Tcl/GSoC 2010

Andreas Kupries ActiveState Software Inc. 409 Granville Vancouver, BC CA
andreask@ActiveState.com

ABSTRACT

As in previous years the Tcl Community took again part in Google's Summer Of Code[1], under the auspices of the Tcl Community Association[2].

1. OVERVIEW

Google's Summer Of Code[1] (short: GSoC) is a global program funded and operated by Google that offers student developers stipends to write code for various open source software projects. The Tcl Community participated again this year, for the third time in a row. As in previous years this participation was managed by the Tcl Community Association[2] (short: TCA) as the mentoring organization, the same organization which runs the US Tcl Conferences.

The main entrypoint to the program for the community itself can be found on the Tcler's Wiki [4].

2. PAST

Starting in 2007, we applied three times, and were accepted two times, with only our very first application not getting accepted by Google. This year was our fourth application and third participation.

Through negotiations by previous program administrators we usually got just shy of 10 slots for our projects[4], with our usual argument the fact that the Tcl Community Association[2] acts as an umbrella for smaller organizations with Tcl related projects. An example for this is the aMSN chat client[3]. The full statistics for the past years[5] are shown in table 1 below.

	2006	2007	2008	2009
Students	630	950	1125	1000
Organizations	102	>130	175	150
Average	6.18	<7.31	6.44	6.66
Tcl	-	-	9	9

Table 1: Statistics of past four years

Tcl '2010 Oakbrook Terrace/Chicago, IL, USA
.

3. PRESENT

Matthew Burke[8], our program administrator of the past years served in that capacity this year as well, with me as backup.

In the slot allocation game/roulette we got seven slots, a loss of two compared to previous years. We considered this to be quite higher than the average, because of Google originally dialing down the total number of slots to 950 in general. In the final statistics[5] we are only average however, as can be seen in table 2 below.

	2010
Students	1026
Organizations	150
Average	6.84
Tcl	7

Table 2: Final statistics for 2010

Our projects for this year are listed in table 3 [1] on the next page, with larger descriptions in the upcoming sections The overall timeline we followed is shown in table 4.

3.1 Base JIT compiler for Tcl

[10] by Guilherme Goncalves
Mentor: Donal Fellows

This project aimed to create a base JIT compiler for the Tcl core, translating bytecodes to (x86) machine code on demand in order to improve the language's performance.

After GSoC's ending, the project still requires several future improvements to be able to land in the Tcl core or be usable in the wild. The first two ideas on the list are: proper register allocation and a finer grained intermediate representation (IR). Currently, the encoding of a very simplified form of the `incr` command requires 51 x86 opcodes, with the current IR this is all done inside a single clause of a switch statement. The idea is to transform this mid-level IR to a lower level one that is closer to a form of a generic-RISC architecture, and then develop and run instruction selection on it. Currently, only toy programs can be tested given the ad-hoc, or inexistent, register allocator, so this is another priority. This second "future" idea is already being developed. Another idea involves the application of another kind of IR, the SSA. It has been used by some known projects: LLVM [18], gcc [19], Jalapeño (or Jikes RVM) [20] and is mentioned as a way to allow some optimizations[21] to take

[1]The same table can also be found at[9]

Student	Project	Mentor
Michael Aram	SCORM Compliant Run-Time Environment for OpenACS	Gustaf Neumann
Andrew Shadoura	OpenStreetMap package and editor in Tcl/Tk	Matthew Burke
Ewa Leszczynska	Themed Tk on Unix	Jeff Hobbs
Ozgur Dogan Ugurlu	Tcl state machine back-end module for XMLVM	Kevin Kenny
Eduardo Santos	OpenACS Abstraction Layer	Colin McCormack
Guilherme Goncalves	Base JIT compiler for Tcl	Donal Fellows
Paraskevi Nikolaidou	MSNP2P refactoring for aMSN	Youness Alaoui

Table 3: 2010 Projects, Students, and Mentors

	February 8	Program announced.
Organizations	March 8	Organization application window opens.
	March 12	Deadline for organization applications.
	March 13-17	Submission review.
	March 18	Publication of accepted organizations.
Students	March 18-29	Discussion of ideas between students and organizations.
	March 29	Student application wind opens.
	April 9	Deadline for student applications.
	April 10-21	Organizations rank and review student applications.
	April 21	Ranking/scoring deadline. Mentor sign-up deadline.
	April 26	Publication of accepted students.
Coding		Community bonding sudents to mentors.
	May 24	Coding period starts.
	July 12-16	Mid-term evaluations.
	August 9	Soft-end of coding. Scrub code, test, document.
	August 16	Hard end of coding period.
Post-Mortem	August 16-20	Final evaluations.
	August 23	Final results announced.
	August 30	Students can begin submitting the require code samples.
	October 23-24	Mentor Summit at Google.

Table 4: 2010 Timeline

place in an efficient manner. Other important improvement is allowing the compilation of non-leaf procedures, which requires a study between the interaction of the JIT compiler and the Tcl VM in order to properly manage the VM state.

3.2 MSNP2P refactoring for aMSN

[11] by Paraskevi Nikolaidou
Mentor: Youness Alaoui

Continuing the series of aMSN projects done in previous summers, this year the current MSNP2P implementation was rewritten to use `snit`[17]. This was done to overcome the problems with the current code base, which was complicated, with code scattered over different files and mixing several different functionalities in the same procedure.

As a bit of background, MSNP2P is the protocol used for P2P data transfers, taking care of webcam video, file transfers, etc.

