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

Report About Milestone/Phase 2

Due date of deliverable: June 2006

Actual Submission date: August 3rd, 2006

Start date of Project: 1st December 2005 **Duration: 2 years**

Lead Contractor of this WP: Change Maker

Authors: Holger Krekel (merlinux GmbH), Beatrice During (Change Maker)

Revision: final, 03-08-2006

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
2006-06-15	Anders Chrigstroem	created document, listed technical topics
2006-07-15	Holger Krekel	draft summary, added and refined technical topics
2006-07-16	Carl Friedrich Bolz	added statistics graphs
2006-07-25	Beatrice Düring	dissemination and consortium section
2006-07-27	Holger Krekel	refinements, outlook + community section
2006-08-02	Hudson/Bolz	clarifications, proofreading
2006-08-03	Krekel/During	finalizing report

Abstract

This document describes the results of work in phase 2 of the PyPy project. The Executive Summary gives an overall view on the results of this second phase of the project. Subsequent chapters provide focused summarizes on technical, research, dissemination and community levels.

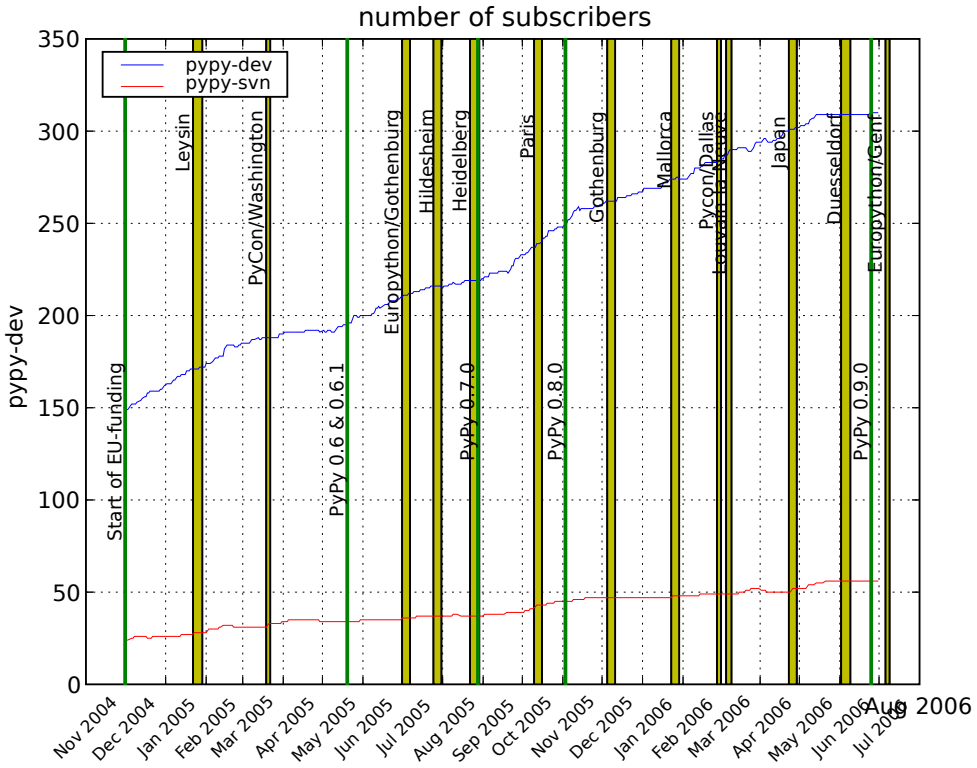
Contents

1 Executive Summary

The major technical milestone of the second phase was reached with the public 0.9 PyPy release in June 2006 with a 20-fold performance improvement by a series of static and core translation optimizations. The release also provides mechanisms to enable massive parallelism and integrates our garbage collection framework allowing the simulation of custom garbage collectors and their incorporation in the translated PyPy executable. Logic Variables and Logic Programming idioms are newly introduced and lastly, the separate extension compiler makes use of PyPy's tool chain and provides a new way for the Python community to produce high-performance extension modules that can be used both for PyPy and the mainstream CPython implementation. Major technical achievements are also documented in the release documentation and also in the interim 7.1 and 9.1 contractual deliverable reports that have been published together with this document.

The project also increased dissemination efforts on all levels: it organised nine international sprints, attended eleven conferences and held three workshops during the second phase of the project. In particular, it continued to involve the Python community itself with several presentations at each of the two major conferences, PyCon 2006 (Dallas) and EuroPython 2006 (CERN/Geneva) in addition to non-domain specific conferences and events such as the largest European "hacker" conference, 22C3, where more than 400 people attended an extensive technical talk.

With the Summer of PyPy call for proposals from students, the project further embarks on its strategies to increase contributions from the community and validate the technical architecture and methodological approaches. With its eight new Video Documentation features the project provides lively insights into its technical and methodological nature and also highlights the appreciation of central figures within the Python community.



The project published the following papers:

- a research paper for the Dynamic Language Symposium at OOPSLA 2006 (VMC), summarizing PyPy's new approaches for the language implementation research community.
- a methodology paper for XP 2006 focusing on PyPy's sprint driven development method (SPRINT)
- a methodology paper for Agile 2006 focusing on PyPy's hybrid development and management approach (AGILE)
- a paper for the Chaos Communication Congress describing PyPy as a new innovative Python implementation (BLOCK)
- a paper, also for the Chaos Communication Congress, highlighting key points of PyPy development (OSEU)

One major example of community contribution during phase 2 worth mentioning was the effort to make use of PyPy's translation tool chain for Microsoft's .NET environment (PYPYNET). This effort was welcomed and supported by the PyPy core group, because it served both as early validation of PyPy's architecture and broadening its overall applicability.

