FEAUTURE DRIVEN DEVELOPMENT

Nama Kelompok :

1. **Azhar Aulia Rahman** **15.12.8701**
2. **Yohanes Dheni Setyawan** **15.12.8711**
3. **Suratno Widhiatmoko** **15.12.8716**
4. **Ryan Gallant Purnomo** **15.12.8741**
5. **Ikhwan Noorsetyo** **15.12.8766**

UNIVERSITAS
AMIKOM YOGYAKARTA
2017/2018

FEAUTURE DRIVEN DEVELOPMENT

Abstract - **Teknologi terus berkembang dan pengguna teknologi semakin banyak sehingga memaksa industri perangkat lunak untuk mengembangkan teknologi baru. Beberapa dekade terakhir, proses development software menghadapi beberapa masalah seperti keterbatasan waktu dan kebutuhan sistem yang terus berubah serta tidak bisa dikerjakan secara konvensional. Beberapa tahun terakhir terdapat pengembangan Agile yang mampu mengatasi permasalahan tersebut. FDD merupakan salah satu model development agile yang berfokus pada development dimana kebutuhan dan fiturnya diekstraksi dalam lingkup bahasa yang mudah dimengerti kliennya. Makalah ini membahas FDD dari sudut pandang kami yang juga masih belajar konsep FDD.**

I. PENDAHULUAN

FDD awalnya dirancang oleh Jeff De Luca, untuk memenuhi kebutuhan spesifik dari satu bulan 15-, 50-orang proyek pengembangan perangkat lunak pada sebuah bank besar di Singapura 1997. Jeff De Luca menyampaikan satu set lima proses yang menutupi pengembangan model secara keseluruhan dan daftar, perencanaan, desain dan bangunan fitur. Proses pertama sangat dipengaruhi oleh pendekatan Peter Coad untuk pemodelan obyek. Proses kedua menggabungkan ide-ide Peter Coad tentang menggunakan daftar fitur untuk mengelola kebutuhan fungsional dan tugas-tugas pembangunan. Proses lainnya dan campuran proses menjadi suatu kesatuan kohesif merupakan hasil dari pengalaman Jeff De Luca. Sejak penggunaan sukses pada proyek Singapura, ada beberapa implementasi dari FDD.

II. FEATURE-DRIVEN DEVELOPMENT

Menurut Palmer (2001), FDD adalah proses yang didesain dan dilaksanakan untuk menyajikan (deliver) hasil kerja secara berulang-ulang dalam waktu tertentu dan dapat diukur. FDD adalah pendekatan yang mengacu pembuatan sistem menggunakan metode yang mudah dimengerti dan mudah diimplentasikan; teknik problem solving; dan pelaporan yang mudah dimengerti dan dikontrol oleh stakeholders.

Pemrogram diberikan informasi yang cukup dan diberikan alat bantu yang baik untuk menyelesaikan aplikasi yang diberikan. Team leader dan manajer proyek diberikan informasi yang baik berdasar waktu mengenai tim dan proyek yang berjalan untuk menghindari resiko yang mungkin terjadi. Pelaporan menjadi lebih mudah, tidak membebani, periodik dan akurat.

Pengguna (sposor, end user, customer) secara langsung dapat melihat bagian mereka sebagai hasil progress proyek dan memberikan umpan balik semasih dalam tahap pengembangan. Dengan hal ini diharapkan proyek dapat menghasilkan sebuah sistem yang benar-benar diinginkan oleh klien.

a. Nilai-nilai Utama

Menurut Calberg, nilai-nilai utama yang terkandung dalam FDD sehingga FDD mampu memberikan nilai lebih bagi proses pengembangan software adalah:

- Tangible results rather than Process Pride
  Proses dalam FDD lebih mengutamakan memberikan nilai-nilai yang dapat diukur daripada sederet proses yang rumit dan menghabiskan banyak tenaga dan sumber daya.

- A system for building system is necessary
  Sangat penting untuk membentuk sebuah sistem yang solid dan rapi untuk membuat sistem (software) bekerja sesuai dengan yang diharapkan.

- Simple is better
  Desain yang dibuat harus sesederhana mungkin namun mampu menggali semua requirement yang disyaratkan oleh klien.

- Process steps should be obviously valuable to each team member

- Good processes move to the background

b. 6 Pemeran Utama

Menurut Palmer dan Flesing (2001), Calberg, Abrahamsson dan Juhani (2002) setiap proses dalam FDD melalui orang-orang di dalamnya dan kelebihan serta kekurangan setiap orang sangat berpengaruh pada keluaran proyek, terdapat 6 pemeran utama proses FDD yaitu:

- **Project Manager** berperan sebagai pemimpin administratif terhadap budget, orang-per-orang dan laporan pencapaian, mengoperasikan sistem proyek serta melindungi pekerja dari gangguan luar.

