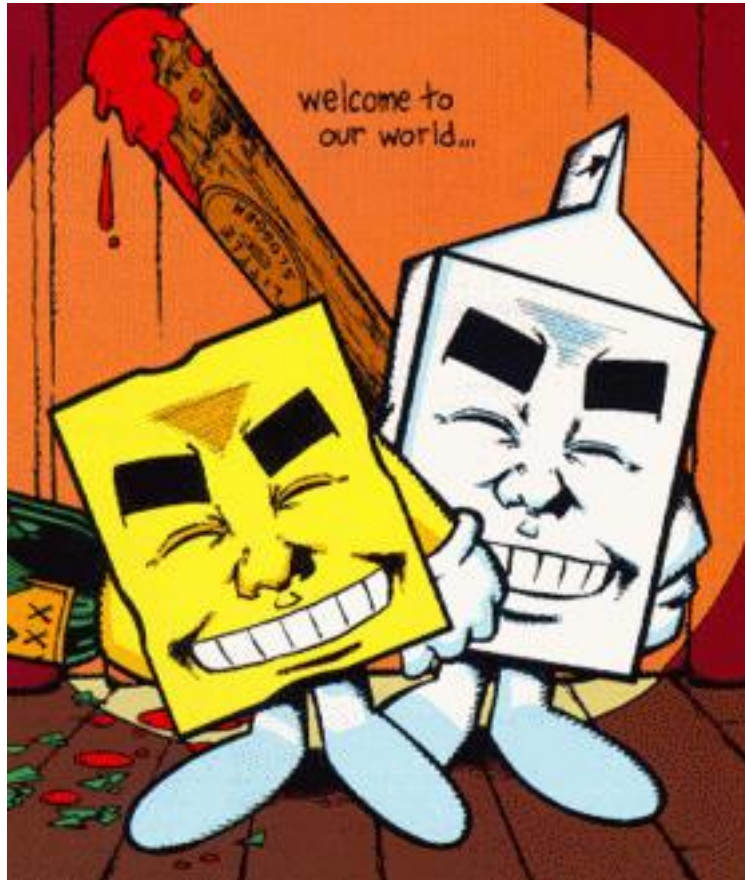# Build Systems, improving development and delivery

Arthur Bergman, Richard Clamp
Fotango

Introductions

What we're going to tell you

- Why we went for a build system

- What it does

- How it does it

- Dancing girls

Before the build system

- System wide perl, apache, perl modules

- System group in charge of this

- Snapshot of CPAN

- Every single time, someone would forget to compile a format option into image magick

Utter chaos

- 1 day to debug
- 3 days to rebuild
- 5 days to sort out what version was where

Like juggling monkeys

That pesky intertwined code

Snapshot of CPAN

- Broken
- Outdated
- Very very inflexible

Image Magick

- DO NOT USE IT
- ARCANE SOURCERY

and tricky to know what changing things will do

Everything must be self contained

Making it easier to manage and deploy

So let's build a project

● It'll be a perl project

```
% cat MadeUp/Example/Build.PL

use strict;
use Module::Build;
Module::Build->new(
    module_name => "DBIx::Potato",
    license     => "perl",
    requires    => {
        "DBD::mysql" => 0,
    },
)->create_build_script;
```

Back up

Have to build a new perl

- otherwise it's not self contained

- we don't trust /usr/bin/perl
    - could be a vendor perl
    - or really old
    - or corrupted
    - or eaten by dogs

