

IBM Research Report

On-line Collaborative Software Development via Wiki

WenPeng Xiao, ChangYan Chi, Min Yang

IBM Research Division

China Research Laboratory

Building 19, Zhouguncun Software Park

8 Dongbeiwang West Road, Haidian District

Beijing, 100094

P.R.C.



Research Division

Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

On-line Collaborative Software Development via Wiki

WenPeng Xiao
IBM China Research Lab
86-10-58748423
xiaowp@cn.ibm.com

ChangYan Chi
IBM China Research Lab
86-10-58748012
chicy@cn.ibm.com

Min Yang
IBM China Research Lab
86-10-58748544
yangmin@cn.ibm.com

ABSTRACT

Wiki is a collaborative authoring system for collective intelligence which is quickly gaining popularity in content publication. In software development communities, especially for open source and global development teams, wiki is already widely used for documentation and coordination purpose but not programming purpose. This paper presents a new programming approach based on wiki technology by which developers will experience “writing wiki page is writing source code”. Moreover, developers are able to compile, execute and debug programs in wiki pages too. A prototype of such on-line collaborative software development environment, Galaxy Wiki, is developed in this environment iteratively to prove the concept.

Categories and Subject Descriptors

D.2.6 [Software Engineering]: Programming Environments – *interactive environments, programmer workbench*. H.5.3 [Information Interfaces and Presentation]: Group and Organization Interface – *collaborative computing, computer-supported cooperative work, Web-based interaction*.

General Terms

Design, Management, Documentation

Keywords

wiki, collaborative programming, software engineering

1. INTRODUCTION

Wiki, one of the most famous Web 2.0 applications, is encouraging the masses to contribute. A large number of participants can create new pages or modify existing pages through their web browser easily in wiki. Besides enabling people to write content to the web, wiki has also been extended to other domains. For instance, researchers are exploring how to use wiki in software development area, to weave source code and related documentation [1].

Although not well recognized, the open source software communities share a similar participation model with wiki. These communities encourage everyone to contribute their code,

documentation or even idea to a project when s/he is attracted. However, nearly all currently available development tools are designed for core developers [2], and little of them take peripheral developers in to account. As a result, there are lots of barriers to peripheral developers when they want to contribute to a project incrementally, and grow to core developers gradually. For example, it is hard to understand source code because documentation is discrete; it is hard to taste functionality quickly with zero-installation; and it is hard to reuse a library as simply as accessing a HTML page.

Web based on-line development is a possible way to solve the mentioned problem, and benefit developers in the following aspects: a) free from download, install and config development environment in local machines; b) always up-to-dated source code; c) easy to use.

Some implementations of wiki have been enhanced to support specific activities in software development, for instance code reviewing, bug tracking, and functional testing. Trac [14] is an enhanced wiki and issue tracking system for software development projects, integrates an issue tracker allowing creating links and references between bugs, tasks, change sets, files and wiki pages [3]. TWiki [15] also has plug-ins to track eXtreme Programming projects.

However, none of them has touched the programming stage in software development lifecycle. In this paper, we propose a wiki based collaborative development environment named Galaxy which leverages some features of wiki: collaboration, rich context, open and easy to access, and wiki names dynamic links. Besides managing source code and document, it also extends wiki with development functionalities like compiling, executing and debugging. One specific point is that Galaxy is under developing in Galaxy.

The remainder of this paper is structured as follows: After introducing some basic concepts of wiki-based programming in section 2, a preliminary prototype, Galaxy Wiki, is discussed in section 3. Related work is summarized in section 4, and our conclusions and future work are outlined in section 5.

2. PROGRAMMING IN WIKI

Beyond using wiki only as a documentation repository or an activities coordinator, we have sketched an on-line collaborative programming environment based on existing wiki system. In this lightweight and highly collaborative environment, user can write, compile, execute and debug programs in a wiki site, just as what we can do in a traditional IDE like Eclipse.

