Liam Brew
SSW 555

Professor Ens
16 November 2020

Homework 09: Feature Driven Development

Description of FDD Roles

The six roles mentioned by FDD are as follow:

1. **Domain Expert**: this role serves as the primary customer representative to the development process. Domain Experts possess intricate knowledge of the problem domain itself as well as knowledge of what the user's requirements are. They ensure that the development process is on track to satisfying the user's needs.
2. **Project Manager**: this person is the leader of the project and serves to organize the overall development process. They fill an administrative role and oversee aspects such as budgeting and equipment management, planning, and progress reporting. They directly facilitate the work of the rest of the team.
3. **Chief Architect**: the Chief Architect is responsible for the overall design and planning of the system under development. They must have a large degree of technical knowledge surrounding the project and be able to plan the implementation of the system both now and down the road.
4. **Development Manager**: this role is responsible for directly overseeing the usual development activities and the typical day-by-day progress of the team. They must also have good technical skills to understand the project and assist the team members in whatever ways they need. The Development Manager serves as a first line manager for helping the team resolve any difficulties or issues that they may face.
5. **Chief Programmer:** a Chief Programmer is an experienced developer who is well versed in both the project itself as well as software development in general. They oversee Feature Teams in the actual implementation of software corresponding to the feature under development and help transpose user features into feature design.
6. **Class Owners**: this person is a developer which oversees a specific class of code. They are responsible for designing, developing, maintaining, documenting, and testing the features and functionality associated with their class.

Applicability to Horn Honking

An example of each role's responsibilities in the new horn honking feature is as follows:

1. **Domain Expert**: for this project, a Domain Expert can be either a professional driver or someone familiar with the overarching self-driving software. The former offers insight into the feature itself and what characteristics it should take on, and the later helps synchronize the requirements of this feature with the greater self-driving car initiative.
2. **Project Manager**: in this scenario, the Project Manager is responsible for dividing the project's available budget and resources among the different components of the team. Additionally, the Project Manager must report to higher ups regarding the horn team's progress, to give higher level managers a better idea as to the overall status of the self-driving car itself.
3. **Chief Architect**: the Chief Architect here is responsible for the design and implementation of the system itself. Some components of this system the algorithm that determines when it is appropriate to honk and the interfacing of this algorithm with the car's physical horn hardware.

4. **Development Manager**: the Development Manager coordinates progress between the smaller development and feature teams, such as the algorithm and interface teams. The Development Manager ensures that both teams are progressing well and resolves any issues or communication errors between the two.
5. **Chief Programmer**: a Chief Programmer leads the implementation of the algorithm that determines when it is appropriate to honk the horn. They are responsible for its planning, development, and testing.
6. **Class Owner**: a Class Owner is responsible for a specific class that comprises a part of the honking algorithm. They must design this code, implement it, test it, and support it, while also documenting every stage of this process.

Comparison with eXtreme Programming

1. **Domain Expert**: this role most closely aligns with eXtreme Programming's Customer role. This is because the Domain Expert of FDD is described as a "voice of the customer", basically one that advocates for the customer's needs and requirements during the development process. An advantage of distinguishing this role is that it ensures that there is always a member of the development team on hand that possesses valuable insight into what the customer may want, and unlike a customer itself, a Domain Expert likely has an amount of development experience as well. A disadvantage of this is that while the Domain Expert may be knowledgeable in the project's field, this is no guarantee that they know exactly what the customer wants. To accomplish this requires direct interaction with the customer, which XP provides.
2. **Project Manager**: this FDD role synchronizes well with XP's manager. Both must deal with high-level development team management and administrative duties such as budgeting and scheduling and are relatively removed from the technical work itself. An advantage of FDD's method is that the Project Manager can assign ownership of code to individuals. This means that for every class and component of code, there is a dedicated individual responsible for working on and maintaining it. A disadvantage of this compared to XP is that developers have less flexibility in choosing what to work on, which can harm productivity as a person's competency may be wasted on something, they are not familiar with.
3. **Chief Architect**: this role does not have an XP counterpart. In XP, the system design is broken down into stories at the discretion of the end user. While developers are expected to collaborate on the implementation of these stories and how they fit into the larger system, there is no central role or authority to facilitate this.
4. **Development Manager**: this role is most like XP's Coach. Both roles are responsible for managing day-to-day development progress and helping to resolve any issues that may arise. Because of this, the Development Manager may also satisfy some of the responsibilities of XP's Tracker as well. Both roles are similar in nature.
5. **Chief Programmer**: this role does not really have an XP counterpart. In XP, programmers are typically equal level and there is not much in the way of a hierarchy. This contrasts with FDD, wherein a Chief Programmer leads a specific group in the design and implementation of certain features. FDD has the benefit of having a lead developer in charge for every aspect of the code which entails benefits such as more efficiency and tighter conformity to standards. However, XP can potentially offer more in the way of programmer flexibility and leverage an individual's experiences better.

6. **Class Owner**: this role is related to XP's Programmer. However, unlike XP, FDD programmers take ownership of their specific segment of work. This is in comparison to XP's more random assignment of stories on an availability basis. A benefit of this is that a Class Owner becomes very adept and familiar with their code thereby centralizing a feature's knowledge in an individual or two. A drawback is that this requires increased cooperation between developers for things like changing a dependency.

Decision

As an overall decision for the project, I would choose FDD. This is due to the importance of the feature under question. With people's lives potentially depending on the success or failure of the automatic honking, it is important to ensure that development is as organized and thorough as possible to reduce the likelihood of any mishaps. FDD facilitates this through dedicated roles such as the Chief Programmer and Chief Architect whose entire purpose is dedicated to specific components of system development. This helps to ensure that there is a level of expertise and experience because of this centralization of knowledge into a single person. Additionally, Class Owners help to make sure that all components of code are well maintained and taken care of, reducing the likelihood of neglect and any issues slipping through the cracks.