STUDI PEMBANGUNAN *FRAMEWORK* BERORIENTASI OBJEK STUDI KASUS PEMBANGUNAN *FRAMEWORK* PENJADWALAN

Nur Afif
Dosen Jurusan Teknik Informatika Fakultas Sains dan Teknologi
Universitas Islam Negeri "Alauddin" Makassar
email : pipo_limpuce@yahoo.co.id

***Abstract**: A framework can be defined as a "semi-complete" appication which can be reuse and specialized to produce many application in a specific domain.*
Building a framework is different from building other kind of software. It needs a special method. This method combines domain engineering and hot-spot driven framework development. Domain engineering is aimed to develop family of system in a domain, which is also suitable to developing framework. This writing also presents study about tools used in domain engineering such as feature modeling and domain specific language (DSL). The proposed method also using hot-spot driven framework development to identify hot-spots and frozen-spots of the framework by using tools such as hot-spot card and framework construction principle (FCP).
Chosen case study for implementing the proposed method is scheduling domain. Scheduling was chosen because scheduling is a well-defined science and has many literature so is relatively easy to understands the domain. Scheduling also has many variety spots which is suitable to be implements as framework.

***Key word**: reuse, domain engineering, hot-spot, frozen-spot, scheduling, flexibility.*

A. PENDAHULUAN

Penggunaulangan komponen perangkat lunak telah menjadi sasaran dalam rekayasa perangkat lunak semenjak bidang ilmu ini muncul. Beberapa penelitian telah mengemukakan solusi untuk penggunaulangan ini. Pada tahun 1970-an dikemukakan teknik pemrograman modular dalam membangun perangkat lunak, dan para rekayasawan perangkat lunak mulai menggunakan modul-modul sebagai komponen yang dapat digunaulang untuk membangun sebuah sistem baru. Akan tetapi penggunaulangan yang diberikan oleh pemrograman modular sangat terbatas, karena adaptasi terhadap sebuah modul tersebut hanya dapat dilakukan dengan mengubah kode pada bagian yang dikehendaki pada modul tersebut. Pada tahun 1980-an pemrograman berorientasi

objek mulai digemari karena pemrograman berorientasi objek diklaim memberikan penggunaulangan yang lebih bagus pada kode dengan adanya fasilitas pewarisan. Pewarisan menawarkan adaptasi yang lebih ampuh dibandingkan penggunaan modul-modul. (Wilson, 1994).

Akan tetapi solusi-solusi tersebut hanya menyediakan penggunaulangan pada level komponen individual berskala kecil yang digunakan dalam membangun sebuah sistem. Adapun masalah yang lebih sulit, yaitu melakukan penggunaulangan pada level yang lebih besar yang digunakan untuk membangun bagian sistem yang lebih luas dimana lebih banyak aspek yang dapat diadaptasi. Hal ini tidak dapat ditangani hanya dengan paradigma berorientasi objek. Adanya permasalahan ini menuntun munculnya pengembangan *framework* aplikasi berorientasi objek, yaitu aplikasi abstrak yang besar dalam domain tertentu yang dapat digunakan untuk menyusun sebuah sistem individual. Sebuah *framework* tersusun dari sebuah struktur besar yang dapat diguna ulang secara keseluruhan untuk konstruksi sebuah sistem baru.

B. TUJUAN DAN KEGUNAAN

Penelitian ini bertujuan untuk menemukan penyelesaian atas permasalahan yang ditemukan.

1. Mengusulkan metode dalam pembangunan *framework* berorientasi objek.
2. Merancang sebuah contoh kasus kedalam sebuah *framework* berorientasi objek dalam hal ini adalah penjadwalan.
3. Mengimplementasikan rancangan contoh kasus kedalam bahasa pemrograman.