Need to satisfy Perl's dependencies

- libgdm

And then we still need DBD::mysql

- DBI, ugly but useful

- mysql, retarded but useful

Needs libmysqlclient.so

- It's XS, it has to link against something
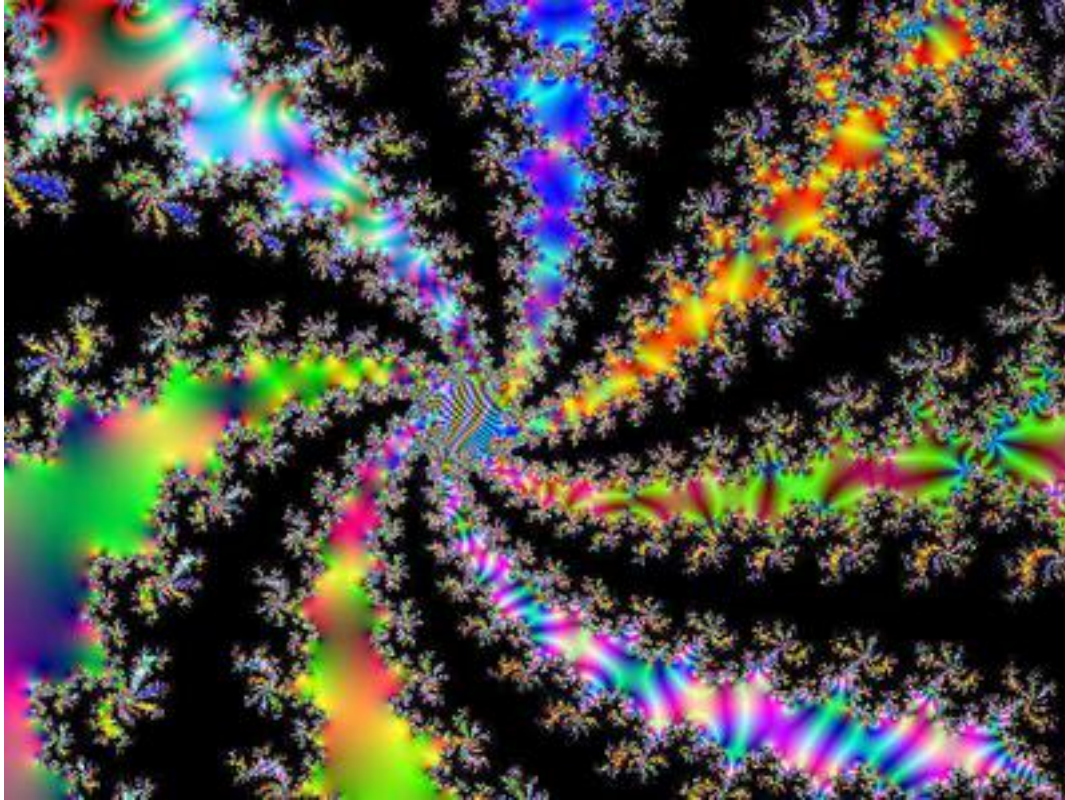
- And it can't be the system lib

Which means we have to build all of mysql

- Which wants zlib too

and then you're off again

- Have to build DBI
  - And anything it decided it needed
- Then you can compile/test DBD::mysql

This can go on for a while

We're currently managing a ton of code

- 187 external packages

- 11,043,011 lines of foreign code

- 152 CPAN distributions

- In CVS

```
Vendor
 |- CPAN
 |- Apache
 |- other
 `- packages
```

Gotchas

- This is by no means an exhaustive list

it links a system library

- ldd is your friend

autoconf

autoconf

- So bad we named it twice

the tests use 'perl' rather than $^X

- Apache::Test
  - finds the apache on the host system, even when you tell it which one   to use

Image Magick/PerlMagick

libpng - no configure

AIX

- libjpeg.h
  - Polluting .h file
  - xlC doesn't let you do that

AIX

- ffmpeg
  - expects gmake
  - expects stuff from glibc

AIX

- ImageMagick

- linker

AIX

- It's just one big gotcha if all you've been using before is x86 Linux, or *BSD

- /opt/freeware/bin may help

- But it's C compiler, linker, and properness will trip some things up

Every other platform

- Darwin
  - OpenSSL doesn't like building here
  - Old things think it's NeXTier than it really is
  - YAFLinker
- Win32
  - Even with cygwin it's a pain
  - Works though
    - We smoke VB and Delphi apps with it