2.1 Literate Programming in Wiki

Nowadays, source code and documentation of a project are often controlled by Software Configuration Management (SCM) tools (e.g. CVS) and Content Management (CM) tools (e.g. wiki) respectively. It's a big challenge for development teams to keep these artifacts synchronized throughout the whole development life-cycle in order to preserve its consistency and value.

Wiki has been recognized as an ideal documentation system for software projects [3], since it's a very handy, simple and appealing collaboration tool, which allows multiple programmers to work on the same document concurrently. Full-featured wiki system which target for software development has devised a wiki-based approach to weave and keep in-sync source code and documents together [1]. By rendering source code into wiki pages dynamically, such wiki system allows user to read source code fragment and document content in same wiki page. However, user still cannot create, modify or delete code freely from wiki site as the source code is under the control of SCM.

Support literate programming [4] in wiki is a possible solution for this problem. The term "literate programming" was coined by Donald Knuth and the main idea is to treat a program as a piece of literature, addressed to human beings rather than to a computer. Wiki is designed to use a simplified markup language for formatted text written, so we can leverage this mechanism and use predefined markup tags for source code written. For instance, in order to inline Java source code into a wiki page, we can use the `#!java` closure in MoinMoin wiki:

```
== Hello World ==
```

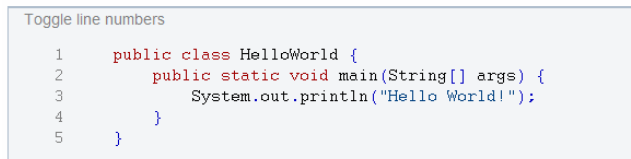
The first program of Java programming.

```
{{#!java
  public class HelloWorld {
    public static void main(String[] args) {
      System.out.println("Hello World!");
    }
  }
}}
```

When the above text is rendered by wiki engine later, source code can be extracted and handled by predefined actions, for example syntax highlight and automatic indenting (Figure 1).

Hello World

The first program of Java programming.



```
Toggle line numbers
1  public class HelloWorld {
2      public static void main(String[] args) {
3          System.out.println("Hello World!");
4      }
5  }
```

Figure 1. Writing code in a wiki page.

By supporting producing, modifying and reviewing source code in Wiki, developers can also benefit from some key features of Wiki, collaboration and easy of access. A developer can contribute code wherever and whenever, as long as he can access the Internet.

2.2 Page Oriented Software Engineering

Currently, although we can download tons of code from an open source portal like SourceForge [16], how to reuse them smoothly in our own project is still a big issue. Firstly, it is not so easy to locate the proper components and classes with reusable potential. Secondly, it is time exhausting to download the whole package and import it into our own projects. Last but not least, it is hard, if possible, to keep the imported code always up-to-date after its author has fixed a critical bug.

In order to solve above problems, page oriented software engineering is proposed. It provides an on-line environment where code and documentation of a class are organized as a wiki page, and a software project is also defined as a wiki page by referencing relevant class pages and resources. All of the pages can be created by one user and improved upon by others. Therefore, in order to locate reusable classes, a user can start from the wiki site's searching engine. After reading and testing, he can import them into his own project by simply referencing corresponding wiki page names. In this way, the problem of up-to-date can also be solved.

The page oriented software engineering is well suited to retrieve, combine and reuse source code available on the Internet, which is especially useful for global software development teams and open source project teams.

2.3 On-line Read-Eval-Print Loop

Many programming languages come with a read-eval-print loop (REPL), which allows you to type in code line by line and see what it does. This feature facilitates incremental program development, and is quite useful for prototyping, experimentation, and debugging code. A REPL provides a conceptually simple, yet powerful framework for interacting with program components as they are being developed [5].

The term REPL is often used to refer to an interactive environment for interpretive languages like Lisp, Scheme and Python, but the same concept can be applied to the similar environments in which programmer can easily access the source code of a component, flexibly debug or test a program, and quickly experiment with a new library. Programming in Wiki show the REPL characters for on-line collaborative software development:

- **Read:** Source code of each component is rendered in a wiki page with rich formatted documentation, so user can easily read and change both source code and documentation within a Web browser.
- **Eval:** Source code of all components are extracted, compiled (if necessary) and executed when receiving user's command from a wiki page.
- **Print:** Yielded output is pretty formatted to make it easier to understand, and sent back to browser as a wiki page.

