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

D01.1: Create QA Plan for the Project

Due date of deliverable: January 2005

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

Lead Contractor of this WP: Change Maker

Authors: Beatrice Düring (Change Maker), Holger Krekel (merlinux)

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

Dissemination Level: PU (Public)

Abstract

The quality assurance plan describes a framework of quality structures being used for implementing the PyPy project. It serves to support the project process by identifying the necessary, minimal procedures needed on both technical and consortium levels of the project.

The agile practices being implemented within the project collide with the traditional project management approach of detailed segmented plans for communication, information flow, quality assurance, scope management, cost management. An important aspect of agile project is to document procedures and features only when deemed necessary - therefore the minimalistic approach in this QA plan.

This also has the effect that this document needs to be under constant revision, quite different from traditional project management strategies regarding quality assurance. The very nature of iterative and agile development in combination with structures for evaluation and "learning loops" will have the result of ever evolving procedures for doing quality assurance better, more automated, more creative and still systematic.

The framework described within this plan covers project standards for:

- communication
- documentation
- tracking actions
- conflict resolution
- evaluation
- corrective actions and exceptions
- roles and responsibilities

This framework represents results of decisions on consortium level of the project but the main creative discussion regarding these aspects are done within the developer community.

Contents

1 Executive Summary

Quality assurance in the PyPy project is mainly done via:

- automated procedures (version control, test suites, technical infrastructure for communication, documenting, reviewing and tracking)
- open procedures (peer review within the project with no/or few limitations regarding access to information and decision-making between the consortium partners and the different individuals working within those partner organizations)
- role procedures (roles and responsibilities, allowing the consortium work to be ongoing in parallel with the technical level of work, integrated when necessary and driven by interest)
- agile procedures (sprint driven development while working distributedly, evaluation routines on different levels of the project is the primary methods for both coordinating, integrating the levels as well as the actual execution of the project)

The cumulative effect of these different procedures is the basis for what we believe to be the successful implementation of PyPy - an agile open source project with partial funding through the European Union's 6th Framework Programme.

2 Introduction

2.1 Purpose of this Document

This document informs project participants and interested parties about definitions of standards within the project process. It serves as a guideline for consortium level decisions on the issues covered within the plan.

2.2 Scope of this Document

This document describes the consortium and main technical procedures for performing, documenting, communicating and evaluating work within the PyPy project. It also describes roles and responsibilities being in effect within the project.

2.3 Related Documents

Before reading this document, a look at the following document is recommended:

- Contract
- Annex 1/Description of Work
- Consortium agreement

3 Document Procedures, Standards and Control; Issue Control for Documents

The primary documents in the PyPy project are:

- Contract (including Annex I and Annex II)
- Consortium agreement
- Amendment 1,2,3
- Proposal
- Project GANTT chart (phase 1-3, external deliverables EU)
- Budget (resource tracking tool)
- Form C statements
- Timesheets
- Description of Work
- Partner information (contact and availability, CPF:s)
- Quality Management Plan
- Dissemination plan (conference and sprint planning)
- Meeting minutes (kick off, sprints, internal reviews, management board, technical board, pypy-sync, Consortium meetings)
- Templates (timesheets, deliverables, sprint report, sprint evaluations)
- Reports (sprint reports, deliverable reports, periodic reports (activity and management), final reports, internal review report)
- Papers (deliverables, conference articles and talks)
- Newsletter (ongoing stakeholder information regarding PyPy project, sprint reports, "this week in PyPy")
- User documentation (development and design descriptions, getting started, how-tos, tool descriptions, coding guide etc.)

All documents in the PyPy-project are under the same standard and control routines. These are:

- All documents are accessible on the project development website http://codespeak.net/pypy
- All documents are saved in a open format (txt, pdf or Open Office format)
- All documents are named in the following way: type_date.format (ex: qmp_041222.txt)
- All documents are put under revision control (Subversion)
- backups of the development and document repository must be kept at a remote site.

- Each author is responsible for adhering to these guidelines (including ones only doing changes)
- The assistant project manager is overall responsible for proper fulfillment of the guidelines and is responsible for possible changes to these guidelines including their communication to the team
- All documents are accessible either in the Subversion directory `extradoc` - dissemination material -- open to the public, or in the directory `eu-tracking` - cost and time tracking, minutes, for consortium members only on. Both of them are part of the central subversion repository.