C. TINJAUAN PUSTAKA

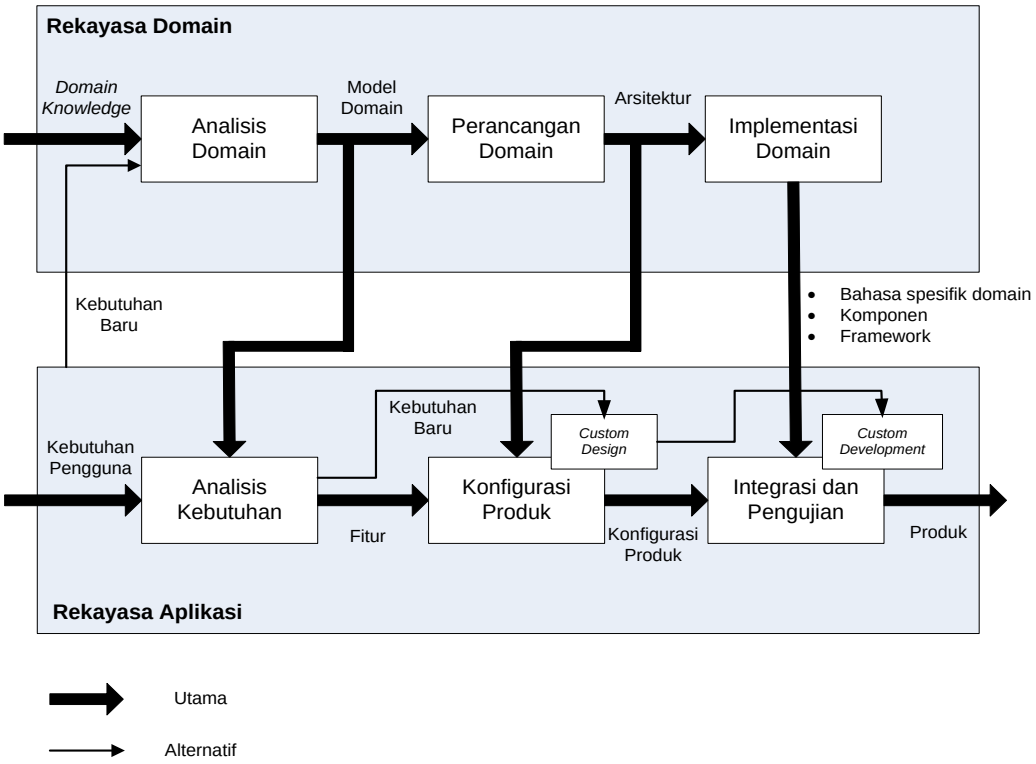
1. Rekayasa Domain

Rekayasa domain adalah aktivitas mengumpulkan, mengorganisasikan, dan menyimpan pengalaman dalam membangun sistem-sistem atau bagian-bagian dari sistem-sistem pada *domain* tertentu dalam bentuk aset-aset yang dapat digunaulang (*reuse*) yaitu produk-produk yang dapat digunaulang, juga menyediakan sarana-sarana yang memadai untuk menggunaulang aset-aset ini (yaitu kualifikasi, penyebaran, adaptasi, perakitan, dan sebagainya) ketika membangun sistem-sistem baru. Rekayasa domain terdiri dari tiga tahap (Czarnceki, 2000)

a. Analisis domain (*domain analysis*), mendefinisikan sekumpulan kebutuhan yang dapat digunaulang untuk sistem-sistem di *domain*. Analisis domain mengidentifikasi lingkup, fitur-fitur, dan titik-titik variasi sistem.
b. Perancangan domain (*domain design*), menetapkan satu arsitektur yang umum (*common*) untuk sistem-sistem pada satu *domain*.

c. Implementasi domain (*domain implementation*), mengimplementasi aset-aset yang dapat digunaulang misalnya *framework*, komponen-komponen, bahasa spesifik domain, *generator-generator*, dan infrastruktur yang dapat digunaulang (*reusable*) .

Rekayasa domain dapat diterapkan untuk berbagai permasalahan, seperti pengembangan *framework*, pustaka-pustaka komponen, bahasa spesifik domain, dan generator-generator.



