Perl 6 Development Grant
Final Report

Patrick R. Michaud
September 10, 2008

INTRODUCTION

This is the final report for the Perl 6 and Parrot development grant provided by the Mozilla Foundation and The Perl Foundation. This report summarizes the work performed under the grant, what has been accomplished, and where things are headed from here. As this report illustrates, we have achieved all of the goals and outcomes intended for this grant.

REPORT

The primary focus of the project was to get Perl 6 on Parrot development "over the hump" and acquire a critical mass of infrastructure, developers, and tools from which sustained further development of Perl 6 can take place. The project proposal identified five specific expected outcomes:

1. A working Perl 6 on Parrot implementation that supports commonly used Perl constructs and operators
2. Review and improvements to the Perl 6 language test suite
3. A substantially complete Parrot Compiler Toolkit, with documentation
4. Ongoing, active development efforts for other languages based on the Parrot Compiler Toolkit
5. An increased number of participants in Perl 6 and Parrot design, implementation, and testing

1. A working Perl 6 on Parrot implementation

We now have a substantial Perl 6 implementation on Parrot. At the beginning of this project, we had a rudimentary Perl 6 compiler that could handle some of the basic features and syntax of Perl. However, some key features such as arrays and hashes were only marginally implemented, and most of the translation components of the compiler were written in Parrot's intermediate assembly language (PIR).

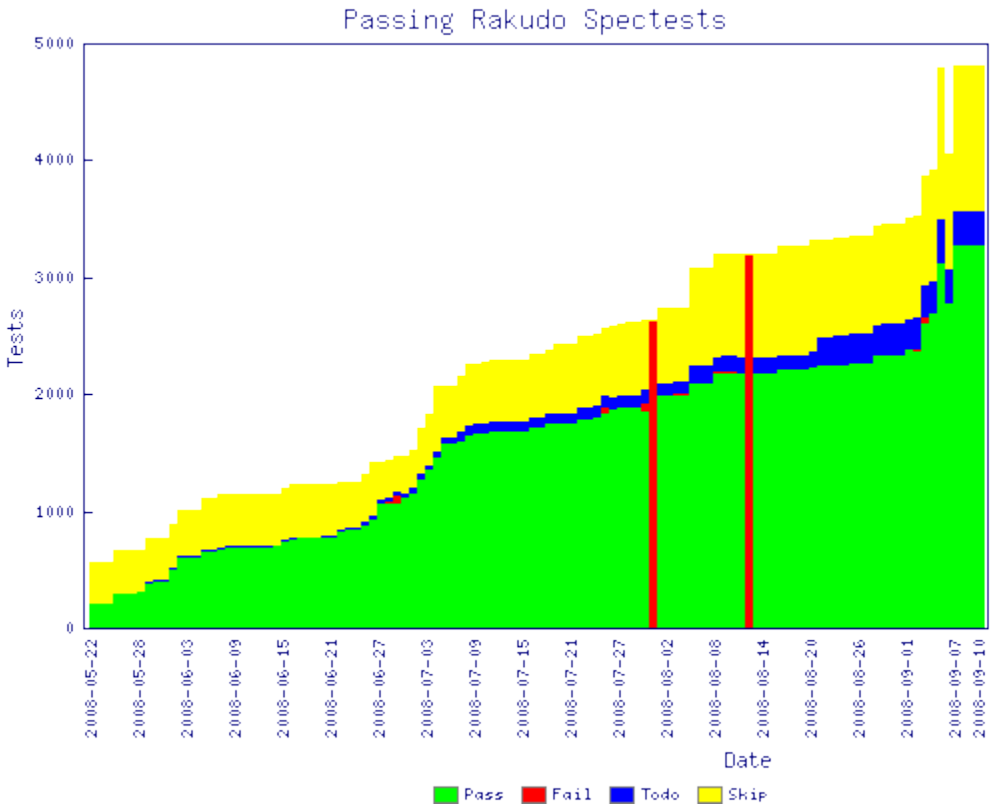
Today the Perl 6 on Parrot compiler has been substantially rewritten using the Parrot Compiler Toolkit (described below), and is now known as "Rakudo Perl" or "Rakudo" [1]. Rakudo currently supports arrays, hashes, classes, objects, inheritance, roles, enumeration types, subset types, role composition, multimethod dispatch, type checking, basic I/O, named regular expressions, grammars, optional parameters, named parameters, slurpy parameters, closures, smart match, junctions, and many other features expected from Perl 6. Rakudo has progressed to the point that others are now using Rakudo as an Apache handler ("mod_perl6") [2], a wiki engine ("November") [3], and recently to build applications on Xlib [4]. The Perl Foundation also recently awarded a grant to Vadim Konovalov to implement a Tk GUI interface for Rakudo [5].