With this wiki based read-eval-print loop (Figure 2), a user is able to interactively create a piece of code, execute it and observe the result.

2.4 Scenarios

Founded on aforementioned concepts, some valuable uses with strong requirements of interaction and collaboration in software development engineering can be implemented.

- **Pedagogy:** A textbook of teaching programming language to beginning student can be organized in a wiki site. Every source code example in the textbook can be immediately read and tested in a wiki page. Throughout the tutorial, the learner is encouraged to come up with own examples and test them interactively in customized wiki pages.
- **Exploration:** A reusable library developed by an open source community or global software development team can be published in a wiki site. Anyone who wants to reuse this library can explore API, documentation or sample code of this library in different wiki pages. The explorer can learn the essential features of this library more quickly if he can conveniently conduct simple experiments by copying and testing sample code in a wiki page.
- **Demonstration:** An agile software development team that advocates delivering valuable software to customers can settle their nightly building demonstration in a wiki site. The customer is able to run and taste it from wiki page with zero configurations.

3. GALAXY WIKI

What we want to figure out is how we might adapt wiki to a lightweight and highly collaborative on-line programming environment. This notion is quite new, and the proper requirements and design for such environment is still not so clear. The lessons we have learned from our preliminary prototype, Galaxy Wiki, have helped illustrate some basic requirements and an architecture design, which we describe in this section.

3.1 Overview of Galaxy

Galaxy Wiki is an on-line collaborative development environment based on wiki and Java technology in which user can conveniently initialize or import a Java project, create or modify a Java class, and execute or debug a Java program. Combined with the existing powerful functions of wiki, e.g. collaboration and documentation, this environment is expected to support software communities to develop Java programs in fully on-line manners.

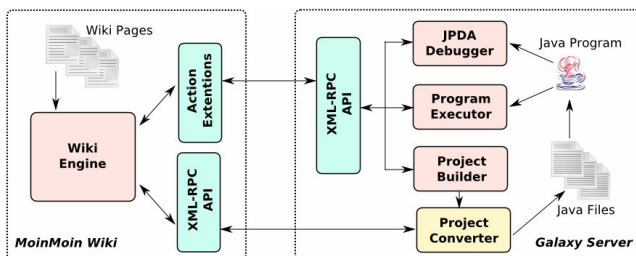


Figure 2. Galaxy architecture.

There are a lot of wikis available to use today and many of them are well designed for extensibility. MoinMoin [10] is chosen by Galaxy as it is an advanced, easy to use and extensible wiki engine with a large community of users. Figure 2 shows the key components of Galaxy as well as their interconnections.

Wiki-based software development is the key idea behind Galaxy. This idea is encouraged by the phenomenon that developers work for the same project are located in different cities, different countries, or different hemisphere [3], and the observation that on-line tools is playing more and more important roles on communication and coordination for such distributed teams.

In current implementation of Galaxy, a programmer is able to manipulate Java project, package, class and library in wiki site from the artifacts perspective. In addition, a programmer is also able to compile, execute and debug Java programs in wiki site from the development perspective.

3.2 Rich Pages

When we use traditional integrated development environment (IDE) like Eclipse [13] to develop a standard Java project, we have to face many concepts such as project, package, class and library. Fortunately, all of them can be congruously mapped to a simple object in Galaxy: wiki page.

There are three types of special pages in Galaxy which are endowed with richer functionalities than other pages:

- A **project page** describes all details information of a project.
- A **class page** defines a class, which is the basic unit of a project.
- A **library page** depicts a library, which can be imported by a project.

A Java project can be easily created by generating a new project page in Galaxy. In this wiki page, user can specify what libraries should be imported into this project, and how all relevant classes are organized into different packages (Figure 3). Moreover, user can also point out the execution entry of this project, which is usually a public class containing the static main function.

