Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 3, No. 3, Maret 2019, hlm. 2862-2866 http://j-ptiik.ub.ac.id

Pengembangan Aplikasi Penentuan Prioritas Kebutuhan Fungsional Perangkat Lunak Berdasarkan Kebutuhan Non-Fungsional

Dini Indah Nurul Rizki Pancawati Raharjo[1], Tri Astoto Kurniawan[2], Denny Sagita Rusdianto[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]diniindah03@gmail.com, [2]triak@ub.ac.id, [3]denny.sagita@ub.ac.id

Abstrak

Kebutuhan perangkat lunak dapat didefinisikan sebagai layanan yang harus disediakan perangkat lunak untuk mendukung tercapainya tujuan pengguna dalam menggunakan perangkat lunak tersebut. Kebutuhan perangkat lunak merupakan dasar dalam proses pengembangan perangkat lunak. Penentuan prioritas kebutuhan merupakan proses yang penting untuk dilakukan terutama dalam situasi dimana waktu dan sumber daya pengembangan perangkat lunak terbatas. Penentuan prioritas ini bermanfaat untuk menyediakan rekomendasi urutan implementasi kebutuhan dan juga kebutuhan yang harus mendapatkan perhatian lebih dalam pengimplementasiannya. Penelitian yang dilakukan oleh Umang Garg dan Abhishek Singhal mengusulkan suatu metode penentuan prioritas kebutuhan fungsional perangkat lunak berdasarkan kebutuhan non-fungsional. Penelitian ini menerapkan metode yang diusulkan Garg dan Singhal dalam bentuk aplikasi berbasis web, dengan tujuan untuk membantu proses penentuan prioritas kebutuhan. Penentuan prioritas kebutuhan dimulai dengan melakukan perbandingan berpasangan antara kebutuhan non-fungsional, kemudian dilakukan perbandingan berpasangan antara kebutuhan fungsional dan kebutuhan non-fungsional, terakhir didapatkan urutan kebutuhan berdasarkan prioritasnya serta nilai prioritasnya. Pengujian sistem dilakukan dengan pengujian unit terhadap tiga method pada sistem, pengujian integrasi, serta pengujian validasi yang terdiri dari 27 kasus uji yang menghasilkan hasil valid.

Kata kunci: penentuan prioritas kebutuhan, kebutuhan fungsional, kebutuhan non-fungsional

Abstract

Software requirements can be defined as services that software must provide in order to support user achieving the goal of using the software. Software requirements are the basis in the software development process. In situation where the source and time are limited it is important to prioritize the requirements. Requirements prioritization shows which requirement should be implemented first and get more attention in the implementation process. A research by Umang Garg and Abhishek Singhal, a method to prioritize software requirements based on non-functional requirements is proposed. In this research, an application to prioritize software functional requirements is developed in order to help the process of requirements prioritization, the method used in this research is the one that proposed by Umang Garg and Abhishek Singhal. Requirement prioritization started with a pair-wise comparison between the non-functional requirements, and then a pairwise comparison between functional requirements with non-functional requirements, lastly the order of requirements' priority and the value of priority are obtained. The system is tested by unit testing on 3 methods of the system, integration testing, and 27 test cases on validation testing which give valid results.

Keywords: *requirement prioritization, functional requirement, non-functional requirement*

1. PENDAHULUAN

Perangkat lunak dihasilkan dari proses rekayasa perangkat lunak, yang merupakan disiplin rekayasa yang berfokus terhadap semua aspek dalam produksi perangkat lunak. Rekayasa perangkat lunak terdiri dari proses spesifikasi kebutuhan, perancangan dan implementasi, validasi, dan evolusi perangkat lunak (Sommerville, 2011). Proses spesifikasi kebutuhan menghasilkan daftar kebutuhan yang mendeskripsikan kebutuhan pemangku

kepentingan yang kemudian menjadi dasar bagi proses rekayasa selanjutnya. Daftar kebutuhan perangkat lunak didapatkan dengan memahami kebutuhan dari peran, sudut pandang, serta objektif pemangku kepentingan (Pandey, et al., 2010).

Penentuan prioritas kebutuhan dapat mengurangi konflik yang muncul diantara pemangku kepentingan, karena melalui penentuan prioritas kebutuhan dapat dibuat susunan kebutuhan yang sesuai dengan urgensinya dan disetujui semua pihak (Liaqat, et al., 2016). Hal ini bertujuan agar waktu dan sumber daya yang dimiliki dapat dialokasikan dengan lebih bijak. Penentuan prioritas kebutuhan juga memegang peranan yang sangat penting dalam perencanaan rilis perangkat lunak, menentukan strategi untuk anggaran biaya dan penjadwalan, juga strategi pemasaran (Perini, et al., 2013).