Several partners began pursuing relations with parties commercially interested in the results from the PyPy project. The commercial interest focuses on all areas, namely the core PyPy implementation with its speed and flexibility gains, its supporting tool chain and its methods for distributed development. The project held workshops and on-site sessions with commercial companies such as IONA and Hewlett Packard and is preparing workshops with IBM and the gaming industry.

The consortium finalized the fourth amendment to its contract with the Commission - implementing a number of EU Review recommendations, particularly a re-packaging of deliverables and the focus on the new extension compiler which responds to commercial and developer community interests.

Some problems arose from delays and resource deployment difficulties of some partners, foremost because Tismerysoft was heavily affected by serious health issues of its key person. This was immediately communicated to the Commission. The main problems arose in Work Package 7, where this partner was not able to continue his work due to the mentioned health incident at the beginning of June. However, other partners collaboratively helped to complete most crucial results in this work package (included in the public 0.9 release). Some works remain to be done and the project is preparing for the situation where this partner cannot return to the project at all.

Finally, the project drew conclusions from the achieved results and resource deployment limitations as well as from feedback from researchers and developer communities and has updated its internal plannings in close co-ordination with all partners. It thus intends to ask for an extension of the project duration in order to deliver all results and also to follow up on the Review recommendations and discussions. The according revised deliverable deadlines will be presented within a separate request for Amendment 5 of the contract. Particularly, regarding Work Package 7 there is only an interim report available, which is now planned to be finalized 2-3 months later than originally scheduled.

The project looks forward to the last phase of the project, being very confident that it will achieve all major technical, methodological and research goals within its contract. Most importantly, there is a strong focus and confidence that the project is going to continue after the EU contract ends - whose funding grant continues to facilitate and enable the research and implementation ground work of a project that would otherwise not have been possible.

2 Introduction

2.1 Purpose of this Document

This document provides a summary for both a domain specific and a non-domain specific readership. It presents a summary of work done during phase 2 of the EU/PyPy project.

2.2 Scope of this Document

This document describes the achievements of phase 2 of the PyPy project, the main technical result and the also related dissemination efforts and their effects. It does not contain financial or related contractual information.

2.3 Related Documents

This document has been prepared in conjunction with the following reports:

- D14.1 report on phase 1.
- D07.1 Translator Optimisations (Interim)
- D09.1 Extend Language with Search and Logic (Interim)

3 PyPy Project Phases 2 and 3

The project started on the 1st of December 2004 with a planned time frame of two years, technically structured into three phases (9 + 9 + 6 months). The first phase - building the core interpreter and translation tool chain - was completed August 2005. The second phase focused on increasing performance and met its major technical milestones with the 0.9 public PyPy release in June 2006. During the second phase the project has already validated a number of architectural aspects and also increased flexibility of the whole system, originally a topic for phase 3.

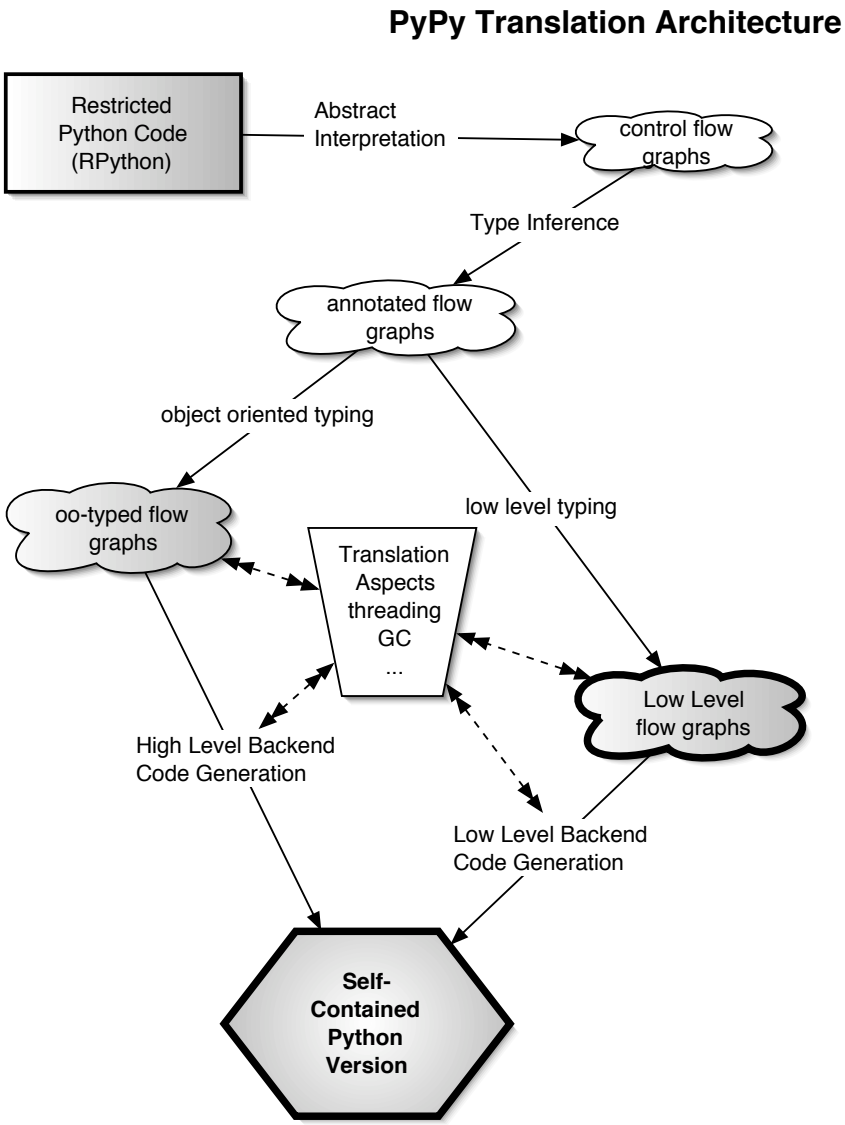
Validation, research and the development of new feature - as well as performance improvements - are interconnected, and it is thus natural that phase 2 and phase 3 are not as strictly separated from each other as phase 1 was from the subsequent periods. This also relates to the project's priority on sustaining the project after the EU contract period by integrating interested parties and by answering to community and commercial interests. For example, the release of the Extension Compiler (Work Package 3) is rather early in terms of the original project plan, but answers to community and commercial interests. Moreover, work on integration and configuration aspects (Work Package 13) and on major validation aspects (Work Package 12) has already started although it was only scheduled for phase 3.

