



Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

BAB II

TINJAUAN PUSTAKA

2.1. Traveling Salesman Problem with Time Windows (TSPTW)

Traveling salesman problem merupakan permasalahan optimasi kombinatorial dimana *salesman* harus mengunjungi setiap kota yang ditunjuk sebanyak N dan menemukan jalan terpendek untuk sirkuit Hamiltonian. Sirkuit Hamiltonian akan mengunjungi setiap node di dalam grafik sebanyak satu kali (Nygard & Yang, 2014).

Kemudian, muncul masalah baru pada bisnis yang memiliki jadwal tetap, seperti: bus sekolah, yaitu adanya jendela waktu yang harus dipenuhi agar dapat melakukan *service*. Sehingga, muncul konsep TSP dengan *time windows* sebuah varian dari *Traveling Salesman Problem* untuk memperhitungkan ketersediaan waktu kota yang dikunjungi (Karaa & Derya, 2015)

Berikut ini merupakan salah satu contoh ilustrasi TSPTW, *sales* datang lebih cepat dari batas *time window* yang ditentukan, sehingga *sales* perlu menunggu untuk melakukan pelayanan (*service*). *Sales* tidak dapat melakukan *service* ketika batas *time window* telah habis.

2.2. PHP: Hypertext Preprocessor

PHP merupakan salah satu bahasa pemrograman berbasis web dimana sistem yang diterapkan adalah pada sisi *server side*. PHP dapat disisipkan diantara skrip bahasa HTML dan arena bahasa *server side* lainnya, dengan itu maka PHP akan dieksekusi secara langsung pada server. Sedangkan *browser* akan mengeksekusi

halaman web tersebut melalui server yang kemudian akan menerima tampilan “hasil jadi” dalam bentuk HTML, sedangkan kode PHP itu sendiri tidak akan dapat terlihat. Berikut ini merupakan kelebihan PHP, yaitu (Haryana, 2015):

1. Web menggunakan PHP dapat dengan mudah dibuat dan memiliki kecepatan akses yang cukup tinggi.
2. Skrip-skrip PHP dapat berjalan dalam *web server* yang berbeda dan dalam sistem operasi yang berbeda pula.
3. PHP dapat berjalan disistem operasi UNIX, Windows dan Macintosh.
4. PHP diterbitkan secara gratis.
5. PHP juga dapat berjalan pada web server Microsoft Personal Web Server, Apache, IIS, Xitami dan sebagainya.
6. PHP adalah termasuk bahasa *embedded* (dapat ditempel atau diletakan dalam *tag* HTML)
7. PHP termasuk *server-side programming*

2.3. React Native

React Native merupakan salah satu *framework* dari JavaScript untuk membangun aplikasi *mobile*. React Native merupakan hasil pengembangan dari React, namun React dikhususkan untuk pembangunan aplikasi berbasis web, dimana React Native dikhususkan untuk pembangunan aplikasi berbasis *mobile*.

Serupa dengan React, syntax di React Native menggunakan format XML, atau yang biasa disebut dengan XML. React Native mendukung platform iOS dan Android sampai saat ini (Eisenman, 2015).

2.4. Algoritma Genetika

Algoritma genetika merupakan metode meta-heuristik global optimasi berbasis populasi yang dapat digunakan untuk menangani masalah yang kompleks dengan ruang pencarian yang sangat besar (Paula, et al., 2015). Algoritma Genetika memiliki tiga fase yang bekerja secara acak (Snaselova & Zboril, 2015):

1. Membuat populasi kromosom secara acak

Setiap kromosom mewakili solusi dari masalah dan memiliki kualitas yang diukur dengan nilai *fitness*. Algoritma Genetika dimulai dengan menghasilkan beberapa solusi secara acak yang disebut sebagai populasi awal.

