Available online at TRANSMISI Website http://ejournal.undip.ac.id/index.php/transmisi

TRANSMISI, 12 (4), 2010, 133-137

Penerapan Konsep *Inversion Of Control* pada Sistem Informasi dengan Java *Framework* Google Guice

Herdhian Cahya Novanto[1], Kodrat Iman Satoto[2], R. Rizal Isnanto[2]

1. *Mahasiswa Jurusan Teknik Elektro Fakultas Teknik Universitas Diponegoro Semarang*
2. *Dosen Jurusan Teknik Elektro Fakultas Teknik Universitas Diponegoro Semarang*

Abstract

Object-oriented concept is a set of methods for the analysis, design, and programming that enables software developers can build software that is reliable, easy to use, easy in maintenance, and well documented. The concept of Inversion of Control (IoC) is able to handle this by allowing class to organize existing objects. Java programming language is one of object-oriented programming language. Google Guice is a Java framework that embraces the concept of the IoC. This framework has several components, such as the binder, annotation, and modules. Information system application that is built to give the IoC concept modeling with Google Guice framework. While the undertaken research steps is the analysis of the required classes, designing system with the Unified Modeling Language, implementation of the concept of IoC with Google Guice framewrok, and testing of desktop applications. From the testing that has been done, it can be concluded that, firtsly, the design produces modeling program with programming concepts IoC, secondly Google Guice framework configuration placing objects in another class, thirdly Google Guice can help software developers in doing the development, modification, testing, and facilitate the reuse , the fourth Google Guice framework can reduce the interference in class user interface.

Keyword *: Inversion of Control, framework, Google Guice, Java, object oriented.*

I. PENDAHULUAN

1.1 Latar Belakang

Saat ini perkembangan sebuah perangkat lunak dengan menggunakan bahasa pemrograman Java sangat pesat. Perancangan yang dikembangkan harus mampu menangani ketergantungan objek yang terdapat dalam sistem yang direancang. Oleh karena itu dibutuhkan konsep yang mampu menangani hal tersebut, yaitu konsep *Inversion of Control* (IoC).

alam Java konsep *Inversion of Control* (IoC) dapat dibangun pada *framework* Google Guice. Sebagai sebuah *framework*, Google Guice menawarkan *loosely coupling* dengan teknik yang menggunakan konsep *Inversion of Control* (IoC). jadi dengan menggunakan *Inversion of Control* (IoC), objek dengan menggunakan entitas luar yang mengkoordinir setiap objek pada sistem. Penggunaan *framework* Google Guice dimaksudkan agar semua pengkodean sistem dilakukan dalam lingkungan Java.

1.2 Tujuan

Tujuan dari penelitian ini adalah menerapkan konsep *Inversion of Control* (IoC) dengan Java *framework* Google-Guice dengan maksud untuk mengurangi ketergantungan antar objek dengan memberikan pemodelan pada sebuah sistem infomasi akademik. Selain itu menjadikan pengkodean lebih efektif dan efisien serta mudah dalam melakukan konfigurasi.

1.3 Batasan Masalah

Dalam penelitian ini pembahasan masalah dibatasi pada permasalahan berikut :

1. Menganalisis konsep dari *Inversion of Control* menggunakan *framework* Google-Guice.
2. Perancangan sistem informasi dengan konsep *Inversion of Control*.
3. Sistem informasi yang dimaksud adalah sistem informasi akademik yang dirancang sendiri dengan data-data diambil dari mahasiswa jurusan Teknik Elektro Universitas Diponegoro
4. Perancangan sistem informasi akademik yang dibuat hanya membahas pada entitas mahasiswa, entitas matakuliah, dan entitas pengambilan matakuliah oleh mahasiswa.
5. Menggunakan bahasa pemrograman Java dengan IDE Netbeans 6.7.1
6. Pengujian hanya dilakukan pada basisdata lokal dan ditampilkan pada aplikasi *desktop*.
7. Aplikasi *desktop* yang dirancang digunakan sebagai contoh permodelan penerapan konsep *Inversion of Control*.
8. Aplikasi *desktop* hanya bisa digunakan oleh pengguna yang telah dimasukkan ke dalam basisdata.

