IST FP6-004779

PYPY

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

STREP

IST Priority 2

Report About Milestone/Phase 2

Due date of deliverable: June 2006

Actual Submission date: Changemaker

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

Lead Contractor of this WP: XXX insert submission date here l

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

Revision: XXX insert revision here

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

Abstract

This document describes the results of work in phase 2 of the PyPy project. The results are summarized on technical, dissemination, research and community levels. In the conclusion we state some of the effects of the work done and we also present briefly work planned for the last phase of the project.

Contents

1 Executive Summary

The major technical milestone of the second phase was marked by the public 0.9 PyPy release in June 2006 implementing a 20-times performance improvement, massive parallelism, full garbage collection integration, Logic Programming idioms and providing a separate extension compiler.

The project increased dissemination efforts on all levels: it organised nine international sprints, attended eleven conferences and held four workshops during the second phase of the project. It particularly continued to involve the Python community itself with several presentations at each of the two major conferences, Pycon 2006 (Dallas) and EuroPython (CERN/Geneva) in addition to non-domain specific conferences and events such as 22C3 publishing about PyPy - the new Python implementation on the block and about Open Source, EU-Funding and Agile Methods.

With the Summer of PyPy call for proposals, the project further embarks on its strategies to raise community contribution, validation of the technical architecture and its methodological approaches. With its new eight Video Documentation features the project provides lively insights into its technical and methodological nature and also highlights the appreciation of central persons from the Python community.

The project moreover published a research paper for the Dynamic Language Symposium, summarizing PyPy's new approaches towards the language implementation research community. Relatedly, it is considered a remarkable success that a student graduated within five months by exploring and porting PyPy's translation tool chain to Microsoft's .NET environment, an effort that was welcomed and supported by the PyPy core group, because it serves both as early validation of PyPy's architecture and broadens its overall applicability. (XXX compare to Iron Python efforts?)

The project also published two methodology papers ("Sprint Driven Development: Agile Methodologies in a Distributed Open Source Project (PyPy)"; "Trouble in Paradise: the Open Source Project PyPy, EU-funding and Agile Practices") for the two major agile software development conferences (XP 2006 and Agile 2006). Papers were accepted as experience reports despite harsh competition due to the showcasing of a hybrid project - integrating open source and distributed development with agile practices, findings of great interest to the agile community.

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 also finalized Amendment 4 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 of the project's execution arose from delays and resource deployment difficulties of some consortium partners, foremost because Tismerysoft was majorly affected by serious health issues of its key person. This has been communicated immediately to the Commission. The main problems arose in Work Package 7, where this partner on the one hand followed commercial opportunities related to his PyPy involvement, and on the other hand his key person suffered a serious health incident early June. However, other partners collaboratively helped to achieve most crucial results in this work package (included in the public 0.9

