ANALISIS DAN IMPLEMENTASI FACTORY METHOD PATTERN

Yudi Yustiranda[1], Shaufiah[2], Kusuma Ayu Laksitowening[3]

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

Abstrak

Design pattern muncul akibat adanya permasalahan yang sama yang sering muncul pada desain pembuatan perangkat lunak. Pada perkembangannya sudah banyak design pattern yang sudah ditemukan oleh para programmer. Saat ini, design pattern dikelompokkan ke dalam tiga tujuan berbeda, yaitu creational, structural, dan behavioral.

Factory Method Pattern merupakan salah satu design pattern yang termasuk dalam kelompok creational pattern. Creational pattern adalah kelompok design pattern yang berhubungan dengan pembuatan objek. Factory method pattern digunakan untuk memisahkan (decouple) proses pembuatan/instansiasi sebuah objek (produk) dari objek lain (klien) yang menggunakannya. Tujuannya supaya perubahan pada product class tidak menyebabkan perubahan kode pada client. Paling tidak akibat dari perubahan itu bisa diminimalisir. Dan juga supaya factory bisa digunakan oleh banyak kelas.

Pada tugas akhir kali ini, dibuat sebuah perangkat lunak yang mengimplementasikan factory method pattern untuk menyelesaikan sebuah kasus. Untuk mengevaluasi factory method pattern, digunakan metrics pengukuran yang khusus menangani perangkat lunak berorientasi objek yaitu object-oriented metrics. Hasil pengujian ini kemudian dibandingkan dengan hasil pengujian perangkat lunak yang tidak menerapkan factory method pattern. Pada akhirnya didapatkan kesimpulan tentang kualitas kedua perangkat lunak dalam hal reusability.

Berdasarkan hasil pengujian didapatkan bahwa perangkat lunak dengan factory method pattern memiliki reusability yang lebih baik bila dibandingkan dengan perangkat lunak tanpa factory method pattern. Hal ini berarti pengembangan perangkat lunak yang menerapkan factory method pattern akan lebih mudah dibandingkan dengan perangkat lunak tanpa factory method pattern.

Kata Kunci : design pattern, factory method pattern, reusability, dan object-oriented metrics.

Abstract

Design pattern appears because of the same problem that usually appears at software design. There are many design patterns created by many programmers lately. Nowadays, design pattern is classified by three different purposes, they are creational, structural, and behavioral.

Factory Method Pattern is one of design patterns that belong to creational pattern purpose. Creational pattern is a design pattern that relates to creating object. Factory method pattern is used to decouple creating object process from another object that's supposed to utilize the object created. This is done to prevent product class change affects client's code change. At least, the affection can be minimalized and the factory can be used by many classes.

In this Final Assignment, a software is created to implement factory method pattern in problem solving. To evaluate factory method pattern, object-oriented metrics is used to measure object oriented software particularly. The result of this measure then compared to result from another experiment that doesn't implement factory method pattern. At last, we get a conclusion about both quality in matter of reusability.

Based on this experiment, it is proved that software with factory method pattern has better reusability than another software without it. This causes software with factory method pattern based will be easier to develop than software without factory method pattern.

Keywords : design pattern, factory method pattern, reusability, and object-oriented metrics.

1. PENDAHULUAN

1.1 Latar belakang masalah

Pemrograman berorientasi objek (PBO) adalah salah satu cara membangun aplikasi, dengan didasarkan pada objek-objek dunia nyata, yang mengandung data dan fungsi/prosedur yang bekerja atas objek tersebut. Dalam perkembangannya, banyak masalah-masalah yang kerap terjadi dalam pembangunan aplikasi berorientasi objek. Kebanyakan masalah tersebut memiliki inti permasalahan yang sama, yang telah diselesaikan oleh para pembuat program [1].

Design pattern muncul akibat adanya permasalahan yang sama yang sering muncul pada desain pembuatan perangkat lunak [4]. Dengan menggunakan *design pattern*, tingkat *reusability* suatu *design* akan semakin tinggi. Pada perkembangannya sudah banyak *design pattern* yang sudah ditemukan oleh pala programmer. Saat ini, design pattern dikelompokkan ke dalam tiga tujuan berbeda, yaitu *creational, structural, dan behavioral* [4].

Factory Method Pattern merupakan salah satu *design pattern* yang termasuk dalam kelompok *creational pattern*. *Creational pattern* adalah kelompok *design pattern* yang berhubungan dengan pembuatan objek. *Factory method pattern* digunakan untuk memisahkan (*decouple*) proses pembuatan/instansiasi sebuah objek (produk) dari objek lain (klien) yang menggunakannya. Pemisahan ini menjadikan desain perangkat lunak dengan *factory method pattern* lebih mudah dikembangkan dan digunakan kembali pada aplikasi lain[12]. Dengan kata lain, penerapan *factory method pattern* membuat desain lebih *reusable*.

Untuk mengukur tingkat reusability ini, digunakan metric pengukuran yang secara khusus menangani pengukuran perangkat lunak berorientasi objek. Metric ini dikenal dengan sebutan *object oriented metrics (OO Metrics)*. Atribut pengukuran yang dapat uji dengan OO Metrics ini antara lain: *complexity, reusability, coupling, dan cohesion.*

1.2 Perumusan masalah

Permasalahan yang akan dibahas pada tugas akhir ini adalah sebagai berikut:

1. Apakah penggunaan factory mehod pattern bisa meningkatkan reusability suatu perangkat lunak?
2. Bagaimana kualitas suatu perangkat lunak yang mengimplementasi factory method pattern dan perangkat lunak yang tidak mengimplementasikan factory method pattern dalam hal reusability berdasarkan oo metrics?