4 Reporting Procedures (Frequency and Format)

Following reports will be produced during the project:

Sprint reports (sprint newsletter):

- Intense one week work meetings ("Sprints") are the primary means for co-ordination of work and tackling new challenges.
- It is a requirement to have a report about each sprint, naming location and participants and detailing achievements as well as general open problems.
- Distribution happens via via pypy-sprint@codespeak.net, pypy-dev@codespeak.net and as a news topic under http://codespeak.net/pypy.
- The assistant project manager makes sure that sprint reports get written and distributed - usually done by the participants.

Internal review reports (every 6 months):

- Report based on the work done in the period of 6 months in the project, summarizing consortium level results as well as development level results, and dissemination done.
- Includes a written evaluation of project, team and development process with suggestions of improvement. This evaluation is a summary of information gathered during the internal review workshop (written by the assistant project manager)
- Distributed on pypy-funding@codespeak.net and archived in the Subversion repository (eu-tracking) on the project website

Periodic report 1, 2 and final report:

- Periodic activity report 1 (15 January 2006)
- Periodic management report 1 (15 January 2006)
- Periodic activity report 1 (15 January 2007)
- Periodic management report 1 (15 January 2007)
- Final report (after project delivery)

- Reports mentioned here (periodic and final) are created by the project manager, with assistance from the project coordinator and the assistant project manager
- Following the template structure created for deliverable reports (ReST auto-generation into pdf)
- Distributed to the Project Officer, to pypy-funding@codespeak.net and archived in the Subversion repository (eu-tracking) on the project website.

Project evaluation report:

- Written evaluation of the total project and product process and all deliverables
- Created by the assistant project manager, with the assistance of the project coordinator, project manager and technical board
- Formally approved by the consortium before distribution
- Distributed on pypy-funding@codespeak.net and archived in the Subversion repository (eu-tracking) on the project website

5 Communication Procedures

The main channels of communication for the project are:

5.1 Websites

Project Development site - http://codespeak.net/pypy: Main place for all documentation and communication on development level as well as on consortium level. IRC and mailing list archives allows people to fully access information from when the project started (early 2003). Codespeak is also the home of the PyPy codebase. News about releases, deliverables, conferences and sprints are announced here.

EU project Summary webpage - http://pypy.org/:

Main summary page for non-developers seeking information about the funded aspects of the PyPy project. Summarizes the main objectives and presents the different partners working in the consortium, with names and roles as well as contact information. News about releases, sprints, deliverables and other consortium achievements are announced here.

5.2 Mailing Lists

- pypy-funding@codespeak.net (closed mailing list focused on consortium co-ordination)
- pypy-dev@codespeak.net (mailing list for the developer community)
- pypy-sprint@codespeak.net (mailing list for organizing sprints)
- pypy-eu-svn@codespeak.net (mailing list for information about changes within the eu-funding repository)
- pypy-dev-svn@codespeak.net (mailing list for information about changes within the PyPy codebase)

5.3 Internet-Relay-Chat

The following IRC channels on the open irc.freenode.net platform are used within the PyPy project:

- #pypy (open channel for communication among developers and interested parties)
- #pypy-funding (open IRC channel for communication inside the consortium structure, the forum for consortium meetings)
- #pypy-tb (IRC channel for meetings of the Technical Board)
- #pypy-sync (IRC channel for weekly meeting between PyPy developers, done via IRC)

5.4 Meetings

5.4.1 Consortium Meetings

Meetings of the consortium and its different representatives (managers as well as developers). Meetings are done via IRC (#pypy-funding on irc.freenode.net) on a monthly basis. The strategy is to hold the meetings the first Friday every month, when timesheets have been updated to the central repository (codespeak/eu-funding).

Consortium meeting is the forum in which the project tracks work on the consortium level, via action lists (which contain the person appointed responsible and the deadline). The assistant project manager prepares the meeting and the agenda is being sent out five days before the meeting (via pypy-funding). The assistant project manager also manages the meeting and writes the minutes (distributed on pypy-funding and archived in `eu-tracking/minute/`) on codespeak.

All partners are obliged to participate, or otherwise communicate the failure to do so via pypy-funding. If that is the case the partner in question need to ratify the minutes after the meeting.

5.4.2 Management Team Meetings

Meetings for coordinating work between project coordinator, project manager and assistant project manager. The meetings are done on a "need to" basis and are prepared, managed and documented by the assistant project manager. Minutes are distributed on pypy-funding and archived in eu-funding/minute/... on codespeak.

