Unconf 2017 - How teams can work in parallel using light dev

Problem Statement:

How can we scale massively and keep short release cycles? Light development being evaluated as a solution (not 100%, but a possibility - need reassurance that is the right thing).

Issue: want to reduce time to implementation.

Light modules can be used to help and reduce pressure on java dev team. You can mix light dev with what already exists, no need to migrate everything to light modules, just create new features as light modules.

You can deploy light modules to a running instance, no restart needed (e.g. use a GIT integration). WAR deployment only needed for back-end changes.

You can have two "release" cycles: one for front end (just push to git), one for back end. First step is to decide what stays in back end and what goes to front end, then start developing in parallel. There is also an area where the two overlap and you need to sync.

Q: Do you recommend splitting into separate teams for both ends?

- Yes.

Issue: If you have special requirements for a light module that need back end involvement - things become chaotic.

We want to use spring and light dev right now, not necessarily in the future. - If you want to go much faster, maybe think about moving to lighter dev. Just refer to your legacy code in the yaml file. Blossom will always require java dev.

All the light dev can be bundled in the backend and then deployed when all is ready(maven bundles).

Q: Another possibility: make more use of headless. Could you give more input? We would need asynchronous event monitoring, REST API is not so interesting for us.

- REST API is being worked on (Adrien)
- Content type (content app, UI, JCR workspace): should come in 2017 and then able to be used with light dev (Christoph)
- Help use front-end tech as Angular/REACH more easily with magnolia (Tomás)

- i.e. no need for Blossom → increase speed of development and deployment

Q: How long will you support Blossom?

- it's being maintained. Everyone is asking how to reduce time to market, Blossom doesn't respond to this need.

Q: What about Java 9 and potentially JIGSAW (OSGI-style)?

- we're waiting to see what it will actually contain (end of year?)

Q: Any customers with large teams relying on light modules?

- yes, one customer is creating more than 140 sites using light modules

Q: Git is a way of bringing something into production. We also want to use Jenkins. Is that the usual way?

- Yes, this is typical - and important to have UAT and/or integration before prod.

Q: Ways of testing what is done in light dev?

- Same as before. Advice: use typescript to validate. UI must be tested manually. The lighter the modules, the less tests needed.

- Please see this section on this page for possible approaches to testing light modules: Sharing modules - Advanced Frontend Topics#AdvancedFrontendTopics-Testing

Note: another nice thing about light dev = magnolia CLI

Next: examine how much could be done in light modules vs back end, then evaluate with Basel Services (Adrien / Jan)