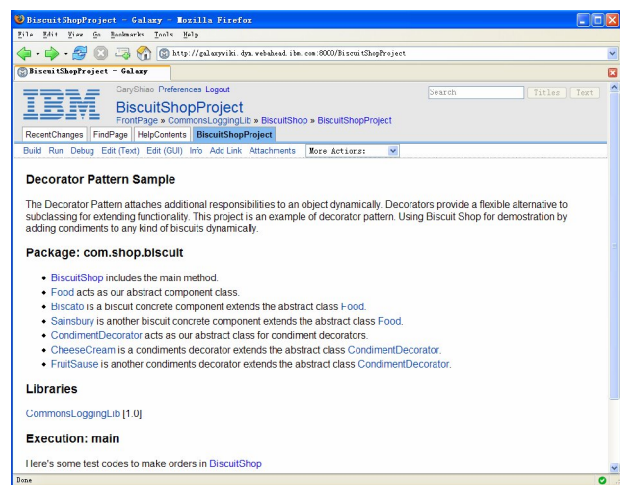


Figure 3. Project page in Galaxy.

Correspondingly, a Java class can be created by generating a new class page in Galaxy, which is often a combination of source code and documentation (Figure 4). In this way, a programmer is able to read the requirement documents or design documents

conveniently while coding in wiki, and the relationship between document and code can be established and persisted.

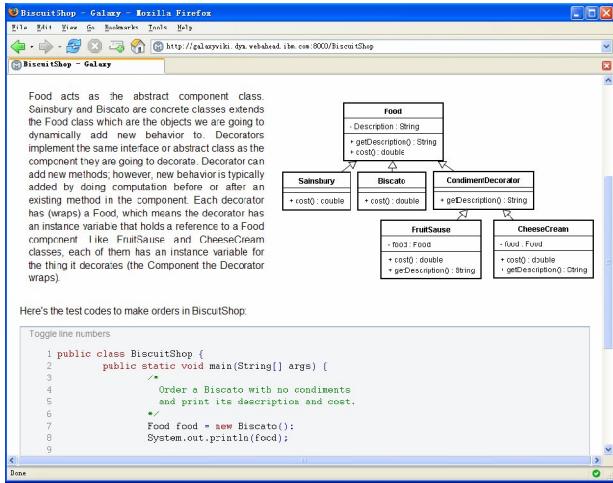


Figure 4. Class page in Galaxy.

We have noticed that some Java archive (JAR) files (e.g. commons-logging.jar) are often required by many projects, and the simplest solution of importing them into each project is not so efficiently. More ideal solution is depicting each reusable JAR file in a library page, and linking with it in the project page. With this approach, a library maintainer can create a library page and upload JARs as attachments of this page, and assign each of them a proper version number (Figure 5). If there is any project that needs this library, the programmer can simply reference the wiki name of this library, and specify the version number in the project page (Figure 3).

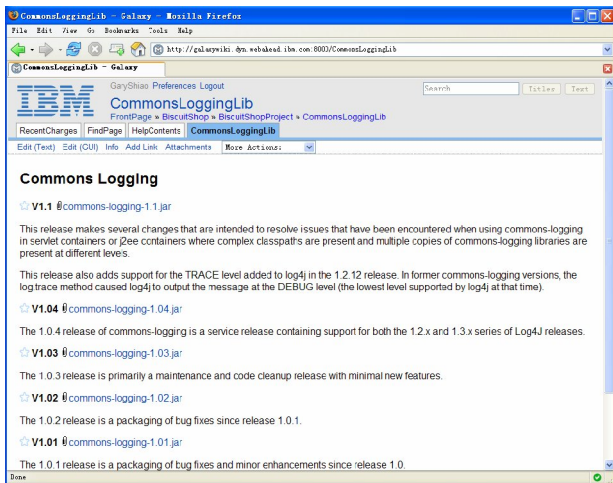


Figure 5. Library page in Galaxy.

3.3 Compilation and Execution

Weaving source code and documentation in single wiki page is not a quite new idea [1]. The further step that Galaxy has moved in wiki aided software engineering is providing an always on-line environment in which programmers are able to not only navigate,

create, modify or delete source code, but also compile, execute and debug Java programs.