2. Operator *Crossover*

Pada tahap ini, akan dilakukan *random crossover* untuk menghasilkan *offspring* baru. Berikut ini beberapa contoh *crossover* yang dapat digunakan pada Algoritma Genetika *partially mapped crossover* (PMX), *order crossover* (OX), dan *cycle crossover* (CX) (Kumar, 2012).

3. Operator Mutasi

Pada tahap ketiga, beberapa kromosom akan bermutasi yang ditandai dengan perubahan atau pergantian beberapa gen di dalam suatu kromosom. Generasi baru yang dihasilkan dari proses diatas akan digunakan untuk iterasi selanjutnya.

2.5. Rapid Application Development

Rapid Application Development (RAD) merupakan siklus hidup yang digunakan untuk mengembangkan software dengan lebih cepat serta menghasilkan kualitas software yang tinggi. RAD juga membantu mengurangi biaya

pengembangan dan tetap mempertahankan kualitas dari software itu sendiri. RAD terdiri dari empat fase, yaitu requirement planning, user design, construction, serta cutover (Naz & Khan, 2015).

2.6. Object-Oriented Systems Analysis and Design

Analisa dan desain berbasis objek merupakan pendekatan yang ditujukan untuk pengembangan sistem yang terus menerus berubah menghadapi lingkungan bisnis yang dinamis. Yang membedakan analisa dan desain berbasis objek berbeda dengan prosedural tradisional adalah penjelajahan objek yang menjadi bagian dari sistem (Kendall & Kendall, 2014).

Analisa dan desain berbasis objek kerap kali bekerja dengan baik pada situasi yang memiliki sistem informasi yang rumit. Analisa dan desain berbasis objek menggunakan *Unified Modeling Language* sebagai standard *modelling* untuk sistem berbasis objek (Kendall & Kendall, 2014).

Setiap objek merupakan representasi komputer dari beberapa kejadian atau hal yang sebenarnya. Contoh dari objek, seperti pelanggan, *items*, *orders*, dan masih banyak lagi. Objek direpresentasikan dan dikelompokkan menjadi beberapa kelas yang optimal untuk digunakan kembali dan mudah dipelihara. Sebuah kelas terdiri dari beberapa atribut dan sifat yang ditemukan di tiap objek dalam kelas (Kendall & Kendall, 2014).

Berikut ini merupakan tahapan dengan deskripsi singkat untuk membuat *UML*:

1. Menetapkan *use case model*

Pada fase ini, *analyst* melakukan identifikasi aktor dan kegiatan yang diinsiasi oleh aktor. Biasanya, *analyst* menggambar diagram dengan *stick figure* yang menggambarkan aktor dan tanda panah yang menggambarkan hubungan antar aktor. Hal ini dinamakan *use case diagram*.

2. Selama fase analisa sistem, mulai gambar diagram UML

Pada fase kedua, *analyst* menggambar *activity diagrams*, yang memberikan ilustrasi seluruh aktivitas utama di *use case*, serta memperlihatkan urutan aktivitas yang terjadi dan *timing*. Saat ini adalah saat yang paling tepat untuk meninjau ulang *use case*, memikirkan kembali, dan melakukan modifikasi apabila diperlukan.

3. Melanjutkan fase analisa sistem, kembangkan *class diagram*

Kata benda pada *use case* adalah objek yang memiliki potensial untuk dikelompokkan ke dalam kelas. Misalnya, setiap *automobile* adalah objek yang memiliki karakter yang sama dengan *automobile*. Mereka kemudian menjadi sebuah kelas.

4. Mulai desain sistem dengan melakukan modifikasi pada diagram UML, kemudian lengkapi spesifikasinya

Desain sistem artinya mulai melakukan modifikasi pada diagram yang sudah digambar pada fase sebelumnya. Diagram-diagram ini dapat digunakan untuk menurunkan kelas, atribut, serta *method*.

5. Kembangkan dan dokumentasi sistem

Dokumentasi sangat penting. Semakin lengkap informasi yang diberikan ke *team development*, semakin cepat pengembangan dan semakin *solid* hasil produksi sistem.

