

AGILE Development Methods

There are many agile development methods; most minimize risk by developing software in multiple repetitions (or 'iterations') of short time frames (known as 'timeboxes'). Software developed during one unit of time is referred to as an iteration, which typically lasts from two to four weeks. Each iteration passes through a full software development cycle, including planning, requirements analysis, design, writing unit tests, then coding until the unit tests pass and a working product is finally demonstrated to stakeholders. Documentation is no different than software design and coding. It, too, is produced as required by stakeholders. An iteration may not add enough functionality to warrant releasing the product to market, but the goal is to have an available release (without bugs) at the end of each iteration. At the end of each iteration, stakeholders re-evaluate project priorities with a view to optimizing their return on investment.

Agile methods emphasize face-to-face communication over written documents. Most agile teams are located in a single open office to facilitate such communication. One agile management methodology, Scrum, advocates a team size of 5 to 9. Larger teams than 9 should be split into smaller teams to help make team communication and team collaboration easier.

Team composition in an agile project is usually cross-functional and self-organizing without consideration for any existing corporate hierarchy or the corporate roles of team members. No matter what development disciplines are required, at a minimum, every agile team will contain a customer representative. This person is appointed by stakeholders to act on their behalf and makes a personal commitment to being available for developers to answer mid-iteration problem-domain questions. This availability is critical to agile project success.

Part of the Scrum methodology is routine and formal daily face-to-face communication between team members. This specifically includes the customer representative and any interested stakeholders as observers. Known as a daily scrum, team members report to each other what they did yesterday, what they intend to do today, and what their roadblocks are. This formalized face-to-face communication prevents problems being hidden, provided that someone with corporate influence is always listening. For the Scrum methodology, this is the Scrum Master. Otherwise it is the agile project manager.

Agile methods emphasize working software as the primary measure of progress. Combined with the preference for face-to-face communication, agile methods usually produce less written documentation than other methods. In an agile project, documentation, Gantt charts and other project artifacts all rank equally with working product. However, when stakeholders are asked to prioritize deliverables for demonstration at the end of the current iteration, they generally prefer to see working product. Stakeholders are encouraged to prioritize iteration outcomes based exclusively on business value perceived at the beginning of the iteration. If documentation represents higher business value than working software in any particular iteration then stakeholders give it a higher priority than working software. The (cross-functional) development team will accordingly produce that documentation instead of lower priority software.

Agile means being able to quickly change direction. In software development, it requires strong discipline to code for agility. It includes writing tests for functionality before coding. It calls for naming of functionality to exactly match the intent and the terminology of the problem domain. It demands cessation of coding



when the tests pass. The sum total of all the disciplines delivers an ability to change direction quickly. New and unexpected functionality required to cope with a sudden change in the business landscape can be inserted in existing code using test-driven development and all the previous tests will pass or fail to instantly indicate where code needs to be refactored to stay functional. If functionality is added before it is required then it becomes "dead weight" when refactoring is called for.

Good Luck!

Many things have changed - especially the mediums - but universal truths of communication tend to survive. **Please feel free to contact us at <http://www.4CInteractive.com>**

Source: http://en.wikipedia.org/wiki/Agile_software_development