LibreOffice Development Infrastructure

State of The Union

LibreOffice Conference, Aarhus 2015

What's 'development infrastructure'

Gerrit, Git, Jenkins, Tinderboxes, Callgrind Perf, UBSAN, Crashtest, Bibisect, Lcov, Cppcheck, ...

Machines Park

TDF Owned:

2 Dual-Socket High End server, running Windows, hosted at Manitu, Germany

1 MacPro 12 cores, Hosted by SQData in Dallas, TX, running MacOSX

1 4-cores Server, volunteer donated, hosted by Suse, running Linux

2 Mac Mini, hosted by various Volunteers

2 Tower PC running Windows, hosted by various Volunteers

Machines Park

TDF Rented

3 64 core, 256GB AMD servers hosted at manitu

1 c4.4xlarge EC2 instance

Few others to insure core infrastructure, like dns, emails, mailing-list, etc..

Machines Park

Provided by Volunteers and Sponsors

1 MacPro 8 core, hosted in Dallas, TX

5 Mac Mini, hosted in Dallas, TX

1 Dual-Socket Server, Hosted and provided by ByteMarks, running Windows

1 Server, running Linux, Hosted and provided by ByteMarks, running Linux

1 Mid-range Server, running Linux, Hosted and Provided by Aquinetic

Machine Park

In the Pipeline, for TDF this year...

2 High-End Dual-Socket Intel servers.

1 Mac Mini, Donated by Volunteer.

Gerrit in Numbers

56 git repos managed by Gerrit (30 GB of storage)

1025 Registered users

148 Committers

12 Months running:

50830 Commits across all managed git repos

6643 Reviewed Patchsets

Gerrit Future

Mostly 'just work'

New features tend to cause performance problem, but overall nothing deadly for us.

Upstream is competent and tries hard to not break things, and is good at addressing breakage when they happen.

Gerrit is a rare example proving that Java is not inherently bad. Competent engineers can make it works.

Gerrit Security

While on the topic of Gerrit.

There are about ~50 registered users' ssh-keys

that are still 1024 bits RSA/DSA...

Please check your keys, and upgrade to at least 2048 RSA.

Jenkins in numbers

14 jobs descriptions

300 job runs a day in average

17 slaves-boxes managed

Performances

Gerrit Build Time

	Linux	MacOSX	Windows
min:	0:05	0:03	0:27
max:	1:31	1:56	2:26
med:	0:21	0:56	0:42
avg:	0:27	0:47	0:54

caveat: these are the build time once scheduled...

There is a finite number of slavebots, so it happens, especially during European work day, that things get queued-up

Jenkins Future

Jenkins is a nice toy, fairly easy to get up and running, but it is a toy.

Completely unreliable, does not scale, wasteful, buggy, upgrade is akin to playing Russian roulette.

Jenkins is the exact opposite of Gerrit and why serious Engineers despise Java.

Jenkins Future ?

Critical breakage stay ignored for years, resource utilization is abysmal (a typical Windows _client_, the thing whose only job is to run script on behalf of the server, is running in the 2GB of ram in less than a day of operation.

Yeah with java you don't have memory management problem.. you just leak GBs of it and get to call that a feature.

Jenkins is susceptible to random fits.. it will hang, start sucking full cpus for extended period of time, with no log, no explanations, no warning

Running a Jenkins is like Running Win98... you'd better just reboot the thing every other day.. just in case.

Jenkins Future (or lack thereof)

Goal: get rid of it.

Candidate: Maybe Mozilla Buildbot, or possibly writing our own. At this point 'anything but' is the target.

Desired feature: lean, fault tolerant, resilient, asynchronous, strong separation between ui and core.

NB: web admin is bug not a feature, especially when, like Jenkins, after a reboot – which you have to do prophylactically regularly – the things start scheduling new jobs 5, 10, 15 minutes before it displays any web pages, let alone let you access the admin section of the website.

Performance Testing using Callgrind

perf.libreoffice.org

We run regularly (that is typically 5-7 times a days)

a job that run callgrind unit-tests and run libreoffice under callgrind for a set of out-of-tree tests.

The results are collected in the Postgres database

which is used to feed the website above.

Demo live perf.libreoffice.org

perfdbmgr

Manages a posgress database containing the callgrind performance results

Organized around the notion of 'suite' of 'test' on 'machine'.

Each execution of a test of a suite on a given machine is stored as a 'run'

perfdbmgr also generates html and json to power

the site perf.libreoffice.org

see git://gerrit.libroffice.org/perfdbmgr

for the code underlying this.

Crashtesting

We have a huge dedicated VM that runs, pretty much continuously. It tries to load and save documents with libreoffice in all kind of formats. There is a collection of 70+ thousands of documents, more like a torture tests collection. Since most of these originate from bug reports there are a lot of pathological cases, designed to try to break the product.

For the past few months that crashtest job, consistently resulted in ~0 crash.

Bibisect

Bibisect are divided by 'epoch' and, of course, platforms

A bibisect epoch is the commits that occurs on master following the tagging of a release branch, until the tagging of the next following release branch, and possibly including the commit that occurs on that release branch.

For example the epoch 5.0 started at the commit that represent the branch point of the 4.4 release branch. which marks the beginning of the work on master toward the 5.0 release.

The 5.1 epoch started at the branch-point of the 5.0 release branch

Bibisect

Starting with the 5.1 epoch both Windows and MacOSX bibisects are built continuously under Jenkins.

On a semi-regular basis, but still a manual process, the bibisect repo on the slave box is being compressed and then uploaded to gerrit.

repos: bibisect-<macosx-64|win32>-<epoch>.git

Bibisect

Bibisect commit messages follow the following form:

```
    source sha:<sha of the source commit used to build>

    source sha:<sha>
[   source sha:<sha>
 ...]
```

List of the source sha included in this build that were not included in any previous build.

Bibisect

as a consequence,

git log --grep <source_sha>

will find the bibisect commit that first contain the given source sha.

This property of the git bibisect messages should allow for easier scripting of bibisect related task.. including the possibility, if epochs are extended to cover the associated release branch, to script the automation of bibisection across multiple epochs.

Questions ?

If you want to help, with any of it…

If you have a great idea you want to implement…

nthiebaud@gmail.com

shmget on Irc #libreoffce-dev or #tdf-infra