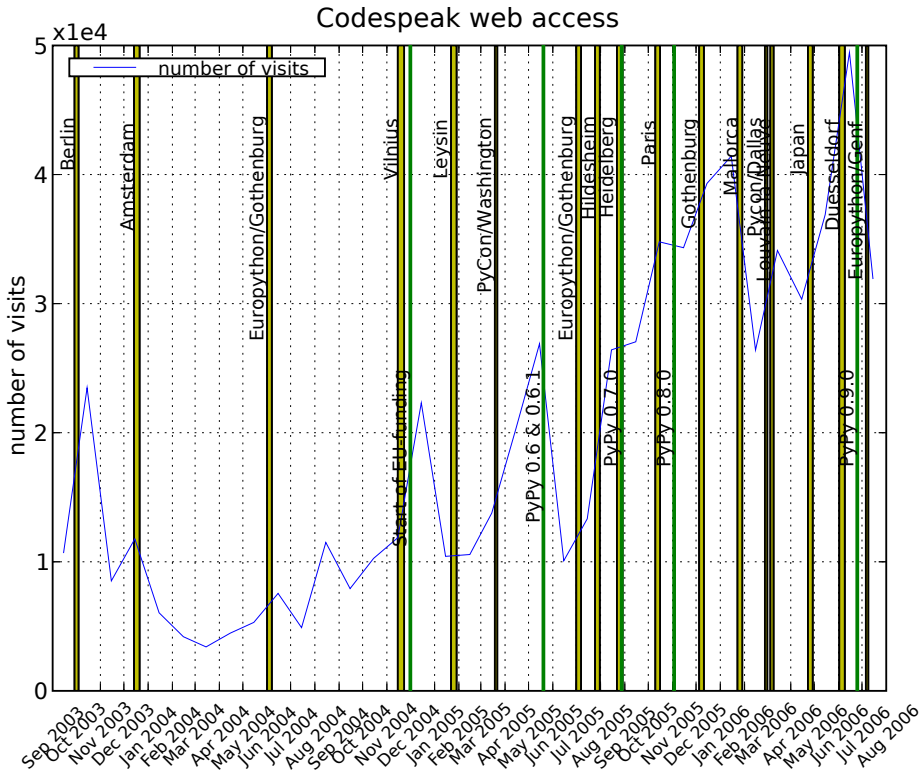


Figure 1: Web visits increasing on PyPy pages

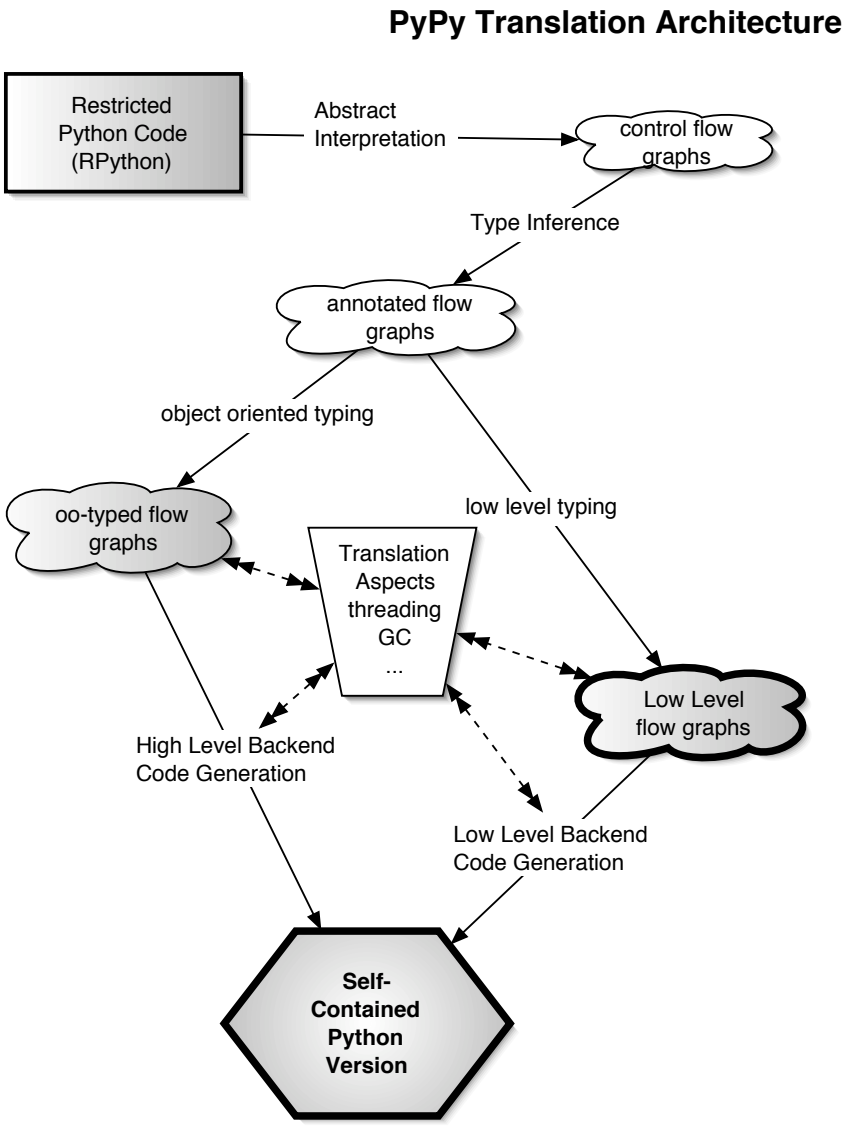


Figure 2: Basic Translation Architecture

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 achived 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 a project extension in order to successfully achieve all promised 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 and 9 there are only interim reports available, and each is intended 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 achieves 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 increasining performance and met its major technical milestones with the 0.9 public PyPy release in June 2006. Within the second phase the project already validated a

number of architectural aspects and also increased flexibility of the whole system, originally a topic for phase 3. In fact, due to the project's goals of connecting research, a practical implementation and leveraging the open source community with agile methods, the project generally weakend (XXXweakened??) and intermingled phase 2 and 3 aspects.

Validation, research and new feature developments as well as performance improvements are dynamically 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) has already started to help everyone to make easier use of PyPy's many options and features.

