IST FP6-004779

PYPY

Researching a Highly Flexible and Modular Language Platform and Implementing it by Leveraging the Open Source Python Language and Community

STREP

IST Priority 2

Configuration and Maintenance of Development Tools and Website

Due date of deliverable: March 2007

Actual Submission date: 12th March 2007

Start date of Project: 1st December 2004 **Duration: 28 months**

Lead Contractor of this WP: merlinux

Authors: Guido Wesdorp (merlinux), Holger Krekel (merlinux)

Revision: Interim

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)

Dissemination Level: PU (Public)

Revision History

Date	Name	Reason of Change
Dec 21 2006	Carl Friedrich Bolz	partial draft of outline
Jan 24 2007	Holger Krekel	abstract, draft exec summary + outline
Feb 28 2007	Guido Wesdorp	draft most of the content
Mar 9 2007	Holger Krekel	full review / partial refactoring of the report
Mar 12 2007	Carl Friedrich Bolz	Publish Interim on the Web Page

Abstract

This Technical Report describes technical configuration and maintenance of the PyPy development, communication and website environment, hosted on codespeak.net and several other servers. It provides an overview on the services and policies configured and used for PyPy's development and distributed work infrastructure. In addition, it presents plans for continued evolvement of the infrastructure past its EU funding period.

Contents

1 Executive Summary

Developing a complex application within an international open source community requires various infrastructural services. For the PyPy project, we managed most of the neccessary infrastructure within the "codespeak" environment, offering a central versioning system, mailing lists, web sites and several additional services. We made use of a variety of existing open source tools, and we developed - and presented at conferences - advanced techniques that assisted with tasks such as synchronizing our repositories to remote places, and versioning of system and project configuration.

The main reason to choose codespeak rather than for instance the common services provided by http://sourceforge.net or http://tigris.org is that it provided and continues to provide the PyPy members more control and transparency over its operation. For example, we wanted to use Subversion which was not readily available otherwise during the first half of the project duration.

Moreover, we employ an open access policy enabling people to easily work in private areas or other projects and then shift to contribute parts to PyPy. Other environments tend to have certain restrictions by default, irrespective of the fact that hardly any contributor can do permanent damage to a fully versioned filesystem. There were no incidents of abuse of this open policy during the EU project duration. In general, we believe that development infrastructures should be very careful to not impose restrictions which lead to unneccessary overheads.

The central services are used for distributed development and testing, supervising translation and build jobs, general mail & communication and for a number of specialized tasks like generating PDF reports and triggering updates to the website from text documents. They are used by participants of the PyPy project and to a lesser extend by people from other projects who make use of the open codespeak platform.

Maintenance and user support was performed on a daily basis. In addition, larger maintenance works like upgrading or migrating services were performed. Overall this led to a dependable environment with very few outages: usually lasting less than an hour, and never resulting in data damage or loss within the 28 months of the EU project duration.

Besides PyPy, several other projects began to use codespeak's infrastructure. Among these are successful open-source projects, depending on the codespeak services for communication and distributed development needs. This makes codespeak a healthy platform, with well over 200 registered users in March 2007, and adds momentum to keep running it much longer than the EU project period.

We aim at appointing a group of volunteers to help with maintenance and improvement efforts. Several People have expressed interest to help. For some time, merlinux will continue to carry hosting costs, with the aim to establish a more collective funding and maintenance.

In any case, the tools developed during the project duration will be of use after the EU project, since they are made available under open source licenses, and may therefore be used and modified further. There is external interest to have some of these tools get a life of their own.

2 Introduction

2.1 Purpose of this Document

This document provides an overview of the infrastructure services and the maintenance and configuration work performed to support the development and management activities of PyPy.

2.2 Scope of this Document

The document gives a high level overview on PyPy's development infrastructure and refers to web pages for more technical detail. It does not aim to describe how to set up a similar system; it can, however, be used by system administrators to get a global idea of what components our infrastructure consists of and why we have chosen these components.