Scope creep merupakan salah satu permasalahan yang terjadi ketika *scope* proyek semakin melebar karena kebutuhan yang ters bertambah (Satzinger, et al., 2012). Proyek FBI *Virtual Case* merupakan contoh kegagalan proyek pengembangan perangkat lunak yang disebabkan oleh kebutuhan yang terus menerus bertambah dan berkembang (Liaqat, et al., 2016). Proyek kemudian diberhentikan setelah 5 tahun pengembangan denga kerugian kurang lebih 170 miliar dollar. Penentuan prioritas kebutuhan merupakan salah satu cara yang dapat dilakukan untuk mengurangi kemungkinan terjadinya kegagalan pengembangan perangkat lunak.

Abhishek Singhal (Garg & Singhal, 2017) pada penelitiannya mengajukan konsep penentuan prioritas kebutuhan perangkat lunak, khususnya kebutuhan fungsional berdasarkan pada kebutuhan non-fungsional. Metode Garg dan Shinghal diajukan agar pengembang perangkat lunak dapat mengambil kesimpulan mengenai kebutuhan fungsional mana yang memiliki prioritas lebih tinggi dengan memperhatikan juga prioritas dari kebutuhan non-fungsional yang ada.

Penelitian ini berjudul "Pengembangan Aplikasi Penentuan Prioritas Kebutuhan Fungsional Perangkat Lunak Berdasarkan Kebutuhan Non-Fungsional" dengan menerapkan metode yang diusulkan Garg dan Shinghal ke dalam bentuk aplikasi web. Harapannya adalah hasil penelitian ini dapat membantu dalam penentuan prioritas kebutuhan fungsional agar lebih mudah dan cepat.

2. PENENTUAN PRIORITAS KEBUTUHAN PERANGKAT LUNAK BERDASARKAN KEBUTUHAN NON-FUNGSIONAL

Penelitian Umang Garg dan Abhishek Singhal (2017) yang berjudul "*Software Requirements Prioritization Based on Non-Functional Requirements*" mengusulkan metode penentuan prioritas kebutuhan perangkat lunak, khususnya kebutuhan fungsional dengan memperhatikan kebutuhan non-fungsional.

Langkah-langkah dalam metode yang diajukan Garg dan Singhal, yaitu:

1. Melakukan perbandingan berpasangan antara kebutuhan non-fungsional dengan menggunakan skala metode AHP pada Tabel 1. Setelah dilakukan perbandingan berpasangan, nilai prioritas kebutuhan non-fungsional didapatkan dengan melakukan pembagian antara jumlah dari perbandingan pada tiap elemen (satu kolom) dengan nilai perbandingan pada elemen individu (nilai setiap baris).
2. Melakukan perbandingan berpasangan antara kebutuhan fungsional dengan kebutuhan non-fungsional. Perbandingan dilakukan dengan memperhatikan prioritas kebutuhan non-fungsional yang bersangkutan. Skala yang digunakan untuk nilai perbandingan ini dapat dilihat pada Tabel 2.

Tabel 1. Skala AHP

Intensitas Kepentingan	Definisi
1	Kedua elemen memiliki nilai kepentingan sama
3	Elemen sedikit lebih penting dari elemen lainnya.
5	Elemen lebih penting dari elemen lainnya.
7	Elemen sangat lebih penting dari elemen lainnya.
9	Elemen mutlak lebih penting dari elemen lainnya.
2,4,6,8	Nilai- nilai antara dua kepentingan yang berdekatan.

Tabel 2. Skala perbandingan berpasangan kebutuhan fungsional dengan kebutuhan non-fungsional

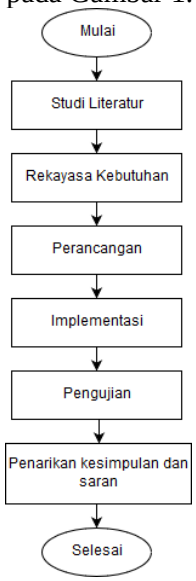
Nilai Kepentingan	Deskripsi
1	Kepentingan sama.
2	Kepentingan sedikit lebih tinggi.
3	Kepentingan lebih tinggi.
4	Kepentingan sangat tinggi.
5	Kepentingan mutlak lebih tinggi.

3. Menghitung nilai prioritas kebutuhan fungsional dengan perkalian matriks. Perkalian dilakukan antara nilai perbandingan dari langkah 2 dengan nilai prioritas kebutuhan non-fungsional yang bersangkutan.

Penentuan nilai pada saat perbandingan berpasangan bersifat subjektif dan berdasarkan pengalaman analis. Tujuan dari metode ini adalah agar dapat ditentukan kebutuhan fungsional mana yang lebih baik diimplementasikan terlebih dahulu dengan mempertimbangkan pula kebutuhan non-fungsional (Garg & Singhal, 2017).

3. METODOLOGI PENELITIAN

Langkah-langkah yang dilakukan dalam penelitan ini dimulai dengan studi literatur, pengembangan aplikasi dengan SDLC model *waterfall* yang dimulai dari rekayasa kebutuhan hingga pengujian, dan terakhir adalah penarikan kesimpulan. Diagram alir dari langkah-langkah ini dapat dilihat pada Gambar 1.