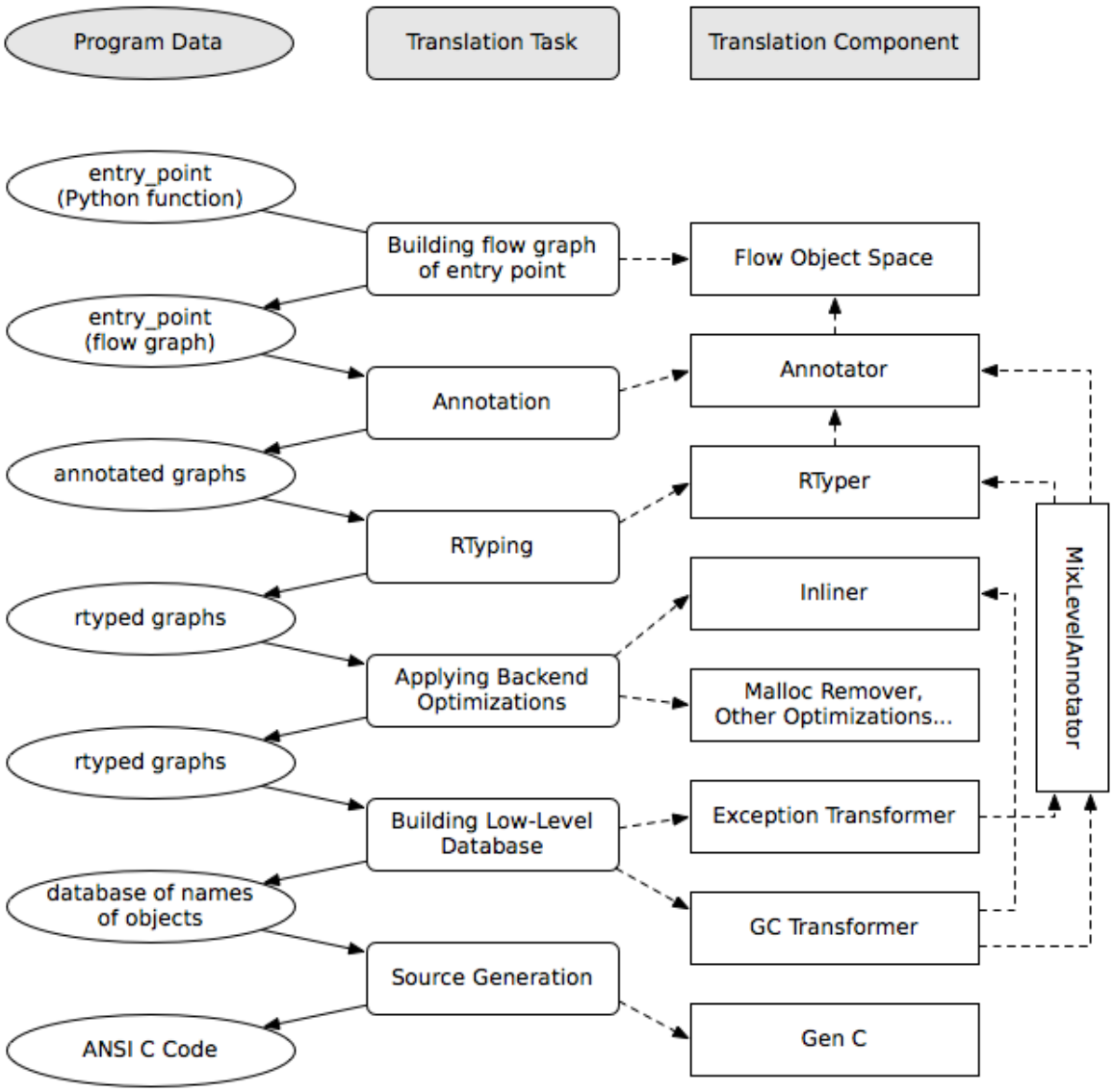
4 Summary of Technical Results

The major achivements 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. It is still 3-10 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 improvements are expected during the execution of the core optimizations.
- An early release of an extension compiler (WP 3) which provides a flexible way to write extension modules that work unmodified both with PyPy and as as a standalone CPython module. Moreover, the translation toolchain is capable to transform 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.
- Several translation aspects (Exception and Garbage collection most noticeably) have been re-implemented as Flow Graph transformations rather then being implemented for backends specifically. (WP07)
- Application level access to massive parallelization features such as tasklets, including tasklet cloning and pickling (WP7).
- 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. (WP7).
- PyPy continues to get more complete in terms of core Python functionality (WP 3), most noticeably there is a new generic implementation of "weakref", providing weak references in context of custom garbage collectors (WP 3 and 7). The core language tests continue to pass at a rate of 95% despite many changes to the code base.
- New Logic and constraint programming features got implemented by means of a "logic" Object Space that provides support for python logic variables and dataflow syncronisation of coroutines. (WP9)
- First results of the work on "Dynamic Optimisations" (WP08) showcase a Just in Time compiler compiler for a turing-complete "toy language" interpreter, re-using PyPy translation architecture.

- Preliminary results on the validation Work Package 12, providing easy paths to Higher Level Backends (.NET, Java, 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.

Also all of these features are described in detail with the public PyPy 0.9 release documentation.



XXX maybe insert and comment on contiued compliance test data?

XXX insert and comment on benchmark/performance (reason is annex 1 B6.2.2 speaks of performance)

XXX - in-depth explanations on community contribution - back ends and commercial interest (in what way extension compiler/Rpython yields interesting results and are being put to use already)

5 Summary of Dissemination Activities

During phase 2 the main strategy for dissemination was to attract increased contribution based on the succesfull results of phase 1 - a self contained PyPy. The primary target group have been the Python community - showcasing PyPy as a platform steady and flexible enough for experimentations of various kinds (back-ends, 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, using the sprint as an extended tutorial (or workshop).

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

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

Due to the ever increasing learning curve and the rapid progress of the code base time have 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, videodocumentation (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 help to "distill" important feature- and architectural knowledge.

Conferences and talks oriented towards the Python community during phase 2: * PyCon,Dallas, USA February 2006 * EuroPython 2006, Cern, Switzerland July 2006

We belivieve that this strategy towards the primary target group have indeed successful, yielding not only increased contribution but also a keen interest in the final results of the project. Increased discussion regarding requirements, recommendations and "external" test feedback suggests this (see metrics and comments below regarding community contribution).

