Quiz 08: FDD

Due Apr 4 at 11pm **Points** 50 **Questions** 11 **Available** until May 15 at 11:59pm
Time Limit None

Instructions

You **may** use the slides from the lecture and other sources to answer these questions. Please be sure to cite any references but be sure to answer the following questions in your own words. Do NOT simply cut and paste the information from the slides. You will receive a score of 0 if you copy the prose from the slides.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	1,155 minutes	5 out of 50 *

* Some questions not yet graded

Score for this quiz: **5** out of 50 *
Submitted Apr 2 at 7:02pm
This attempt took 1,155 minutes.

Question 1

Not yet graded / 5 pts

Choose one of the quotations from the Palmer paper and explain how it relates to Software Engineering or Agile Methods.

Your Answer:

"Kanban is the science of not trying to do much at once"

It's more like dividing a tiny amount of work into each sprint and having each developer focus on one or two features from that sprint, in my perspective. During each sprint, the team will consult with a few users to see if there are any new requests for a feature or if any revisions are required. It would be more easier to complete the job in this manner rather than doing it all at once and then making revisions.

Ex: "Human brain capacity is more or less fixed, but software complexity grows at least as fast as the square of the size of the program." Gerald M. Weinberg, Quality Software Management: Volume 1 Systems Thinking, 1992

This quote is related to Agile Methodology, since it points out the problem with planning out too much at one time. The human brain is unable to think that far in advance, when it comes to software advancements, and since software advances so quickly, this proves that planning in small iterative bursts is best. This also implies that the more software that is designed in one period of time, the more and more complex it becomes, so working in iterative bursts also reduces the complexity of the software system, keeping it maintainable.

Question 2

Not yet graded / 5 pts

Describe three items that FDD suggests be kept under configuration management.

Your Answer:

Three items that FDD suggests being kept under configuration management are: -

Firstly, keep requirements documents under configuration management: It guarantees that the documentation is usable, and that the team can better comprehend the requirements.

Secondly, keep test cases under configuration management: It aids in improving the test coverage, which in turn aids in enhancing the software's quality.

Thirdly, Analysis and design artifacts: A good analysis and design improves a team's development efficiency and takes into account all of the potential problems that the team may face during the development process.

Code to to manage conflicts and maintain consistency. Requirements documents. Analysis and design artifacts. Test cases, test harnesses and scripts and even test results. Version of the process you are using, any changes and adjustments that may be made during the construction and maintenance of the system which is useful for audits and measuring process effectiveness

Question 3

Not yet graded / 5 pts

Describe two things that FDD shares with other agile methods.

Your Answer:

Two things that FDD shares with other agile methods are: -

Firstly, FDD and Agile allows teams to update the project regularly which is called continuous integration in agile and identify errors quickly and early. Plus, clients can be provided with information and substantial results at any time which is also done in agile.

Secondly, Feature-Driven Development (FDD) and Agile methodology both are customer-centric, iterative, and incremental, with the goal of delivering tangible software results often and efficiently.

References: (**https://www.planview.com/resources/articles/fdd-agile/#:~:text=An%20Agile%20methodology%20for%20developing,to%20track%20progress%20and%20results** (https://www.planview.com/resources/articles/fdd-agile/#:~:text=An%20Agile%20methodology%20for%20developing,to%20track%20progress%20and%20results) .)

Short, time-boxed iterations, Lightweight process, Requirements described as features, Customer participation in planning (but not as often, and at a higher level)

Question 4

Not yet graded / 5 pts

Explain how FDD addresses the Agile Manifesto's concern about working software over comprehensive documentation.

Your Answer:

The typical method to software development is to collect all customer requirements at the start of the project and then plan the rest of the development phases without involving the customers. Months are spent writing requirements, design, analysis, and test case documents by the team. When the team is finished with the requirements, these documents might become obsolete. This is where FDD comes in. In FDD the documentation is important and essential. In FDD a feature/ documentation needs to be given every 8 – 10 days which make made after taking the customer into consideration. This way when the team is finished with the requirements the documents are not obsolete. This is how FDD addresses the Agile Manifesto's concern about working software over comprehensive documentation.

Working software over comprehensive documentation doesn’t mean you shouldn’t write documentation. Instead, documentation is important and required, but documentation without working code is a path to failure

Question 5

Not yet graded / 5 pts

Explain how FDD addresses Glenford Myers' quotation, "We try to solve the problem by rushing through the design process so that enough time is left at the end of the project to uncover the errors that were made because we rushed through the design process"

Your Answer:

Glenford Myers' quote suggests that the team should finish the job as quickly as feasible, then they will have plenty of time to fix errors and make changes and improvements.
However, by the time the project is finished, a new core feature will need to be introduced, or an existing core feature will need to be removed since it has become obsolete. It may be tough to make such modifications.
FDD is an iterative and incremental software development approach that comes in to solve the problem. It is more common to plan every step of the procedure at the start of a project. Instead of simply preparing, we must be willing to adapt the plan in response to the needs of the consumer.

FDD's view on the Agile Manifesto is Responding to change over following a plan doesn't mean you shouldn't plan. Instead, planning is important but you must be willing to change the plan

Question 6

Not yet graded / 5 pts

Describe two things about FDD that are different from other agile methods.

Your Answer:

Two things about FDD that are different from agile methods are: -

Firstly, during FDD as its name implies, it's a "feature-focused method" they are the foundational piece of FDD as opposed to a delivery-focused method. A feature to FDD is what user stories are to Scrum. A feature in FDD should be delivered every 2-10 days which is different from Scrum, in which sprints typically last about two – four weeks.

Secondly, A key difference between FDD and Agile is the end user. In FDD, the actual user or customer is viewed as the end user, whereas in Scrum it's typically the Product Owner who is seen as the end user.

References: (**https://www.planview.com/resources/articles/fdd-agile/#:~:text=An%20Agile%20methodology%20for%20developing,to%20track%20progress%20and%20results** **(https://www.planview.com/resources/articles/fdd-agile/#:~:text=An%20Agile%20methodology%20for%20developing,to%20track%20progress%20and%20results)**.)

One aspect that differs FDD from Agile is that FDD has initial planning and design, unlike Agile, even if it is only JEDI (Just Enough Design Initially). FDD's approach was created to prevent having to change the solution along the way, like Agile does because the initial team takes the time to learn more about the domain it is in.

Another difference is that the feature teams do not own their work as a whole, unlike in Agile which does do this. In FDD, developers own their own classes which they are experts on, but are not responsible as a whole team.

Question 7

5 / 5 pts

FDD relies on which technique rather than pair programming?

- pair programming
- Correct! [✓ inspections]
- collective code ownership
- documentation

Question 8

Not yet graded / 5 pts

Describe one advantage and one disadvantage of individual class ownership.

Your Answer:

Advantage of individual class ownership is Owners can take pride in the classes he/she has made because they are the only one who have put all their effort in making the class which is difficult. There is no pride or a sense of accomplishment when a class is made in collaboration with other person.

Disadvantage of individual class ownership is that we cannot make changes in the class without asking or taking the permission of the other person. This is done because even though both the owners are working on the same class owner A does not know what exactly owner B is working on. So every time asking the other person for permission becomes a tedious job. Because if this is not done it might rise problems or errors in the class and might start an argument between the owners.

Advantages: Owner maintains conceptual integrity of class, Each class has an expert associated with it, Owner can implement changes faster than anyone else, Owners can take pride in their classes

Disadvantages: Dependencies between classes (owner A needs owner B to make changes to their classes), Risk of loss of owners during development

Question 9

Not yet graded / 5 pts

Describe FDD's Process 2: build a features list. Who's involved? What's the goal?   What are the outcomes?

Your Answer:

FDD's Process 2: build a features list: - By functionally partitioning the domain into topic areas, knowledge collected during the initial modeling is used to identify a list of features. Business activities are contained in each subject area, and the steps within each business activity serve as the foundation for a categorized feature list. Small chunks of client-valued functions stated in the form "action> result> object>". For example "Calculate the amount of a sale" or "Validate the password of a user." If a feature takes more than two weeks to complete, it should be split down into smaller chunks.

Chief Programmers and the developers under them are involved in building a feature list. Their goal in this process is to Build the project backlog and stating the specification of each feature and its purpose and how it would help the project. The outcome of this process is to produce a Hierarchy of feature list which is oriented to the business needs, including Functional areas, Business activities and Business features.

Building the features list is led by the chief programmers. The goal of this process is to have all of the features of the system defined by the end of it. The feature list is hierarchical, unlike user stories, and oriented to the needs of the business. Each feature that is created by the feature list team can not take more than 10 days to complete, and follow the outline <action><result><object>. The outcome of this is the feature list to be implemented which is the start of defining the classes and the methods that should be a part of each class.

Question 10

Not yet graded / 5 pts

Describe three of the six roles in FDD

Your Answer:

Three roles in Feature-Driven Development are: -

1. The Project Manager - It's a role which makes good progress reports, to work for the well-being of the team, to deal with budgets, recruitment, and logistics. If we

examine this description thoroughly, it is often the role attributed to Scrum Masters in large companies who have trouble becoming 100% Scrum Master.

2. The Domain Experts - This role is represented by Key Users, Sponsors and Business Analyst. They are present to identify needs and allow developers to work. Their presence is essential to have a quality product.
3. The Chief Programmers – In FDD, The analysis and technical specifications will be handled by experienced developers. They will oversee a team of three to six developers.

One of the roles in FDD is the domain expert. The domain expert is the champion of the customer. They speak on the customers behalf because they not only have direct access to the customer, they are also experts in the domain the software is being built.

Another role is the feature teams. Feature teams are groups of developers created to implement certain features in the system. These teams only last as long as it takes to finish the defined feature, and then they are broken apart. The teams are self reliant and formed based on what classes are needed for what features and who owns them. This means developers can be on multiple teams at one time.

A third role is the class owner. The class owner is a developer who take ownership of a certain class that will be implemented in the system. The class owners are the experts in their class, and are responsible for maintaining them and helping others integrate them.

Also Project Manager (DeLuca) –administrative lead, Chief Architect (Coad) –responsible for overall design, Development Manager (Palmer) –team management, Chief Programmers– lead teams in design of features

Question 11

0 / 0 pts

“I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. I further pledge that I have not copied any material from a book, article, the Internet or any other source except where I have expressly cited the source.”

Correct!

[✓ True]

False

Quiz Score: **5** out of 50