4 Summary of Technical Results

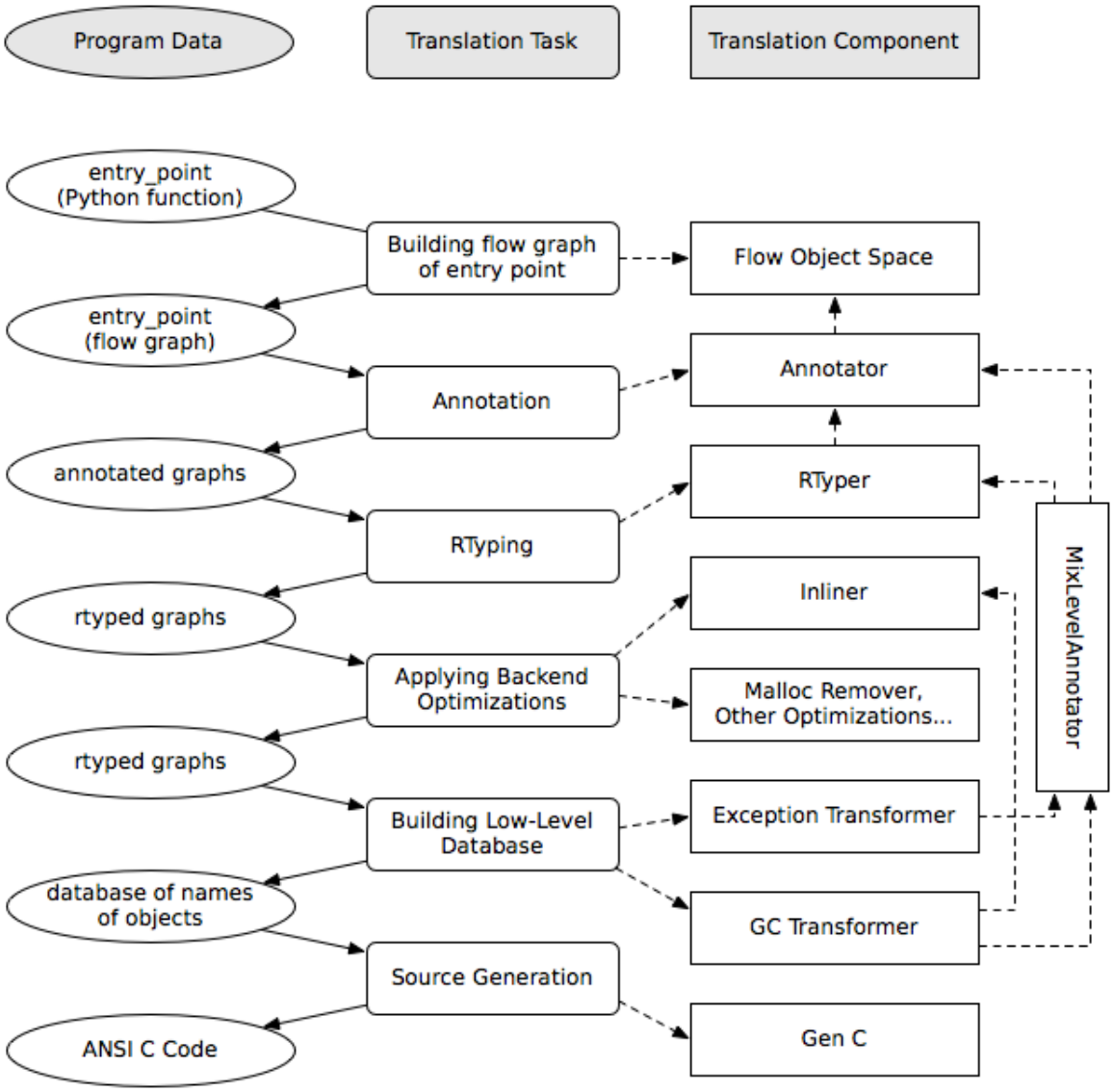
The major achievements during phase 2 were:

- With the 0.9 PyPy release the speed of the translated PyPy interpreter has improved by a factor of 20 since the 0.7 release (the major phase 1 release). It is still 2.5-10 times slower than CPython on popular benchmarks. However, early experiments with type optimizations (WP06 core optimizations) show that PyPy can perform faster on specific operations. More detailed performance comparisons can be found in the Interim D07 report.
- An early release of an extension compiler (WP03) which provides a flexible way to write extension modules that work unmodified both with PyPy and as a standalone CPython module. Moreover, the translation toolchain is capable of transforming dynamic calls into c libraries into static ones, emulating the popular ctypes API which is becoming the mainstream (with CPython 2.5) approach for performing foreign function calls dynamically.
- The re-implementataion of translation aspects (Exceptions and Garbage collection most noticeably) as generic flow graph transformations as opposed to being implemented for specific backends (WP07). This makes it possible to share code between various backends and it enhances flexibility and maintainability.
- Application level access to massive parallelization features such as tasklets and coroutines, including tasklet cloning and pickling (WP07). In addition the interface of 'greenlets' (a coroutine implementation for CPython) was implemented, so that existing code can be run on top of CPython and PyPy unmodified.
- Integration of several alternative implementations of memory management strategies to choose from. Exploration on implementing tagged pointers which have been used to implement int objects for small valued integers. (WP07).
- PyPy continues to get more complete in terms of core Python functionality (WP03), most noticeably there is now a generic implementation of "weakrefs", providing weak references in context of custom garbage collectors (WP03 and 07). The core language tests continue to pass at a rate of 95% or better despite many changes to the code base.
- New Logic and constraint programming features were implemented by means of a "logic" Object Space that provides support for python logic variables and dataflow synchronization of coroutines. More detailed information is found in in the Interim D09 report.
- First results of the work on "Dynamic Optimisations" (WP08) showcase the automatic generation of a (static) compiler for a Turing-complete "toy language" interpreter, reusing the PyPy translation architecture.
- Preliminary results on the validation Work Package 12, providing easy paths to higher level backends (for example .NET, Java or Smalltalk). and discussing security prototype approaches with IBM researchers. Also some experiments with transparently distributing objects have been performed and demonstrated at the EuroPython 2006 conference.
- Extensions and improvements to the testing and development tools as well as to the infrastructure (WP02) supported an open and agile development process. Particularly the generic py.test tool is independently used in dozens of other projects and companies.
- Integration and configuration related refactoring has been started (Work Package 13) to ensure a consistent and integrated access to features and options.

Many of these features are described in detail with the public PyPy 0.9 release documentation. Moreover the (VMC) paper contains in-depth information about architecture results and aspects.



Translation Components: high-level overview on translation architecture.



Basic PyPy Translation Architecture: detailed component and processing view on translation architecture.

During phase 2 of the project we continued to receive high-quality contributions from the growing PyPy community. Among them was a JavaScript backend for PyPy and a tool to use it together with the translation toolchain to produce AJAX applications by translating RPython code. In addition, a .NET, a Squeak and a Lisp backend were experimented with, several extension-modules were ported to PyPy and work was started to make the widely-used Twisted network framework work on top of PyPy.

5 Summary of Dissemination Activities

5.1 Commercial Interest

Dissemination in phase 2 has seen an increase of commercial interest in several aspects of the project - both of the R&D results, the technical infrastructure in use as well as the methodology - sprint driven development. The strategy to support this growing interest have been to firmly establish the results of the project within the eyes of the Python community and the SME's in various European countries using Python as their primary or secondary language.

During PyPy sprints in phase 2 more than 11 different companies using Python have participated and contributed to PyPy. These companies have mainly been involved in web application development and financial application development (modelling and data simulation) - their main interest in PyPy have so far been the performance increase as well as the flexibility offered by the translation tool chain. Sprints have in these cases been functioning as extended tutorials with a full team of core developers present for mentoring. Due to this several of the companies and their developers have already participated or plan to participate in more PyPy sprints.

Besides sprint participances several companies are seeking co-operation with PyPy core members and partners. They are interested in the unique set of skills acquired in the partner companies (presenting opportunities for support and consulting), mainly in areas such as optimizations and performance, but also of PyPy itself, tools, infrastructure and methodology used in the project: Hewlett-Packard, Phillips Medical System, Canonical, CCP Games, Greenpeace International, Iona Technologies, EWT LLC, Next Limit Technologies, IBM etc. In most of these cases project partners have arranged workshops (see "Dissemination through workshops" below) or are planning workshops in phase 3 as means to instigate collaboration. In one case, a company offered the project a sponsored sprint (travel and accommodation) in return for performance and optimization oriented efforts from the team. PyPy could not participate in the sponsored sprint in question due to differences in objectives and focus at the time but there is still an ongoing dialogue about future collaboration possibilities.

The project will continue with supporting commercial inquiries with sprint locating and participation (arranging sprints geographically near commercial interest, mentoring during sprints) and workshops customized towards the specific companies interest and needs throughout the project duration. The strategy behind these techniques is to cater to interest by reducing the learning curve through more customized and accelerated learning sessions. However, these more specialized arrangements require more specialized preparation and mentoring in order to secure sustainability and commercial interest during and after the project. The changed nature of sprints and workshops (influenced by expressed interest from commercial parties) is a contributing reason why the project is requesting an extension of the project.

Below we present the various activities done during phase 2 in order to reach out to both commercial and non-commercial parties, python and non-python communities - covering both the technical results as well as disseminating the successful methodological aspects of the project.

5.2 Dissemination through Sprints

During phase 2 a main strategy for dissemination was to attract more contributions based on the successful results of phase 1 - a self contained PyPy. The primary target group has been the Python community - showcasing PyPy as a platform steady and flexible enough for experimentation of various kinds (backends, optimizations etc). In order to achieve this

result the project have invested extensive time to support and mentor newcomers to the PyPy community - with successful results.

Again, as in phase 1, sprints and conferences such as PyCon 2006 and EuroPython 2006 have been primary forums for raising awareness and interest. Sprints arranged in conjunction with these conferences have shown an unusual high amount of newcomers, making the sprint in effect an extended tutorial (or workshop).

