Universidad ORT Uruguay

Facultad de Ingeniería

–Metodología FDD–

Cátedra de Ingeniería de Software.

Docente Responsable: Gastón Mousques.

Autor:
Luis Calabria – 122919

2003

Índice General

Abstract

FDD con sus siglas en inglés Feature Driven Development es un enfoque ágil para el desarrollo de sistemas. Dicho enfoque no hace énfasis en la obtención de los requerimientos sino en como se realizan las fases de diseño y construcción. Sin embargo, fue diseñado para trabajar con otras actividades de desarrollo de software y no requiere la utilización de ningún modelo de proceso específico. Además, hace énfasis en aspectos de calidad durante todo el proceso e incluye un monitoreo permanente del avance del proyecto. Al contrario de otras metodologías, FDD afirma ser conveniente para el desarrollo de sistemas críticos.

Consiste en cinco fases secuenciales que son:

1- **Develop an Overall Model**: Los expertos del dominio presentan un walkthrough inicial de alto nivel sobre el alcance del sistema y su contexto. A continuación los expertos del dominio y los desarrolladores construyen el esqueleto de un primer modelo del sistema bajo la tutela del Chief Arquitect. Luego, el dominio es dividido en distintas áreas y a cada subgrupo se le asigna un área de dominio a desarrollar. Una vez finalizada cada área, el grupo se reúne para realizar un modelo global en base a todas las alternativas.

2- **Build a Feature List**: El equipo identifica las features[1], las agrupa, las prioriza y las pondera. En iteraciones subsecuentes del proceso, el equipo se divide en subgrupos que se especializan en áreas relacionadas a las features.

3- **Plan by Feature**: En base a la features list[2] de la etapa anterior, el Project Manager, el Development Manager y el Chief Programmer establecen hitos y diseñan un cronograma de diseño y construcción.

4- **Design by Feature**: El Chief Programmer toma la próxima feature a ser diseñada, identifica las clases involucradas y contacta al Class Owner correspondiente. Luego el equipo, trabaja en la realización del diagrama de secuencia correspondiente. El Class Owner hace una descripción de la clase y sus métodos.

5- **Build by Feature**: En esta etapa cada Class Owner construye los métodos de clase para cada feature correspondiente y luego realiza el testing unitario para cada una de las clases. A su vez, se realiza una inspección del código y luego que el código fue implementado e inspeccionado, el Class Owner realiza un check-in al CMS (Configuration Management System)[3]. Luego se realiza un main build en el cual se hace la integración con la funcionalidad antes realizada. También se realizan los testing de integración correspondiente.

En conclusión FDD:

- ✓ Es un proceso que ayuda al equipo a producir resultados periódicos y tangibles.
- ✓ Esta metodología utiliza pequeños bloques que contienen la funcionalidad del sistema, llamados features.
- ✓ Organiza los bloques que están relacionados entre sí, en una lista llamada feature set.
- ✓ Hace énfasis en la obtención de resultados cada dos semanas.
- ✓ Incluye estrategias de planificación que hacen que las features puedan desarrollarse en dichos lapsos.

[1] ***Feature***: Son pequeñas funcionalidades que el cliente quiere.
[2] ***Feature List***: Lista que agrupa toda la funcionalidad del sistema
[3] ***CMS***: Es un repositorio en el cual se almacena toda la información del proyecto como ser documentación, código fuente, etc.

La filosofía de FDD

A pesar de los muchos avances en el desarrollo de software, no es muy común para los proyectos de los últimos años el uso de function-driven process[1]. No obstante, muchos proyectos de software exceden el presupuesto, fallan con el programa y entregan menos de lo deseado. Como si fuera poco, el crecimiento de la tecnología hace que los proyectos más modernos sean poco exitosos ya que los cambios tecnológicos se realizan con tanta rapidez que es imposible poder planificar a largo plazo.

Lo normal para los proyectos con iteraciones cortas (fast-cycle-time projects) es un feature-driven iterative process, comenzando por los requerimientos y el modelado, siguiendo por el diseño y construcción en base a incrementos. En este trabajo formalizaremos el proceso llamado "Feature-Driven Development" (FDD).