5.4.3 PyPy-sync Meetings

Weekly short coordination meetings between developers, open to all developers active in the PyPy community, usually but not necessarily involving aspects of EU-funded work on deliverables. These 30 minute IRC-meetings serve a weekly synchronisation for regular discussions and integration of ongoing work. Meetings are prepared with an agenda sent out to pypy-dev and minutes are distributed on pypy-dev and archived in the repository with publicly accessible links. Preparing, managing and documenting the meetings is rotated between active developers and is supervised by the Technical Director.

5.4.4 Technical Board Meetings

Meetings of the technical board are the formal forum of EU-project related development planning as well as decisions about acceding new partners to the project. The meetings are done on a "need to" basis and are prepared, managed and documented by the Technical Director. Minutes are distributed on pypy-funding and archived in the repository - for access by all consortium partners. However, most of the direct co-ordination between active developers and researchers is done during pypy-sync meetings (and only discussed within the Technical Board if necessary or if crucial problems arise).

6 Corrective Actions and Exception Control

Corrective actions and exception control is being implemented on different levels in the project:

6.1 Consortium Level

The consortium agreement specifies the project coordination committee as the organizational body entitled to make decisions on a consortium level. The monthly consortium meetings acts as the primary forum for these decisions. The project management board prepares and executes these decisions. This work is delegated to the management team.

- Resource consumption in man-months are tracked on a monthly basis, all partners submit their timesheet to eu-funding/timesheet/... on codespeak. This is then summarized in the resource tracking tool (archived in the same repository) on a WP level and on a partner level, planned versus consumed hours. Deviations are raised by the management team to consortium level for discussions and justifications. Corrective actions are explored by the partner in question and for larger deviations by the technical board who suggests strategies. Deviations larger than 10% need to be explained on consortium level.
- Resource consumption in costs are tracked every 6th month by internal form c claims. These are then connected to installments of prefunding being distributed. This is then summarized in the resource tracking tool (archived in the same repository) on a partner level, planned versus consumed cost. Deviations are raised by the management team to consortium level for discussions and justifications. Corrective actions are explored by the partner in question and for larger deviations by the technical board who suggests strategies. Deviations larger than 10% need to be explained on consortium level.
- Resource consumption of results (work achieved) is coordinated by the partners involved in a specific workpackage and communicated by the lead partner during status of work in the consortium meetings.

6.2 Technical Level

- Peer Reviewing and Pair-Programming are primary means at our regular one-week Sprint workshops. Developers usually sit together and discuss and program together on all parts of the system. Newcomers are often paired with more experienced people, providing a helpful and motivating learning experience. Apart from working together at sprints, Peer Reviewing also takes place through reading changes that other developers did - a common and proven technique in alive open-source environments.

- The automated test suite being used in PyPy development and the different levels of the technical work being tested is the main tool for handling exceptions and for very quickly identifying and implementing corrective actions. The various test suites are: regression tests of the PyPy codebase, compliance tests (originally written for CPython).

  Developers should - before committing changes to the main repository - run tests and make sure that no regression occurred. Failures can always be tracked to a particular change to source code which broke expected (tested) behaviour. This provides direct feedback with detailed information helping the developer to identify failures. This test driven development environment is the primary quality assurance tool and strategy for making sure that PyPy maintains a high-quality code base. It is a requirement to add new tests for each newly added features and for each fixed bug.
- Version control is the primary means of providing a secure platform for incremental development. In PyPy, Subversion is being used, covering both program files and documentation (also consortium level documentation). Several mailing lists on development and consortium level makes sure that every person involved receives instant notification on changes - support Peer Review on all levels.
- Metrics regarding different trends on codespeak are being monitored and published on codespeak. Examples of metrics being covered are:
  - lines of code
  - volume of tests in the codebase
  - number of atomic changes (covering both program files and documentation files)
  - amount of subscribers on the main development list (pypy-dev)
  - #IRC pypy
  - web-access on codespeak

  The assistant project managers uses these metrics to evaluate trends in the project and raises interesting observations for discussion among the core developers. These metrics are also used in periodic reports and for wider dissemination purposes.
- An extensive coding guide, published on codespeak.net (development website) serves as an introduction to the set of specifications being used while coding PyPy. The coding guide covers not only naming conventions but also design, test and documentation guidelines.