This rewrite of the current code base will also help with implementing MSNP2Pv2, which is the version used in the latest builds of Windows Live Messenger and supports concurrent login from different locations. Implementing MSNP2Pv2 will in turn enable full support of protocol version 18 and audio/video calls.

3.3 OpenACS Abstraction Layer

[12] by Eduardo Santos
Mentor: Colin McCormack

The basic idea behind this project was to rewrite the server integration of the OpenACS' web framework to use Colin's Wub web server packages, extending OpenACS' portability.

3.4 OpenStreetMap package and editor

[13] by Andrew Shadoura
Mentor: Matthew Burke

A set of packages for the convenient handling OpenStreetMap data was created, both for its storage and rendering into a Tk canvas. On top of that a visual editor for OpenStreetMap data was made, using Tk for the interface and allowing the creation and editing of ways and tags.

Future work may include expansion of the editor's modular approach into a full plugin system.

3.5 SCORM Compliant Run-Time Environment for OpenACS

[14] by Michael Aram
Mentor: Gustaf Neumann

This was a follow-up project to last year's successful SCORM project, which focused on input, export, and presentation of SCORM packages. During the last project it turned out that implementing the runtime environment (RTE) is a relatively big task. As the RTE implemented only the most basic RTE-API functions it was therefore only capable of importing simple packages.

This year's project overcame these limitations, implementing a fully standard conformant RTE according to the min-

imal compliance level defined by ADL. In future subsequent projects reaching even higher compliance levels, which enable further SCORM features, could be a sensible goal. Also, incorporating newer versions of the SCORM or alternative packaging standards (e.g. SCORM 2004, IMS CC) should be considered.

3.6 Tcl state machine back-end module for XMLVM

[15] by Ozgur Dogan Ugurlu
Mentor: Kevin Kenny

Originally planned was the creation of a back-end for the XMLVM bytecode cross-compiler, enabling all languagues producing JVM or .Net bytecodes to be translated to and executed by Tcl.

This project mutated over the summer and became a "Tcl Bytecode Assembler" instead, extending the Tcl core with a command taking a textual description of Tcl bytecodes and creating the executable internal representation.

3.7 Themed Tk on Unix

[16] by Ewa Leszczynska
Mentor: Jeff Hobbs

This project first and foremost updated the Qt and Gtk bindings for Themed Tk to work with the latest versions of their respective GUI toolkits. Additionally some shortcomings and general code quality issues in the previous versions of those packages were addressed.

4. FUTURE

For a mentoring organization Google's Summer Of Code[1] is pretty much a year-round operation. Simply look back at the timeline (Table 4 on the previous page).

Next up in this cycle is starting the preparations for 2011, i.e., updating our application[6], restarting the collection of project ideas, and reaching out to prospective students and mentors in general. The last point is one of the more important things to do, not only for us as a mentoring organization, but for the Tcl community at large too, to make a general effort of spreading awareness of Tcl and its community as a viable (and fun) scripting language which doesn't has to hide.

APPENDIX

A. REFERENCES

[1] Google, GSoC.
http://socghop.appspot.com/

[2] Tcl Community Association.
http://www.tclcommunityassociation.org/

[3] aMSN.
http://www.amsn-project.net/,
http://wiki.tcl.tk/12783

[4] Google Summer Of Code on Tcl'ers Wiki.
http://wiki.tcl.tk/25801

[5] GSoC Statistics. http://code.google.com/p/google-summer-of-code/wiki/ProgramStatistics

[6] Tcl Org Application 2010.
http://wiki.tcl.tk/25802

[7] Andreas Kupries.
http://wiki.tcl.tk/26

[8] Matthew Burke.
http://www.seas.gwu.edu/~mmburke/

[9] GSoC 2010 Executed Projects.
http://wiki.tcl.tk/26114

[10] Base JIT Compiler, Guilherme Goncalves.
http://code.google.com/p/tcljit

[11] MSNP2P refactoring for aMSN, Paraskevi Nikolaidou.
http://www.amsn-project.net/

[12] OpenACS Abstraction Layer, Eduardo Santos.
http://code.google.com/p/portable-nsd/

[13] OSM/Tk, Andrew Shadoura.
http://bitbucket.org/andrew_shadoura/tcl-osm/

[14] SCORM, Michael Aram.
http://wiki.tcl.tk/26120

[15] Tcl ByteCode Assembler, Ozgur Dogan Ugurlu.
http://wiki.tcl.tk/26123

[16] Themed Tk on Unix, Ewa Leszczynska.
http://hefka.pl:1027/index,
http://github.com/aweelka/,
http://wiki.tcl.tk/26122

[17] Snit's Not IncrTcl. Will Duquette.
http://wiki.tcl.tk/3963

[18] LLVM: A Compilation Framework for Lifelong Program Analysis & Transformation, Chris Lattner and Vikram Adve. CGO '04: Proceedings of the international symposium on Code generation and optimization, 2004, ISBN 0-7695-2102-9, page 75, IEEE Computer Society

[19] GNU Compiler Collection Internals – SSA, Free Software Foundation.
http://gcc.gnu.org/onlinedocs/gccint/SSA.html

[20] The Jalapeño virtual machine, B. Alpern and C. R. Attanasio and others. IBM Syst. J. vol 39.1, 2000, ISSN 0018-8670, pages 211 – 238, IBM Corp.

[21] Efficiently Computing Static Single Assignment Form and the Control Dependence Graph, Ron Cytron and Jeanne Ferrante and Barry K. Rosen and Mark N. Wegman. ACM Transactions on Programming Languages and Systems vol 13.4, 1991, pages 451 – 490.