2.3 Related Documents

- D02.3 Release automated testing framework
- D13.1 Integration and Configuration

3 Hosts and Services

To support distributed development properly, several services were made available that allow multiple people to work on the PyPy code base at the same time, provide easily accessible communication channels - some of which store discussions for later reference - and to test the code. These services were mostly hosted by volunteers from the PyPy community, although some hosts were also made available by other (third) parties as test platforms.

3.1 Codespeak services

Most of the services were made available on the 'codespeak' domain (URL: http://codespeak.net), an environment maintained by members of the PyPy community and driven by merlinux employed personnel. Here is a list of noteworthy technologies used for the PyPy development infrastructure:

- Gentoo Linux: flexible Linux Sourcecode-based Distribution
- XEN kernel-level virtualization: http://www.xensource.com
- VServer advanced jailing: http://linux-vserver.org
- EVMS Volume management: http://evms.sourceforge.net
- DRBD filesystem synchronisation: http://drbd.org
- BitTorrent for large-file distribution: http://en.wikipedia.org/wiki/BitTorrent

- Subversion revision control system: http://subversion.tigris.org/
- Roundup bug/issue tracking: http://roundup.sf.net/
- Nagios system monitoring: http://nagios.org
- BIND/Postfix/Mailman/Apache: standard DNS, mail and web servers

3.2 Other Hosts

In addition to the codespeak services, there are several other machines that PyPy uses. One of those (pypy2) was maintained by merlinux as a large virtual domain, the others (snake, cobra, tuatara and wyvern) are non-virtualized machines kindly provided by the Heinrich-Heine Universität Düsseldorf. In addition some community developers provide private machines to regularly perform testing.

3.3 Operating Systems

All the hosts except for Tuatara run the open-source Linux operating system, Tuatara runs Apple's OS-X. The latter allows us to test the PyPy code base on PPC/OS-X, which forms a good testing ground for our code and is additionally a popular platform to compile for.

3.4 Maintenance

Apart from maintenance tasks of the seperate services, which are described below, certain effort was spent on general installation and maintenance tasks, to get the virtual domains installed and upgraded and to install and maintain the base systems. Linux kernels as well as basic operating system tools have been installed and upgraded, and also several migrations were done to keep the services available and up-to-date.

There were several system crashes, mostly caused by hardware failures, but the services remained thoroughly available and had only very few downtimes, at most 2 hours at a time.

3.5 Automated Test and Translation Runs

The main services running on the hosts donated by the University of Düsseldorf perform automated test runs and translations. Both automated tests and automated translations are run each night to identify integration problems early. The related tools and processes were refined multiple times during the EU project period by PyPy developers and are mostly based on the `py.test` testing tool. The Status section on the main PyPy documentation web page links to all testing sites and information.

3.6 Additional Tools

Other administrative tools developed for PyPy include `py.rest` as part of the py library, a tool used to convert ReStructured Text to PDF, with graphviz and latex support. This is used to build PDF reports and documents, using the same base format as the web pages.

4 Subversion Repository

All code and documentation of the PyPy project is stored in the open-source Subversion revision control system (URL: http://subversion.tigris.org/). Additionally, important EU-project related documentation and contracts are stored in the repository for central and controlled access by all partners.

Several support tools were written and set up for PyPy usage by the project, for sending notification mails, notifying IRC channels, generating website documentation or running automated tests. During the full project period, Subversion has been updated and re-configured multiple times, ranging from the early 0.24 version to today's 1.4.

4.1 Mirror Repository

To make sure the Subversion services are always available, we have set up a 'mirror repository' for which a separately developed "svn-sync-repo" tool (URL: http://codespeak.net/svn/admin/bin/svn-sync-repo.py) pulls new revisions from the remote master repository, usually with a delay of 1 minute, and replays them locally. With this approach, we get some certainty that modifications to the repository will not get lost, and that we can quickly switch to the slave host to continue providing the services when problems occur on the master.