Gambar 1 Rekayasa Domain (Czarnceki, 2000)

2. Pemodelan Fitur

Model fitur merepresentasikan fitur-fitur umum dan beragam dari sebuah konsep serta kebergantungan antara fitur-fitur variabel tersebut. Model fitur dibuat pada tahap pemodelan fitur. (Czarnceki, 2000)

Fitur muncul pada sembarang level seperti level kebutuhan sistem level tinggi, level arsitektur, level subsistem, level komponen, bahakan pada level implementasi (yaitu level objek atau prosedur). Pemodelan terhadap semantik fitur biasanya memerlukan formalisasi pemodelan yang lain seperti diagram objek, diagram interaksi, atau diagram *state-transition*.

Model fitur berisi diagram fitur dan beberapa informasi tambahan antara lain sebagai berikut (1) deskripsi semantiks ringkas masing-masing fitur, (2)

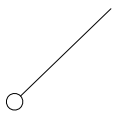
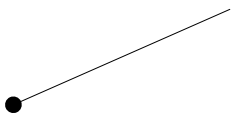
alasan untuk masing-masing fitur, (3) pihak-pihak yang berkepentingan terhadap fitur, (4) contoh-contoh sistem dengan fitur yang dikaji, (5) konstrain-konstrain terhadap fitur, (6) aturan-aturan kebergantungan *default*, (7) dimana, kapan dan pada apa fitur tersedia dan diikatkan, dan (8) prioritas-prioritas fitur. Diagram fitur menyediakan notasi grafis seperti pohon. Diagram fitur menunjukkan hirarki fitur-fitur. Akar pohon diagram fitur merepresentasikan satu simpul konsep. Semua simpul lain merepresentasikan tipe-tipe fitur berbeda. Fitur-fitur merepresentasi karakteristik-karakteristik yang dapat dibedakan dari satu konsep. (Hariyanto, 2008)

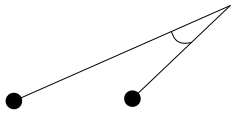
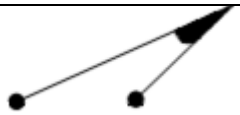
Sebuah diagram fitur terdiri dari kumpulan simpul (*node*), kumpulan sisi (*edge*), dan kumpulan dekorasi sisi (*edge decoration*). Simpul-simpul dan dan tepi-tepi yang ada membentuk sebuah pohon (*tree*). Dekorasi sisi adalah garis yang digambarkan sebagai busur yang menghubungkan sebagian atau semua sisi yang berasal dari simpul yang sama. Akar (*root*) dari sebuah diagram fitur merepresentasikan sebuah konsep, yang disebut simpul konsep (*concept node*). Simpul-simpul lainnya dalam diagram fitur merepresentasikan fitur-fitur dan disebut simpul fitur (*feature node*). (Czarnceki, 2000). Berikut ini adalah tabel notasi diagram fitur :

Tabel 1. Notasi diagram fitur (Czarnceki, 2000)

Notasi	Nama
C	Simpul konsep (*concept node*)
f_1	Simpul fitur (*feature node*)

Tabel 1. notasi diagram fitur (Czarnceki, 2000) (lanjutan)

Notasi	Nama
	Dekorasi sisi untuk fitur opsional (*optional feature*)
	Dekorasi sisi (*edge decoration*) untuk fitur mandatori (*mandatory feature*)

	Dekorasi sisi untuk fitur alternatif (*alternative feature*)
	Dekorasi sisi untuk fitur *or* (*or feature*)

