REDESIGN DAN REIMPLEMENTATION MY DATABOOK DENGAN DESIGN PATTERNS

Aris Setyawan[1], Dana Sulistyo Kusumo[2], -[3]

[1]Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

Abstrak

Maintenance aplikasi My DataBook (MDB) sangat sulit. Pengembangan aplikasi ini masih menggunakan metode struktural sehingga tidak dapat dipungkiri lagi bahwa desain ini masih kurang optimal dari beberapa sisi jika dibandingkan dengan metode berorientasi objek. Mendesain perangkat lunak dengan metode berorientasi objek sangat susah, dan mendesain perangkat lunak yang reusable dengan metode berorientasi objek lebih susah lagi. Harus dicari objek-objek yang saling berkaitan, memecahnya menjadi beberapa kelas dengan granulatity yang tepat, mendefinisikan interface kelas dan hirarki inheritance, dan menentukan relasi diantara kelas-kelas tersebut. Di suatu sisi desain yang dihasilkan harus spesifik pada permasalahan, tapi pada sisi lain harus cukup general untuk mengatasi kebutuhan di masa yang akan datang.
Pada Tugas Akhir (TA) ini akan dilakukan redesign dan reimplementation suatu perangkat lunak yang disebut dengan MDB dengan design pattern. Patternpattern yang digunakan meliputi Creational Pattern, Structural Pattern, dan Behavioral Pattern. Sedangkan Interpreter yang digunakan adalah PHP 5.
Kesimpulan yang diharapkan dari pembuatan TA ini antara lain adalah Menjadikan kode program MDB menjadi maintainable dan memiliki tingkat reusability dan readability yang tinggi sehingga mudah diubah sesuai dengan kebutuhan sistem.

Kata Kunci : Design patterns, reusability, readability

Abstract

Maintaining MDB application is very dificult. This application development still uses structural method, so its design is not optimal yet compared to objectoriented method. Designing software in object-oriented is very hard, and designing reusable object-oriented software is even harder. You must find pertinent objects, factor them into classes at the right granularity, define class interfaces and inheritance hierarchies, and establish key relationships among them. The design should be specific to the problem at hand but also general enough to address future problems and requirements.
This final project will redesign and reimplementation an application which called MDB using design patterns. The patterns which used are Creational Pattern, Structural Pattern, and Behavioral Pattern. And the interpreter is PHP 5's interpreter.
The conclusions expected from the final project are convert MDB's code maintainable and has high level reusability that easy to change to address future requirements.

Keywords : Design patterns, reusability, readability

1. Pendahuluan

1.1 Latar belakang

MDB adalah aplikasi berbasis web yang berfungsi untuk manajemen *appointment*, *task*, dan penjadwalan. Untuk mengatasi kemungkinan pengembangan system, MDB harus memiliki desain yang fleksibel, *reusable*, dan *extendable*.

Maka harus dipikirkan bagaimana sistem akan berubah. Yaitu desain yang tidak memerlukan *redesign* pada masa yang akan datang. Perubahan itu meliputi *redefinition, reimplementation* dan *retesting.* Intinya *redesign* yang memerlukan biaya yang mahal harus diminimalkan.

Dibawah ini adalah beberapa hal yang kurang optimal pada MDB yang memperbesar kemungkinan MDB akan didesain ulang pada masa yang akan datang.

1 *Readability* dari program rendah, sehingga sulit untuk dikembangkan secara estafet.
2 Implementasi MDB masih menggunakan metode struktural sehingga desainnya tidak modular antara data dan *behaviour*.
3 Sulit dirubah menjadi *web* yang lebih dinamik, karena beberapa implementasi masih statis. Hal ini meliputi susunan menu hirarki dan layout web.
4 Sulit untuk dievolusikan untuk memenuhi *requirement* yang kemungkinan besar akan muncul, seperti mendukung *user spcific task*/modul.
5 Tidak mendukung *customizable layout* sesuai dengan keinginan *user*.