4.2 Daily Backups

As an additional measure, backups are taken on a daily basis. For this purpose, we used a tool called 'vsync' (URL: http://codespeak.net/svn/vsync/dist/) that has some advanced features, such as the ability to keep less backups over time (for instance you can have one backup per day for the last week, but only one per week for the last month), on top of the basic backup functionality. This tool was developed outside of EU funding scope.

4.3 Access to the Repository

Access to Subversion is provided over http, https and ssh. The user can choose which access method to use depending on his preference. Special hooks are implemented that allow synchronizing passwords across services (single login) and to regulate a few access restrictions.

4.4 vadm - Versioning System Files

Normally Subversion is used for source code, but because the versioning and backup features are useful for system administration too, we used and improved, outside of EU funding scope, the 'vadm' tool (http://codespeak.net/svn/vadm) which allows to version files without requiring them to exist in a "working copy" structure. This tool is available as an open-source package, and was also presented at the EuroPython conference in Geneva, 2006.

5 Communication Channels

In an 'agile' environment, communication is a key issue. As in common these days in open source projects, we made use of asynchronous and synchronous digital channels, i.e. Mailing Lists and IRC channels. Voice over IP Communication was only used by the management team and very rarely by the developers.

5.1 Mailing Lists

Codespeak hosts mailing lists for developers, but also for the consortium and community. This way communication is clear and accessible by anyone who it concerns, and available at any given time.

The mailing lists provided are:

- pypy-dev - developer mailing list (also suitable for users)
- pypy-svn - coding and documentation commit notifications
- pypy-eu-svn - documents and resource tracking commits
- pypy-sprint - sprint preparation communications
- pypy-funding - EU project related communication

Also the 'py-lib', a library developed for and used extensively by PyPy, has two mailing lists available:

- py-dev - developer mailing list (also suitable for users)
- py-svn - coding and documentation commit notifications

5.2 IRC

For real-time discussions the PyPy community mostly relies on the IRC protocol, we have a channel called '#pypy' on the free 'freenode.net' service (URL: http://freenode.net). There are several related administrative tools, such as a number of 'IRC bots' that announce commits and log the conversations (see http://tismerysoft.de/pypy/irc-logs/pypy/).

5.3 Maintenance

The mailing list software used by PyPy is 'Mailman' (URL: http://www.gnu.org/software/mailman/), an open-source product written in Python. This software was relatively care-free and easy to maintain, apart from performing several updates during the EU project period and dealing with anti-spam measures. Adding users to restricted lists also required manual effort, as did dealing with particular problems of users interacting with the lists (attachments, blocked email addresses).

Since we used a third-party service (http://irc.freenode.org) for IRC, there was relatively little maintenance required there. The main development and maintenance tasks were related to the 'bots' and done in the spare time of their authors.

6 Web Site

For promotion purposes and to provide information to the 'outside world', PyPy has two websites hosted on codespeak, the first is accessible as http://codespeak.net/pypy and targets the PyPy developer and user communities, the second is available as http://pypy.org and provides EU-project related information. The websites provide news sections with release and sprint information, documentation about PyPy and links to other resources such as the mailing lists and Subversion repository. Also accessible via the web server are the PyPy issue trackers that keep track of bugs and feature requests, and Bittorrent trackers that provide access to the PyPy video documentation.

6.1 Maintenance

The websites run on the open-source Apache web server (URL: http://httpd.apache.org), which is relatively easy to maintain, although setting it up was not trivial. To allow easy updating and improve creation of new parts of the web site, several tools and practices were developed, such as a tool to generate HTML, the presented format of the pages, from specifically formatted plain text ('ReStructured Text'), and tools to automate releases and uploading of different versions of PyPy.