Esta metodología propone tener etapas de cierre cada dos semanas, lo cual implica que los desarrolladores tendrán nuevas actividades que realizar en dicho período de tiempo. Esto hace que la motivación del equipo se mantenga durante todo el proyecto debido a que se ven los resultados periódicamente.

A los gerentes les fascina tener resultados a la vista cada cierto período de tiempo ya que reducen el riesgo del proyecto por tener entregas periódicas y tangibles. Con FDD, se obtienen porcentajes reales del proceso, lo cual ayuda a demostrar al cliente donde se encuentra el proyecto.

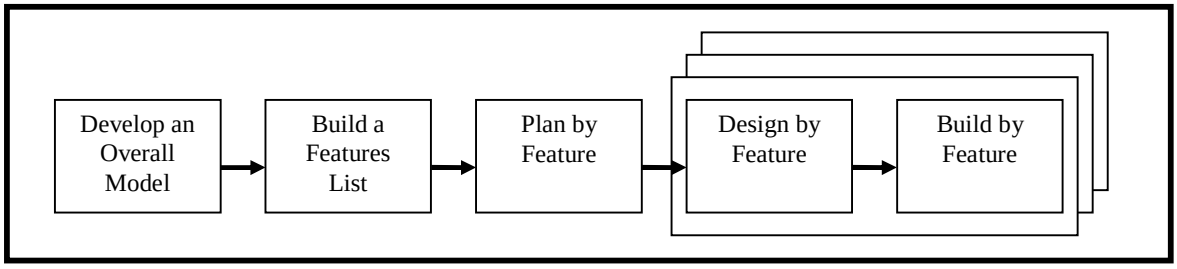
A los clientes también les gusta esta idea ya que pueden tener planes con entregas y resultados frecuentes. Además, los clientes saben cuan lejos está el proyecto de su finalización.

Con FDD se pretende dejar satisfechos a los desarrolladores, gerentes y clientes sin afectar al proyecto.

[1] Function-driven process refiere a un proceso guiado por las funciones ya que el desarrollo se hace en base a los casos de uso. Se utiliza para priorizar los requerimientos y las clases que sirven para realizar los casos de uso.

El Proceso

Dicho proceso consiste de cinco fases secuenciales durante las cuales el diseño y la construcción del sistema se llevan a cabo.



La parte iterativa del proceso de FDD (Diseño y Construcción) soporta un desarrollo ágil, con adaptaciones rápidas para cambios de último momento en los requerimientos.

1- Develop an Overall Model

Cuando esta fase comienza, los expertos del dominio ya tienen una idea del contexto y los requerimientos del sistema. Es probable que el documento de especificación de requerimientos ya exista. Sin embargo, FDD no hace énfasis en la recolección y administración de los requerimientos. Los expertos del dominio presentan un informe llamado walkthrough en el cual los miembros del equipo y el Chief Arquitect son informados a través de una descripción de alto nivel del sistema.

El dominio global es dividido en diferentes áreas y se realiza un walkthrough detallado para cada una de dichas áreas por parte de los expertos del dominio. Luego de cada walkthrough, un grupo de desarrolladores realizan un modelo de objetos para el área del dominio. Además, el equipo de desarrollo discute y decide cual es el modelo de objetos más apropiado para cada área del dominio. Simultáneamente, el modelo global del sistema es construido.

2- Build a Features List

Los walkthroughs, el modelo de objetos y los requerimientos existentes ofrecen una buena base para construir una features list que resuma la funcionalidad del sistema a ser desarrollado. En dicha lista, el equipo de desarrolladores presenta cada una de las funcionalidades evaluadas por el cliente. Las funcionalidades son presentadas por cada área del dominio y éstas forman un Major List Sets. Dicha lista es divida en subconjuntos en base a la funcionalidad. Estas representan diferentes actividades con un área específica del dominio. La features list es revisada por los usuarios y sponsors del sistema para su validación y aprobación.