- **Chief Architect** berperan sebagai penanggung jawab desain sistem secara keseluruhan, menjalankan workshop desain, dan mengarahkan proyek terkait kendala teknis.

- **DevelopmentManager** berperan sebagai pemimpin pengembangan dari hari-ke-hari, mengatasi konflik sumberdaya, biasanya juga dikombinasikan dengan Project Manager dan Chief Architect.
- **Chief Programmer** adalah developer yang berpengalaman memimpin grup kecil dari sekumpulan developer individual.

- **Class Owner** adalah developer individual yang mendesain, mengkodekan dan menguji dan mendokumentasikan fitur.

- **Domain Expert** yaitu user, klien, sponsor dll. Dikenal baik oleh developer.

Selain enam pemeran utama tersebut diatas juga terdapat pemeran pembantu seperti domain manager, release manager, language guru, bild engineer, toolsmith, dan system administrator. Selain itu juga terkadang cukup membantu adalah tester, deployer, dan technical writer.

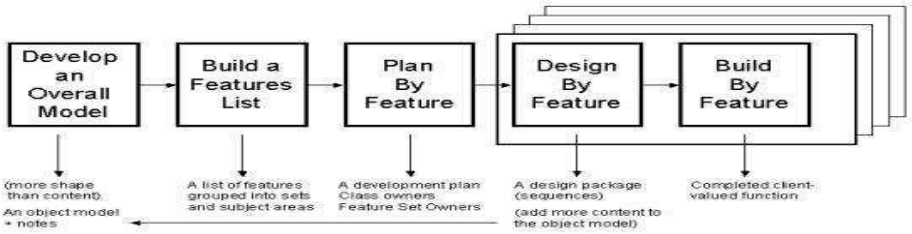
c. 5 Proses FDD

FDD terdiri dari 5 proses berurut selama mendesain dan mebangun sistem. Proses FDD yang iteratif dalam mendesain dan membangun (design and build) mendukung metode Agile dengan adaptasi yang cepat terhadap perubahan requirement dan kebutuhan bisnis. Kelima proses tersebut adalah:

· Build an Overall Model

Ketika fase ini dimulai, Domain Expert telah menyadari scope, konteks dan requirement dari sistem yang akan dibangun. Pembuatan dokumen requirement seperti use case atau spesifikasi fungsional ada dalam fase ini. Namun FDD tidak secara eksplisit menggali, mencari dan mengatur requirement ini. Domain expert menyajikan apa yang disebut " walkthrough" yang mana anggota tim dan Chief Architect diinformasikan dengan deskripsi level tinggi dari sistem.

Domain keseluruhan (overal domain) lebih lajut dibagi kedalam area domain yang berbeda sedangkan walkthrough yang lebih detail deberikan oleh anggota domain. Kemudian anggota tim



Gambar 3.1 Lima Proses FDD

developer bekerja dalam grup-grup kecil untuk mengerjakan project model dari domain area yang telah diterima.

· Build a Feature List

Walkthrough, object model dan dokumentasi requirement yang ada memberikan dasar yang kuat dalam membangun feature list yang komprehensif untuk sistem yang dikerjakan. Dalam daftar (list), tim menyajikan masing-masing **client valued functions** ke dalam sistem. Fungsi-fungsi tersebut dibagikan kepada masing-masing domain area dan masing-masing grup dari fungsi tersebut disebut sebagai major feature set. Sebagai tambahan, major feature sets kemudian dibagi lagi menjadi feature sets. Ini merepresentasikan aktifiti yang berbeda di setiap domain area. Feature list adalah yang dilihat oleh user atau sponsor untuk validitas dan kelengkapan mereka.

Feature dalam hal ini adalah langkah-langkah aktifitas bisnis, berbasis customer bukan teknologi. Bahasa yang digunakan mencakup bahasa yang dimengerti oleh cutomer. Nomenklatur untuk menunjukkannya terdiri atas:

<aksi> <hasil> <obyek>

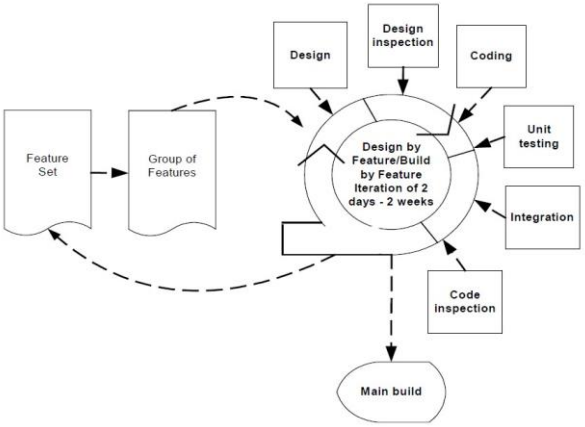
Misalnya:

<berikan> <nomor akun> untuk <anggota baru>. [5]

· Plan by Features

Plan by feature mencakup perencanaan pada level yang lebih tinggi, dimana feature set diatur

sedemikian rupa sesuai dengan prioritas dan hubungannya. Prioritas ditentukan sesuai dengan kebutuhan customer yang disetujui oleh Chief Programmer. Dalam fase ini, Project Manager, Development Manager dan Chief Programmer merencanakan urutan feature-feature yang akan dikerjakan dengan demikian class owenership telah dilengkapi.



Gambar 3.2 Proses Design by Feature dan Build by Feature[7]

- Design by Feature dan build by Feature

  fitur diambil dari feature set dan diperlukan feature team untuk membangun fitur terpilih disebut sebagai class owner. Proses design by feature dan build by feature bersifat iteratif selama fitur yang dipilih tersebut diproduksi. Satu kali iterasi memerlukan waktu beberapa hari. Proses iteratif ini mencakup beberapa tugas seperti inspeksi rancangan, pengkodean, pengujian unit, integrasi dan inspeksi kode seperti yang dijelaskan dalam gambar 3.2.

d. Project Tracking Methodology

   Project Tracking diperlukan untuk mengetahui sejauh mana proyek telah berjalan dan mengevaluasi strategi dan kemungkinan yang akan dijalankan terkait situasi itu

e. Karakteristik

   Menurut Calberg, penggunaan FDD sebaiknya digunakan jika; memperkerjakan 10 – 250 developer yang memiliki kemampuan teknis lebih dari rata-rata, dan jangan digunakan jika; jumlah tim kurang dari 10, tim sedang belajar menguasai pekerjaan dan jika kurang dukungan dari sistem. FDD lebih terhirarki daripada Extreme Programming, memiliki class owenership yang terpisah-pisah, sukses jika dalam rentang jumlah developer diatas rata-rata, klien tidak dilibatkan dalam seluruh urutan proses (hanya dalam proses 1, 2 dan 4) dan menghargai developer sebagai individu manusia yang menentukan sukses atau tidaknya pengembangan.

III.KELEMAHAN

Dalam sebuah metode tentu saja ada sebuah kekurangan. Kekurangan yang terdapat dalam metode FDD adalah sebagai berikut :

1. FDD memerlukan jumlah pekerja/personel yang banyak, yang dipecah-pecah ke dalam masing masing bagian. Jika dilihat dari segi biaya, semakin banyak pekerja, maka semakin banyak pula biaya yang dikeluarkan.
2. Apabila seorang Class owners melakukan perubahan pada feature yang ditanganinya, anggap sebagai A. Dan ternyata perubahan tersebut berpengaruh pada feature B, yang dikerjakan oleh Class owners lainnya. Hal ini akan berdampak buruk apabila ternyata feature B memerlukan feature A. Dan pada akhirnya pengerjaan feature B harus menunggu kepastian feature A terselesaikan. Jelas ini akan berdampak pada mundurnya jadual kerja dari project.
3. Karena jumlah jam kerja dari FDD yang tidak terikat, maka kemungkinan pada saat di tengahtengah jadual, para developer bekerja tidak optimal dan di akhir jadual akan bekerja extra keras.

V. KESIMPULAN

Metode FDD memiliki proses dengan tingkat hirarkis yang tinggi sehingga setiap proses merupakan rangkaian tugas yang baku dan klien hanya berperan dalam sebagian proses saja tidak dalam keseluruhan proses. Fleksibilitas dan kemampuannya menghadapi perubahan masih bisa dilakukan walaupun melalui proses iteratif yang panjang karena melalui beberapa prosedur sampai feature diberikan ke klien.

Referensi :

1. Scott Ambler, ”Survey Says: Agile Works in Practice ” , 2006.

2. Philadelphia SPIN, “Agile Methode a Practical Persp ective” (Software Process Environtment Network), 2006

3. David J. Anderson, “Feature-Driven Development: Tow ard a TOC, Lean and Sigma Solution for Software Engineering”, Microsoft Corporation, 2004.

4. Stephen R. Palmer dan John M. Felsing, Practical Guide to Feature-Driven Development, Prentice Hall, 2001.

5. Reid S. Calberg, Featur Driven Development, SE470, www.fivesticks.com/intoo/fdd

6. Pekka Abrahamsson and Juhani Warsta , Agile Software Development Methods Review and Analysis, Julkaisija-Utgivare-Publisher, 2002.

7. Amith Pulla, Feature Driven Development (FDD), a presentation, NJIT.