Makefiles

- People scatter -j4 around
  - But with broken makefiles
  - And we actually have dual-proc boxes for building on
- Or they assume gmake and gcc

reporting tools

- logs to a wiki

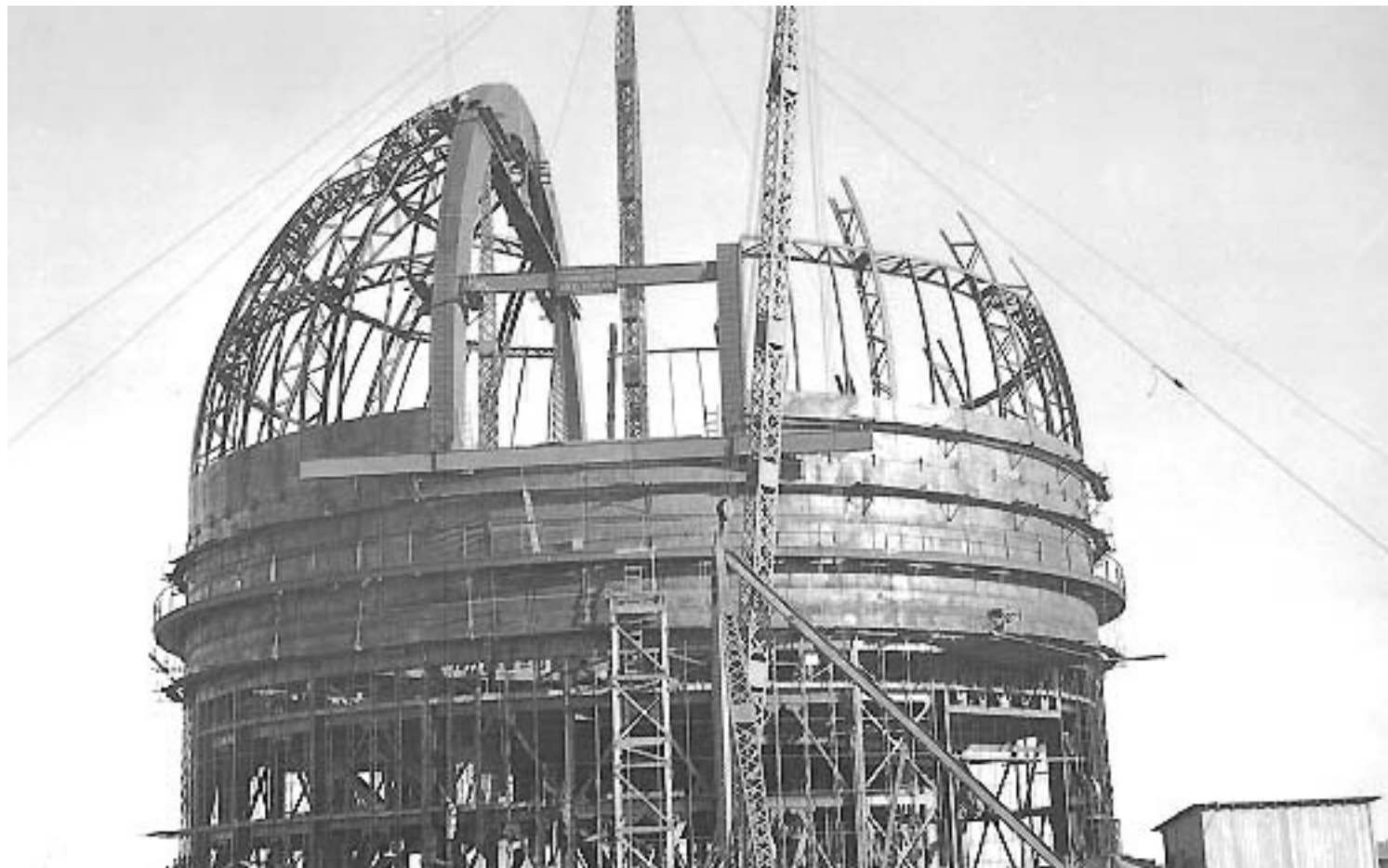- cgi parses and gives an overview

- daily smoke emails

Modules we used

- Module::Depends::Intrusive

- Parse::CPAN::Packages

- IPC::Run
  - best of a bad lot

- WWW::Mechanise

- Parse::Rcs

Modules we didn't use

- CPAN.pm

- CPANPLUS

- Module::Install

- ExtUtils::AutoInstall

- We even use LWP to grab things from search.cpan.org

use this picture

Result

- The developers love the build system

- The developers hate the build system

- Everything is now managed

- Everything is now in CVS

- Well, not glibc and the linux kernel, but we are working on it!

- Life is simply more orange

Future

- More modules to CPAN

- Opensource larger parts of it

- Automatic testing of new versions of modules, perl and libraries

- Upstream patch mangement

- Robots

Robots!