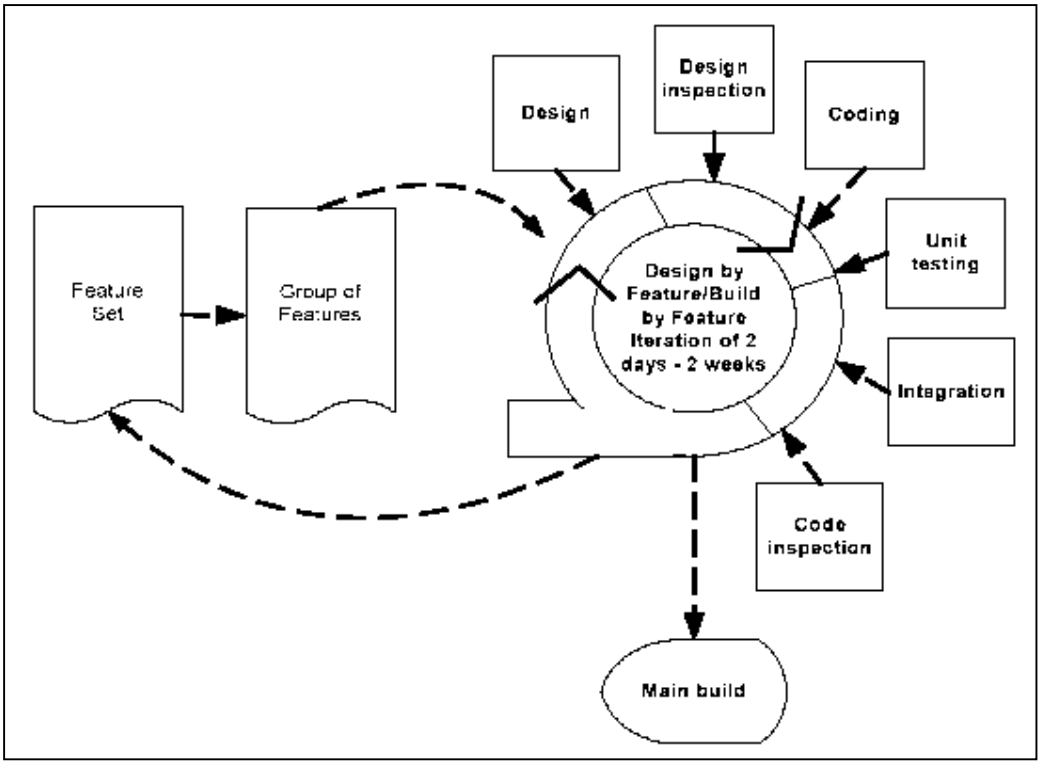
3- Plan by Feature

En esta etapa se incluye la creación de un plan de alto nivel, en el cual la features list es ordenada en base a la prioridad y a la dependencia entre cada feature. Además, las clases identificadas en la primera etapa son asignadas a cada programador.

