope Image Processing. 1 ed. : Elsevier Academic Press, 2008, p. 113-158.

[13] R. C. Machado. "Adesso: Ambiente para Desenvolvimento de Software Científico". Dissertação de Mestrado, Faculdade de Engenharia Elétrica e Computação, UNICAMP, Brasil, Junho de 2002.

[14] R. C. Machado, R. A. Lotufo, A. G. Silva, A. V. Saúde. "Adesso: Scientific Software Development Environment". Journal of Computer Science and Technology, April 2003.

[15] A. G. Silva, R. A. Lotufo, R. C. Machado, A. V. Saúde. "Toolbox of Image Processing Using the Python Language". In: ICIP 2003 - IEEE International Conference on Image Processing, 2003, Barcelona. IEEE Proceeding of ICIP 2003, p. 1049-1052.

[16] Edward R. Dougherty, Roberto A. Lotufo. "Hands-on Morphological Image Processing". SPIE Publications. July 24, 2003.