3. Framework Berorientasi Objek

Framework berorientasi objek adalah sebuah teknologi yang menjanjikan untuk membentuk rancangan dan implementasi perangkat lunak yang handal sehingga dapat mengurangi ongkos pembangunan dan meningkatkan kualitas perangkat lunak. Sebuah *framework* dapat diartikan sebagai aplikasi "setengah jadi" yang dapat digunaulang dan dapat dikhususkan untuk menghasilkan bermacam-macam aplikasi dalam sebuah domain tertentu. Berbeda dengan teknik gunaulang pada paradigma berorientasi objek yang berbasis pada pustaka kelas, *framework* lebih ditujukan untuk unit-unit bisnis tertentu (misalnya pemrosesan data atau komunikasi selular) dan untuk domain-domain aplikasi (misalnya antarmuka pengguna). (Fayed, 1999)

Manfaat-manfaat utama dari *framework* berorientasi objek sangat beragam yaitu modularitas, penggunaulangan, dan perluasan. (Fayed, 1999)

a. Modularitas (*Modularity*)
b. Penggunaulangan (*Reusability*)
c. Perluasan (*Extensibility*)

Framework memberi peningkatan penting pengembangan komponen gunaulang berskala besar dibanding komponen gunaulang pada pendekatan tradisional. Pada penggunaan pustaka (*library*), pengembang (*developer*) aplikasi menulis program utama, menentukan aliran kendali dan kode pengembang memanggil kode pustaka. Pada penggunaan *framework*, pengembang aplikasi menulis kode implementasi spesifiknya, dan *framework* menentukan aliran kendali dan kode *framework* yang memanggil kode implementasi tersebut.

Framework dimaksud untuk mereduksi usaha pengembangan produk perangkat lunak serta menghasilkan perangkat lunak berkualitas. Aplikasi dibangun menggunakan *framework* sebagai basis dan memperluasnya dengan fungsionalitas spesifik aplikasi. *Framework* mempermudah pengembangan aplikasi, memungkinkan penulisan kode sesedikit mungkin, memungkinkan pemrogram pemula menulis program bagus.

Secara umum framework dapat dibagi menjadi dua bagian yaitu (Fayed, 1999):

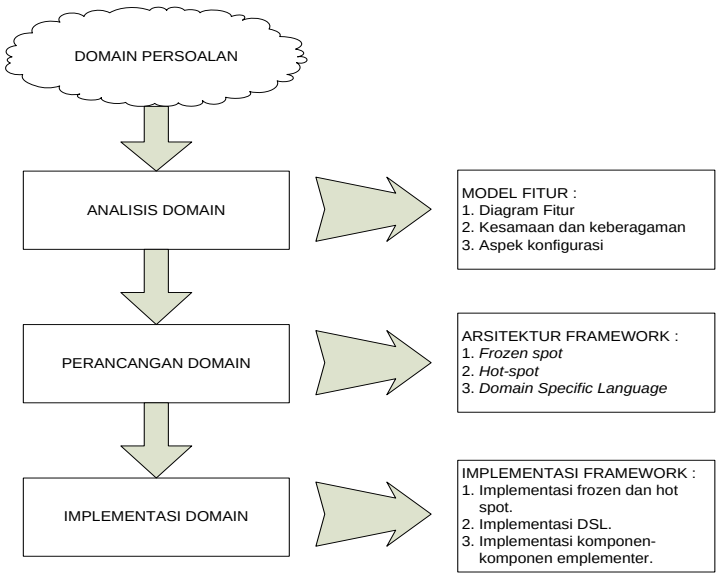
1) Kernel (*frozen spot*)
   Frozen-spot adalah bagian dari *framework* yang cenderung tetap dan sulit untuk diubah. *Frozen-spot* telah dimplementasikan sebelumnya di dalam *framework* dan akan selalu ada dalam setiap instan dari *framework.*
2) *Hot-spot*
   Hot-spot adalah titik fleksibilitas dari *framework*. *Hot-spot* akan diimplementasikan oleh pengguna sesuai kebutuhannya. *Hot-spot* ini nantinya akan dipanggil oleh *frozen spot* sesuai dengan *event* yang terjadi dalam *framework*

