Technical Articles

Lars Hvam
November 5, 2020 4 minute read

CTS is beautiful

Every now and then I've been asked if abapGit can be used to deploy software to production, while it might be technically possible I heavily discourage doing it.

abapGit and CTS

SAP Change and Transport System(CTS) is used in ABAP systems to deploy software between systems, I guess most ABAP developers are familiar with transactions like SE09 and STMS

abapGit is a git client for ABAP written in ABAP, it is open source, and developed by the community on GitHub. It has been around for ~5 years is in active development, and works with versions 702 and up.

Introduction

abapGit is meant for helping software development, not software deployment, the requirements for these two areas are very distinct:

Software Development

- Agility
- Speed
- Flexibility

Software Delivery

- Reliable
- Stable
- Auditable

In software development, one developer might choose to switch from emacs to vim, this should not impact software delivery. A lot has happened in version control software, CVS has moved to SVN, SVN has moved to git, and git will move to something else sometime.

However, in the normal world(outside ABAP), the software delivery method has typically stayed the same with distributing executable/compiled files.

- Maven is used for Java development
- NPM is used for Javascript
- And Crates for Rust

A lot of things break during software development, but it is not good to have errors during software delivery, as it can possibly impact production systems and users.

Parallel Development

Having ABAP code in git does not enable parallel development.

ABAP systems have one active version of a program, and abapGit works on package level, ie. it's possible to have one branch checked out at a time. Developer1 can work on feature1, but developer2 cannot work on feature2 if it's the same package, in the same system.

Some features might be added in abapGit to have one branch checked out and commit to a different branch, but it will not be very transparent.

To have parallel development, multiple ABAP systems/containers are required, as suggested by Ethan Jewett back in 2015.

Branching Strategy

In a typical 3 tier setup, DEV -> TST -> PRD, I don't have an overview of how to do the branching properly. Assuming a setup with weekly fixed releases is quite easy, it will work with trunk based development.

But an agile setup where every feature can move through git and the environments gives more complexity and a lot of CI runs. I assume some manual testing will happen in TST, so some changes might be 1 hour on the way to production, and some might take months.

It feels like it would end up with the same as STMS, a list of branches for each system that has not been merged to that system/branch yet, with similar pitfalls as CTS.

Each traditional CTS transport can be seen as a "mini" branch for the objects that it contains, which is automatically rebased when imported to the target system. It would also be possible to put every object in its own package to simulate this behavior in git, but it would be a lot of work and not help developers.

Performance

abapGit works on package level, CTS on the objects that is part of the transport. If doing software delivery with abapGit I see the following three possibilities:

1. Make assumptions regarding the target system state, look at the git history to find the limited list of objects to import
2. Import all objects in the package every time
3. Compare all objects in git vs the system and import everything that does not match

Making assumptions regarding ABAP system state is not something I'd recommend, everything can happen in ABAP 🙂 It would also go against the reliability requirement for software deployment listed in the beginning.

abapGit is slower than CTS, and say the package contains 10k objects, and aiming to have an "agile" setup with changes constantly moved to production, I'm afraid this would end up in a deadlock situation where features cannot be imported because the previous imports are still running.

CTS on the other hand, works on object level, so if a developer changes a single report, only that report is part of the transport, making it the fastest import setup possible.

Rollback

With git it's easier to do rollback, yes, especially during development on a feature branch, as it would typically not have any consequences.

Rolling back changes to production is more difficult, assuming it's not the latest commit. Say, three unrelated changes, feature1 + feature2 + feature3 have been deployed successfully, and after a while a logical error is found in feature1, with git the rollback would cascade to feature2 and feature3, assuming it's in the same package.
And before doing the rollback it must be manually analyzed if all 3 features can be rolled back, in case of DDIC changes.

Many customer custom developments are decoupled, consisting of small individual programs in a big package. With CTS it is possible to create a new transport rolling back feature1, not having to consider feature2 and feature3, there are also partner solutions that can do this automatically.

Stability

During my last 15 years of using CTS for ABAP deployment, I don't recall having experienced a bug in CTS. Having bugs in the software deployment software, possibly causing requirements to upgrade to the latest S/4 HANA would be bad, software deployment needs stability.

Software development, on the other hand, needs agility, the possibility for new tools and processes. Hence abapGit can be upgraded without impact on the landscape, across a wide range of versions(702+), assuming it's used for software development and not deployment.

Keeping the core clean

I see CTS as part of the core, it's a prerequisite for running an ABAP installation. CTS is a good example of keeping the core clean and stable.

abapGit on the other hand is an option, with rapid development and changes, and should not be part of the core.

CTS could provide an API, so that new innovations can be built outside the core, utilizing the API. Integrations and APIs are also very important in the lower levels of the technology stack.

I'm not a git or CTS expert, so comments and corrections are welcome 🙂

Alert Moderator

Assigned tags

ABAP Development | DevOps | Open Source | abapgit |

Related Blog Posts

Parsing JSON in ABAP
By **Kerem Koseoglu** , Aug 03, 2017

How to use ATC and Code Inspector with CTS/QGM/ChaRM
By **Mahadevan Venkata** , Apr 10, 2015

New Configuration Editor in 7.31
By **Former Member** , Oct 28, 2012

Related Questions

Change cts..
By **Former Member** , Mar 04, 2009

cts
By **Former Member** , Feb 24, 2007

What is CTS ?
By **Former Member** , Mar 16, 2006

Be the first to leave a comment

Add Comment

Find us on

Privacy	Terms of Use
Legal Disclosure	Copyright
Trademark	Cookie Preferences
Newsletter	Support