II. DASAR TEORI

2.1 Bahasa Pemrograman Java

Salah satu keunggulan Java adalah sifatnya yang *platform independence*, yaitu memungkinkan Java baik pada kode program maupun hasil kompilasinya sama sekali tidak bergantung kepada sistem operasi dan *platform* yang digunakan. Kemampuan Java berjalan di lebih dari satu sistem operasi tidak lepas dari keberadaan Java *Virtual Machine*(JVM) yang menjembatani antara *bytecode* dan hardware. Maka dari itu diperlukan instalasi JVM pada setiap platform yang berbeda agar *bytecode* dapat berjalan. Jika hanya untuk keperluan pengoperasian, hanya diperlukan Java *Runtime Environment* (JRE). Namun untuk membuat program, tidak bisa hanya mengandalkan JRE, dibutuhkan Java SDK (Software *Development Kit*) yang mempunyai *compiler* di dalamnya. mengandalkan JRE.

J2SE atau Java 2 *Standard Edition* merupakan bahasa pemrograman Java untuk aplikasi *desktop* yang mampu menerapkan *object-oriented programming*. Pada J2SE, terdiri dari dua buah produk yang dikeluarkan untuk membantu dalam membuat aplikasi tanpa tergantung dari platform yang digunakan, yaitu JRE dan JDK.

2.2 Pemrograman Berorientasi Objek

Demi menjawab semua tantangan pemrograman yang semakin komplek, maka muncul gagasan konsep pemrograman yang mengorganisasi program dengan memodelkan objek-objek dunia nyata, seperti benda, sifat, sistem dan lainnya ke dalam sebuah bahasa pemrograman. Konsep seperti ini dikenal dengan pemrograman berorientasi objek atau *Object Oriented Programming* (OOP). Dalam pemrograman berorientasi objek ini diaplikasikan dengan sebuah kelas, *method*, dan properti/variabel.

2.3 MySQL

MySQL merupakan perangkat lunak yang dibuat untuk Sistem Menejemen Basisdata (DBMS). Suatu relasional basisdata menyimpan data dalam tabel-tabel terpisah. Hal ini memungkinkan kecepatan dan fleksibelitas. MySQL merupakan turunan salah satu konsep utama dalam basisdata sejak lama yaitu SQL (*Structured Query Language*).

MySQL menggunakan standar SQL (*Structured Query Language*), yaitu bahasa standar yang paling banyak digunakan untuk mengakses basisdata dan SQL dirancang khusus untuk berkomunikasi dengan basisdata.

2.4 JDBC

Java merupakan sebuah bahasa pemrograman yang sangat bertenaga dan handal dalam hal konektifitas basisdata. Aplikasi Java tidak dapat berkomunikasi secara langsung dengan basisdata karena DBMS/RDBMS hanya dapat mengerti perintah SQL. Untuk itu diperlukan sebuah mekanisme untuk menterjemahkan pernyataan Java menjadi pernyataan SQL. Pada Java dikenalkan dengan istilah yang biasa disebut dengan Java Databace Connectivity (JDBC).

JDBC adalah Application Programming *Interface* (API) yang dirancang untuk mengakses basisdata universal berdasarkan SQL. JDBC API mempunyai beberapa komponen yang berhubungan denngan koneksi driver, koneksi basisdata, eksekusi perintah SQL dll. Adapun komponen komponen yang dimaksud:

1. *Driver* merupakan komponen yang bertugas untuk menangani permasalahan komunikasi dengan basisdata server. Pada IDE NetBeans telah disediakan JDBC driver untuk MySQL.
2. *DriverManager* merupakan *class* yang berfungsi untuk membuat objek *connection* sesuai dengan parameter *connection String.*
3. *Connection* adalah bagian yang menangani koneksi ke basisdata.
4. *Statement* merupakan bagian yang menangani pengiriman perintah SQL ke basisdata.
5. *ResultSet* merupakan komponen yang menangani penyimpanan data yang didapat dari basisdata setelah perintah SQL dieksekusi oleh komponen Statement.
6. *SQLException* merupakan komponen yang digunakan untuk menangani kesalahan-kesalahan yang mungkin terjadi dalam pengolahan basisdata.

