Where do we go from here?

Darcs in 2010

Haskell libraries increasingly using git(hub)

GHC finally moving away

At least some of our remaining userbase views darcs as “legacy”

So why don’t we just give up?

If Darcs didn't exist, it would be worth writing

First-class patches are a fundamental innovation

UI simplicity: a natural consequence

Deliver on the promise: Richer patch types

We need to play catch-up

github ➔ darcsden/patch-tag?

multi-head repos

conflict handling

repo identifiers

rebase

speed

How do we grow?

We do have a small loyal userbase…

…but a more widespread bad reputation…

…but still a reservoir of affection

To the majority we are unknown or irrelevant

<u>Need good interoperability</u>

What should we do

Fix our problems

Reengineer our codebase

Innovate!

Need a healthy mix of all three

darcsden/patch-tag

Momentum for a merger/transfer to darcsden?

Darcsden is not author's primary focus

Should we help improve it as part of our "core" work?

Help with hosting?

Multi-head repos

Often touted as a major weakness

Sharing build products is important

We need to get the design right

Can we do better than just allowing multiple branches in a repo?

A branch is a **set** of patches

Conflict handling

Don’t underestimate how much this sucks

No patch names

Unpredictable ordering of hunks

No markup at all unless it’s hunk-hunk

Conflict fights

Can’t associate the resolution with the merge

Darcs 1 and darcs 2 patches broken

What should be in darcs 3 patches?

Fix setpref
Hunk move
Character patches
Indentation patches
UUID based file tracking
(fix add-add conflicts)
Refer to entire directories
Refer to groups of files
Improve replace-replace conflicts

Binary diffs
(simple byte ranges)
Permission handling
(perhaps just execute bits)
Subrepos
(falls out from UUIDs?)
Plugins
Patch “lenses”
Semantic patches

Derive commute rules
Verify correctness
How do we manage upgrades?
UI challenges
Interoperability is important for darcs ⇔ darcs!

Patches and “changes”

Patches are an approximation of logical changes

Bridge the gap without compromising the UI

Superseding published patches

Conflict resolution between X+Y is a null change

Should be activated implicitly when both X and Y are live

But what if there’s a disagreement?

Versioning patches??

What have I been doing?

Rebase

Going slower than I hoped

Somewhat hampered by the messy codebase

And a bit by a baby ☺

Will it make 2.8?

Less incentive now GHC has gone

Rebase

Basic UI is there

Big items left:

Managing conflicts better

Dealing with explicit dependencies

Review it

And lots of small UI tweaks

Refactoring Darcs.Patch

Now more modular

Specific patch types in their own namespaces

Primitive and conflict layers decoupled

Still a few direct legacy imports

Still a lot to do

Incremental progress driven by other work

Conflict naming

Commute conflicts back until they disappear

Attach names to primitive patches

Commute forward again

Adding them temporarily leads to messy types

Performance / how far back to commute

Repo format change?

“Graphictors”

... because every proper darcs hacker needs their own conflict representation

Remove remaining performance blowups

Fix “conflict fights”

Commute out depended-on patches

Still just on paper

What's a conflict fight?