Pier – The Meta-Described Content Management System*

ESUG Innovation Technology Awards 2007

Lukas Renggli, renggli@iam.unibe.ch
Software Composition Group, University of Bern, Switzerland

March 14, 2008†

Abstract

Pier is the second generation of an industrial strength content management and application framework. Pier is written with objects from top to bottom and it is easily customized to accommodate new requirements. Pier is based on Magritte, a powerful meta-description framework. Pier has proven to be very powerful in the combination with Seaside, to enable easy composition and configuration of interactive web sites through a convenient web interface without having to write code.

Keywords. Content Management, Meta-Driven Application Development, Web Application, Visual Programming, Seaside

Description

Pier is the second generation of a fully object-oriented implementation of a meta-described content management and Wiki system.

Pier still inherits a lot of Wiki functionality from its first version called SmallWiki [1, 2]. Over the years Pier has grown into a fully-fledged application and content management framework. It is written with extensibility in mind and it can be customized easily to accommodate new needs. Pier is based on the meta-framework Magritte [3, 4] to enable building end user interfaces declaratively, to provide sophisticated search queries and automatic persistency.

The object model of Pier does not depend on a specific web or user interface framework. In fact, the model is completely separated from any view related code. This means that you can write different user interfaces that work on the same model data merely by writing a new meta-interpreter.

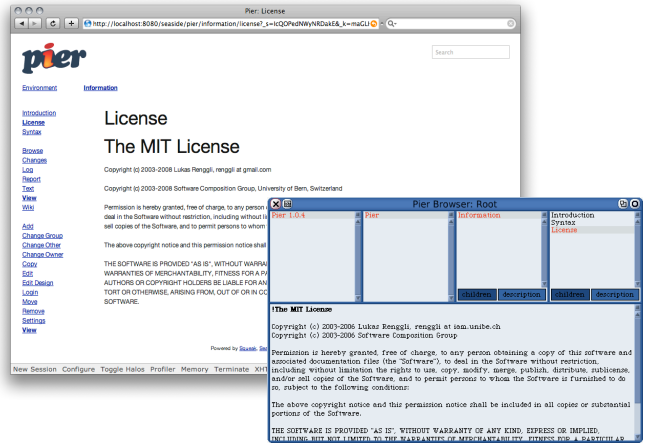

Figure 1: Pier after the installation in the web browser (left) and in Squeak Morphic (right).

The most popular user interface is the one built on top of Seaside [5, 6], a framework for developing sophisticated web applications in Smalltalk. As seen in Figure 1, there is also a simple OmniBrowser [7] based view running in Squeak Morphic. It is possible to use the different views on the same model simultaneously.

It is easy to build new views or to extend existing ones, because every part of Pier is meta-described. If new output formats or views have to be implemented, all that has to be done is to write a new meta-interpreter and some glue code to connect the different software components. This meta driven architecture makes the system easy to extend and change, as we don't have to repeat the generation of different user interfaces over and over again. Changing the meta description of an entity is enough to make all its users aware of the change. Meta-descriptions are used for almost all parts of the Pier framework: to build of viewers, editors

*We thank Orla Greevy for her feedback on a draft of this proposal.

†This is an updated version of the original proposal from June 29, 2007.

and reports, to validate data, to process queries, to implement object persistency, and to initialize objects.

Figure 2: The web site of Seaside. It is built on top of Pier and features interactive Seaside example applications. The news feed is automatically built from community blogs.

The following section summarizes the most important features of the Pier content management system:

Object-Oriented Design

Pier features a fully object oriented and meta-described domain model. As an example, the content of the pages is parsed and stored as a tree of different entities representing text, links, tables, lists, etc. All these entities are meta-described, allowing for example the search engine to display all pages containing a table.

Meta Driven

Pier is built on top of Magritte, a key component to avoid the duplication of code and to enable adaptability, extensibility and (end user) customization of the model.

Model-View Separation

The model of Pier is cleanly separated from its views. All functionality can be easily reused in different contexts, by writing a meta interpreter building a new user interface as specified through the meta-descriptions.

Extensibility

Pier is easy to extend: page types, storage mechanism, actions, security mechanism, web server, etc. There is big collection of ready made plug-ins available that can be loaded independently of each other into the system.

Test Suites

Magritte and Pier are both heavily tested. There are more than 1'800 unit tests covering Magritte, and more than 1'300 covering Pier. This makes it easy to change and verify the code and comes in extremely useful when porting to other Smalltalk dialects or when writing extensions.

