Agile “Berlin Bunker” Exercise

1.Michael Mullarkey (16204743)
2.Xiaosheng Liang (15211913)
3.Niranjan Shankar (15202453)

View the Youtube clip here: http://www.youtube.com/watch?v=bRSFuXfNM0E. The Iteration #6 planning meeting features a number of terms related to software development in general and Agile methods (mainly Extreme Programming) in particular:

- **iteration**: Iteration is a single development cycle used in agile software development. Iteration is used to gradually piece together the product advancing to the project’s each iteration. Iteration is the main concept in the video, the manager is furious when he’s told testing hasn’t been done since iteration #3

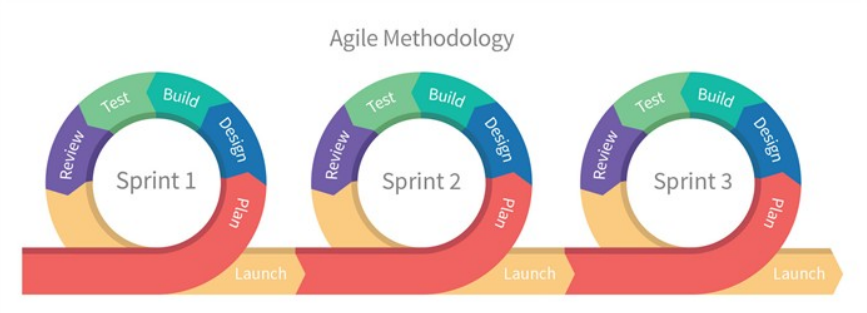
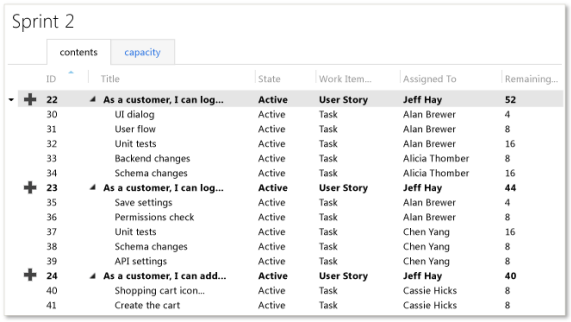


Figure 1: Iterations in an agile methodology

- **user story**: User stories are short simple descriptions of a feature told from the perspective of the customer/user. Senior developer mentions that they’re behind on work so unfinished user stories are carried over from the last iteration (iteration #5)



	ID	Title	State	Work Item...	Assigned To	Remaining...
✚	**22**	**As a customer, I can log...**	**Active**	**User Story**	**Jeff Hay**	**52**
	30	UI dialog	Active	Task	Alan Brewer	4
	31	User flow	Active	Task	Alan Brewer	8
	32	Unit tests	Active	Task	Alan Brewer	16
	33	Backend changes	Active	Task	Alicia Thomber	8
	34	Schema changes	Active	Task	Alicia Thomber	16
✚	**23**	**As a customer, I can log...**	**Active**	**User Story**	**Jeff Hay**	**44**
	35	Save settings	Active	Task	Alan Brewer	4
	36	Permissions check	Active	Task	Alan Brewer	8
	37	Unit tests	Active	Task	Chen Yang	16
	38	Schema changes	Active	Task	Chen Yang	8
	39	API settings	Active	Task	Chen Yang	8
✚	**24**	**As a customer, I can add...**	**Active**	**User Story**	**Jeff Hay**	**40**
	40	Shopping cart icon...	Active	Task	Cassie Hicks	8
	41	Create the cart	Active	Task	Cassie Hicks	8

Figure 2: Example: As a customer, I can log into the portal and view my profile

- **use case**: A use case is a description of steps needed to complete an action by a user. In the beginning of the meeting the manager is informed of stories carried over from iteration #5, as there are many alternate paths through the use cases.

