Trim your toolkit with this one weird trick!

DITA-OT Day, 2019

Robert D. Anderson, IBM

@robander

Why me?

- I do lots of DITA stuff
- Also lots of DITA-OT stuff
- For quite a long while now

What I expect you are expecting

- Historical travels though DITA-OT
- How we got where we are
- What we’ve lost (and gained) along the way
- How best to keep up?

Agenda

- History of DITA-OT
- Early focus and major changes
- Our troubles with technical debt
- *How much can a tool like ours change?*
- Ways to manage *our* changes with *your* extensions

Who are you?

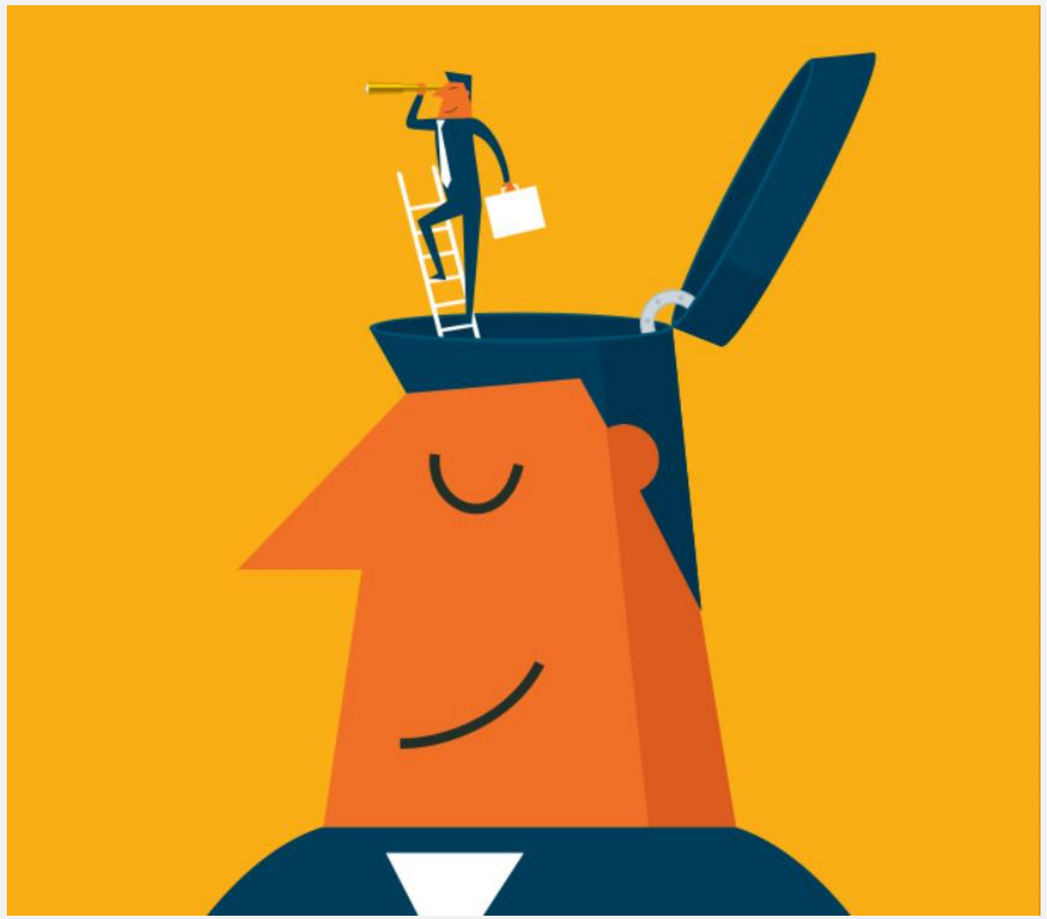
- How long with DITA-OT?
  - New to the tool?
  - Years and years?
  - Customizing / configuring / registering plug-ins?
- Currently running:
  - 1.8.5 or earlier?
  - 2.x?
  - 3.x?
- Experience with upgrades?

Some of this history is approximate

- Because which 1.x added what is not so important at this point?
- Aiming for quick overview of progression

DITA-OT 1.0: What happened there again?

- Beta implementation => Real Official Tool
- Mixed set of developers
- Acceptance criteria:
  - *It is possible to make something build*
  - *That “something” is mostly HTML*



“Ease of use”

- Just attach this wire here
- And that one goes over there
- AND WHATEVER YOU DO
  DON’T CUT THE RED WIRE

Customization expected!

- *But….*
  - No defined mechanism except `args.xsl` parameter
  - No upgrade path
  - Anything and everything is available!
  - *…which means nothing can change*

1.1, 1.2: Early focus on usability

- *As in, make it possible to use…*
  - Maybe, include required tools?
  - Maybe, give a better command line?
  - And a bit later … `startcmd`

- Toolkit lived in multiple worlds:
  - *Walk-up-and-use for any author*
  - *Embedded library in a product*
  - *Collection of resources to pick apart*

Eventually: back to one package

- Multiple packages = too confusing
- Few MB savings = less urgent
- “Add your own dependencies” = additional required testing and expectations of support
- Everyone got the everything-package anyway

PDF2 and configuration directories

- “Baby steps”
- Extension mechanism developed independently
- Only applies to PDF

Here come plugins

- Plugin directory structure came from Deborah Pickett

- Ant, XSL, libraries did not move
  - Largely my fault

Eventually: even the base code as plugins

- Base processing code: `org.dita.base`
- Base XHTML XSL: `org.dita.xhtml`
- Make plugins removable, keep root directories clear

`plugin:` referencing syntax

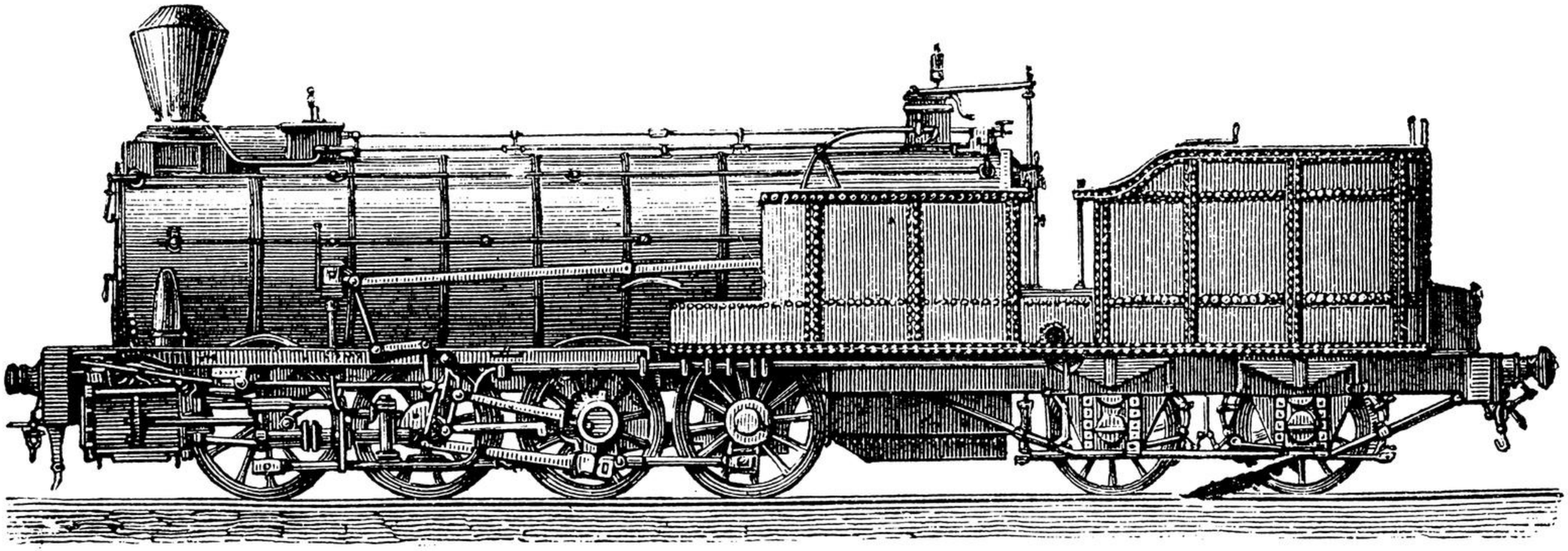
- Gives core code freedom to move
- Gives plugins freedom to move
- Finally removes dependencies on knowing / depending on DITA-OT file structure

Have to pay down technical debt

Remove old or poorly designed function

- Cut down on command lines
- Simple, documented interfaces
- Reorganize for easy maintenance
- Plan before implementing
  - *Seems obvious, but…*

Eventually: left with no choice



Otherwise, you probably won’t get here…

Or here ...

And *definitely* not here

Examples

- Java and Ant command lines
- Move files around
- Ant targets move or merge
- New ways to call Java code
- Remove XSLT templates, variables
- Remove obsolete plugins

Who has another favorite example?

So … how can you keep up?

Trick #1: Read the release notes

- What’s new, what’s different
- Link to “Migrating to X.y” topic
  - Available since 1.5.4
  - Much more comprehensive since Roger took over
- In particular, tracks deprecated functions, “this-might-break” changes

Trick #1.a: Don’t *just* read them…

Update customizations as you go

Trick #2: Upgrade as often as you can

- OK, easier said than done
- Easier to find info about, get help for recent changes than for old
- Break the work into small pieces

Trick #3: stay out of the core code

- Feel free to play around
- Change things if you're careful
- *But once you're done…*
  - Find a fix? Submit a pull request!
  - Changing function? Extract to plugin!
  - Broadly useful? Add it to the registry!
- ***Changing core code, without using plugins, eventually makes updates cost-prohibitive***

Trick #4: Use the productivity enhancements

- ***Use the plugin: syntax***
- ***Declare your parameters***
- ***For Ant extensions, use <ditafileset>***
- ***If something doesn't work...***
  Let us know!

Trick #5: Manage your own technical debt!

- Don’t need it? Remove it!
- Have a workaround in place for a bug? Remove it when upgrading
- *Full disclosure: with as many plugins as I manage, this is often difficult*
- *See also: Roger’s presentation from DITA Europe*

Trick #6: work with us!

- Regular contributor calls
- Monitor GitHub issues, pull requests
- Join DITA-OT Slack channel
- *Let us know what you're doing!*

Final trick: be here, and talk to us

- We do like to hear from you
- We are probably in a good mood today
- You may not get another chance to discuss your ideas one-on-one

Thank you for being here!

Thank you

Any questions?