A secondary target for dissemination have been to increase awareness in other communities about the objectives and on-going results of the project. PyPy have been targeting conferences such as ACCU 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 (XXX short words to summarize this) which we believe is of interest to other language communities as well as research bodies.

Conferences,papers and talks oriented towards other software communities during phase 2: * 2CC3 conference, Berlin, Germany December 2006 * Solutions Linux, Paris, France January 2006 * ACCU, Oxford, United Kingdom April 2006 * OOPSLA 2006, Portland, USA 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 papers accepted and published at the two main agile conferences (XP 2006 and Agile 2006). There is an 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.

Conferences and talks oriented towards the agile community during phase 2: * 2CC3 conference, Berlin, Germany December 2006 * XP Day France, Paris, France March 2006 * XP 2006, Oulu, Finland June 2006 * EuroPython 2006, Cern, Switzerland July 2006 * Agile 2006, Minneapolis, USA July 2006

There have also been several opportunities during phase 2 to cooperate with our sister project Calibre - their primary objective being research on agile, distributed and open source practices, business models and implementations.

- Calibre, 2nd International conference, Limerick, Ireland, September 2005
- Joint seminar with Calibre for the European Commission,Bruxelles, Belgium December 2005
- 7th CALIBRE Workshop: Distributed Software Development, Skövde, Sweden March 2006
- FRCSS at ETAPS 2006, Wienna, Austria April 2006
- PMI Sweden seminar, Linköping, Sweden October 2005

Commercial impact is slicing vertically through both the primary and secondary target groups for dissemination. Within the primary target group - many of the requirements raised as well as mentoring being done have been tied to commercial interest (primarily perfomance). The Python community is a network of people, but people who work in companies using Python for commercial purposes and who are keenly interested in PyPy results due to bottlenecks in the current main language implementation CPython. There have also been a commercial interest in the development infrastructure being used, especially in the automated test-framework py.test which also have been disseminated during phase 2. (XX should we be more specific here - mention companies and more details of their areas of interest and practical usage etc?)

Within the secondary target group XXX should we mention oz, smalltalk, squeak, jython - maintanability - other interests (optimizations) ??? Finally there have also been a commercial interest in the methodology - especially oriented towards how to make development more efficient and "secure" when coordinating work between distributed teams. The sprint-driven approach as well as the supporting infrastructure haven been viewed as of direct practical usage to companies we have been supporting and mentoring in connection with talks and workshops.

Workshops: XXX Bea, can you describe each of the these workshops in a small paragraph, possibly with separate section headers?

IONA/PyPy Dublin, Ireland February 2006 Oz/PyPy Lovain-la-Neuve, Belgium March 2006

All documentation (talks, papers, EU-reports, diagrams, tutorials, sprint reports, architectural documentation, videodocumentation) have been published on the PyPy project development website:

http://codespeak.net/pypy

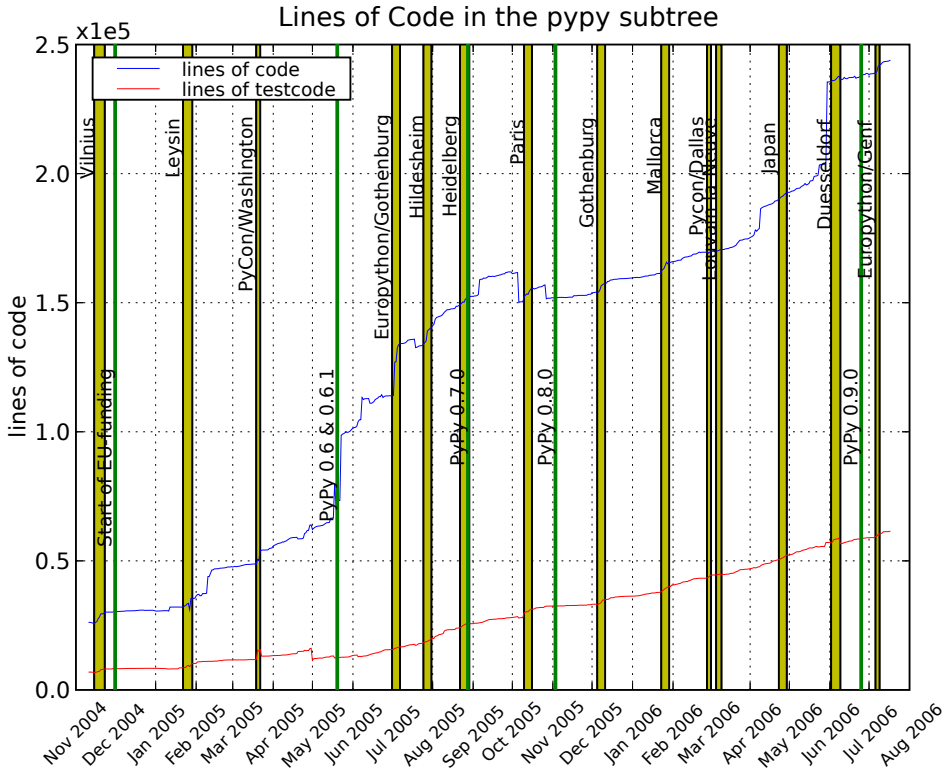
This website is also the primary communication channel (mailinglist) for the PyPy community, combined with IRC channels on freenode (for details - see http://codespeak.net/)