2.5 Unified Modeling Language (UML)

UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML memiliki beberapa diagram untuk pemodelan perangkat lunak yang berbasiskan objek oriented yaitu *use-case diagram, class diagram, sequence diagram,* dan *activity diagram.*

2.6 *Inversion of Control* (IoC)

Prinsip yang paling mendasar dari IoC adalah "jangan memanggil kami, kami yang akan memanggil anda" (Do not call us we will call you).. Ada dua prinsip utama dari IoC

1. Mengumpulkan konfigurasi objek dalam satu tempat konfigurasi dengan tujuan agar lebih mudah untuk memanage objek.
2. Abstraksi tidak boleh bergantung pada rincian, rincian harus tergantung pada abstraksi.

Inversion of Control (IoC) merupakan prinsip desain yang diklaim untuk meningkatkan kualitas desain perangkat lunak seperti mudah dikembangkan (extensibility), mudah dimodifikasi (modifiability), mudah dalam pengetesan (testability), dan dapat digunakan lagi (reusability).

2.7 *Framework* Java Google Guice

Framework adalah sekumpulan fungsi, kelas, dan aturan-aturan. Berbeda dengan *library* yang sifatnya untuk tujuan tertentu saja, *framework* bersifat menyeluruh mengatur bagaimana mempermudah membangun aplikasi.

Google Guice adalah kerangka kerja perangkat lunak *open source* untuk platform Java yang dirilis oleh Google di bawah lisensi Apache. Google Guice menyediakan dukungan untuk *Inversion of Control* (IoC) menggunakan *annotation* untuk mengkonfigurasi objek pada Java.

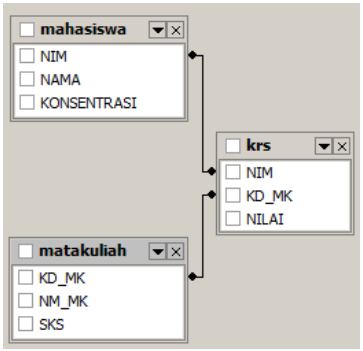
Google Guice memungkinkan pengikatan pada kelas implementasi dengan *interface* nya, kemudian disuntikkan ke dalam konstruktor, metode atau bagian yang menggunakan *annotation* @Inject. Bila lebih dari satu implementasi membutuhkan *interface* yang sama, maka dapat melakukan kustomisasi *annotation* yang mengidentifikasi suatu implementasi, yang kemudian *annotation* tersebut digunakan menyuntikkan.

III. PERANCANGAN SISTEM

Dalam penelitian ini dibuat sebuah aplikasi yang mampu mengakases basisdata sederhana yang di dalamnya ditanamkan sebuah *framework* Guice. Aplikasi yang dikembangkan adalah sebuah aplikasi yang bertujuan memberikan gambaran mengenai penggunaan *framework* Guice. Sehingga basisdata yang dikembangkan basisdata sederhana dan hanya mencakup entitas yang diperlukan saja.

3.1 Perancangan Basisdata

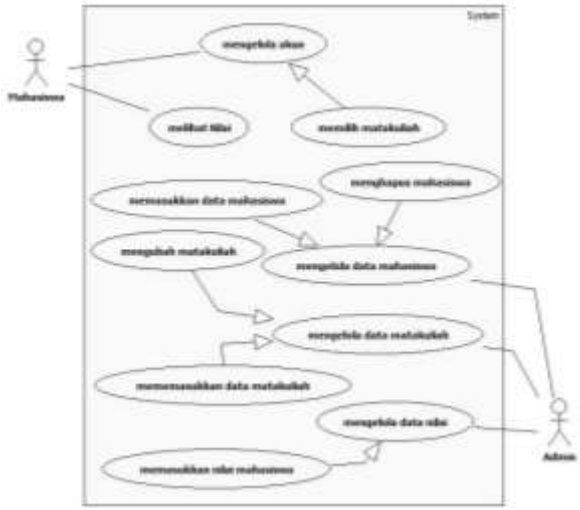
Proses normalisasi adalah proses untuk mencegah penciptaan struktur tabel yang kurang fleksibel atau kurang efisien serta terhindar dari redudansi yang akan mengakibatkan pemborosan memori penyimpanan data. Proses normalisasi dalam pembuatan aplikasi ini melewati beberapa tahapan penormalan yaitu bentuk 1 NF, 2NF dan 3NF.



