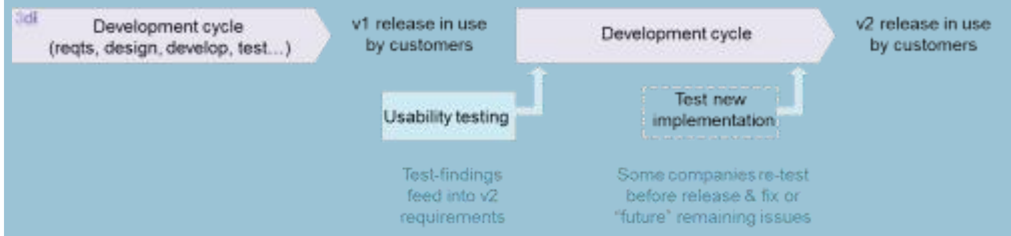
When’s the best time to usability-test software?

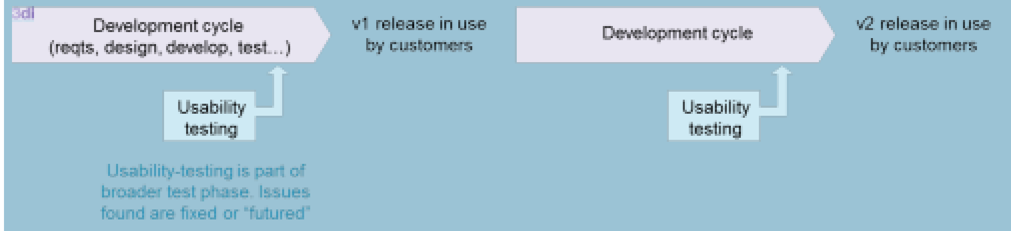
I’ve been talking to people at a lot of software companies recently. On the one hand, I’ve been talking to managers and members of User Experience teams about how they “do” usability for software. At the same time, I’ve been talking with software product managers, development managers and others who want to know how my company (3di Software Usability) suggests fitting usability testing with their development processes. I thought it might be useful to put some thoughts down about the latter, based on what I’ve found out from the former. So here goes…
Most development teams fit usability testing into their processes in one of three ways:

1. **Usability testing as part of requirements gathering phase**
In this approach, a thorough batch of testing on a product that’s already released and in use, as part of the requirements gathering for the next version. This may or may not be followed up by integrating usability testing more tightly into development teams in future phases.

Over the course of a couple of development cycles, this looks something like the following:

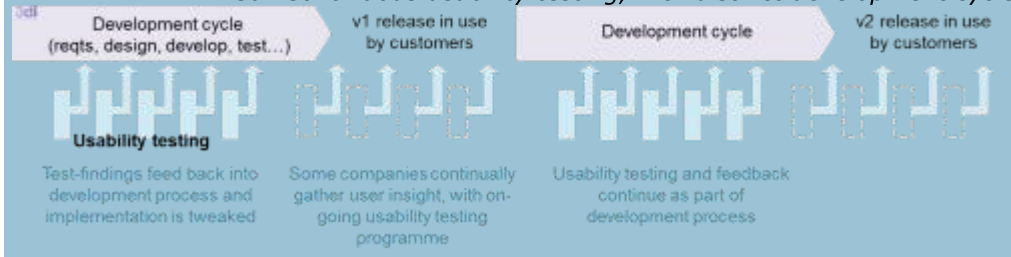

3di
Development cycle (reqts, design, develop, test…)
v1 release in use by customers
Development cycle
v2 release in use by customers
Usability testing
Test new implementation
Test-findings feed into v2 requirements
Some companies re-test before release & fix or ‘future’ remaining issues

2. **Usability testing as part of the testing phase of development**
A batch of testing on all new features towards the end of the development phase – before code freeze – to enable changes to be made in response to issues discovered. This is sometimes followed by a small number of tests to confirm improvements have been effective.