4. Penjadwalan (*Scheduling*)

Penjadwalan (*scheduling*) merupakan proses pembuatan jadwal-jadwal [6]. Jadwal adalah rencana atau dokumen yang memberitahu kapan sesuatu akan terjadi, menunjukkan satu rencana pewaktuan aktivitas-aktivitas tertentu yang merupakan jawaban pertanyaan: "Kapan sesuatu akan berlangsung?". Pada proses penjadwalan, kita perlu menspesifikasi tipe dan jumlah masing-masing sumber daya sehingga dapat menentukan kapan operasi-operasi dapat dilakukan secara layak. Saat menspesifikasi sumber daya, kita mendefinisi batasan penjadwalan. Kita mendeskripsi masing-masing operasi dalam kebutuhan sumber daya, durasi, waktu dapat dimulai, durasi pengolahan, waktu seharusnya telah selesai, dan sebagainya.

D. HASIL DAN PEMBAHASAN

Usulan Metode Pembangunan Framework



Gambar 2 Usulan metode pembangunan *framework*

Usulan metode pembangunan framework yang diajukan yaitu dengan menggunakan rekayasa domain (*domain engineering*). Rekayasa domain berbeda dengan rekayasa aplikasi dimana rekayasa aplikasi ditujukan hanya untuk membangun aplikasi tunggal sedangkan rekayasa domain ditujukan untuk pembangunan keluarga sistem (*family of system*) dalam suatu domain. *Framework* adalah aplikasi setengah jadi yang dapat digunaulang dan dikhususkan untuk menghasilkan bermacam-macam aplikasi dalam suatu domain. Karena itu *framework* tidak dibangun dengan rekayasa yang spesifik untuk satu jenis aplikasi melainkan *framework* mencakup sebuah domain dan diperlukan rekayasa domain untuk membangun *framework* tersebut.

Rekayasa domain untuk pembangunan framework ditujukan untuk menghasilkan *framework* yang memenuhi kebutuhan domain dan dapat digunaulang dalam lingkup domain tersebut. Rekayasa domain untuk membangun framework terdiri dari tahapan berikut:

1. Analisis domain

Analisis domain dalam pembangunan *framework* menggunakan pemodelan fitur, pemodelan fitur dipilih dengan alasan berikut:

a) Pemodelan fitur menggambarkan konsep. Konsep disini tidak memiliki semantik yang didefinisikan sehingga konsep bisa berupa apapun. Karena itu sangat cocok menggunakan pemodelan fitur untuk menggambarkan elemen-elemen dan struktur-struktur yang terdapat dalam domain, dan bukan hanya objek-objek.
b) Model fitur yang dihasilkan oleh pemodelan fitur akan menggambarkan kesamaan (*commonalities*) dan keberagaman (*variabilities*) dari sebuah domain sehingga akan memudahkan dalam mengidentifikasi hot-spot pada langkah selanjutnya.
c) Pemodelan fitur tidak memerlukan semantik seperti halnya model objek sehingga lebih mudah bagi seorang pakar domain yang tidak memahami mengenai kelas, objek, pewarisan, *design pattern* dsb.

Hasil utama dari tahap analisis domain adalah model fitur yang menyatakan kesamaan dan keberagaman serta aspek konfigurasi yang ada dalam sebuah domain, kesamaan dan keberagaman ini diekspresikan menggunakan diagram fitur.

2. Perancangan domain

Setelah model fitur di dapatkan dari hasil analisis domain, maka model fitur tersebut digunakan untuk membentuk *frozen/hot spot* dalam *framework*, dengan bantuan kartu *hot-spot* (*hot spot card*). Selain itu dalam tahap ini dihasilkan bahasa spesifik domain (*Domain Specific Language /DSL*) dari domain tersebut. DSL ini adalah bahasa yang digunakan untuk mengkonfigurasi *framework*. Dengan DSL maka penggunaan *framework* menjadi lebih mudah dan cepat.