7 Conflict Resolution

Conflict resolution can take place on different levels in the project:

7.1 Consortium Level Conflict Resolution

In the case of conflicts within the consortium, that can not be solved by consensus and open discussions, are solved by voting in the management board. Each representative has 1 vote, in case of a tie the project manager has a casting vote. The consortium can also invoke the decision structure described in the consortium agreement (In case of conflicts that cannot be resolved in the usual decision procedure and amicably, the rules of arbitration will apply - see section 16.1 in consortium agreement and 5.2.6 - rules of voting).

7.2 Technical Level Conflict Resolution

In the case of conflicts among developers which can not be solved by consensus and open discussions at pypy-sync meetings, they are solved by voting in the Technical Board. Each representative has 1 vote, in case of a tie the technical director has a casting vote. The consortium can also invoke the decision structure described in the consortium agreement (In case of conflicts that cannot be resolved in the usual decision procedure and amicably, the rules of arbitration will apply - see section 16.1 in consortium agreement and 5.2.6 - rules of voting).

8 Meeting Draft Agenda

Consistent agendas are being implemented on different levels in the project:

- Consortium level; consortium meeting agenda:

```
1. Agreement on agenda
2. Status of work (tracking action points form previous
   meetings Done, To do, person responsible and time to
   completion)
3. Amendments
4. Deliverables and reports
5. Decision point (with prepared decision sent out in the agenda)
6. Other topics
7. Next meeting (date, location, content) and deadline for agenda
8. Meeting ends

Management board meeting draft agendas do not have a specific structure.
```

- Technical level: PyPy-sync meeting agenda:

```
1. Regular Topics:

  - Roll call (meeting opens and the agenda is being agreed upon)

  - Activity reports (3 prepared lines of info)
    Everybody submitted activity reports

  - Re-assigning / adjusting planning / resolve conflicts/blockers

2. Topics of the week
```

```
    - Specific development work

    - Sprint planning (goals and content for upcoming sprints)

  3. Closing the meeting
```

- Technical board meeting draft agendas do not have a specific structure.

9 Format of Meeting Minutes

Consistent minutes are written on different levels in the project. All minutes are saved in txt format and being subjected to peer review and version control:

- Consortium level; consortium meeting minutes:

```
Minutes of consortium meeting: date
Form: IRC
Attendees:
Protocol:
Manager:

Agenda:

1. Agreement on agenda
2. Status of work (tracking action points from previous meetings)
3. Amendments
4. Deliverables and reports
5. Decision point (with prepared decision sent out in the agenda)
6. Other topics
7. Next meeting (date, location, content) and deadline for agenda
8. Meeting ends
9. Summary of status of action points
```

- Management board meeting minutes do not have a pre-decided structure but covers the following information: date, place, attendants, moderator, minutes, agenda, topics, action list).
- Technical level: PyPy-sync meeting minutes:

```
pypy-sync meeting: date, place, moderator

Attendants:
1. Regular Topics:

  - Roll call (meeting opens and the agenda is being agreed upon)

  - Activity reports (3 prepared lines of info)
    Everybody submitted activity reports

  - Re-assigning / adjusting planning / resolve conflicts/blockers
```

```
2. Topics of the week

  - Specific development work

  - Sprint planning (goals and content for upcoming sprints)

3. Closing the meeting

4. Full IRC log from meeting
```

- Technical board meeting minutes do not have a pre-decided structure but covers the following information: date, place, attendants, moderator, minutes, agenda, topics, action list.

10 Tracking Systems for Actions

Tracking systems for actions are being implemented on different levels in the project. Both methods being described here are being subjected to peer review and version control:

10.1 Consortium Level

On consortium level, action points in consortium meeting minutes are used. A regular topic on the agenda and in the meeting is the status of work. During that topic every action point is checked for status (done, to-do, delayed), person responsible and time to completion. Status of these action points are communicated via email before the meeting. The consortium then focuses on the action points which have identified blockers and try to resolve these. The minutes act as the documentation of the changing status of the different action points. The assistant project manager is responsible for tracking action points.

10.2 Technical Level

On the technical level, development and bug requests are monitored through a public "Issue Tracker". It helps to create, change status and nosy-lists (people being interested in a particular issue) as well as to assign features to particular releases. This is a primary tool for tracking work among developers. The WP 2 lead partner is responsible for maintaining and improving the issue-tracker.