During the project there have been many contributors to the development of Rakudo Perl; some of the major contributors include chromatic, Vasily Chekalkin (bacek), Jerry Gay, Jeff Horwitz, Moritz Lenz, Carl Mäsak, Cory Spencer, Stephen Weeks (Tene), and Jonathan Worthington. Jonathan Worthington's efforts deserve special mention: he is primarily responsible for the bulk of the work on classes and types in Rakudo Perl, and much of his work was supported by a grant from Vienna.pm [6].

2. Review and improvements to the Perl 6 test suite

At the beginning of this project, the test suite that existed for Perl 6 had been implemented as part of the Pugs effort and contained approximately 16,000 tests. However, in many cases the tests were out of date with respect to the current Perl 6 language specification, and we needed a way for multiple independent Perl 6 implementations to be able to easily make use of the test suite.

In December 2007 I proposed a structure for reorganizing the test suite [7]; Jerry Gay and Larry Wall then extended this proposal and developed tools to make it easier to share the tests among multiple implementations. In May 2008 Moritz Lenz refactored Rakudo's test harness to provide a "make spectest_regression" target -- since then this has become our primary measure of Rakudo progress [8]. Moritz Lenz also developed a tool to display the progress in graphical form:



Adrian Kreher received a Google Summer of Code grant (with Moritz Lenz and Jerry Gay as mentors) to continue the test suite refactoring [9]. As of early September 2008, the official, refactored test suite contains a little over 8,000 tests, or approximately half the size of the original Pugs test suite. Furthermore, the official suite includes many new tests for object-oriented and typing features of Perl 6 that weren't present in the original test suite. As the graph above indicates, Rakudo is currently passing over 3,200 of the tests in the official suite, and work is continuing on refactoring the suite and increasing Rakudo's pass rate.

3. A substantially complete Parrot Compiler Toolkit, with documentation

As mentioned previously, the compilers that existed for Parrot at the beginning of the project (such as Perl 6) were primarily written in Parrot's intermediate assembly language. This made working on the compilers less accessible to new developers, as well as increasing the time needed to build a compiler. Therefore, the first couple of months of this project were spent on developing the Parrot Compiler Toolkit (PCT) and a simple Parrot language called Not Quite Perl (NQP). PCT provides an abstract syntax tree representation and code generator for Parrot languages; NQP enables compiler writers to easily create compilers and builtin functions for Parrot using a simplified Perl 6 syntax.

Once PCT and NQP were substantially complete, we were then able to convert Perl 6 (now "Rakudo Perl") and many other Parrot compilers to use the new toolkit. This went surprisingly quickly -- most of the existing compilers were converted within just a couple of weeks. In addition, a few new compilers and languages arrived on the scene: Will Coleda and Simon Cozens quickly created an implementation of LOLCODE [10], and Klaas-Jan Stol created a language called "Squaak" as a demonstration and tutorial for the toolkit [11]. A couple of quotes from the period give a sense of how these tools opened up Parrot development to others:

> "The real fun, though, has been digging into perl6, the Parrot Perl 6 implementation. Recently, Patrick Michaud has been doing some incredible work building NQP (Not Quite Perl 6), a bootstrapping language for implementing Perl 6, and extensively refactoring the existing Perl 6 on Parrot compiler to fit with it. I'm still very much getting my head wrapped around the whole thing, but it's been easy enough to start digging into and implementing and fixing a few things."
>
> - Jonathan Worthington, December 2007 [12]

> "It's really, really true. Parrot lets you implement your own languages using Perl 6 rules for the grammar and Perl 6 for the compiler."
>
> - Simon Cozens, January 2008 [13]

Both PCT and NQP stabilized early in the project; recent changes have been primarily to optimize existing features or make other minor improvements. We expect these tools to continue to evolve as needed to support compiler development; however, given the wide variety of languages being implemented using the toolkit surprisingly few changes have been needed. Primary documentation for the toolkit consists of a Parrot Design Document (PDD26) for the abstract syntax tree representation [14], the Squaak tutorial [11], and numerous example languages in the Parrot repository. More detailed documentation and examples for the toolkit are expected to be developed over the coming months.

4. Ongoing, active development efforts for other languages based on the Parrot Compiler Toolkit

Currently there is active development on at least three languages for Parrot; these include Perl 6 (Rakudo Perl), PHP (Pipp), and Ruby (Cardinal). In addition, there is ongoing but less active development on implementations of Python (Pynie) and Smalltalk (ChitChat). All of these are based on the tools from the Parrot Compiler Toolkit. The Perl 6 and PHP implementations are usable from mod_parrot [2], and our next steps are to improve the toolkit and library conventions to support loading multiple languages simultaneously in Parrot.

5. An increased number of participants in Perl 6 and Parrot design, implementation, and testing

There can be little question that momentum for Perl 6 and Parrot development continues to grow, and the work supported by this grant has been a major catalyst for that growth. For the twelve months prior to September 2007 the Parrot subversion repository had a total of 6,634 commits; in the subsequent twelve months Parrot had 9,686 commits -- a year-over-year increase of 46% in the commit rate.