3. Implementasi domain

Pada tahap ini dilakukan implementasi terhadap *frozen/hot spot* dalam *framework* serta mengimplementasi DSL. Selain itu komponen-komponen implementer dari *framework* juga diimplemetasikan.

a. Analisis Domain

Analisis domain melibatkan penetapan lingkup, menganalisis kesamaan, keberagaman, dan kebergantungan di dalam satu keluarga sistem, dan pengembangan kebutuhan-kebutuhan yang dapat dikonfigurasi dan digunaulang. Analisis domain akan dilakukan dengan menggunakan pemodelan fitur, dimana konsep penjadwalan akan di dekomposisi menjadi fitur-fitur dan digambarkan kedalam diagram fitur.

Algoritma Penjadwalan.

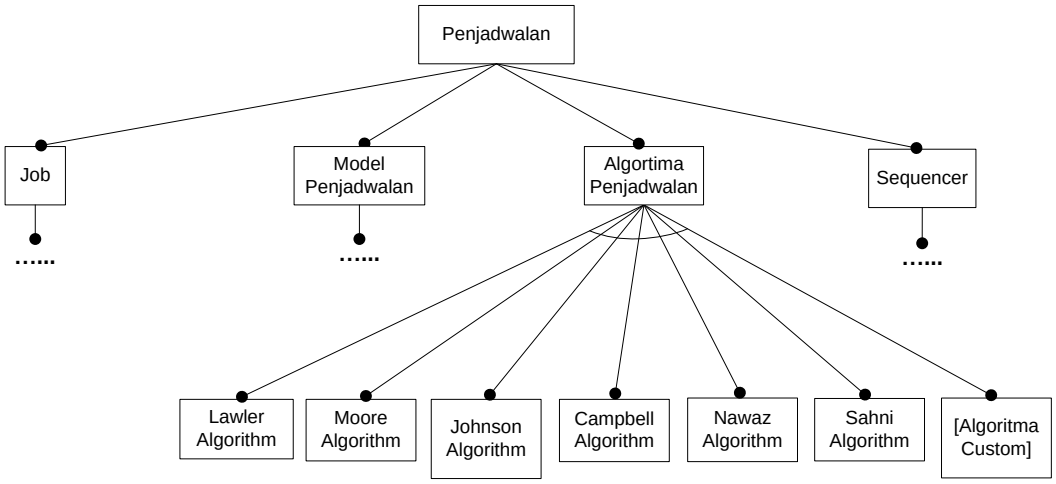
Model Penjadwalan mendefinisikan algortima yang digunakan dalam melakukan penjadwalan terhadap Job yang ada. Ada beberapa algortima yang umum digunakan dalam penjadwalan seperti Lawler Algorithm, Moore Algorithm, Johnson Algorithm, dan sebagainya.

Tabel 2 Fitur Algoritma Penjadwalan

No	Nama Fitur	Deskripsi
1.	Algortima Penjadwalan	Algortima Penjadwalan adalah algortima yang digunakan untuk menjadwalkan operasi-operasi pada Job.
3.	Lawler Algorithm	Algortima Lawler untuk menyelesaikan masalah $1\|prec\|f_{max}$
4.	Moore Algorithm	Algortima Moore untuk menyelesaikan masalah $1\|d_u\|U$

Tabel 2 Fitur Algoritma Penjadwalan (lanjutan)

5.	Johnson Algorithm	Algortima Johnson untuk menyelesaikan masalah $F2\|prmu\|C_{max}$
6.	Campbell Algorithm	Algortima Campbell untuk menyelesaikan masalah $F\|prmu\|C_{max}$
7.	Nawaz Algorithm	Algortima Nawaz untuk menyelesaikan masalah $F\|prmu\|C_{max}$
8.	Sahni Algorithm	Algortima Sahni untuk menyelesaikan masalah $P\|pmtn,d_i\|f_{max}$
9.	Algortima Custom	Algoritma Custom adalah algortima dengan implementasi custom.