MoinMoin provides extension mechanism to extend actions. Galaxy defines three subsidiary actions and binds them to each project page, namely **Build**, **Run** and **Debug**. They can be triggered by user from the action bar of MoinMoin (Figure 6). These actions either produce some output based on page contents (run), or implement functions that are not related to viewing a page (build and debug).

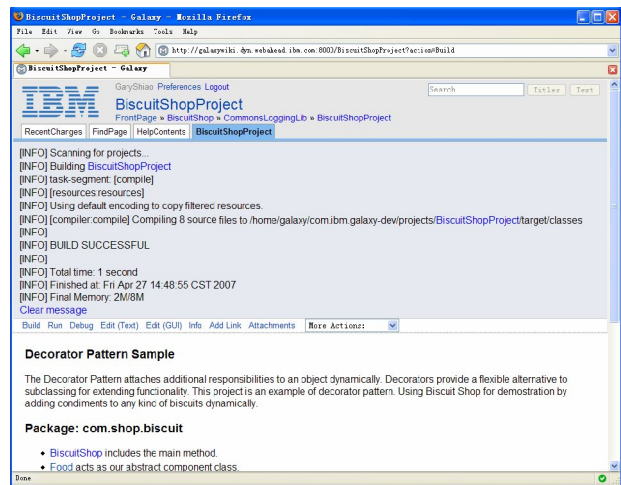


Figure 6. Build project in Galaxy.

After putting all source code and documentation into class pages, and properly organizing them in a project page, user can trigger the build action from his project page in order to compile it and observe the outputs in the same page. The requested project page will be parsed at server side, and all relevant class pages will be located and converted into Java files. There are different approaches to convert a class page into a Java file: (a) simply extracts source code from specific markup and remove other wiki contents, (b) fully converts all wiki contents except code into Java comments and keeps source code untouched. Galaxy uses the latter one because it is possible to convert a Java file back to a wiki class page.

One mission of Galaxy is proposing a clear definition of what the project consisted of, an easy way to share classes and JARs across several projects, and a standard way to build projects on-line (specific in wiki now). Apache Maven [11], a powerful Java building system, has the similar goal for traditional off-line Java projects and is used by Galaxy to simplify the build processes when all Java files have been converted. The build action requested by user on a Galaxy project page activates the compilation chain at server side. First, a corresponding project object model (POM) required by Maven will be generated, and converted Java files will be copied into proper directories. Next, Maven will be executed to perform real building process. Last, the compilation output yielded by Maven will be redirected to browser and showed in project page, which is illustrated in Figure 6.

A successfully building denotes that user can emit the run action from project page subsequently. When the request is received and handled by Galaxy server, a proper runtime environment will be

constructed, in which the specified execution entry is able to be launched according to the description of project page. Similarly, the output of this program is also redirected to browser and showed in project page (Figure 7).

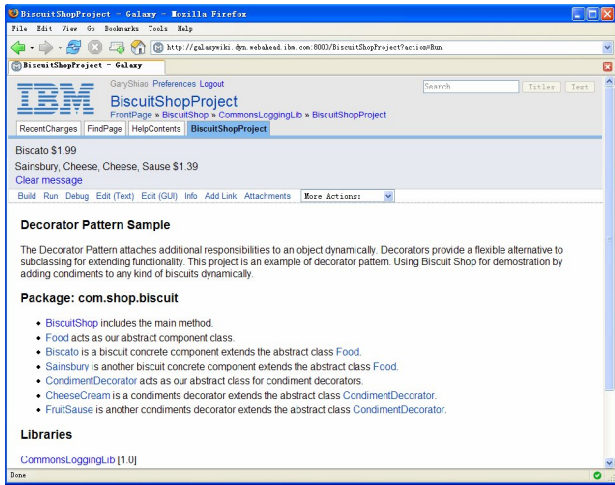


Figure 7. Run project in Galaxy.