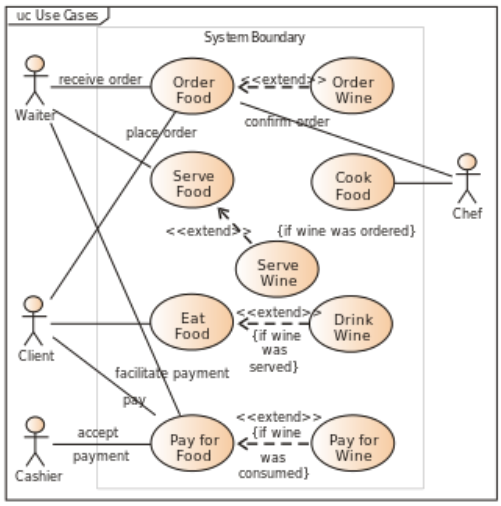


Figure 3: A use case for order management in a restaurant

- **automated unit testing**: A unit test is an automated piece of code that invokes a unit of work in the system and then checks a single assumption about the behavior of that unit of work. The automated unit testing is also being used as an excuse as to why stories are being carried over from iteration #5, as they were "more challenging than anticipated".

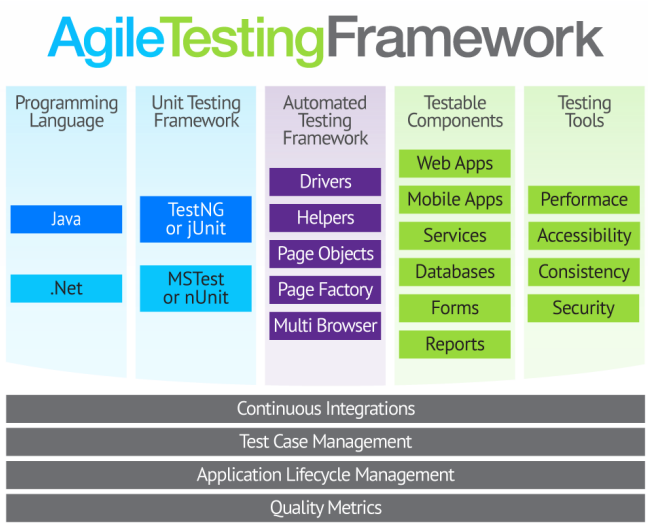


Figure 4: Agile Testing Framework

- **code coverage**: The measurement/completion of the source codes (executed commands) while automated tests are running. The manager rebuttals the reasons as to why they're behind on work, as he tries to gain reassurance by insinuating that 90% of code coverage has been completed.

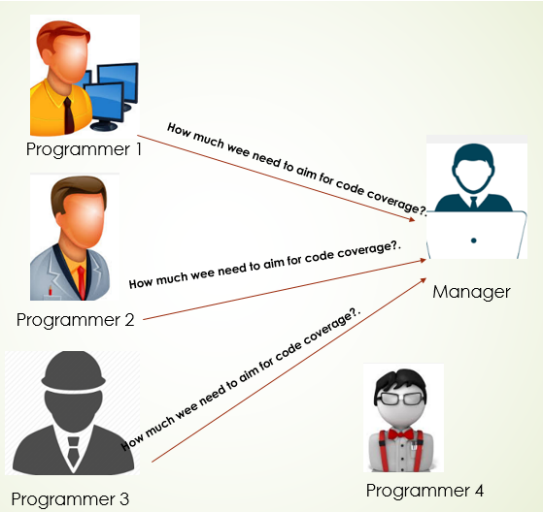


Figure 5: Code Coverage

- **mock object**: The aim is to simulate desired behavior of real objects under a controlled environment. The manager chastises the senior developers as mock objects haven't been produced.

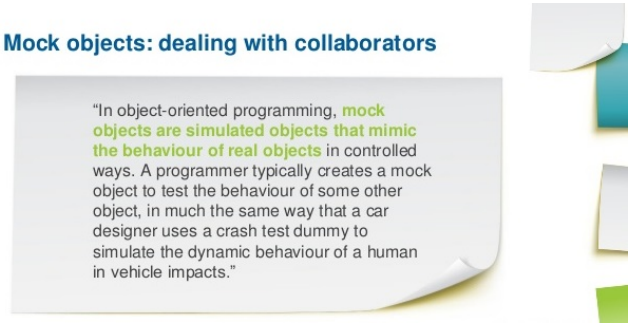


Figure 6: Definition of mock object from Wikipedia

- **stand-up**: A stand-up meeting often takes place in an agile environment where the participants attend meetings while standing. Due to this discomfort, the meetings are often shorter in durations. The senior developer gives excuses for not having mock-objects (test cases) ready and the manager asks why they never mentioned this during daily stand-ups.