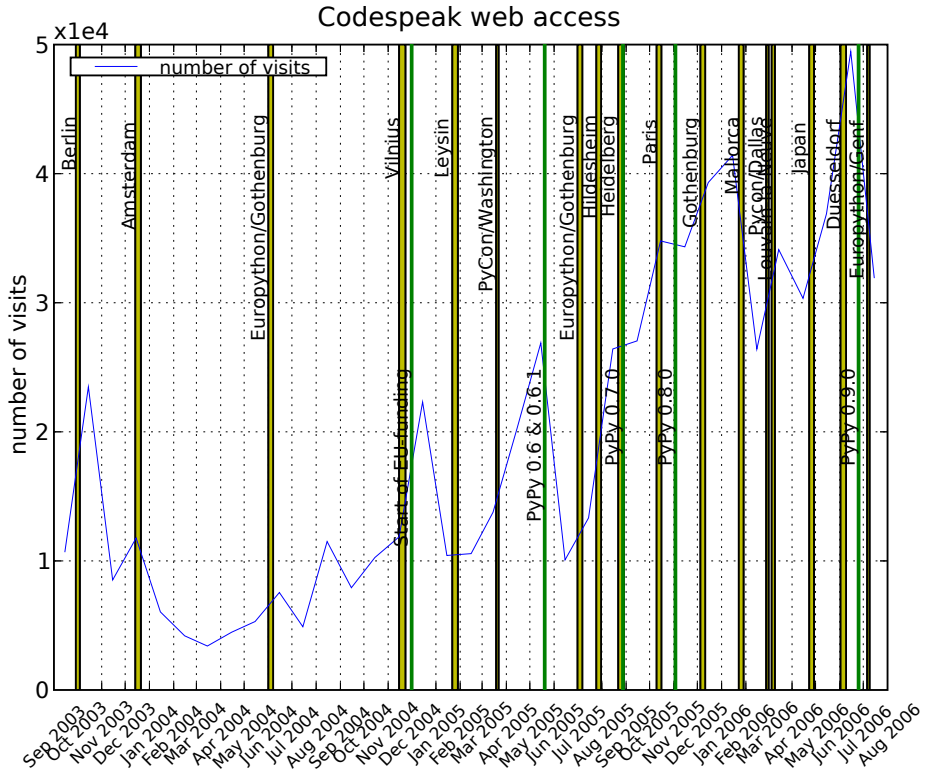
6 Summary of Consortium activities

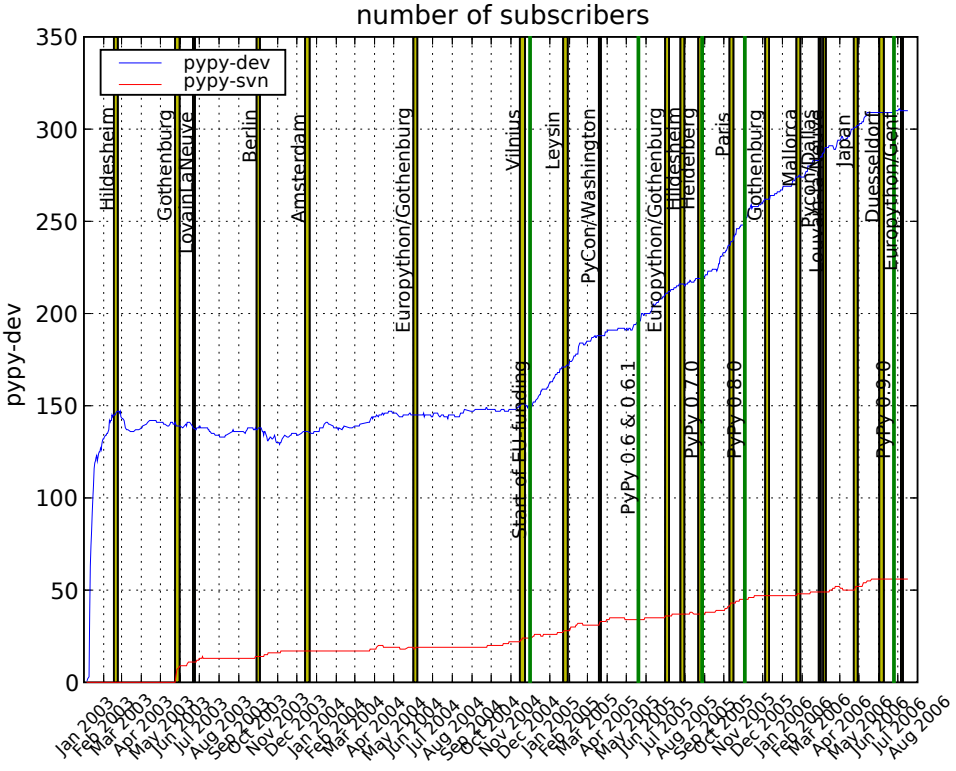
XXX very short - just a listing of partners and amendments /direction of project - finish on sunday - latest next week.