In order to support programming in the large, we are working on building a conventional debugger base on the Java Platform Debugger Architecture (JPDA) in Galaxy, which also can be easily invoked by user from a project page. When a project page is executed in the debugging mode, more actions will be dynamically bound to this page which allows user to control behaviors of the program to be debugged. Except that the debug command is inputted in a wiki page (by actions) and the debug information is outputted to a wiki page, there is no any other difference for a programmer to debug a program in Galaxy and to debug a program in console based debugger like JDB.

3.4 IDE Integration

Although Galaxy provides wiki based interfaces for users to access most of its functionalities conveniently, it is not very efficient for core developers who always work in client-side IDEs like Eclipse. IDE integration is a high-priority planned activity of Galaxy. We have developed a plug-in for Eclipse to access the project page, class pages and library pages stored in Galaxy, and convert them into Java project, Java files and JARs respectively in Eclipse (Figure 8).

XML-RPC [12] is a remote procedure call protocol which uses XML to encode its calls and HTTP as underlying transport mechanism. MoinMoin Wiki exposes an API via XML-RPC which allows programmatic inspection, retrieval, and modification of its contents. Using this API, the plug-in running in Eclipse is able to download a project page from Galaxy Wiki, and create an equivalent Eclipse Java project in current workspace. Simultaneously, the relevant class pages will be addressed and downloaded according to the package sections of a project page. Currently, a class page is transformed into a Java file by simply converting all wiki contents except source code into Java comments or Javadoc tags.

As long as a Galaxy project is imported into local workspace successfully in Eclipse, we are able to operate on it freely in

Eclipse. For example, we can create new packages or classes, modify existing source code, and even change settings of projects. Our seamless integration between Eclipse and Galaxy is able to synchronize the changes happened in Eclipse to Galaxy Wiki and vice versa.

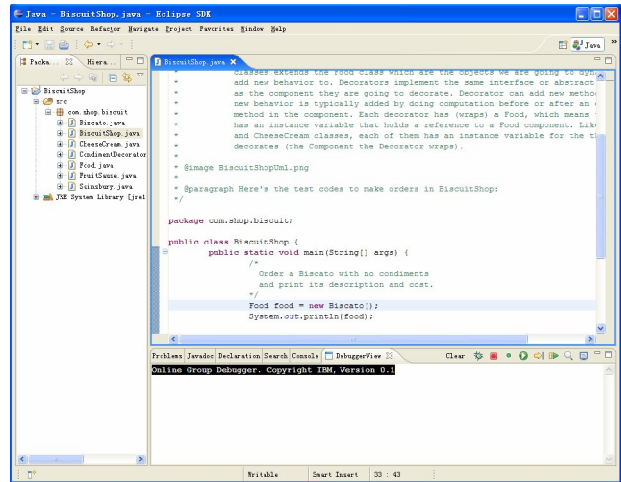


Figure 8. Access and convert wiki pages in Eclipse.

3.5 Iteratively Develop Galaxy

In order to experiment Galaxy in a real software project, we are developing Galaxy project in Galaxy Wiki. A mentioned project page is created as the homepage of this project, where programmers can see what packages and classes are included in this project, and what libraries are imported into this project. Each library specified in the project page can be addressed by a library page, in which a JAR file is uploaded as an attachment and a specify version number is assigned. Equivalently, each class referenced in the project page can be accessed from a class page which includes not only documentation but also source code.

Programmers of Galaxy use either Eclipse or browser to manipulate their code and documentation. Through this approach, the functionalities of Galaxy are developed iteratively by the whole team on-line.

4. RELATED WORK

Global Software Development (GSD) is a kind of software work undertaken at geographically separated locations across national boundaries in a coordinated fashion involving real time and asynchronous interaction [6].

GENESIS (GEneralised eNvironment for process management in cooperative Software Engineering) intends to develop an Open Source platform that supports co-operation and communication among software engineers belonging to distributed development teams involved in modeling, controlling, and measuring software development and maintenance processes [2].

Many open source projects have a web-based, portal-style Collaborative Development Environment (CDE) which integrates project tools [2]. Such tools including project homepage, discussion forum, mailing list, bug tracking, file release system, and version control system. These web-based CDEs are a key part of active open source projects.