Open Source

Magritte and Pier are both released under the MIT license which grants unrestricted rights to copy, modify, and redistribute as long as the original copyright and license terms are retained.

Development

Pier is developed in Squeak. It is known to run on Squeak 3.7, 3.8, 3.9 and 3.10. As only prerequisite it depends on the Magritte meta-framework. The web interface requires the latest version of Seaside.

Portability

Pier has been programmed with portability in mind. There are regular ports to *Cincom Smalltalk* and *GemStone Smalltalk* available.

Installation

Pier can be installed with a single click from *SqueakMap* or from *Package Universes*. The installer pulls in all the prerequisites and asks a few questions such as on what port the web server should be started. Within a few minutes users get a working system and are ready to build their web site and fill it with contents.

Plugins

The architecture of Pier is designed for extensibility, therefore a lot of plugins for different purposes exist enhancing the existing functionality or adding new features to the system. Most plugins can be loaded independently of each other[1], so everybody

[1]The official repository provides a vast collection of ready made Pier plugins: `http://source.lukas-renggli.ch/pieraddons`.

is able to configure the system exactly the way it is required. This is a non exhaustive list of Pier plugins:

ACL Security. A security framework that manages permissions using Access Control lists.

Aggregation. Aggregates the title, the link and a short summary of a freely configurable list of RSS feeds.

Blog. A generic blog extension supporting multiple blogs, RSS feeds, posts with embedded media, public comments, etc.

Bookmark. Manages a personal list of bookmarks.

DiggFeed. Aggregates Digg stories.

Form. Enables end users to define custom forms and reports, and enter structured data into Pier.

LightBox. Display images or specific pages within a LightBox instead of directly embedding or linking them.

Logo. Let you embed and run programs written in the LOGO programming language.

Math. Embeds mathematical expressions and formulas into the page contents that will be formatted nicely using a LaTeX rendering engine.

Poll. Runs polls and visualizes voting results.

Randomizer. Randomizes the display of specific parts in the page.

RelatedLinks. Displays a list of related links to the current page.

Shout Pier. Syntax highlights Smalltalk expressions and methods. The color scheme can be configured using a CSS style-sheet.

Sitemap. Generates an XML site-map that declares public visible page to improve Google indexing.

StellDichEin. Manages vCard compatible address data, iCal events and todo entries.

Unix Security. A security framework that manages permissions using a Unix File System.

View Monticello. Provides a freely configurable listing of recent commits to a Monticello repository. Also displays the commit comments and a link to download the version.

Moreover any existing Seaside application or component can be directly embedded into Pier from its user interface. There is no need to learn Seaside and to change anything in the source code. Therefore Pier is often seen as an abstraction layer over Seaside, allowing end users to simply plug together a REST-ful web application from existing components.

Community

The official web site of Pier can be found at `http://www.lukas-renggli.ch/smalltalk/pier`.

References

[1] Lukas Renggli. SmallWiki: Collaborative content management. Informatikprojekt, University of Bern, 2003. http://smallwiki.unibe.ch/smallwiki.

[2] Stéphane Ducasse, Lukas Renggli, and Roel Wuyts. SmallWiki — a meta-described collaborative content management system. In *Proceedings ACM International Symposium on Wikis (WikiSym'05)*, pages 75–82, New York, NY, USA, 2005. ACM Computer Society.

[3] Lukas Renggli. Magritte – meta-described web application development. Master's thesis, University of Bern, June 2006.

[4] Lukas Renggli, Stéphane Ducasse, and Adrian Kuhn. Magritte — a meta-driven approach to empower developers and end users. In *ACM/IEEE 10th International Conference On Model Driven Engineering Languages And Systems*, LNCS. Springer-Verlag, September 2007. To appear.

[5] Stéphane Ducasse, Adrian Lienhard, and Lukas Renggli. Seaside — a multiple control flow web application framework. In *Proceedings of 12th International Smalltalk Conference (ISC'04)*, pages 231–257, September 2004.

[6] Stéphane Ducasse, Adrian Lienhard, and Lukas Renggli. Seaside — dynamic language power for web development. *IEEE Software*, 2007.

[7] Alexandre Bergel, Stéphane Ducasse, Colin Putney, and Roel Wuyts. Meta-driven browsers. In *Advances in Smalltalk — Proceedings of 14th International Smalltalk Conference (ISC 2006)*, volume 4406 of *LNCS*, pages 134–156. Springer, 2007.