Sprints, their locations and number of participants, arranged in phase 2:

- Paris, France, October 2005 (22 participants)
- Göteborg, Sweden, December 2005 (16 participants)
- Palma University, Mallorca, January 2006 (15 participants)
- PyCon, Dallas, USA, February 2006 (20 participants)
- Louvain-la-Neuve University, Belgium, March 2006 (9 participants)
- Leysin, Switzerland, April 2006 (10 participants)
- Tokyo, Japan, April 2006 (12 participants)
- Heinrich-Heine Universität Düsseldorf, Germany, June 2006 (12 participants)
- EuroPython (CERN), Switzerland, July 2006 (24 participants)

Sprint locations during phase 2 have been the result of dissemination activities and networking during phase 1 of the project. In several cases the location of a sprint has been a means of connecting with a local community of open source developers and other language communities in order to disseminate through participation. Some examples of this:

- PyPy was invited and hosted by the Palma University, their computer science department and local F/OSS group. A PyPy tutorial and and open technical presentations for students and interested parties were arranged.
- PyPy was invited by the Free Software Initiative of Japan after contact was established during a Calibre conference. The sprint was hosted by the Advanced Institute of Science and Technology - piloting their interest in hosting F/OSS events. Again tutorials and technical presentations where done as well as by participation supporting Python in a country where Ruby much more established.
- PyPy instigated a collaborative sprint arranged at the Louvain-la-Neuve University, focused on meeting experts from the OZ community in order to explore how constraints and logic programming could be implemented in PyPy/Python. During the sprint a workshop was arranged to further explore various implementation strategies.

5.3 Dissemination through Conferences (talks and papers)

Participating in and publishing at conferences have been a major effort during phase 2 - in total project partners participated on various levels (talks, papers) in 11 conferences. Conferences were targeted for different purposes - reaching out to different audiences. Towards the primary dissemination group - the Python community - the two main language conferences in that community were the focus for both several talks and sprints for the PyPy team. At PyCon the project presented the following talks: "Agile Open-Source Methods, Businesses and

EU Funding", "PyPy -- Where We are Now" and "PyPy Architecture Session". At EuroPython the project presented the following talks:"An Introduction to PyPy", "PyPy architecture session", "What can PyPy do for you?" and "Kill -1: process refactoring in the PyPy project". These talks cover both the technical progress of the project as well as methodological progress. The project publishes selected talks on

http://codespeak.net/pypy/dist/pypy/doc/extradoc.html

A secondary target for dissemination has been to increase awareness in other communities about the objectives and on-going results of the project. PyPy has been targeting conferences such as the ACCU conference and OOPSLA, arranging workshops as well as presenting talks at universities for computer science students and postgraduates during sprints. We believe these have been efficient ways to present the architectural approach (JIT/VM, translatability and flexibility), which we believe is of interest to other language communities as well as research bodies.

Conference papers and talks oriented towards other software communities during phase 2 have been delivered at various events. The 2CC3 conference (a hacker conference and a meeting place for various different F/OSS communities focusing on different implementations, platforms and languages) had 2 published PyPy papers (and talks) - covering the technological approach as well as the methodological approach: "PyPy - the new Python implementation on the block" and "Open Source, EU-Funding and Agile Methods".

During Solutions Linux in Paris a talk about open source, EU-funding and Python/PyPy - "PyPy - un projet libre doté d'un financement européen" - was presented. At the ACCU conference in Oxford the primary focus of the conference has for long time been C/C++ and Java. A growing interest in Python is evident - this was the second year PyPy visited ACCU and this talk was entitled "PyPy - a progress report". Finally a research paper was accepted at the Dynamic Language Symposium at OOPSLA (the main object oriented software development conference in the world). The title of the paper is "PyPy's Approach to Virtual Machine Construction", and the paper will be presented at the conference in October 2006.

In this secondary target group we have also focused on reaching out to the agile community, presenting the results of our methodological objective - showcasing sprint-driven development. The project had experience reports accepted and published at the two main agile conferences, XP 2006 and Agile 2006. The titles of the reports are "Sprint Driven Development: Agile Methodologies in a Distributed Open Source Project (PyPy)" (SPRINT) and "Trouble in Paradise: the Open Source Project PyPy, EU-funding and Agile Practices" (AGILE).

There is interest in our findings and experiences as a hybrid project - combining distributed open source development with agile practices. Data from PyPy regarding these findings have been used to support research in the area and there are further plans and opportunities for supporting research in phase 3. Finally, a talk was presented at XP Day France, reporting on the practices of the distributed and agile companies participating in the PyPy project.

5.4 Dissemination through Workshops

Besides participating in workshops (such as Calibre and FRCSS) PyPy also arranged two workshops during the period - one methodological and one technical. After initial contacts with IONA at a Calibre conference both parties identified the need to arrange an experience workshop - covering best practices and recommendations regarding distributed and agile practices (sprints, sync-meetings) as well as infrastructure and automated testing. IONA is a leading software development company in Ireland - working from 4 different locations, also

touching aspects of F/OSS with their projects and products. The workshop was held in IONA's facilities in February 2006 and had in total 7 participants (mainly team leaders and middle managers from IONA - two participants from PyPy).

The technical workshop was arranged during the Louvain-la-Neuve sprint in Belgium, March 2006. The main focus was to meet up with experts from the Oz language community in order to discuss the PyPy architecture, the current implementation of a constraint store in Python and the dataflow paradigm and basic mechanisms. Another important topic was the integration of multiple programming paradigms into one language especially looking at Oz, but also at Soul/Smalltalk language symbiosis, which is the combination of a logical and an object-oriented language. The workshop was arranged at the Louvain-la-Neuve University and had 16 participants.

Another project related workshop took place in San Diego where Hewlett Packard invited merlinux to train and explain to them the usage of the new "py.test" testing framework that is used and has been developed in conjunction with the PyPy project. As a result of this four-day training and discussion session, Hewlett Packard is considering deployment of this testing tool on a larger scale, and there is ongoing discussion about an according support and sponsoring contract.