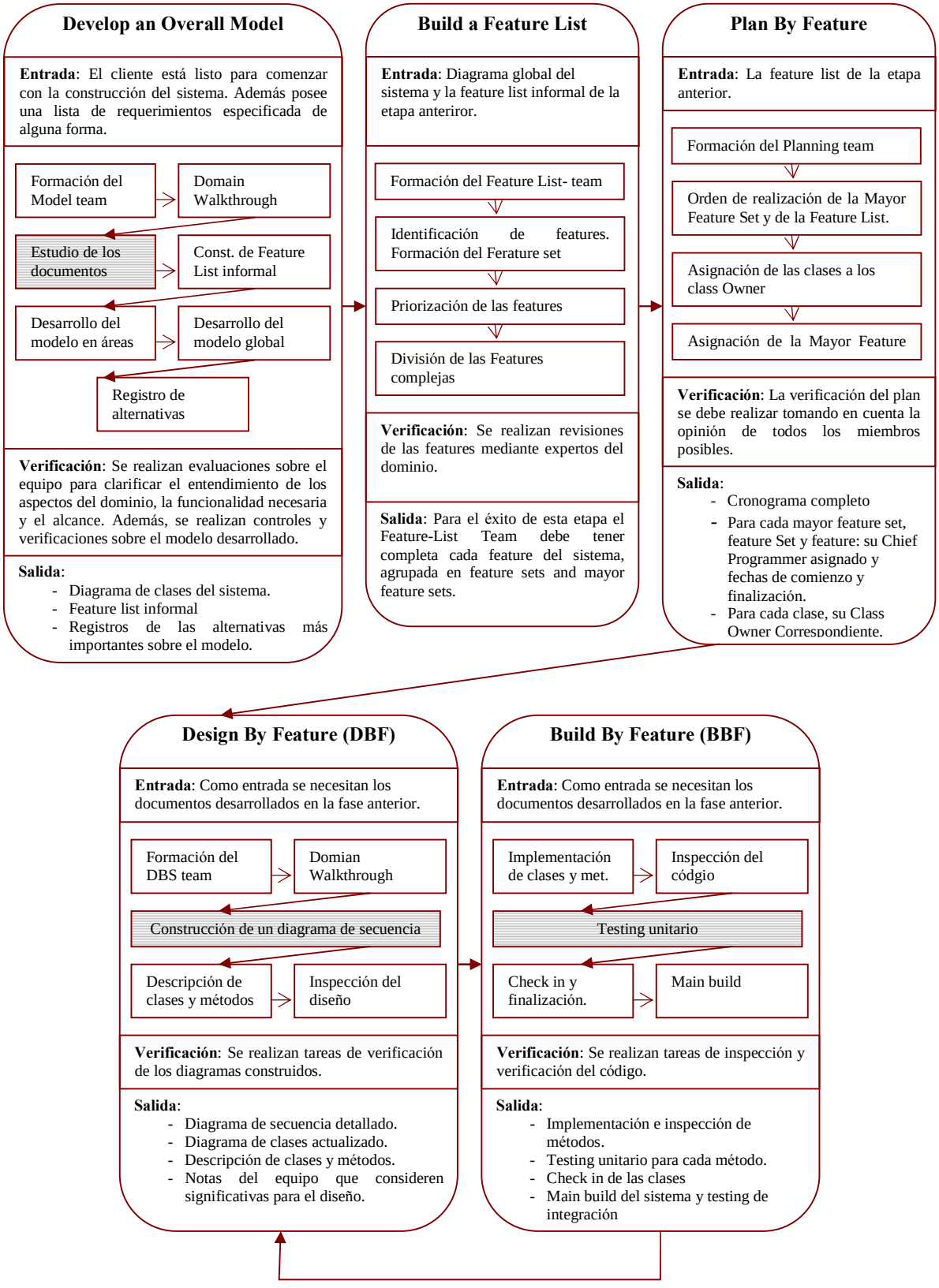
4 y 5- Design and Build by Feature

Un conjunto de features es seleccionada de la features list.
El diseño y construcción de la funcionalidad es un proceso iterativo durante el cual las features seleccionadas son producidas. Una iteración puede llevar desde unos pocos días a un máximo de dos semanas. Este proceso iterativo incluye tareas como inspección del diseño, codificación, testing unitario, integración e inspección del código. Luego que la iteración llega a su fin se realiza un main build de la funcionalidad en el cual se integra la funcionalidad. Dicho main build se realiza mientras la siguiente iteración comienza.

La siguiente imagen detalla como se desarrollan las iteraciones 4 y 5.



Resumen del proceso



Nota: Las tareas marcadas con gris son opcionales.

Develop an Overall Model

Resumen: Los expertos del dominio presentan un walkthrough inicial de alto nivel sobre el alcance del sistema y su contexto. A continuación los expertos del dominio y los desarrolladores construyen el esqueleto de un primer modelo del sistema bajo la tutela del Chief Arquitect. Luego el dominio es dividido en distintas áreas y a cada subgrupo se le asigna un área de dominio a desarrollar. Una vez finalizada cada área, el grupo se reúne para realizar un modelo global en base a todas las alternativas.

Entrada: El cliente está listo para comenzar con la construcción del sistema. Además posee una lista de requerimientos especificada de alguna forma.