Gambar 3 Diagram fitur Algoritma

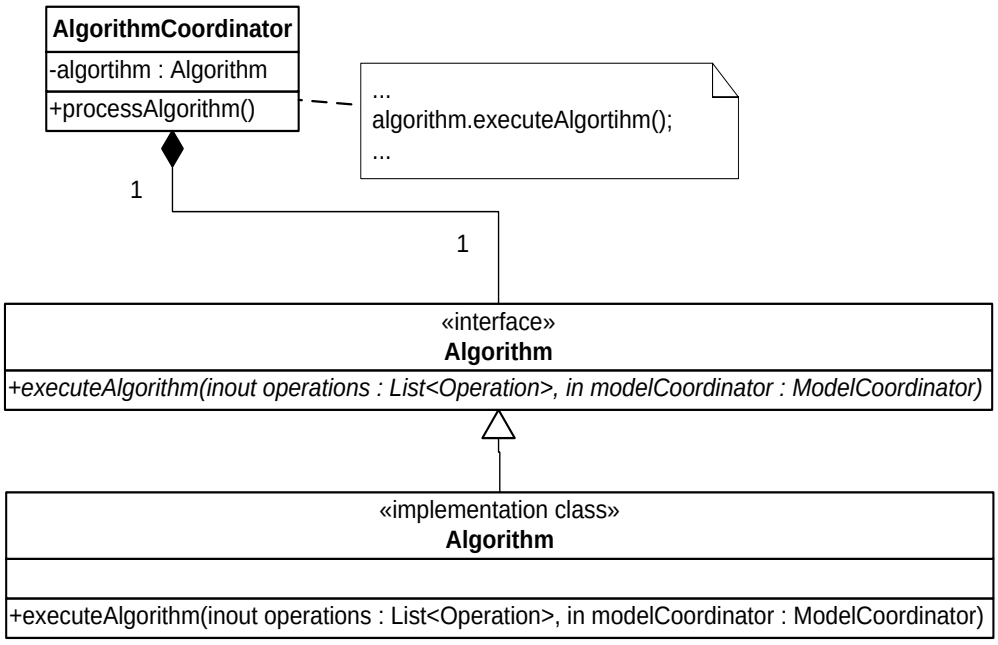
b. Perancangan Domain

1) Identifikasi Hotspot

Nama **: Algortima** derajat fleksibilitas ☒ adaptasi tanpa *restart* ☒ adaptasi oleh *end-user*
Deskripsi: Algortima Penjadawalan adalah algortima yang digunakan untuk menjadwalkan operasi-operasi pada Job.
Variabilitas: Algortima mempunyai alternatif sebagai berikut : 1. Lawler Algorithm 2. Moore Algorithm 3. Johnson Algorithm 4. Campbell Algorithm 5. Nawaz Algorithm 6. Sahni Algorithm Algoritma juga dapat diimplementasi dengan *custom*.

Gambar 4 Kartu Hot-spot Algortima

2) Perancangan Hotspot



Gambar 5 Perancangan Hot-spot Algortima

Implementasi Algoritma dalam *framework* menggunakan *design pattern* Strategy (*separation principle*). *Template method* terdapat pada kelas AlgorithmCoordinator yaitu metode processAlgorithm(). Metode ini nantinya akan memanggil *hook method* pada attribut algorithm yang merupakan tipe *interface* Algorithm. Kelas yang mengimplementasi *interface* Algorithm akan mengimplementasikan *hook method* tersebut yaitu executeAlgorithm ().