5.5 Collaboration with Other Parties

There have been several opportunities during phase 2 to cooperate with our sister project Calibre - their primary objective being (FP6-funded) research on agile, distributed and open source practices, business models and implementations. PyPy presented a case-oriented talk during 2nd International conference, Limerick, Ireland with the title "Agile Methods in the PyPy Project" - contacts during this conference led to both sprint contacts (Tokyo and a planned Limerick sprint in August as well as potential support to researchers in the F/OSS area), a workshop (IONA) and talks at UCD in Dublin. PyPy also co-arranged with Calibre a seminar for the Commission in December 2006 named "Best Practice in the Use and Development of Free and Open Source Software" in order to support Commission staff with domain knowledge from the F/OSS area. PyPy also participated in the "7th CALIBRE Workshop" in Skövde - covering distributed practices.

Other dissemination activities during the period were:

- a PyPy presentation at the FRCSS at ETAPS 2006, Wien, Austria April 2006
- a PyPy talk and participation in the IST exhibition area at the II Open Source World Conference in Málaga 2006
- a talk at PMI (Project Management Institute) Sweden seminar 2005

Besides sprints, conferences and workshops the project has also had to work with supporting dissemination for on-line interactions. Due to the ever increasing learning curve and the rapid progress of the code base time has also been spent in order to communicate results and ongoing efforts with core Python developers and the community at large in order to help them follow the progress of the project. Sprint reports, diagrams, video documentation (of sprints, tutorials, design discussions and talks) as well as extended tutorial material and demos have been produced and made available on the development web in order to summarize the rapid progress.

Further documentation (talks, papers, EU-reports, diagrams, tutorials, sprint reports, architectural documentation, video documentation) has been published on the PyPy project development website:

http://codespeak.net/pypy

6 Summary of Consortium Activities

During phase 2, in December 2005, the project submitted contractual deliverables from the first reporting period, primarily work finalized during phase 1. These deliverables were:

- D01.1 Create QA plan for the project
- D04.1 Partial Python Implementation on top of CPython
- D04.2 Complete Python Implementation on top of CPython
- D04.3 Report about the parser and bytecode compiler
- D04.4 Release PyPy as a research tool
- D05.1 Publish on translating a very-high-level description
- D05.2 A compiled, self-contained version of PyPy
- D05.3 Publish on implementation with translation aspects
- D05.4 Publish on encapsulating low level language aspects
- D14.1 Report about Milestone/Phase 1

PyPy also submitted a periodic management report (presenting status of costs and budget of the first year) and a periodic activity report (presenting a summary of work done during the first year). These deliverables and reports were the topic for the review from the Commission which took place in Brussels in January 2006. The main feedback from the reviewers, besides an acceptance of all deliverables from year 1, were incorporated into Amendment 4 which restructured the 58 deliverables (of the entire project) into 21 contractual deliverables (10 already submitted). There was also a change, based on commercial community demands, towards including work on the extension compiler (an early version of which was already released with PyPy 0.9 in June), explained in detail in the Amendment 4 documentation.

It is interesting to note is that the reports finalized and submitted to the Commission have also been used by people in the community in order to prepare for sprint participation when they felt in need of more extended documentation. This led the project to complete and publish interim reports in phase 2 (see D07.1 and D09.1), endorsed by the reviewers, in order to provide status and documentation before an actual deliverable is due. They are published on the project website so that the community can use and feedback on them as well. Moreover, we invite particular experts to scrutiny our results, and thereby also disseminate to interesting multipliers.

The project has also performed its planned assessment of work, resources and costs at the end of phase 2 in order to make adjustments for the last phase of the project - anticipating change instead of reacting late phase 3 to possible changes already incurred. These plans were also endorsed by the reviewers. This work will result in amendment 5 which will be submitted early August. The main focus of amendment 5 is to adapt and refine plans for reaching all goals of the project, and accounting for resource problems and changes of certain partners' main resources. At the same time, we try to reach out more strongly to community and commercial interests, e.g. by deploying the Summer of PyPy model as a way to fund sprint participation and increase contributions to the project.

In preparation for the review - anticipating the work needed in order to prepare Amendment 4 and 5 - there was a need to create a more agile structure of managing consortium level issues. The consortium decided to organize the work in a way that better followed the pace and need of the development process, still keeping formal roles and responsibilities staked out in Annex I. An agile management team works on a need-to-work basis, identifying consortium level issues (such as amendments) and involves people from the different partner companies based on the necessary skills needed for the specific tasks. The team then usually prepares the issue as a decision item by talking to individual partners first and then the consortium only gathers to make a decision. This more efficient and focused structure is also more fitting to the more diverse nature of work during the second year (covering both phase 2 and 3).

In the second phase the consortium had 14 meetings, 3 of which were physical meetings - the rest of the meetings was done via IRC in the #pypy-funding channel on freenode.net.

Moreover, the Technical Board of the project met 10 times in order to co-ordinate technical work, releases, sprints and to prepare consortium decisions from a technical perspective. The Board members and particular invited participants discussed crucial technical problems, approaches and reached consensus agreement on virtually all topics.

