AGILE DEVELOPMENT TECHNIQUE

Feature-Driven Development

Feature-Driven Development (*FDD*) is another iterative and incremental development model. Unlike most of the other agile models that use risk, customer value, waste, or some other theme to guide development, FDD focuses on application features. At a high level, an FDD project builds a list of wanted features and then iteratively adds those features to the application until every feature has been added.

FDD Phases

FDD projects move through five phases. The first three occur at the start of the project (during iteration 0, if you like that term) and the last two phases are repeated iteratively until the application is complete.

The following sections describe the five FDD phases:

- **Develop a Model**

When the project starts, the team builds an object model for the application. To some, this may smell suspiciously like big design upfront. In FDD, however, the model is built quickly and iteratively with the assistance of *domain experts.*

The model's goal is to give the customers and the development team a common vision of the application's scope and goals. It should help everyone understand the domain's key concepts, interactions with other systems, potential problems, and ways the application will be used. Then provide the detail needed to actually implement the project's features.

- **Build a Feature List**

The next step is to build a list of the features that make up the application. FDD technically defines a feature as an *action/result/object* triple where the *action* generates the *result* related to the *object*. For example, a feature might be "calculate the customer's outstanding balance." Here the action is "calculate," the result is "outstanding balance," and the object is "the customer."

To help with large projects, FDD organizes the feature into a three-level hierarchy with the levels corresponding to areas, activities, and features. The top-level groups the features by domain area. The areas are divided into activities that represent things the user might need to do. The activities are divided into features. These are the atomic operations that make up the activities expressed in FDD's action/result/object style. The resulting hierarchy is called the *feature list* .

- **Plan by Feature**

During this phase, the planning team prioritizes the features and builds an initial schedule.

Because FDD uses class ownership, the features assigned to a team also assign the classes that provide those features to the team. For example, if you assign a bunch of customer-related features to a team, then you'll probably want to assign the Customer class to that team. In that case, it also makes sense to assign the other Customer class features to the team. In that way the object model helps group the features.

- **Design by Feature**

In this phase, the chief programmer selects a collection of features to be implemented in the next two-week iteration. For each selected feature, the chief programmer gathers the owners of the classes that will be involved into a feature team. Next, the feature team creates sequence diagrams representing the feature.

The class owners then write method prologues. A method prologue is a description of a method that includes its purpose, input and output parameters, return type, possible exceptions (ways the method can fail), and assumptions. During the design phase, the feature team hopefully adds new details to the project's classes. The chief programmer updates the project's object model to show the new detail. Finally, the chief programmer holds a design inspection to make sure the new feature design didn't omit anything.

When this phase is finished, the result is a design package that includes:

- A description of the package
- Sequence diagrams showing how the features will work
- Alternatives (if any)
- Updated object models
- Method prologues

The result is sort of like the start of a big design upfront project, but on a smaller scale.

- **Build by Feature**

With the completed design packages, the class owners build the methods that implement the iteration's selected features. The new code is

unit tested and run through a code inspection (held by the chief programmer). If everything looks good, the code is promoted to the project's build.

The following figure shows the relationships among the FDD project's five phases at a high level.

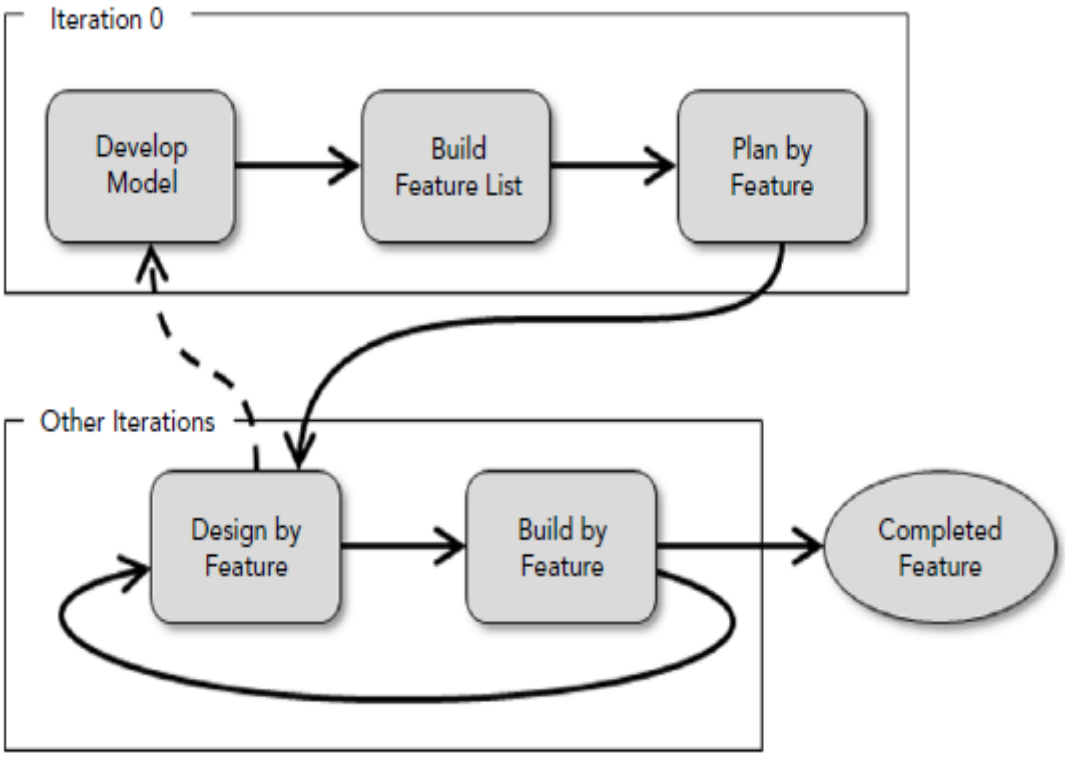

Figure: In FDD, the last two phases repeat for each feature iteration. The Design by Feature phase feeds changes back to the object model.

FDD Iteration Milestones

In an FDD project, the features must be implemented in no more than two weeks, so they are necessarily fairly small. Although the iterations are quick, they include a fair number of tasks. To keep track of everything that's going on during an iteration, FDD defines six milestones.

The following table shows the six milestones and the completion percentage each represents.

Table: FDD Milestones

PHASE	MILESTONE	PERCENTAGE
Design by Feature	Domain Walkthrough	1%
	Design	40%
	Design Inspection	3%
Build by Feature	Code	45%
	Code Inspection	10%
	Promote to Build	1%

For example, after the domain walkthrough (which is optional), design, and design inspection, an iteration is considered 1 + 40 + 3 = 44% complete.

Notice that the percentages give almost one-half of the iteration's credit to the Design by Feature phase. This reflects the importance of design, particularly for large projects with developers of varying skills. As mentioned earlier, the chief programmer (who is more experienced) uses the design phase to set up the team members for success during the build phase.