Gambar 1. Perancangan basisdata

3.2 Perancangan Diagram *Use-Case*

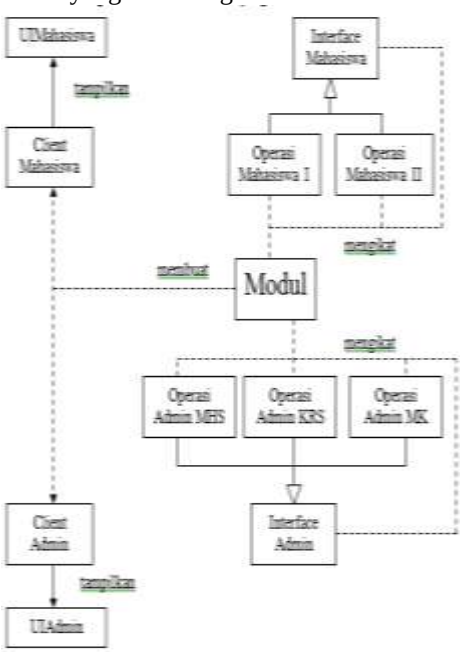
Diagram *use-case* diagram *functional* yang dibuat untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan dan diharapkan oleh pengguna.



Gambar 2. Perancangan diagram *use-case*

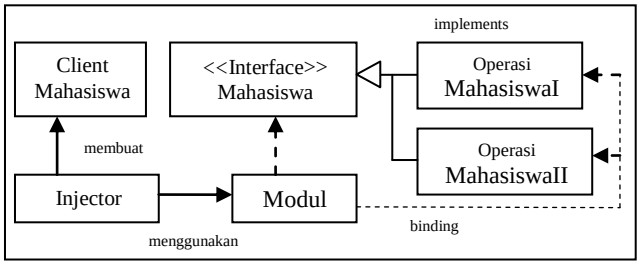
3.3 Perancangan Kelas Diagram

Kelas adalah jantung dari setiap pemrograman berorientasi objek, karena itu diagram UML yang paling memberikan gambaran adalah diagram kelas. Struktur dari sebuah diagram kelas adalah penggambaran hubungan dari kelas-kelas yang dirancang.



Gambar 3. Perancangan kelas diagram

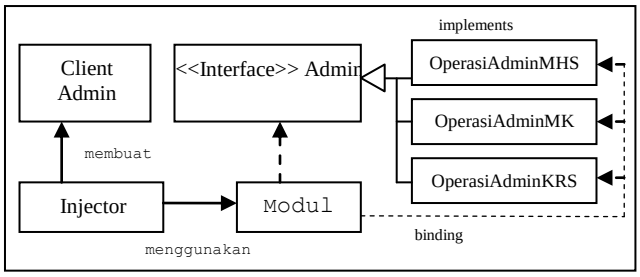
3.4 Penerapan IoC dengan Google-Guice pada level mahasiswa



Gambar 4. Penerapan IoC pada level mahasiswa

Pada sisi mahasiswa modul melakukan pengikatan antara *interface* dengan kelas implementasinya yaitu OperasiMahasiswa.

3.5 Penerapan IoC dengan Google-Guice pada level admin



Gambar 5. Penerapan IoC pada level admin

Pada sisi admin modul melakukan pengikatan antara *interface* dengan beberapa kelas sekaligus, yaitu OperasiAdminMHS, OperasiAdminMK, dan OperasiAdminKRS. Sehingga dibutuhkan penamaan pada setiap penginjeksiannya.