At the beginning of this project there were perhaps three or four active contributors to the Perl 6 on Parrot compiler (i.e., Rakudo Perl); over the course of the past year that number has increased to at least ten active contributors and at least as many more occasional contributors to Rakudo Perl. Parrot and the Parrot Compiler Toolkit have also garnered new contributors, including several new committers. There are also new teams of developers working on applications based on Rakudo Perl and Parrot, such as the November wiki engine and the Rakudo/Tk GUI interface.

Lastly, recent improvements in the Rakudo compiler architecture will allow much of the Perl 6 runtime library currently written in PIR to be rewritten substantially in Perl 6. This will enable even more new contributors to participate in Perl 6 development.

PROJECT EVALUATION

The project proposal identified five criteria by which the success of this project could be measured:

- Ability to write and test Perl 6 programs and language features
- Passing rate for Perl 6 language test suite
- Improved coverage and accuracy of the Perl 6 language test suite
- Increased number of participants in Perl 6 and Parrot development
- Active development of at least two other languages using Parrot compiler tools

This report demonstrates the success of this project along all of these measures. Perl 6 on Parrot ("Rakudo") is being used to write and test Perl 6 programs and language features, and has become a strong driver for improving the coverage and accuracy of the test suite (and indeed, of the Perl 6 language specification itself). Continual reporting and publication on Perl 6 and Parrot activities continues to attract increased interest and participation from the wider community.

Moreover, this project helped jump-start even larger fund-raising efforts. In May 2008 the Perl Foundation received a $200,000 philanthropic donation from Ian Hague; roughly half of this donation is intended to continue the Perl 6 development efforts that have been part of this project [15]. And, as mentioned earlier, Jonathan Worthington and others are receiving grants for continued work on Rakudo Perl and Parrot [6,8,16].

CONCLUSION AND FUTURE WORK

The funding provided by the Mozilla Foundation and The Perl Foundation has indeed enabled us to get Perl 6 on Parrot development "over the hump" in development. We now have a robust platform for higher-level language development on Parrot, along with an active and growing development and support community (which is the hallmark of any successful open source project). All of the desired outcomes of this project have been realized.

Our next steps will be to continue to extend and build upon the work of this project -- increasing Rakudo's coverage of the Perl 6 language and bringing all of our efforts much closer to production releases. In fact, we recently developed a "road map" for Rakudo Perl that identifies the major steps to be taken in the upcoming months and assists in coordinating the remaining development activities [17].

Lastly, I want to express my sincere appreciation to the many people who contribute time and energy to Perl 6 and Parrot, and to give special thanks to the people of the Mozilla Foundation and The Perl Foundation for their ongoing support and enthusiasm for this project. It is a great honor to work and correspond with such a terrific and professional group of individuals.

REFERENCES

1. P. Michaud (January 16, 2008), "The compiler formerly known as 'perl6'", http://use.perl.org/~pmichaud/journal/35400
2. J. Horwitz, "Mod_parrot website", http://www.smashing.org/mod_parrot/
3. C. Mäsak, "Announcing November, a wiki in Perl 6", http://use.perl.org/~masak/journal/37212
4. http://svn.perl.org/parrot/trunk/examples/nci/xlibtest.p6
5. Alberto Simões (August 30, 2008), "2008Q3 Grants Results", http://news.perlfoundation.org/2008/08/2008q3_grants_results.html
6. "Vienna.pm funds Jonathan Worthington to work on Ra[kudo]" (April 23, 2008), http://use.perl.org/article.pl?sid=08/04/23/2314234
7. P. Michaud (December 20, 2007), “Proposal: refactor the test suite according to synopsis”, http://groups.google.com/group/perl.perl6.compiler/msg/ed748490f030da2f
8. P. Michaud (June 16, 2008), "Rakudo test suite progress", http://use.perl.org/~pmichaud/journal/36695
9. A. Kreher, "Auzon's Blog: gsoc2008", http://auzon.blogspot.com/search/label/gsoc2008
10. "I HAZ A PARROT" (January 3, 2008), http://lolcode.com/news/i-haz-a-parrot
11. K. Stol (March 9, 2008), "PCT Tutorial Episode 1: Introduction", http://www.parrotblog.org/2008/03/targeting-parrot-vm.html
12. J. Worthington, "Chipping away at perl6", http://www.jnthn.net/cgi-bin/blog_read.pl?id=589
13. S. Cozens (January 3, 2008), "Parrot is really quite wonderful", http://blog.simon-cozens.org/post/view/132
14. “Parrot Abstract Syntax Tree (PDD 26)”, http://www.parrotcode.org/docs/pdd/pdd26_ast.html
15. R. Dice (May 16, 2008), "TPF receives large donation in support of Perl 6 development", http://news.perlfoundation.org/2008/05/tpf_receives_large_donation_in.html
16. J. Worthington (August 5, 2008), "Multiple Dispatch Design Work", http://use.perl.org/~JonathanWorthington/journal/37101
17. “Perl 6 Development Roadmap”, http://svn.perl.org/parrot/trunk/languages/perl6/ROADMAP