Figure 7: a stand-up meeting in an agile team

- **continuous integration**: It is a coding practice where all developers integrate their code with a shared common repository like GitHub or Visual Studio, at the end of the day. After everyone has checked in, the code is sent to a build to detect any bugs/issues. The developer says they do not have test cases updated and the manager blasts at them by saying "we agreed for continuous integration, test driven development".

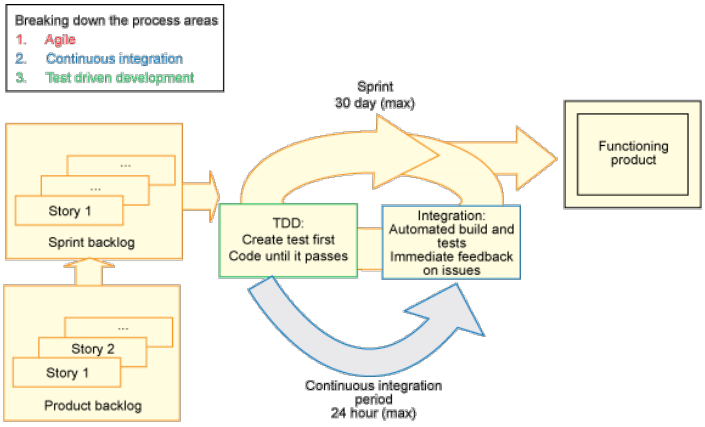


Figure 8: Continuous Integration as an important process area

- **test-driven development**: TDD is an approach where tests are written based on requirements, before writing actual code and the code is written just to pass this test case only. This happens in an iteration until all the specifications are met. The developer says they do not have test cases updated and the manager blasts at them, "we agreed for continuous integration, test driven development".

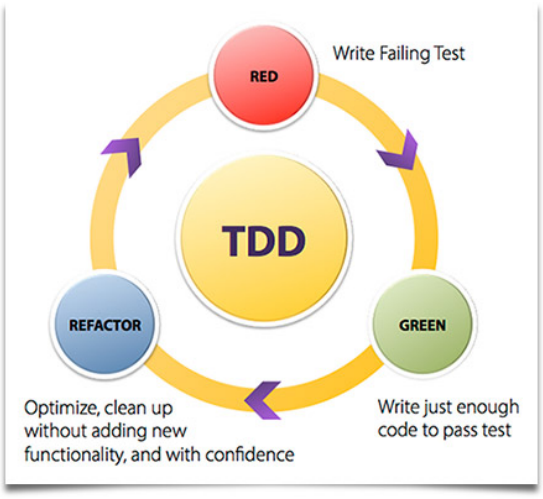


Figure 9: Steps in Test Driven Development

- **"test behavior, not just state":** Test cases should not test deep functionalities/implementations of a method or a class because later, even for minor changes of code, test cases must be modified accordingly, and this increases time, effort and bugs. For example, a test case could only test the inputs given and outputs returned by a method. One need not test how it achieves it. A team member complains that "the customer has many alternate paths through the use cases", to which the manager replies "[We agreed] that we would test behavior, not just state".

- **“embrace change”**: Be open to any change in requirements at any time and accept that the process is dynamic. Our goal is to serve the customers and they are the center of everything we do. The senior developer accuses customers of making too many changes in requirements and the manager shouts back “That is why we call it Agile. We were supposed to embrace change”

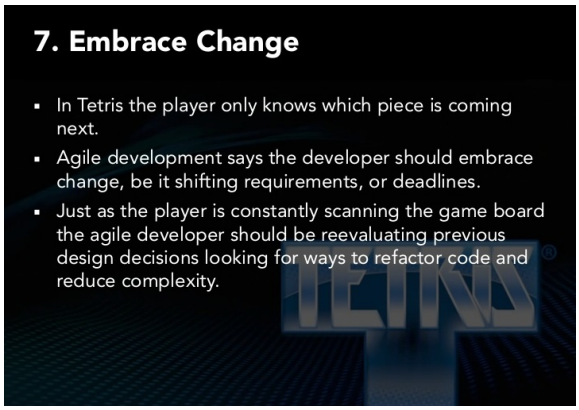


Figure 10: Embracing change in an agile compared to TETRIS game

- **velocity**: The velocity is how much work (or how many tasks) have been done in a certain time-period. The tasks are counted by the number of same units of project which have been decided at the beginning of the project. The manager asks about the velocity, indicating that he wants to know how long it would take to finish the project.