11 Risk Assessment

The primary strategy for risk assessment and risk mitigation is the process of cyclic, iterative approaches on:

- project level: sprints (planning, implementing, closure)
- consortium level: monthly consortium meetings
- technical level: weekly pypy-sync meeting, Technical Board meetings

These loops of iterative cycles allows for early risk identification and also increased levels of learning and awareness on all levels in the project. Risks are documented and turned into action points in minutes (research information, communicate issues etc.) as well as created as issues in the PyPy-dev issue tracker (research, prototype, bug fixing, refactoring etc.).

12 Evaluation Routines

Evaluation routines are being implemented on different levels in the project:

12.1 Project Level: Sprint Closure Meetings and Sprint Evaluations

During sprint closure meetings the team evaluates the work done and plan the work for the work to be done in between sprints, Improvements to sprint organization and execution is suggested and discussed. If a consensus is being reached the improvements are implemented. The external participants receive an individual evaluation via email after the sprint, covering the different activities during the sprint (information, tutorials, methodology, management, documentation etc). These evaluations are the responsibility of the assistant project manager to communicate to the consortium if necessary, in most cases to the core developers.

12.2 Consortium Level: Internal Review Workshops ("Learning Loops")

Every 6th month of the project (roughly, also depending on current work plans etc.) an internal review workshop is performed on the consortium level, all partners required to attend. The agenda is prepared beforehand with a minor written evaluation - choosing topics relevant to the consortium at that specific time of the project. The main purpose of the internal review workshop is to identify and discuss potential learnings and improvements to the project process. This is a technique being part of the agile portfolio. The assistant project manager is responsible for preparing the agenda, executing the workshop and documenting the results. Decisions of specific improvements are turned into topics for upcoming consortium meetings. Minutes from the internal review workshop (with a summary of work achieved in a consortium and technical level in the covered period) are distributed on pypy-funding and archived in `eu-funding/minute/` on codespeak (subjected to peer review and version control).

13 Specific Responsibilities within the Project

The formal roles and responsibilities in the project are:

- Project coordinator:
  - The coordinator role is shared by Stephan Busemann and Alastair Burt, DFKI (Stephan Busemann as primary resource)
  - Manage dialogue with the Commission and the Project Officer
  - Inform the project about and attend Commission tracking and concertation meetings.
  - Manage resource tracking (collect, summarize and communicate data on a monthly basis)

  - Report deliverables to EU and participate in preparing deliverables (templates, content and research)
  - Contract administration (prepare amendments and coordinate with all parties)
  - Participate in consortium meetings and management meetings

- Project manager:
  - Jacob Hallén, AB Strakt
  - Support resource tracking and analyzing
  - Manage (prepare and track) periodic and deliverable reports to the EU
  - Market project
  - Manage physical person process
  - Manage sprint coordination (planning venues and support sprint organization)
  - Participate in consortium meetings and management meetings

- Assistant project manager:
  - Beatrice Düring, Change Maker
  - Tracking actions and deliverables within the consortium
  - Manage consortium meetings and management meetings
  - Track sprints and support sprint organization through communication with the developer group
  - Market and document project through documents and web
    http://codespeak.net/pypy, http://pypy.org
  - Project administration (minutes, documentation, templates, reports)
  - Manage quality assurance of the project in cooperation with Technical Board (the technical director)
  - Participate in consortium meetings and management meetings

- Management team:
  - Stephan Busemann, DFKI, Jacob Hallén AB Strakt, Beatrice Düring, Change Maker
  - execute the work of the management board (mentioned in Annex 1)

- Technical director:
  - Holger Krekel, merlinux GmBH
  - Lead the work of the Technical Board

- Technical board:
  - Holger Krekel, merlinux GmBH, Armin Rigo HHU, Samuele Pedroni, AB Strakt, Christian Tismer, Tismerysoft GmBH
  - Recruiting and justifications for accessions of physical persons.
  - Review of the technical content in the reporting of deliverables
  - Development planning

- Lead partner:
  - Coordinate the deliverables within the workpackage with the other partners (this should as much as possible happen on a developer level)

  - Fill out the EU-report for the deliverables and inform project coordinator
  - Estimate achieved result in percentage at the end of the month and report

Changes to these roles and responsibilities are subject to consortium level decisions.

14 Glossary of Abbreviations

The following abbreviations may be used within this document:

14.1 Technical Abbreviations:

AOP	Aspect Oriented Programming
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 DirectMedia 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.

14.2 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