During phase 2 the following partners contributed to the project:

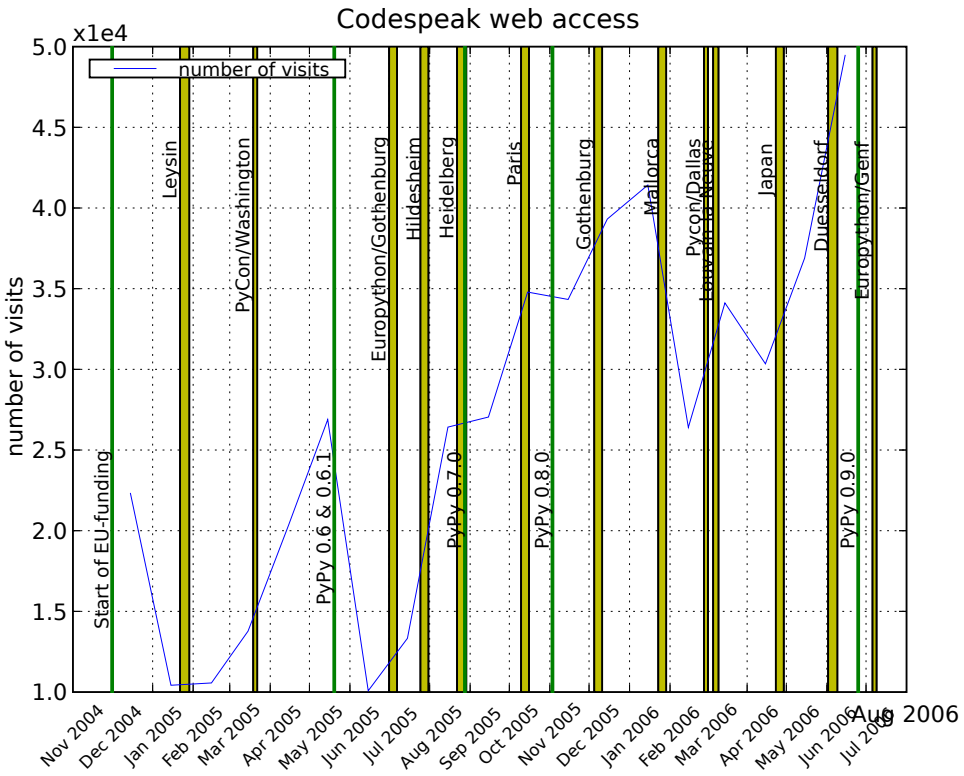
- DFKI Saarbrücken, Germany
- AB Strakt, Göteborg, Sweden
- Logilab, Paris, France
- Change Maker, Göteborg, Sweden
- merlinux GmbH, Hildesheim, Germany
- tismerysoft GmbH, Berlin, Germany
- Heinrich-Heine-Universität Düsseldorf, Düsseldorf, Germany (HHU)
- impara GmbH, Magdeburg, Germany
- Laura Creighton
- Eric Van Riet Paap
- Richard Emslie
- Niklaus Haldimann

7 Community Aspects / Statistics

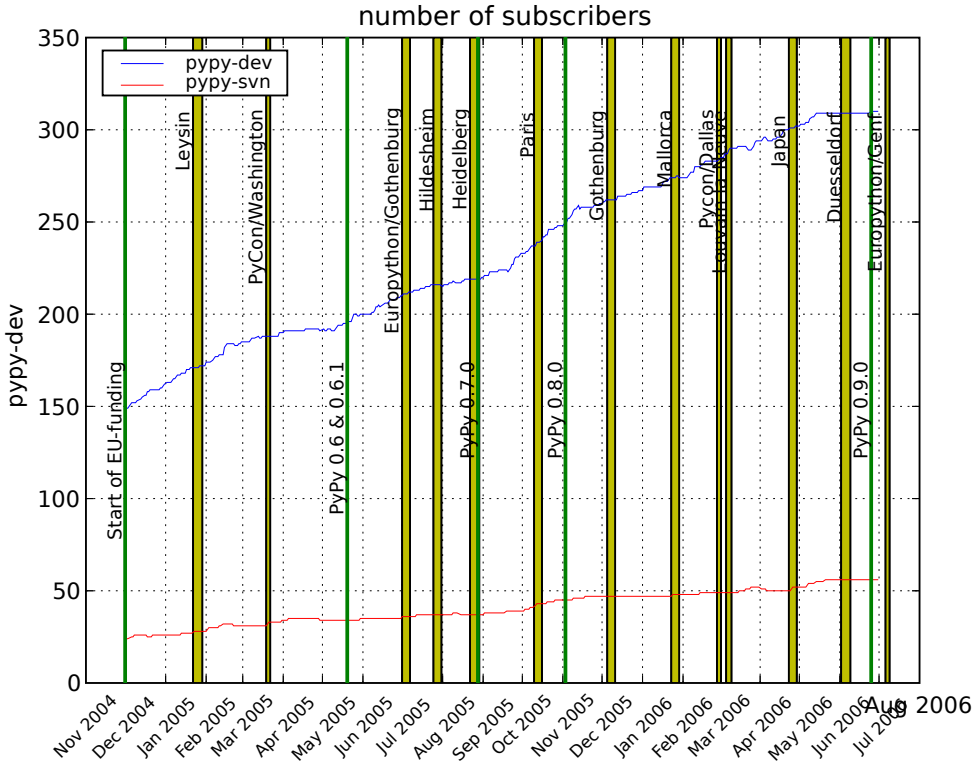
During phase 2 and continuing into phase 3 the project has increased its efforts of leveraging the community in order to support sustainability with again participating in Google's Summer of Code program of mentoring. The PyPy team have also instigated it's own model for mentorship and sponsoring contributions through funding sprint participation - the "Summer of PyPy" which was announced during July 2006.