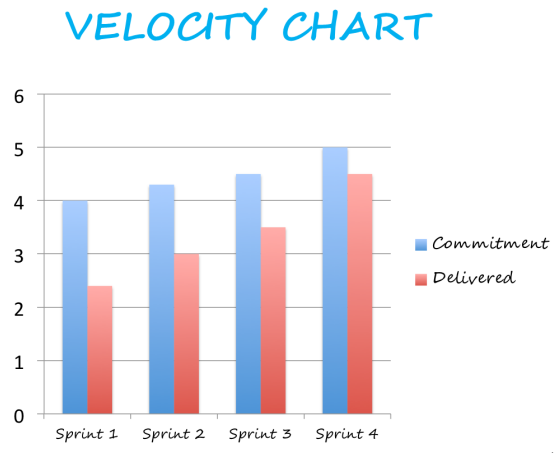


Figure 11: Velocity chart across sprints

- **hourly build**: Hourly build means frequent and regular builds to make sure the integration occurs frequently and avoid the problem. The manager asks his team to run the build once an hour and integrate frequently.

- **story point**: Story points are used to calculate the effort, complexity and risk needed for user stories. It’s a unit of estimating the effort it would take to finish a story. The manager requires his team get credit for story points after all the test done.

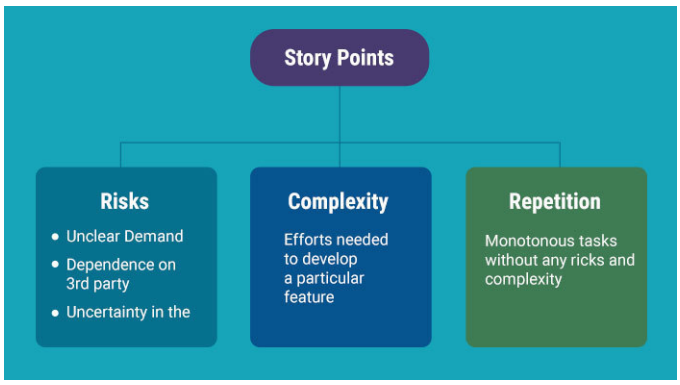


Figure 12: Factors contributing to a Story point

- **Agile Manifesto**: Agile Manifesto officially announces the four core values and twelve principles that guide the software development approach of people-centered iteration. It's foundation of agile development, which the manager still believes, despite current failure to deliver.

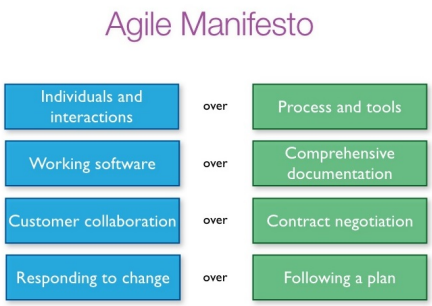


Figure 13: Agile manifesto

- **waterfall**: The "traditional" approach to software development involving a sequence of phases, analysis, design, implementation, testing, maintenance. It's the antithesis of Agility, so the manager mocks his team that they want to "go back to waterfall."

- **burndown chart**: Burndown chart is a visual representation of the work that needs to be done before the project is completed. Burndown charts do look hopeless means all the works are almost done.

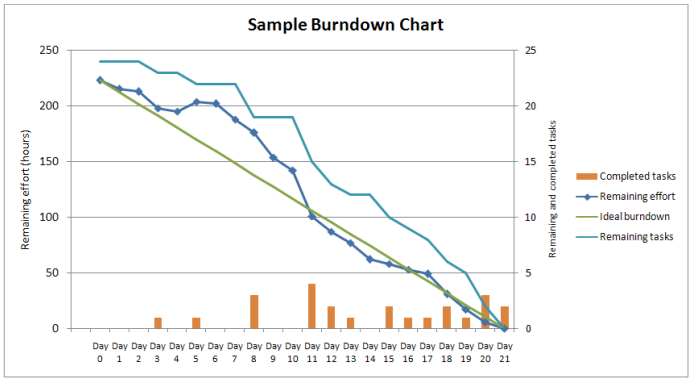


Figure 14: Sample burndown chart