3) Perancangan DSL

Algoritma {

// Kelas Algortima – dependency injection
[kelas-algortima <string>]
// Pilihan Algortima
[algoritma-predifined [Lawler-algorithm| Moore-algorithm |
Campbell-algorithm]]
}

c. Implementasi Domain

Kelas CustomAlgortihm

```
public class CustomAlgorithm implements Algorithm {
  @Override
  public String getAlgorithmId() {
    return "algorithmId01";
  }
  @Override
  public String getAlgorithmDescription() {
    return "CustomAlgorithm";
  }
  @Override
  public void executeAlgortihm(List operations, Batch batch,
          Precedence precedence, Sequencer sequencer) {
    sequencer.executeSequencer(operations);
    int startTime = 0;
        for (int i=0;i<=operations.size();i++) {
          Operation operation = (Operation) operations.get(i);
      operation.setStartTime(startTime);
          operation.setDeadlineTime(startTime);
        }
  }
}
```

Gambar 5 Implementasi Kelas CustomAlgortihm

`CustomAlgortihm` adalah kelas implementasi pengguna untuk *hot-spot* Algorithm. Kelas ini harus mengimplementasikan beberapa *method* yaitu getAlgortihmId() yang akan mengembalikan *string* untuk id Algortihm, *method* getAlgortihmDescription() yang mengembalikan deskripsi Algortihm, dan *method* executeAlgortihm(). *Method* `execute Algortihm`() ini adalah tempat pengguna mendefinisikan algoritma yang diinginkan. *Method* ini memiliki parameter yaitu list dari operasi-operasi yang akan dimanipulasi, parameter batch, precedence dan sequencer yang telah didefinisikan dalam DSL. Beberapa algortima membutuhkan bacth, precedence, dan sequencer untuk prosesnya.

E. KESIMPULAN

1. Metode yang diajukan untuk pembangun *framework* adalah perpaduan antara rekayasa domain dan pengembangan *framework* yang dituntun *hot-spot.* Rekayasa domain digunakan untuk membangun keluarga sistem karena itu cocok untuk membangun *framework*. Rekayasa domain terdiri atas tiga tahap yaitu (1)

analisis domain, (2) perancangan domain dan (3) implementasi domain. Sedangkan pengembangan *framework* dituntun *hot-spot* digunakan dalam rekayasa domain yaitu pada tahap perancangan domain untuk menerapkan hasil pemodelan fitur kedalam prinsip konstruksi *framework* dengan menggunakan kakas yaitu kartu hot-spot

2. Fitur modeling adalah teknik dalam rekayasa domain yang digunakan untuk menangkap kesamaan dan keberagaman dari sebuah domain. Dari fitur modeling dapat diidentifikasi titik-titik variasi dalam sebuah domain. Kakas yang digunakan adalah diagram fitur.

3. Domain penjadwalan adalah contoh yang baik untuk membangun *framework* karena memiliki titik-titik perluasan untuk pengguna seperti, Job, Operation, Algortihm dan Sequencer. Dengan adanya titik-titik perluasan ini maka memungkinkan domain penjadwalan untuk diterapkan kedalam *framework* sebagai hot-spot dari *framework* tersebut.

DAFTAR PUSTAKA

Baker, Kenneth R., (2001), Elements of Sequencing and Scheduling Kenneth R. Baker, 2001

Batory, Don, Gang Chen, Eric Robertson, Tao Wang, (2000), Design Wizards and Visual Programming Environments for GenVoca Generators, IEEE

Czarnceki, Krzysztof, Ulrich W. Eisenecker (2000), Generative Programming: Methods, Tools, and Applications, Addison-Wesley, 2000

Fayad, Mohamed E., Douglas C. Schmidt, Ralph E. Johnson (1999), Building Application Frameworks: Object-Oriented Foundations of Framework Design, John Wiley & Sons, Inc., 1999

Fontoura, Marcus Pree, Wolfgang Rumpe, Bernhard . (2001). UML Profile For Framework Architecture. Addison-Wesley, 2001

Hariyanto, Bambang (2008), Penerapan Generative Programming Pada Sistem Autonomic Untuk Memperoleh Keluwesan Dan Kinerja Tinggi, Disertasi Doktoral, Institute Teknologi Bandung, 2008

Transactions on Software Engineering, Vol. 26, No. 5, May 2002

Wilson, Dave. (1994) Designing Object-Oriented Frameworks. Personal Concepts, Palo Alto, CA 1994.