1.3 Batasan masalah

Batasan masalah pada tugas akhir ini adalah :

1. Komponen OO metrics yang di analisis :
   a. Coupling Between Object Classes (CBO)
   b. Lack of Cohesion of Methods (LCOM)
   c. Weighted Methods per Class (WMC)

2. Menggunakan bahasa pemrograman Java dan NetBeans IDE Dev sebagai aplikasi GUI nya serta Microsoft Windows 7 sebagai system operasinya.
3. Desain perangkat lunak dimodelkan dengan menggunakan Microsoft Visio 2003.
4. Aplikasi yang dibuat berbasis desktop, studi kasus: sistem informasi kos-kosan.

1.4 Tujuan

Tujuan yang ingin dicapai dalam pembuatan Tugas Akhir ini antara lain :

a) Menerapkan *factory method pattern* pada sebuah perangkat lunak berbasis desktop.
b) Menganalisa dan membandingkan *reusability* dari kedua perangkat lunak yang menerapkan *factory method pattern* dan yang tidak menerapkan *factory method pattern* berdasarkan *OO metrics*.

1.5 Metodologi penyelesaian masalah

1. Studi Literatur :
   Mencari informasi dari berbagai sumber seperti: jurnal, buku, pencarian melalui internet, dan artikel-artikel mengenai beberapa topik seperti: *design pattern* dan *factory method pattern*.
2. Analisis dan Desain :
   Menganalisa dan merancang desain untuk membangun aplikasi desktop yang menerapkan *factory method pattern* dan yang tidak menerapkan *factory method pattern*.
3. Implementasi sistem:
   Membangun prangkat lunak yang menerapkan *factory method pattern* dan yang tidak menerapkan *factory method pattern*.
4. Analisis hasil implementasi:
   Analisis OO metrics terhadap aplikasi yang telah menerapkan *factory method pattern* dan yang tidak menerapkan *factory method pattern*.
5. Penyusunan laporan
   Hasil penelitian akan disusun menjadi suatu laporan yang meliputi aspek-aspek dalam penelitian yaitu teori dan implementasinya.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan rangkaian analisis, desain, implementasi, pengujian dan pengukuran yang telah dilakukan, maka dapat disimpulkan bahwa:

1. Penggunaan *factory method pattern* pada perangkat lunak sisfo kos-kosan dapat memperbaiki nilai coupling perangkat lunak tersebut. Hal ini mengindikasikan bahwa perangkat lunak sisfo kos-kosan dengan factory method pattern memiliki reusability yang lebih baik.
2. Jika dilihat dari segi kohesi, kedua perangkat lunak baik yang menerapkan factory method pattern maupun yang tidak menerapkan factory method pattern memiliki kohesi yang sama hal ini disebabkan karena tidak ada perubahan struktur kelas dari kedua perangkat lunak. Dengan kata lain penerapan factory method pattern pada perangkat lunak sisfo kos-kosan tidak berpengaruh pada kohesifitas perangkat lunak tersebut.
3. Dari segi kompleksitas, perangkat lunak yang menerapkan *factory method pattern* justru memiliki kompleksitas yang lebih tinggi dibadingkan dengan perangkat lunak tanpa *factory method pattern.* Hal ini dikarenakan ada penambahan kelas pada perangkat lunak dengan factory method pattern yang sekaligus menambah jumlah method.

5.2 Saran

1. Selain dapat dimplementasikan dengan *factory method pattern*, sisfo kos-kosan juga bisa dikombinasikan dengan *design pattern* yang lain, misalnya *abstract factory pattern*.
2. Untuk menghasilkan nilai *object-oriented* yang lebih akurat, dapat digunakan tools untuk menghitung parameter-parameter yang ada secara otomatis, sehingga dapat membantu mempercepat dalam proses analisis.

DAFTAR PUSTAKA

[1] Anonymous. 2008. Design Pattern (Computer Science). Dikutip : 14 Maret 2010, [online]. Available : http://en.wikipedia.org/wiki/Design_ pattern_(computer_ science).

[2] Bratadinata, Anggie. 2009. *Factory Pattern*. Dikutip : 29 November 2009, [online]. Available : http://masputih.com/2009/05/factory-pattern

[3] Chidamber, Shyam R. and Kemerer, Chris F. 1994. A Metrics Suite for Object Oriented Design. Transaction on Software Engineering.

[4] Freeman, Eric & Freeman, Elisabeth. 2004. Head First Design Pattern, First Edition, O'Reilly Media, Inc.

[5] Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John (1994). *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley. ISBN 0-201-63361-2.

[6] Jamali, Sayed M. 2006. *Object Oriented Metrics*. Department of Computer Engineering Sharif University of Technology.

[7] Jasnowski, Mike. 2004. *Java, XML, and Web Service.* Third Edition, Hungry Minds.

[8] Larman, Craig. 1999. *Applying UML and Patterns.* Third Edition, Prentice Hall PTR.

[9] Martin, Robert. 1994. *OO Design Quality Metrics.* Cranbrook Road.

[10] Rosenberg, Linda H. 1998. *Applying and Interperting Object oriented Metrics.* Sowftware Assurance Technology Center.

[11] Salea, Trisya Kansya. 2008. *Tugas Akhir: Analisis dan Implementasi Refactoring Untuk Mendapatkan Decorator Design Pattern pada Kode Pemrograman Berbasis Objek.* Fakultas Teknik Informatika Institut Teknologi Telkom.

[12] Schach, S. R. 1997. Software Engineering with Java. Times Mirror Higher Education Group.

[13] Steven, John Metsker. 2002. Design Pattern Java Workbook. Addison Wesley.

[14] Wicaksono, Ady. 2005. *Dasar-Dasar Pemrograman Java 2.* PT. Elex Media Komputindo.