Wiki-based software documentation is the key idea behind XSDoc [7]. It is an open and extensible documentation infrastructure based on wiki and XML technologies that ensures the semantic consistency between different kinds of contents, namely source code, models and documents.

JOSH [8] develops a web-based interpreter which executes Java fragments and relieves the learner from programming all the extra code. The JOSH approach can be summarized as follows: first compile code fragments externally, then execute them externally.

Pages in WubHub [9] containing content, presentation logic, data conversion, or computational functions which can be woven together in an iterative way by distributed community. In contrast with traditional wikis, pages are executable, and can interoperate with each other by passing and returning data structures of known type, such as messages, URLs, or locations.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a wiki based environment in which programmers can collaboratively write source code, build projects, execute programs, and debug defects. From our experience with wiki-based programming and the preliminary prototype, we are convinced that the idea of page oriented software engineering is very attractive and promising for large software communities. Galaxy Wiki also helps on stepping forward to the direction of collective programming.

The design and prototype we have outlined for the wiki-based programming leaves much room for future work. Current version of Galaxy requires improvements on execution phase to support programs with rich user interfaces and heavy interactions. We plan to pilot how to use Galaxy to develop more complex Java applications, for example Eclipse's plug-in. We also plan to provide more facilities and functionalities in Galaxy to enhance the on-line programming user experiences, such as automatic indentation, keyword highlighting and code formatting. Another concern is to enable different projects to be executed at the same time.

Same concept and infrastructure can be applied to other languages like Python, or application frameworks like Ruby on Rails. We are very interested in exploring how to use Galaxy to develop applications with dynamic languages for maximum development efficiency.

6. REFERENCES

[1] Aguiar, A., and David, G. WikiWiki weaving heterogeneous software artifacts. In *Proceedings of the 2005 International*

Symposium on Wikis. ACM, San Diego, California, USA, 2005, 67-74.

- [2] Boldyreff, C., Nutter, D., Rank, S., Smith, M., Wilcox, P., Dewar, R., Weiss, D. and Ritrovato, P. Environments to Support Collaborative Software Engineering. In *2nd Workshop on Cooperative Supports for Distributed Software Engineering Processes*. Benevento, Italy, 2003, 25-28.
- [3] Al-asmari, K. R., and Yu, L. Experiences in Distributed Software Development with Wiki. In *Proceedings of the International Conference on Software Engineering Research and Practice & Conference on Programming Languages and Compilers (SERP 2006)*. CSREA Press, Las Vegas, Nevada, USA, 2006, 389-293.
- [4] Knuth, D. E. Literate Programming. *Comput. J.*, 27, 2 (1984), 97-111.
- [5] Allen, E. E., Cartwright, R., and Stoler, B. DrJava: a lightweight pedagogic environment for Java. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2002)*. ACM, Cincinnati, Kentucky, USA, 2002, 137-141.
- [6] Carmel, E., and Agarwal, R. Tactical Approaches for Alleviating Distance in Global Software Development. *IEEE Software*, 18, 2 (2001) 6.
- [7] Leuf, B., and Cunningham, W. *The Wiki Way: Quick Collaboration on the Web*. Addison-Wesley, 2001.
- [8] Diehl, S., and Bieg, C. A new Approach for Implementing stand-alone and Web-based Interpreters for Java. In *Proceedings of the 2nd international conference on Principles and practice of programming in Java*. ACM, Kilkenny City, Ireland, 2003, 31-34.
- [9] Cheyer, A., and Levy J. A Collaborative Programming Environment for Web Interoperability. In *1st Workshop on Semantic Wikis (SemWiki '06)*. Budva, Montenegro, 200.
- [10] MoinMoinWiki. <http://moinmoin.wikiwikiweb.de/>
- [11] Apache Maven. <http://maven.apache.org/>
- [12] XML-RPC. <http://www.xmlrpc.com/>
- [13] Eclipse. <http://www.eclipse.org/>
- [14] Trac. <http://trac.edgewall.org/>
- [15] TWiki. <http://twiki.org/>
- [16] SourceForge. <http://sourceforge.net/>