IV. IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Pengujian IoC dengan Google-Guice pada level mahasiswa

Gambar 6. Tampilan desktop pada sisi mahasiswa

Sistem melakukan pembacaan bahwa yang melakukan login adalah seorang mahasiswa. Mahasiswa dapat menambahkan matakuliah yang diinginkan dengan memasukkan kode matakuliah dan kemudian menekan tombol **tambah**.

Pada Gambar 6 menunjukkan tabel nilai yang memberikan keluaran berupa nilai huruf tanpa nilai tengah. Karena modul mengikat *interface* mahasiswa dengan kelas OperasiMahasiswaI. Jika modul melakukan pengikatan antara *interface* Mahasiswa dengan kelas OperasiAdminII maka tabel akan memberikan keluara nilai angka dengan nilai tengah.

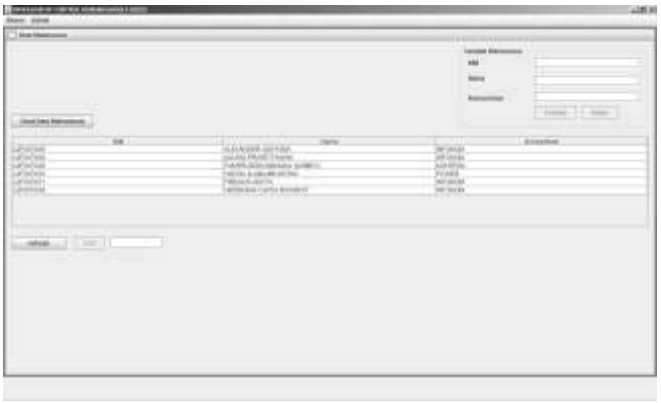
4.1 Pengujian IoC dengan Google-Guice pada level admin

Sistem melakukan pembacaan bahwa yang melakukan login adalah seorang admin. Pada sisi admin terdapat beberapa menu yang dapat diakses oleh seorang menu, yaitu menu **input data mahasiswa**, **input data matakuliah**, dan **input nilai.**

```
public class ModulGuice implements Module
{
   public void configure(Binder binder)
   {
     binder.bind(Admin.class).annotatedWith(AdminMat
     akuliah.class).to(OperasiAdminMK.class);

     binder.bind(Admin.class).annotatedWith(AdminMah
     asiswa.class).to(OperasiAdminMHS.class);

     binder.bind(Admin.class).annotatedWith(AdminKRS
     .class).to(OperasiAdminKRS.class);
   }
}
```

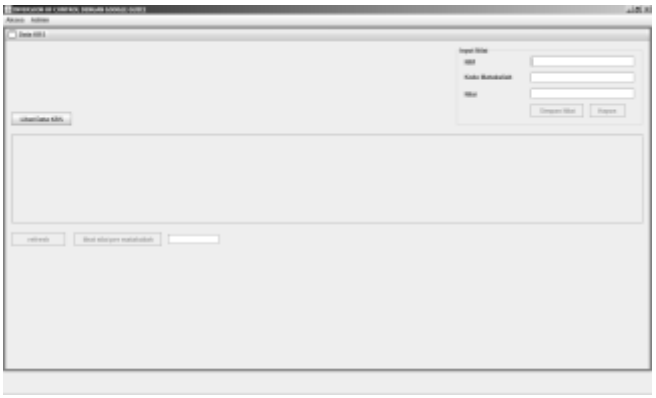


Gambar 7. Tampilan desktop pada sisi adminMHS

Data yang dimasukkan pada *text field* dapat ditampilkan pada Jtabel karena Modul dikonfigurasi dengan melakukan penamaan *annotation* @adminMahasiswa antara *interface* Admin dengan kelas OperasiAdminMHS.