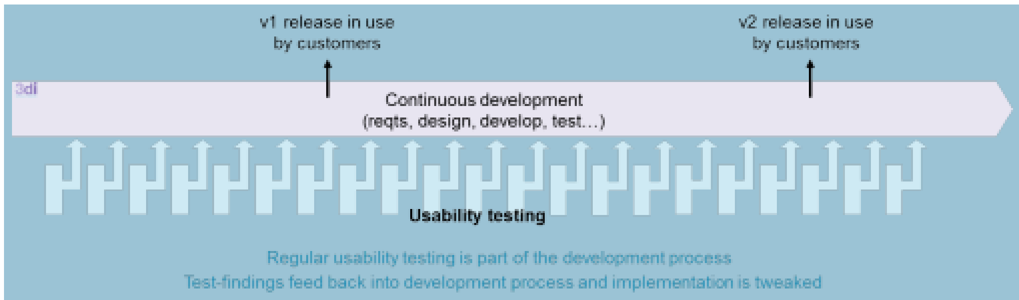

3di
Development cycle (reqts, design, develop, test…)
v1 release in use by customers
Development cycle
v2 release in use by customers
Usability testing
Usability testing
Usability-testing is part of broader test phase. Issues found are fixed or “futured”

3. Usability testing fully integrated with development process (alongside the development of new features)
As each feature or user story is designed or implemented, one or two tests are done to ensure it is effective. There are a couple of variations of this approach, to fit with different development processes, as shown in the following illustrations.

3a: Continuous usability testing, with distinct development cycles


3di
Development cycle (reqts, design, develop, test…)
v1 release in use by customers
Development cycle
v2 release in use by customers
Usability testing
Test-findings feed back into development process and implementation is tweaked
Some companies continually gather user insight, with on-going usability testing programme
Usability testing and feedback continue as part of development process

3b: Continuous usability testing as part of a continuous development model


v1 release in use by customers
v2 release in use by customers
3dl
Continuous development (reqts, design, develop, test…)
Usability testing
Regular usability testing is part of the development process
Test-findings feed back into development process and implementation is tweaked

Realistically, usability testing tends not to be as regular as these illustrations suggest; instead it happens in waves, depending on the stage of development and other demands on the usability resource's time.

Incidentally, only one company I talked to actually does #3b – though several aspired to it.

Which is the best way to fit usability testing into the software development process?
Unsurprisingly, the answer is…

"it depends"

…on your development model, type of product, availability of resources, and so on. Sorry for not being able to supply a more definitive answer.

On the other hand, a pattern did seem to emerge from the conversations I had with the more mature user experience teams:

Many teams started with something like #1. It's a nice way to get experience of what you can get out of adding usability testing to development processes. In addition, the insight and test videos it produces help smooth the way for development teams to be motivated to find ways to include usability in their processes – rather than resisting the introduction of this new (to them) specialism on their "turf".

The product is re-tested towards the end of the development process, as in #1. Testing at this stage then becomes a normal part of the development process – i.e. model #2 is adopted.

Testing uncovers issues on areas that have had a lot of development input. It may even feel too late to make changes to these areas so the issues get "futured". Eventually, the lesson learned is that spending time developing problematic UIs is expensive, so the process moves towards model #3.

In the companies I spoke to with in-house user experience specialists, most were in the process of transition from #2 to #3, which is probably not coincidental. Approach #3 fits well with many agile development teams, who do one or two tests per sprint that involves UI development … and these teams were also in the process of transition to agile development. It also works well when the user experience resource is also able to get involved in reviewing and commenting on designs before they're implemented. Ultimately, these teams often aim to move to testing on designs or wireframes, rather than on fully coded features – to reduce code-rework.

How about you?
Are these models and transitions familiar to you, or do you have an entirely different experience of usability testing in the software development process? I'd love to hear…

Source: http://communicationcloud.wordpress.com/2011/12/16/how-usability-testing-fits-with-software-development-processes/