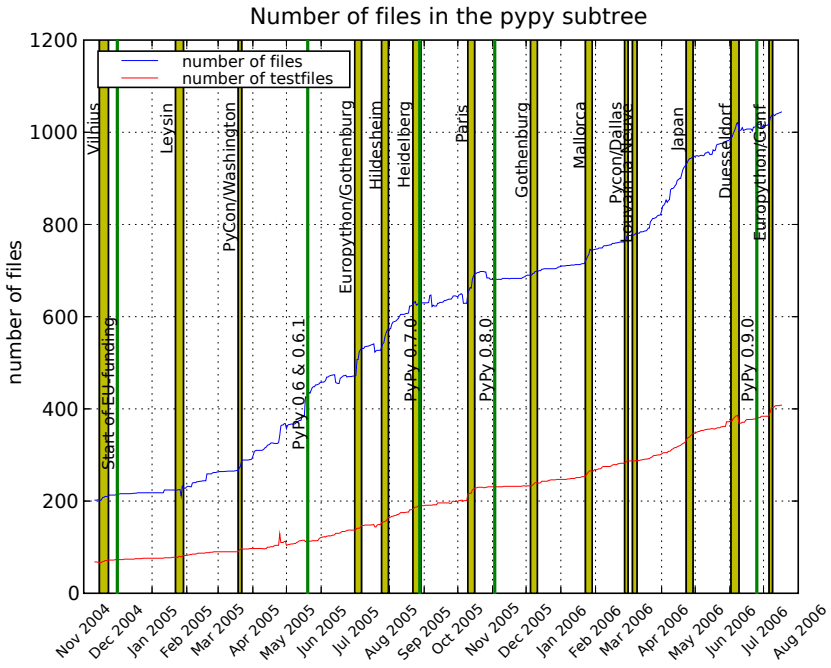
7 Community Aspects

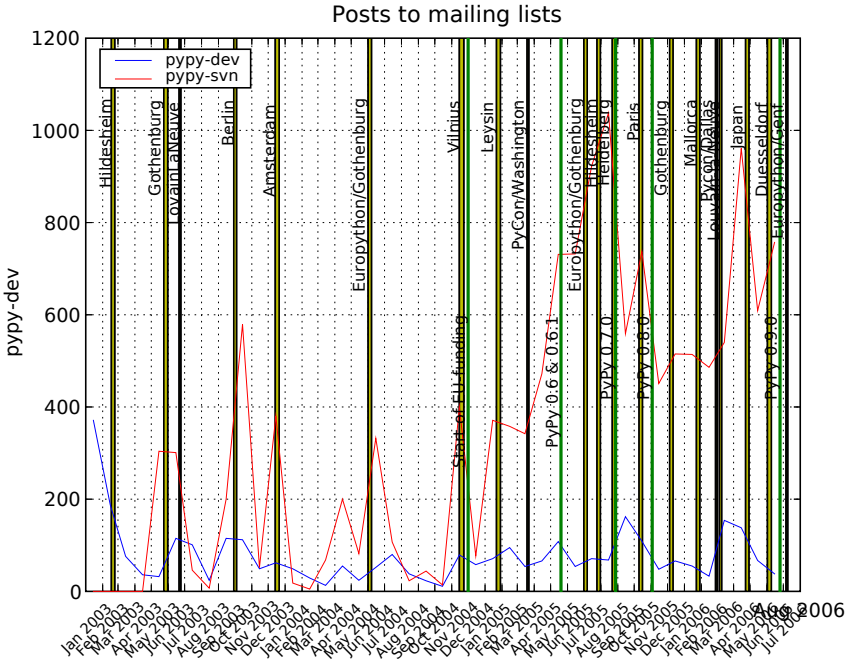
XXX Will work on this during next week (during conference) - have to study the metrics.




Codespeak web access
x1e4
number of visits
Berlin
Amsterdam
Europython/Gothenburg
Vilnius
Start of EU-funding
Leysin
PyCon/Washington
PyPy 0.6 & 0.6.1
Europython/Gothenburg
Hildesheim
Heidelberg
PyPy 0.7.0
Paris
PyPy 0.8.0
Gothenburg
Mallorca
Pycon/Dallas
Louvain-la-Neuve
Japan
Duesseldorf
PyPy 0.9.0
Europython/Gent


number of subscribers
pypy-dev
pypy-svn
Hildesheim
Gothenburg
Louvain-la-Neuve
Berlin
Amsterdam
Europython/Gothenburg
Vilnius
Start of EU-funding
Leysin
PyCon/Washington
PyPy 0.6 & 0.6.1
Europython/Gothenburg
Hildesheim
Heidelberg
PyPy 0.7.0
Paris
PyPy 0.8.0
Gothenburg
Mallorca
Pycon/Dallas
Louvain-la-Neuve
Japan
Duesseldorf
PyPy 0.9.0
Europython/Gent





8 Conclusion and Outlook

XXX Will finalize this on sunday or latest - during next week.

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.
docutils	The Python documentation utilities.
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.

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

XXX REFERENCES: .. research paper for the Dynamic Language Symposium: