What are your efficiency gains by removing XML in java projects? [closed]

Asked 16 years, 1 month ago Modified 12 years, 4 months ago

Viewed 270 times

3

As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, visit the help center for guidance.

Closed 12 years ago.

I have just moved from phase 1 to phase 2 of a project. In this transition we got rid of all of the XML configuration for spring and struts2, switching to a fully annotated regime.

All of the participants are amazed at the actual effect this has on development speed. There are a number of reasons for this:

- Structured thinking about which roles are needed lead to clear understanding that we needed @StubRepository, @TestService and @NotTestService annotations to facilitate all the

different run-time models that our project uses (runtime, unit test, integration test, selenium test)

- These annotations replaced a *large* number of duplicate xml definitions.
- All dependencies are fully trackable in the ide.
- Simpler to work with code because dependencies are clear

Some of these gains could probably have been realized by re-structuring the xml as well, but that would not have been a trivial task.

I honestly believe this has given somewhere around 10% boost in productivity for our 20,000 man hour project. I almost find it hard to believe.

Anyone else sharing this same experience ? What's your gain, and how do you explain it ?

java **xml**

edited Aug 7, 2012 at 14:39

Bill the Lizard

405k ● 211 ● 572 ● 889

asked Nov 12, 2008 at 10:36

krosenvold

77k ● 33 ● 156 ● 209

Mystical Man-hour ... :) – Robert Gould Nov 12, 2008 at 11:18

I believe the overall size of the project is relevant because it reflects the complexity (and amount of) XML involved. These are budget figures anyway ;) – krosenvold Nov 12, 2008 at 11:35

3 Answers

Sorted by: Highest score (default)

2

Basically, this could be viewed as a question about config files vs. annotation.

If you experienced an amelioration in productivity boost when putting some data as annotations, that means they were not "configuration" material in the first place.

The difference between the two:

- annotation: everything is in one place and the configuration information is associated directly with the Java component. Many types of refactorings of the source code are transparent to annotations for this reason -- the annotations will always apply to the component they're attached to, even when that component is moved or renamed. For other refactorings that require new or modified annotations, everything is in the same location, assuring that the annotations are visible to the developer and increasing the likelihood that they'll remember to make the necessary changes.

- Configuration files can provide an organized view of a web of relationships that span different components of an application. Because they're separate from the actual source code, they don't interfere with the readability of the Java source code.

What you experienced is the disparition of the need to maintain config files in parallel with the source code of an application, with no obvious connection between the two.

BUT: annotations have their drawbacks: source code can become cluttered with all sorts of annotations that are irrelevant to the actual program logic, interfering with the readability of the code. And that while annotations are ideal for metadata that relates to a particular component, they're not well suited to metadata with cross-component application.

Share Improve this answer

Follow

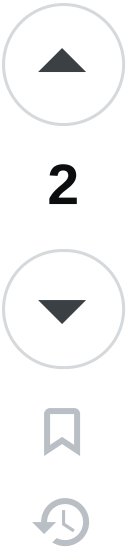
answered Nov 12, 2008 at 12:17

VonC

1.3m ● 558 ● 4.7k ● 5.6k

Our lunch discussion recently has been if those centralized configuration files ever had any real benefits. They seem like a good idea at the "Pet Shop" level, but when your project reaches a certain size they're more or less useless.
– krosenvold Nov 12, 2008 at 12:28

Nice lunch ;) Config files are very useful for large projects, where modifying a value does not require updating/re-compiling the code! But used incorrectly and config files can be a drag. – VonC Nov 12, 2008 at 12:48



(I really didn't mean to respond with a bag of cliches, but XML seems to bring out the proverb collection... ;-)

I have found XML configuration (and config files in general, but that's another question) to introduce multiple levels of your question, because, "To a small boy with a hammer, everything looks like a nail." It is easy to become enthusiastic about a tool (such as XML) and start using it much more widely than it deserves, or to keep adding "just one more" feature until the result becomes too unwieldy.

Also, as the old saying goes, "Everything in programming is a trade-off."

So, to play devil's advocate, have you reached the point where you need to do maintenance or tweaks on the running system? In the new scheme of life, how often do you have to make adjustments by re-building all or part of the system, instead of being able to tweak a config file?

Finally, there's rough equivalent of the "Hawthorne effect" (see below). Sometimes a team gets a burst of excitement when they are allowed to clean up some minor issue, and they achieve a burst of productivity out of proportion to the actual improvement, triggered by the relief at getting rid of a long-standing annoyance.

Footnote:

The Hawthorne effect is named after a study of factory worker productivity relating to changes in lighting. The

researchers found that increased light was followed by an increase in productivity. But then they found that ***decreased*** light was also followed by an increase in productivity. (Both increases were typically short-term.) The final conclusion was that the workers were more productive because someone was paying attention to them.

So, sometimes an otherwise trivial change triggers a boost in morale (sense of empowerment, etc.) that causes improvements in performance.

There's seldom just one factor.

Share Improve this answer

Follow

answered Nov 12, 2008 at 13:02

joel.neely

30.9k ● 9 ● 57 ● 64

> I find the reality of enterprise systems is that these configurations NEVER get changed without rebuilding & testing anyway. I've never seen anybody actually USE the flexibility of xml for anything. Automated builds & automated teests make you WANT to rebuild with these changes.
> – krosenvold Nov 12, 2008 at 14:28

0

I don't have productivity figures like you but I have seen significant improvement when moving from xml configurations to annotations. I think its because the configuration information is in the same place with the code.

Before you had to look at a separate file to find configurations and that slowed things down.

Share Improve this answer

Follow

answered Nov 12, 2008 at 10:51

Vincent Ramdhanie

103k ● 23 ● 142 ● 194

Well our scrum numbers indicate 30% improvement, but I cannot attribute all of that to this one change. It just feels so much simpler, complexity is a bigger killer than I was aware of. – krosenvold Nov 12, 2008 at 11:01