Tareas

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
Formación del model team	El model team es un equipo permanente formado por expertos del dominio y desarrolladores.	Project Manager	Requerida
Domain Walkthrough	Un experto del dominio realiza un pequeño tutorial del área a ser modelada.	Modeling Team	Requerida
Estudio de los documentos	El equipo investiga los documentos disponibles incluyendo si es necesario modelo de componentes, requerimientos funcionales, user guides, etc.	Modeling Team	Opcional
Construcción de una Feature List informal	El equipo construye una Features List antes de comenzar con la Fase 2 y especifica que documentos serán necesarios para dicha etapa.	Chief Arquitect, Chief Programmers	Requerida
Desarrollo del modelo en áreas	El dominio global es dividido en diferentes áreas. Cada subgrupo construye un diagrama para el área de dominio especificada, bajo ciertas consideraciones del Chief Arquitect, haciendo énfasis en las clases y asociaciones, luego en los métodos y finalmente en los atributos. Los subgrupos agregan métodos en base a lo extraído de los walkthrough y de la feature list. Los subgrupos también realizan uno o más diagramas de secuencias informales.	Modeling Team divido en pequeños grupos	Requerida
Desarrollo del Modelo Global	Cada subgrupo presenta su propuesta para el área de dominio especificada. El Chief Architect puede proponer una alternativa adicional si así lo desea. El Model Team selecciona una propuesta como base y se va mejorando con las demás. También se realiza un diagrama de secuencia informal de dicho modelo.	Chief Arquitect, Modeling Team	Requerida
Registro de alternativas	Un miembro del equipo es asignado para registrar las alternativas más importantes al modelo que el equipo evaluó para referencias futuras en el proyecto.	Chief Arquitect, Chief Programmers	Requerida

Verificación: Se realizan evaluaciones sobre el equipo para clarificar el entendimiento de los aspectos del dominio, la funcionalidad necesaria y el alcance. Además, se realizan controles y verificaciones sobre el modelo desarrollado.

Salida: Al final de la etapa, el equipo debe obtener los siguientes resultados, sujetos a revisiones y a la aprobación del Developer Manager y el Chief Arquitect:

- ✓ Diagrama de clases del sistema.
- ✓ Feature list informal
- ✓ Registros de las alternativas más importantes sobre el modelo.

Build a Features List

Resumen: El equipo identifica las features, las agrupa, las prioriza y las pondera. En iteraciones subsecuentes del proceso, el equipo se divide en subgrupos que se especializan en áreas relacionadas a las features.

Entrada: Diagrama global del sistema y feature list informal de la etapa anterior.

Tareas.

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
Formación del Feature-List Team	Consiste en un grupo permanente formado por expertos del dominio y desarrolladores.	Project Manager, Development manager	Requerida
Identificación de features. Formación del Feature Set	El equipo comienza con la features list informal creada en la etapa anterior. Luego de esto: -Transforman los métodos del modelo a features. -Transforman funcionalidad común en feature sets. -Discuten que otras features pueden ser agregadas para satisfacer las necesidades del cliente.	Feature-List team	Requerida
Priorización de las features	Un subgrupo del equipo establece las prioridades de las features: A – (debe estar) B – (deseable que esté) C – (deseable si se puede) D – (deseable a futuro) El equipo considera cada feature en base a la aprobación del cliente (si se incluye la feature) y la desaprobación del cliente (si no es incluida)	Feature-List team	Requerida
División de las features complejas	Los desarrolladores liderados por el Chief Architect discuten sobre aquellas que pueden llegar a tomar más de dos semanas en ser desarrolladas. El equipo divide esas features en otras más pequeñas.	Feature-List team	Requerida

Verificación: Se realizan revisiones de las features mediante los expertos de dominio.

Salida: Para el éxito de esta etapa el Feature-List Team debe tener completa cada feature del sistema, agrupada en feature sets and mayor feature sets.

Plan By Feature

Resumen: En base a las features list de la etapa anterior, el Project Manager, el Development Manager y el Chief Programmer establecen hitos y diseñan un cronograma de diseño y construcción.

Entrada: La feature list de la etapa anterior.

Tareas.

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
Formación del Planning Team	Está formado por el Project Manager, el Development Manager y el Chief Programmer.	Project Manager	Requerida
Orden de realización de la Mayor Feature Set y de las feature lists.	El Planning Team determina en que orden serán desarrolladas las features y se establecen los plazos de inicio y de fin.	Planning Team	Requerida
Asignación de las clases a los Class Owners	Usando el orden de desarrollo de las features y la ponderación de las mismas, el Planning Team asigna clases a los distintos Class Owners.	Planning Team	Requerida
Asignación de las Mayor Feature Sets y features a los Chief Programmers	Usando el mismo criterio que antes el Planning Team asigna las features a los Chief Programmers.	Planning Team	Requerida

Verificación: Se realiza una verificación del plan tomando en cuenta la opinión de todos los miembros posibles del equipo.