Gambar 8. Tampilan desktop pada sisi adminMK

Data yang dimasukkan pada *text field* dapat ditampilkan pada Jtabel karena Modul dikonfigurasi dengan melakukan penamaan *annotation* @adminMatakuliah antara *interface* Admin dengan kelas OperasiAdminMK.



Gambar 9. Tampilan desktop pada sisi adminKRS

Data yang dimasukkan pada *text field* dapat ditampilkan pada Jtabel karena Modul dikonfigurasi dengan melakukan penamaan *annotation* @adminKRS antara *interface* Admin dengan kelas OperasiAdminKRS.

V. PENUTUP

5.1 Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan maka dapat disimpulkan beberapa hal sebagai berikut:

1. Dihasilkan aplikasi dengan menerapkan konsep *Inversion of Control* yang mengubah pola pembentukan objek dengan menerapkan *construction* injection dan *setter* injection, sehingga dapat mengurangi ketergantungan objek yang ada.
2. *Framework* Google Guice merupakan salah satu framework IoC yang sangat sederhana dalam penggunaannya karena menempatkan konfigurasi objek pada kelas tersendiri
3. Penggunaan *framework* Google-Guice dapat mempermudah dalam melakukan pengem-bangan (extensibility), pemodifi-kasian (modifiability) dan mempermudah dalam pemanfaatan kembali (reusability).
4. Pemindahan pembentukan objek dari *user interface* ke modul Google Guice dapat mengurangi interfensi pada *user interface*, sehingga jika sistem diubah sewaktu-waktu maka cukup melakukan pengubahan pada modul.

5.2 Saran

Untuk pengembangan sistem lebih lanjut, maka dapat diberikan saran-saran sebagai berikut.

1. Aplikasi perlu ditambahkan fungsi untuk penghitungan Indeks Prestasi (IP) sehingga mahasiswa dapat mengetahui IP masing-masing.
2. Aplikasi masih dapat dikembangkan dengan merancang sistem informasi akademik dalam lingkup yang nyata.
3. Aplikasi perlu dikembangkan agar dapat dimasukkan tentunya fasilitas lain bisa dimasukkan untuk ditambahkan data informasi mengenai identitas mahasiswa dan matakuliah secara lengkap sehingga didapatkan data yang lebih lengkap.
4. Aplikasi ini perlu dikembangkan tidak hanya untuk menangani sistem informasi akademik saja melainkan dapat digunakan pada sistem informasi perpustakaan, sistem penjualan dan sebagainya.

Daftar Pustaka

[1] Chonoles , Michael Jesse and James A. Schardt, *UML 2 for Dummies*, Wiley Publishing, Inc, New York, 2003.

[2] Hamilton, Kim dan Russell Miles, *Learning UML 2.0*, O'Reilly, California, 2006.

[3] Prasanna, Dhanji R., *Design patterns using Spring and Guice*, Manning Publications Co., 2009.

[4] Rachim, Arif, *Dependency Injection dengan Google Guice*, JAMU 0607, Jakarta, 2007.

[5] Pillay, Anban, *Object Oriented Programming using Java*, University of KwaZulu-Natal, Durban, 2007.

[6] Tirsen, Jon dan Aslak Hellesoy, *JavaPolis: Inversion-of-Control made easy*, Belgia, 2003.

[7] Vanbrabant, Robbie, *Google Guice: Agile Lightweight Dependency Injection Framework*, Springer-Verlag New York Inc., 2008.

[8] Wahana, *Pengembangan Aplikasi Database Berbasis JavaDB*, Penerbit Andi, Yogyakarta, 2010.

[9] Wilson , Jesse dan Dhanji Prasanna, *Big Modular Java with Guice*, Google I/O, San Fransisco, 2009.

[10] Yang , Hong Y., Hayden Melton, dan Ewan Tempero, *An Empirical Study into Use of Dependency Injection in Java*, The University of Auckland, New Zealand, 2006

[11] Yulianto, Ardhian Agung, *Analisis dan Desain Sistem Informasi*, Politeknik Telkom, Bandung, 2009.

[12] -------, Google Guice, http://code.google. com/p/google-guice/, Agustus 2011.