Haskell Summer of Code

February 3, 2010

Summer of Code

As part of Google's **Summer of code** program, Google sponsors 5–10 **projects for Haskell**.

The Haskell Summer of Codes have often produced excellent results, but how excellent is excellent? Are there any features or commonalities between successful projects or unsuccessful ones?

(This questions are particularly important as SoC 2010 isn't too far away; yet we don't have even a general sense of where we are.)

Example retrospective: Debian

An energetic blogger & Debian developer has produced **a three part** series on the many Debian-related Summer of Code projects.

The results are interesting: some projects were a failure and the relevant student drifted away and had little to do with Debian again; and some were great successes. I don't discern any particular lessons there, except perhaps one against hubris or filling unclear needs. Let's see whether that holds true of Haskell.

Haskell retrospective

Haskell wasn't part of the first Summer of Code in 2005, but it was accepted for 2006. We start there.

2006

The 2006 **homepage** lists the following projects:

- "Fast Mutable Collection Types for Haskell"

  Caio Marcelo de Oliveira Filho, mentored by Audrey Tang

  unsuccessful. This ultimately resulted in the **HsJudy** library ('fast mutable collection' here meaning 'array'; see the **application**). HsJudy was apparently used in Pugs at one time, but no more.

- "Port Haddock to use GHC"

David Waern, mentored by Simon Marlow

Successful. Haddock has used the GHC API ever since.[1]

- "A model for client-side scripts with HSP"

  Joel Björnson, mentored by Niklas Broberg

  Successful? Was initially unsuccessful, but seems to've been picked up again.

- "GHCi based debugger for Haskell"

  José Iborra López, mentored by David Himmelstrup

  Successful. The GHCi debugger was accepted into GHC HEAD, and by now is in production use,

- "HaskellNet"

  Jun Mukai, mentored by Shae Erisson

  Unsuccessful. HaskellNet is dead, and nothing of it has propagated elsewhere. (I'm not entirely sure what happened with the HaskellNet code - I know of **two repos**, but that's about it.) Shae tells me that this poor uptake is probably due to a lack of advertising, and not any actual defect in the HaskellNet code.

- "Language.C - a C parser written in Haskell"

  Marc van Woerkom, mentored by Manuel Chakravarty

  Unsuccessful. According to **Don Stewart's outline** of the 2006 SoC, this project was not completed.

- "Implement a better type checker for Yhc"

  Leon P Smith, mentored by Malcolm Wallace

  Unsuccessful. See Language.C

- "Thin out cabal-get and integrate in GHC"

  Paolo Martini, mentored by Isaac Jones

  Successful. Code is in Cabal, which we all know and love.

- "Unicode ByteString, Data.Rope, Parsec for generic strings"

  Spencer Janssen, mentored by Don Stewart

  Successful. (Again, per Don.)

[1] I can hear the wankers in the peanut gallery - "Yeah, and it's been buggy ever since!" Hush you. (You can read Waern's reply to this **here**.)

4 successful; 2 unsuccessful; and 2 failures.

2007

The **2007 homepage**

- "Darcs conflict handling"

  Jason Dagit, mentored by David Roundy

  Successful. The work was successful in almost completely getting rid of the exponential conflict bug, and has been in the regular releases of Darcs 2.x for some time now.

- "Automated building of packages and generation of Haddock documentation"

  Sascha Böhme, mentored by Ross Paterson

  Successful. The auto build and doc generation are long-standing and very useful parts of Hackage.

- "Rewrite the typechecker for YHC and nhc98"

  Mathieu Boespflug, mentored by Malcolm Wallace

  Successful. According to the TMR writeup, the type-checker code has made it into (at least) YHC.

- "Cabal Configurations"

  Thomas Schilling, mentored by Michael Isaac Jones

  Successful. Cabal configurations have been around for a while and are very useful for enabling/disabling things

- Update the Hat tracer

  Kenn Knowles, mentored by Malcolm Wallace

  Unsuccessful. The update apparently happened, since the **Hat homepage** says "Version 2.06 released 2nd Oct 2008", but it is **described** as unmaintained, and I can't seem to find any examples of people actually using Hat.

- Generalizing Parsec to ParsecT and arbitrary input (ByteStrings)

  Paolo Martini, mentored by Philippa Jane Cowderoy

  Successful?. The performance is still so terrible that few people use it.

- Shared Libraries for GHC

  Clemens Fruhwirth, mentored by Simon Marlow

  Successful?. The situation is unclear to me, but I know that for some period dynamic linking worked for some platforms. However, it's 2010 and I still have static linking, although GHC 6.12 apparently gets dynamic linking; so I'm going to chalk this one up as a mixed success.

- "Libcurl"

  Mieczysław Bąk, mentored by Bryan O'Sullivan

  Successful. Libcurl does exist and has been used, and is maintained. I understand it has been used, even if I personally don't know of any examples.

- "Extending GuiHaskell: An IDE for Haskell Hackers"

  Asumu Takikawa, mentored by Neil David Mitchell

  Unsuccessful. GuiHaskell does not exist in any usable form. (The homepage summarizes the situation thusly: **"Warning: This project is fragile, unfinished, and I do not recommend that anyone tries using it."**)

6 successes; 1 unsuccessful; 1 failure.

See also

- **The Monad.Reader's issue 9** covers SoC projects
- **http://www.serpentine.com/blog/2007/04/12/haskellorg-and-googles-summer-of-code/**

2008

The **2008 homepage** isn't so kind enough as to list all the projects as before, but it does tell us that only 7 projects were accepted by Google.