Salida: El Planning Team debe producir un plan de desarrollo, sujeto a la revisión y aprobación del Development Manager y del Chief Architect. Se debe entregar:

- ✓ Una fecha global de terminación
- ✓ Para cada mayor feature set, feature Set y feature: su Chief Programmer asignado y fechas de comienzo y finalización.
- ✓ Para cada clase, su Class Owner Correspondiente.

Design By Feature (DBF)

Resumen: El Chief Programmer toma la próxima feature a ser diseñada, identifica las clases involucradas y contacta los Class Owner correspondientes. Luego el equipo, trabaja en la realización del diagrama de secuencia correspondiente. El Class Owner hace una descripción de la clase y sus métodos.

Entrada: Como entrada se necesitan los documentos desarrollados fase anterior.

Tareas.

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
Formación del DBS Team	El Chief Programmer identifica las clases involucradas en el diseño de la feature. Luego contacta a los Class Owners correspondientes y también a los expertos del dominio, si es necesario.	Chief Programmer	Requerida
Domain Walkthrough	Los expertos del dominio entregan un overview del domino referida a la feature en consideración.	Feature Team, Expertos del dominio	Opcional
Construcción de un diagrama de secuencia	Aplicando sus conocimientos de las features, el Feature Team construye un diagrama de secuencia detallado para la feature. Luego el Chief Programmer añade el diagrama de secuencia (con su correspondiente diagrama de clases al Modelo Global).	Feature Team	Requerida
Descripción de clases y métodos	Cada Class Owner actualiza la descripción de sus clases correspondiente en base al diagrama de secuencia. El incluye tipos de parámetros, tipos de retorno, mensajes envidados, etc.	Class Owner	Requerida
Inspección del diseño	El Feature Team realiza una inspección del diseño. El Chief Programmer invita a expertos fuera del equipo para que participen, cuando siente que la complejidad lo requiere.	Feature Team	Requerida

Verificación: Se realizan tareas de verificación de los diagramas construidos.

Salida: Para el éxito de esta fase el DBS team debe obtener los siguientes resultados:

- ✓ Diagrama de secuencia detallado
- ✓ Diagrama de clases actualizado
- ✓ Descripción de clases y métodos
- ✓ Notas del equipo que consideren significativas para el diseño.

Build By Feature (BBF)

Resumen: En esta etapa cada Class Owner construye los métodos de las clases para cada feature correspondiente y luego realiza el testing unitario para cada clase. El Feature Team realiza una inspección del código antes del testing unitario. Luego que el código fue implementado e inspeccionado, el Class Owner realiza un check-in al CMS[1] (Configuration Management System). Luego se realiza un main build en el cual se hace la integración con la funcionalidad antes realizada. También se realizan los testing de integración correspondiente.

Entrada:
- ✓ Diagrama de secuencia detallado
- ✓ Diagrama de clases actualizado
- ✓ Descripción de clases y métodos
- ✓ Notas del equipo que consideren significativas para el diseño.

Tareas.

Nombre de la tarea	Descripción	Responsables	Requerida/Opcional
Implementación de clases y métodos	Cada Class Owner implementa los métodos de sus correspondientes clases en base al diagrama de secuencia. Además realiza testing de los métodos.	Feature Team	Requerida
Inspección del código	El Chief Programmer establece en el cronograma una inspección del código que puede realizar antes o después del testing unitario. El Feature Team conduce dicha inspección y si lo requiere puede pedir la ayuda a expertos externos al equipo.	Feature Team	Requerida
Testing unitario	Cada Class Owner realizar un testing unitario de las clases asignadas. El Chief Programmer ayuda al Class Owner sobre los datos de prueba que deben utilizarse.	Feature Team	Requerida
Check in y finalización de la iteración	Una vez que el código fue implementado, inspeccionado y testeado, cada Class Ower realizar un check-in al CMS. Cuando todas las clases están ingresadas el Chief Programmer informar acerca de la finalización.	Feature Team	Requerida
Main Build	Luego que la iteración llega a su fin se realiza un main build de la funcionalidad en el cual se integra la funcionalidad. Dicho main build se realiza mientras la siguiente iteración comienza.	Feature Team	Requerida

Verificación: Se realizan tareas de inspección y verificación del código.