1.2 Perumusan masalah

Masalah yang akan dibahas dalam TA ini adalah mendesain MDB dengan desain *pattern. Pattern- pattern* yang akan dipakai sebagian dari *pattern-pattern*:

1 ***Creational Pattern***. *Abstract Factory, Builder, Factory Method, Prototype,* dan *Singleton*
2 ***Structural Pattern***. *Adapter, Bridge, Composite, Decorator, Facade, Flyweight,* dan *Proxy*
3 ***Behavioral Pattern***. *Chain of Responsibility, Command, Interpreter***,** *Iterator, Mediator, Memento, Observer, State, Strategy, Template Method,* dan *Visitor*

1.3 Tujuan

Tujuan dari pembuatan TA ini adalah:

1 Menjadikan kode program MDB memiliki tingkat *reusability* yang tinggi sehingga mudah berevolusi sesuai dengan kebutuhan sistem
2 Menjadikan MDB kode program lebih *readable* sehingga dapat diprogram secara estafet
3 Menganalisis performansi setelah dilakukan desain ulang

1.4 Batasan masalah

1 Tidak membahas tentang analisis sistem atau dengan kata lain tidak menganalisis ulang sistem.
2 Evolusi yang dimaksudkan dibatasi pada mendukung *user spcific task*/modul dan mendukung *customizable layout* sesuai dengan keinginan *user*
3 Performansi yang akan dianalisis dibatasi pada waktu eksekusi aplikasi

1.5 Metodologi penyelesaian

Metodologi penyelesaian masalah yang digunakan dalam TA ini terdiri dari tahapan-tahapan berikut ini :

1 **Studi Literatur**. Pada tahap ini dilakukan pencarian dan pengumpulan literatur-literatur yang berhubungan dengan masalah yang sedang dikerjakan dari buku-buku referensi yang menunjang.
2 ***Analisis***. Tahap ini terdiri dari proses m*apping* dari kode program MDB ke *diagram* dan menganalisa kode program MDB untuk menemukan kandidat *pattern.*
3 ***Redesign***. Desain ulang MDB dengan menerapkan desain *pattern.*
4 ***Reimplementation***
5 ***Testing***
6 **Dokumentasi**

5. Kesimpulan dan Saran

5.1 Kesimpulan

Kesimpulan yang dapat diambil dari TA ini adalah:

1 Desain *pattern* dapat jika digunakan dengan tepat dapat meningkatkan *reusability* dan *readability* aplikasi berbasis *web*

2 *Trade off* dari peningkatan *reusability* dan *readability* MDB adalah waktu proses perhalaman yang lebih lama bahkan berlipat secara numerik

5.2 Saran

1 Diperlukan *pattern* untuk mengatasi waktu proses perhalaman yang lebih lama yang disebabkan karena akses *template HTML*

2 Diperlukan *pattern* untuk mem-*filter* inputan dari user dimana *pattern* tersebut tidak merubah tapi menambahi desain awal

3 Perlu dilakukan implementasi terhadap aplikasi selain MDB, sehingga desain ini dapat dipakai sebagai *Framework* untuk sebagian besar aplikasi berbasis *web* karena keumumannya

Daftar Pustaka

[1] Achour M, Betz F., Dovgal A, Lopes N., Olson P., Richter G., Seguy D., and Vrana J. *PHP Manual.* PHP Documentation Group ,1997-2004.

[2] Boggs, Wendy, Michael Boggs. *Mastering UML with Rational Rose 2002.* SYBEX Inc, 2002.

[3] Fathansyah. *Basis Data.* Bandung: Penerbit Informatika 2004.

[4] Fowler, Martin. R*efactoring, Improving the Design of Existing Code.* Kanada: Addison-Wesley, 1999.

[5] Gamma, E., R. Helm, R. Johnson, dan J. Vlissides. *Design Patterns: Elements of Reusable Object Oriented System.* Kanada: Addison-Wesley, 1995.

[6] Kafura, Dennis. *Object-Oriented Software Design and Construction with C++.* New Jersey: Prentice Hall, 1998.

[7] Ramakrishnan R., Gehrke J. *Database Management System.* Singapore: McGraw-Hill 2003.

[8] R. Isak. *Dasar Pemrograman Berorientasi Objek dengan JAVA 2 (JDK 1.4).* Yogyakarta: ANDI Yogyakarta, 2003.

[9] Shalloway, A. dan R. T. James. *Design Patterns Explained: A New Prespective on Object-Oriented Design.*Kanada: Addison Wesley 1999.

[10] Sklar, Joel. *Principles of Web Design*. Kanada: Thomson, 2006.