This increased effort led to a continued growing interest among various communities most notably the Python and PyPy community itself. An estimated 1200 downloads of the new video documentation features, the increased number of people following code changes and

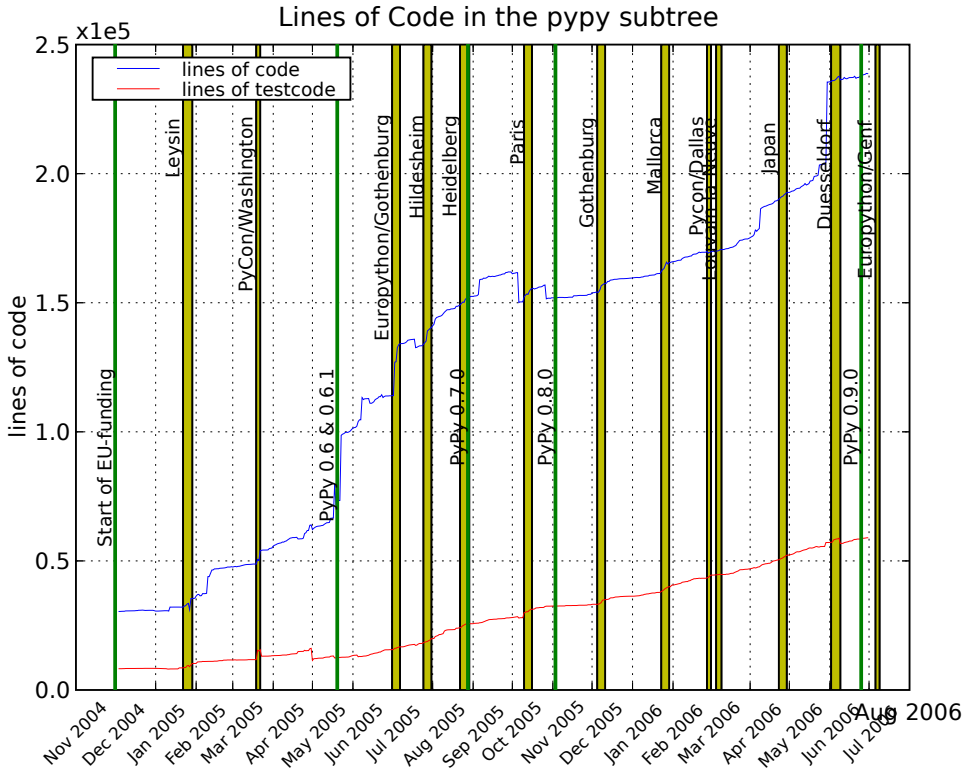
the general mailing list, and the increased web page access underline the raised interest. However, it is also true that due to the project's fast pace and its many developments, it requires substantial effort for the average community member to contribute to the project. Also the mentoring and supporting activities from the EU project members have increased accordingly. We consider this a worthwhile effort as it majorly contributes to the "Leveraging the Open Source community" major goal of the project and, eventually, to sustaining the project beyond the funding period.



Web visits increasing on PyPy pages: With the releases and sprints the web page visits usually peaks - but the graph also shows a sustained raise in interest.



Number of visible subscriptions increasing: In June 2006 more than 50 developers are following changes to the code base, from which only around 20 people are benefiting from EU funding. Also the number of subscriptions to the developer mailing list crossed the "300" barrier, showing a growing interest in the project discussions and ongoings.



Lines of Code: PyPy's code base has reached 250 KLOCs with a large fraction (60Klocs) of test code. The project continues to rely on test-driven development and now has around 3000 automated tests.

8 Conclusion and Outlook

Despite resource constraints and external factors the project kept its pace and produced the major technical, dissemination and research results scheduled for phase 2. It has continually adapted its management and co-ordination style - true to the agile principles.

Some delays arose due to illness and resource constraints but all risks have been considered and are tackled by the partners in a co-operative and effective manner.

The project now implemented almost all Review recommendations from January and has performed an internal assessment and re-planning phase which will allow it - pending acceptance of the planned Amendment 5 - to successfully complete the project withing budget.

With its focus on involving the community and its increased dissemination activities, also towards commercial entities, and its promising technical and research results, there is a strong perspective for sustaining the project after the EU funding period.

9 Glossary of Abbreviations

The following abbreviations may be used within this document:

9.1 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.
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.
Graphviz	Graph visualisation software from AT&T.
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 Library - a cross-platform multimedia library designed to provide fast access to the graphics framebuffer and audio device.
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.

9.2 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

References

(VMC) *PyPy's approach to virtual machine construction*, Armin Rigo, Samuele Pedroni, draft paper, accepted to OOPSLA, to appear in 2006

(AGILE) *Trouble in Paradise: the Open Source Project PyPy, EU-funding and Agile Practices"*, Beatrice During p.221-230, Proceedings Agile 2006, Minneapolis MN, 23-28 July 2006,Editors Joseph Chao, Mike Cohn, Frank Maurer, Helen Sharp, James Shore, IEEE Computer Society Conference Publishing Service

(SPRINT) *Sprint Driven Development, Agile Methodologies in a Distributed Open Source Project (PyPy)*, Beatrice During p. 191-195, Extreme Programming and Agile Processes in Software Engineering, 7th International Conference, XP 2006, Oulu, Finland, June 2006, Proceedings, Editors Pekka Abrahamsson, Michele Marchesi, Giancarlo Succi, Springer-Verlag Berlin Heidelberg 2006

(PYPYNET) *Implementing Python in .NET*, Antonio Cuni, Thesis Universita' Degli Studi Di Genova, July 2006

(BLOCK) *PyPy - the new Python implementation on the block, Language/VM R&D, whole program type inference, translation to level backends, fun*, Armin Rigo, Carl Friedrich Bolz, Holger Krekel, p.273-278, 22. Chaos Communication Congress Proceedings, Verlag Art d´Ameublement Bielefeld, 2005

(OSEU) *Open Source, EU-Funding and Agile Methods*, Beatrice Düring, Holger Krekel, p.267-272, 22. Chaos Communication Congress Proceedings, Verlag Art d´Ameublement Bielefeld, 2005