Salida: Para el éxito del proceso, el Feature team debe obtener los siguientes resultados, sujetos a la revisión y la aprobación del Chief Programmer:
- ✓ Implementación e inspección de métodos.
- ✓ Testing unitario para cada método.
- ✓ Check-in de las clases.
- ✓ Main build del sistema y testing de integración.

[1] Es un repositorio en el cual se almacena toda la información del proyecto como ser documentación, código fuente, etc.

Reportes de avance de las tareas

El Release Manager se reúne con los Chief Programmers para que éstos reporten como es el avance de las tareas. En esta reunión, que tiene una duración de 30 minutos o menos, cada Chief Programmer informa de manera verbal el avance de sus features. Hacer esto de forma verbal es bueno para que cada uno de los Chief Programmers se tome el tiempo de escuchar a los otros y saber donde están situados los demás en el proceso de desarrollo.
Al final de la entrevista, el release manager anota los resultados, actualiza la base de datos y genera los reportes.

El release manager reporta los resultados obtenidos semanalmente, tanto para el equipo, para el cliente y para el Project Manager. Para los clientes y los gerentes, el reporte debe incluir el porcentaje de avance de cada feature. Para el equipo se informa cual es el desempeño del mismo.

Roles y Responsabilidades.

El FDD clasifica sus roles en las siguientes tres categorías:

- ✓ **Key roles**
  - Project manager
  - Chief architect
  - Development manager
  - Chief programmer
  - Class owner
  - Expertos del dominio
- ✓ **Supporting roles**
  - Realese manager
  - Languaje lawyer/language guru
  - Build engineer
  - Toolsmith
  - Sistem administrator
- ✓ **Additional roles**
  - Testers
  - Deployers
  - Techical writers

Vale la pena aclarar que un miembro de un equipo puede ejercer varios roles y un rol pude ser compartido por varias personas.

Project Manager
Es el líder administrativo y financiero del proyecto. Una de sus tareas principales es proteger al equipo de distracciones externas y permitir que el equipo de pueda trabajar en las condiciones apropiadas. En el FDD el Project Manager tiene la última palabra sobre temas referidos al alcance, tiempo y personal.

Chief Architect
Es el responsable por el diseño global del sistema y de la ejecución de todas las etapas del diseño. También tiene la última palabra sobre las decisiones del diseño de todo el sistema. Si es necesario, este rol puede ser dividido en dos roles como ser Domain Architect y Tecnical Architect.

Development Manager
Lleva diariamente las actividades de desarrollo y resuelve cualquier conflicto que pueda ocurrir con el equipo. Además, este rol incluye la responsabilidad de resolver problemas referentes a los recursos. Las tareas de este rol pueden ser combinadas con las del Chief Architect o el Proyet Manager.

Chief Programmer
Es un desarrollador con experiencia, el cual participa en el análisis de los requerimientos y el diseño del proyecto. El Chief Programmer es el responsable de guiar a pequeños equipos en el análisis, diseño y desarrollo de nuevas funcionalidades. Además, selecciona las funcionalidades a desarrollar de la lista de funcionalidades de la próxima iteración en la última fase del FDD, identifica las clases y el Class Owner que se necesita en el equipo para la iteración. Con la ayuda de otros Chief Programmers resuelve problemas técnicos y de recursos, y reporta el progreso del equipo durante la semana.

Class Owner
Trabaja bajo la guía del Chief Programmer en las tareas de diseño, codificación, testing y documentación. Él es responsable del desarrollo de las clases que se le asignaron como propias. Para cada iteración los Class Owner participan en la decisión de que clase será incluida en la lista de funcionalidades de la próxima iteración.

Expertos del dominio
Los expertos del dominio pueden ser un usuario, un cliente, un sponsor, un analista del negocio o una mezcla de estos. Su tarea es poseer el conocimiento de los diferentes requerimientos del sistema. El experto del dominio pasa el conocimiento a los desarrolladores de manera tal que asegure que estos entreguen un sistema completo.