So we can work from the **code.google.com** page which lists 6:

- "C99 Parser/Pretty-Printer"

  by Benedikt Huber, mentored by Iavor Diatchki

  Successful. The first try failed, but the second won through, and now people are doing things like **parsing the Linux kernel** with it.

- "GMap - Fast composable maps"

  by Jamie Brandon

Successful. GMap is on **Hackage**, and I believe I've seen it used.

- "Haskell API Search"

  Neil Mitchell

  Successful. The improved performance and search capability have made it into Hoogle releases.

- "Cabal 'make-like' dependency framework"

  Andrea Vezzosi

  Unsuccessful? (**His code wound up** becoming **hbuild**, which is not on Hackage or apparently used by anyone.)

- "GHC plugins"

  Maximilian Conroy Bolingbroke

  Successful? (As of **January 2010**, the patch adding plugins functionality has yet to be accepted & applied.)

- "Data parallel physics engine"

  Roman Cheplyaka

  Unsuccessful? It seems to be finished but no use made of the actual engine (that I can see mentioned on the **engine's blog**).

- "GHC API".

  Thomas Schilling

  Unsuccessful. Schilling's fixes went in, but they were in general minor changes (like adding the GHC monad) or bug-fixes; the GHC API remains a mess.

4 successful, 3 unsuccessful.

See also

- The Monad.Reader's **Issue 12**

2009

5 projects were **accepted** this year; Darcs tried to apply in its own right was rejected.

In general, these looked good. Most of them will be widely useful — especially the Darcs and Haddock SoCs — or address longstanding complaints (many criticisms

of laziness revolve around how unpredictable it makes memory consumption). The only one that bothers me is the EclipseFP project. I'm not sure Eclipse is common enough among Haskellers or potential Haskellers to warrant the effort, but at least the project is focused on improving an existing plugin than writing one *ab initio*. The 5 were:

- "Optimising Darcs for medium to large repositories",

  by Petr Ročkai; Eric Kow, mentor

  Unknown. **hashed-storage** exists and is used in Darcs, but from watching the bugtracker traffic, it's unclear whether Darcs saw a net gain from it.

- "haskell-src-exts -> haskell-src"

  by Niklas Broberg; Neil Mitchell, mentor

  Unknown. Niklas added a large number of **patches** but it's unclear to mean what practical benefit it adds. But it *does* handle comments now.

  "This project aims to improve on my haskell-src-exts library for working with Haskell source code. The goal of the project is to have haskell-src-exts supercede the haskell-src library. The main thing involved is to add a mechanism that allows the user to conditionally choose what extensions the parser should be aware of. Further, I would extend haskell-src(-exts) with functionality to handle comments and exact layout, to allow for things like automatic code generation and refactoring.""

- "Haddock improvements"

  by Isaac Dupree; David Waern, mentor

  Successful?. Dupree's **patches** have been applied to head and apparently make cross-package links **usually work**.

- "Improving space profiling experience"

  by Gergely Patai; Johan Tibell, mentor

  Successful. **hp2any** seems quite alive and usable.

- "Extend EclipseFP functionality for Haskell"

  by Thomas ten Cate; Thomas Schilling, mentor

  Unsuccessful. See **Cate's summing-up**.

2 unknown, 2 successful, 1 failure.

Haskell roundup

So, what lessons can we learn from the 3 SoCs? It seems to me like there are roughly 3 groups explanations of unsuccess or failure fall into. They are:

1) *Hubris*. GuiHaskell is probably a good example; it is essentially a bare-bones IDE, from its description. It is expecting a bit much of a single student in a single summer to write *that*!

2) *Unclear use*. HsJudy is my example here. There are already so many arrays and array types in Haskell! What does HsJudy bring to the table that justifies a FFI dependency? Who's going to use it? Pugs initially did apparently, but perhaps that's just because it was there - when I looked at Pugs/HsJudy in 2007, certainly Pugs had no need of it. (The data parallel physics engine is probably another good example. Is it just a benchmark for the GHC developers? Is it intended for actual games? If the former, why is it a SoC project, and if the latter, isn't that a little hubristic?)

3) *Lack of propaganda*. One of the reasons Don Stewart's bytestring library is so great is his relentless evangelizing, which convinces people to actually take the effort to learn and use Bytestrings; eventually by network effects, the whole Haskell community is affected & improved[2]. Some of these SoC projects suffer from a distinct lack of community buy-in - who used HaskellNet? Who used Hat when it was updated? Indifference can be fatal, and can defeat the point of a project. What good is a library that no one uses? These aren't academic research projects which accomplish their task just by existing, after all. They're supposed to be useful to real Haskellers.

Don Stewart's view

Don Stewart writes in reply to the foregoing:

> We explicitly pushed harder in 2008 to clarify and simplify the goals of the projects, ensure adequate *prior Haskell experience* and to focus on libraries and tools that directly benefit the communtity. [*sic*]
> And our success rate was much higher.
> So: look for things that benefit the largest number of Haskell developers and users, and from students with proven Haskell development experience. You can't learn Haskell from zero on the job, during SoC.

[2] Many good and worthwhile projects suffer this fate because of their academic origins. There's no reward for someone who creates a great technique or library and gets the wider community to adopt it as standard. As far as the Haskell community is concerned, one Don Stewart is worth more than a dozen Oleg Kiselyovs; Oleg's work is mindblowingly awesome in quantity and quality, everyone acknowledges, but how often does anyone actually use any of it?