6.2 Issue Tracker

To keep track of bugs and issues, both PyPy and the py lib have an issue tracker running on codespeak. For this, the open-source Roundup product is used, which turned out to be relatively hard to set up but also relatively easy to maintain.

6.3 BitTorrent

Another service that codespeak runs for PyPy is a Bittorrent 'tracker' that provides access to the PyPy video documentation material. Bittorrent provides a way to, in a peer-to-peer manner, allow distributed downloads: clients share the parts that they have downloaded with others immediately so that stress on the main server is reduced. This means that if there are more clients connected, the stress on the server is not increased, which makes it an ideal solution for sharing large files such as the videos.

To ensure availability of the downloads, the PyPy system administrators have set up some backup clients, running on remote machines, besides the main tracker. The clients were started as seeds, meaning they had the full data readily available and served to serve rather than download data. The software used for both the tracker and the client is Bittornado (URL: http://www.bittornado.com/), a popular open-source application written in Python. Custom scripts aided to upload videos to the trackers.

7 Statistics

During the whole project, all services where monitored and usage statistics produced. For website statistics, we used a third-party open-source product called 'awstats' (URL: http://awstats.sourceforge
along with some scripts we wrote ourselves.

To monitor the Bittorrent service, we wrote a Nagios (URL: http://nagios.org) plugin which shows the total amount of downloads - counting well over 7500 before March 2007.

To analyze the development of the codebase, we derived several statistics from the subversion repository. To analyze community size and involvement we analyzed various mailing lists.

The resulting statistics are:

- Lines of code and lines of test code
- Development list subscribers
- Development list activity
- IRC channel activity
- Mentioning of PyPy on comp.lang.python
- Web access statistics

8 Glossary of Abbreviations

The following abbreviations may be used within this document:

8.1 Technical Abbreviations:

AST	Abstract Syntax Tree
CPython	The standard Python interpreter written in C. Generally known as "Python". Available from www.python.org.
codespeak	The name of the machine where the PyPy project is hosted.
CCLP	Concurrent Constraint Logic Programming.
CPS	Continuation-Passing Style.
CSP	Constraint Satisfaction Problem.
CLI	Common Language Infrastructure.
CLR	Common Language Runtime.
docutils	The Python documentation utilities.
F/OSS	Free and Open Source Software
GC	Garbage collector.
GenC backend	The backend for the PyPy translation toolsuite that generates C code.
GenLLVM backend	The backend for the PyPy translation toolsuite that generates LLVM code.
GenCLI backend	The backend for the PyPy translation toolsuite that generates CLI code.
Graphviz	Graph visualisation software from AT&T.
IL	Intermediate Language: the native assembler-level language of the CLI virtual machine.
Jython	A version of Python written in Java.

LLVM	Low Level Virtual Machine - a compiler infrastructure available from University of Illinois at Urbana-Champaign
LOC	Lines of code.
Object Space	A library providing objects and operations between them, available to the bytecode interpreter via a well-defined API.
Pygame	A Python extension library that wraps the Simple Direct-Media Layer - a cross-platform multimedia library designed to provide fast access to the graphics framebuffer and audio device.
pypy-c	The PyPy Standard Interpreter, translated to C and then compiled to a binary executable program
ReST	reStructuredText, the plaintext markup system used by docutils.
RPython	Restricted Python; a less dynamic subset of Python in which PyPy is written.
Standard Interpreter	The subsystem of PyPy which implements the Python language. It is divided in two components: the bytecode interpreter, and the standard object space.
Standard Object Space	An object space which implements creation, access and modification of regular Python application level objects.
VM	Virtual Machine.

8.2 Partner Acronyms:

DFKI	Deutsches Forschungszentrum für künstliche Intelligenz
HHU	Heinrich Heine Universität Düsseldorf
Strakt	AB Strakt
Logilab	Logilab
CM	Change Maker
mer	merlinux GmbH
tis	Tismerysoft GmbH
Impara	Impara GmbH