Domain Manager
Lidera al grupo de expertos del dominio y resuelve sus diferencias de opinión concernientes a los requerimientos del sistema.

Realese Manager
Controla el avance del proceso mediante la revisión de los reportes del Chief Programmer y realiza pequeñas reuniones con él. Además, reporta el progreso del proyecto al gerente.

Language Lawyer/Language Guru
Es un miembro del equipo responsable de poseer un vasto conocimiento en, por ejemplo, un específico lenguaje de programación o tecnología. Este rol es particularmente importante cuando el equipo trabaja con una nueva tecnología.

Build Engineer
Es la persona responsable de preparar, mantener y correr el proceso de construcción, incluidas las tareas de mantenimiento de las versiones y la publicación de la documentación.

Toolsmith
Es un rol para la construcción de herramientas específicas para el desarrollo, conversión de datos y testeo. Además, puede trabajar en la preparación y mantenimiento tanto de bases de datos o sitios web destinados al proyecto.

System Administrator
La tarea del administrador del sistema es configurar, administrar y reparar los servidores, estaciones de trabajo y equipos de desarrollo y testeo utilizados por el equipo.

Tester
Verifica que el sistema recién creado cumpla con los requerimientos del cliente. Puede llegar a ser una persona independiente del equipo del proyecto.

Deployer
Es el encargado de convertir la información existente requerida por el nuevo sistema y participa en el lanzamiento de los nuevos productos.

Technical Writer
Se encarga de preparar la documentación para los usuarios, quienes pueden formar parte o no del equipo del proyecto.

Conclusiones

- FDD es un proceso que ayuda al equipo a producir resultados periódicos y tangibles.
- Esta metodología utiliza pequeños bloques que contienen la funcionalidad del sistema, llamados features.
- Organiza los bloques que están relacionados entre sí, en una lista llamada feature set.
- Hace énfasis en la obtención de resultados cada dos semanas.
- Incluye estrategias de planificación que hacen que las features puedan desarrollarse en dichos lapsos.

Palabras Claves

- ✓ Build a Feature List
- ✓ Build engineer
- ✓ Chief architect
- ✓ Chief programmer
- ✓ Class owner
- ✓ CMS (Configuration Management System)
- ✓ Deployers
- ✓ Design and Build by Feature
- ✓ Develop an Overall Model
- ✓ Development manager
- ✓ Expertos del dominio
- ✓ Feature
- ✓ Feature List
- ✓ Feature Set
- ✓ Language lawyer/language guru
- ✓ Main Build
- ✓ Mayor feature set
- ✓ Model Team
- ✓ Plan by Feature
- ✓ Project manager
- ✓ Realese manager
- ✓ Sistem administrador
- ✓ Techical writers
- ✓ Testers
- ✓ Toolsmith
- ✓ Walkthrough

Glosario

- ✓ ***CMS***: Es un repositorio en el cual se almacena toda la información del proyecto como ser documentación, código fuente, etc.
- ✓ ***Feature***: Son pequeñas funcionalidades que el cliente quiere.
- ✓ ***Feature List***: Lista que agrupa toda la funcionalidad del sistema
- ✓ ***Feature Sets***: Es una lista de features agrupadas por la misma funcionalidad.
- ✓ ***Mayor Feature Sets***: Son una lista de feature agrupada por cada área del dominio. Puede estar compuesta por varias feature Sets.
- ✓ ***Walkthrough***: Es un documento que especifica a grandes rasgos cual es el alcance del sistema y la funcionalidad básica del mismo. Sirve entre otras cosas para informar a los integrantes del equipo cual es el objetivo del sistema.

Sources

Artículos

"Agile software development methods. Review and analysis" – Pekka Abrahamsson, Outi Salo & Jussi Ronkainen. Editado en 2002. www.info.vtt.fi/pdf/

Libros

"JAVA Modelling in Color with UML. Enterprise Components and process" – Peter Coad, Eric Lefebvre, Jeff De Luca. 1ra Edición 1999. www.oi.com

Sitios Web

www.featuredrivendevelopmente.com