Gambar 1. Digram Alir Metodologi Penelitian

Studi literatur bertujuan untuk mendapat pemahaman yang lebih dalam mengenai konsep serta teori yang berkaitan dengan penelitian ini dan juga mengenai teknologi pengembangan perangkat lunak.

Rekayasa kebutuhan merupakan langkah yang terdiri dari elisitasi kebutuhan, deskripsi umum sistem, dan pemodelan kebutuhan. Elisitasi kebutuhan dilakukan dengan menganalisis *paper* Garg dan Shinghal yang berudul "*Software Requirement Prioritization Based on Non-Functional Requirement"*. Kebutuhan kemudian dimodelkan dalam bentuk *use case diagram*.

Langkah selanjutnya, yaitu perancangan. Perancangan merupakan langkah untuk membuat rancangan sistem dari hasil rekayasa kebutuhan yang sebelumnya dilakukan. Perancangan sistem menghasilkan perancangan arsitektur berupa pemodelan *class diagram* dan pemodelan *sequence diagram*, perancangan komponen berupa *pseudocode*, perancangan data berupa *Conceptual Data Model* (CDM) dan *Physical Data Model (*PDM), dan perancangan antarmuka berupa *wireframe.* Hasil perancangan sistem kemudian diimplementasikan dengan menggunakan *framework* CodeIgniter, MaterializeCSS, dan basis data MySQL.

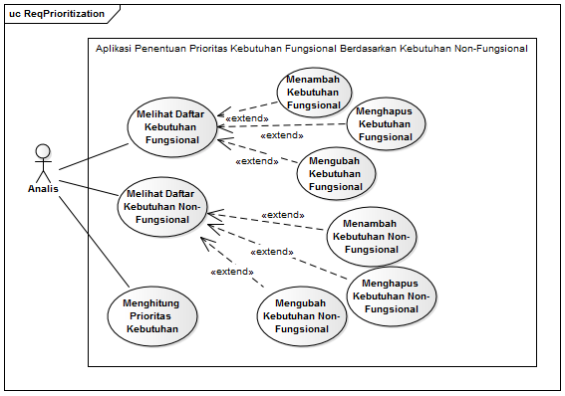
Langkah selanjutnya adalah pengujian. Pengujian dilakukan untuk memastikan semua kebutuhan sudah terpenuhi dan memperbaiki kesalahan yang ditemukan. Pengujian yang dilakukan adalah pengujian integrasi dan pengujian unit yang menggunakan teknik pengujian *white box* dan metode *basis path testing*, juga pengujian *black box* berupa pengujian validasi dengan metode *scenario based testing*.

Langkah terakhir adalah penarikan kesimpulan berdasarkan hasil dari setiap langkah sebelumnya dan diberikan saran untuk pengembangan selanjutnya.

4. REKAYASA KEBUTUHAN

Rekayasa kebutuhan bertujuan untuk mendapatkan kebutuhan-kebutuhan yang diperlukan dalam pengembangan aplikasi. Elisitasi kebutuhan dilakukan dengan menganalisis *paper "Software Requirement Prioritization Based on Non-Functional Requirements"* (Garg & Singhal, 2017). Hasil dari rekayasa kebutuhan, didapatkan aktor yang berinteraksi dengan sistem, yaitu analis, serta Sembilan kebutuhan fungsional. Tidak terdapat

urgensi untuk menambahkan kebutuhan non-fungsional. *Use case diagram* sistem terdapat pada Gambar 2.

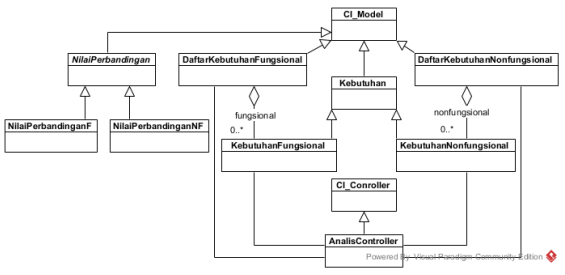


Gambar 2. *Use Case Diagram* Sistem

5. PERANCANGAN

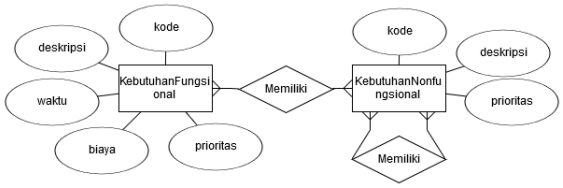
Perancangan dilakukan berdasarkan hasil rekayasa kebutuhan untuk menyediakan gambaran implementasi. Perancangan sistem terdiri dari perancangan arsitektur, perancangan komponen, perancangan data, serta perancangan antarmuka. Pendekatan dalam penelitian ini adalah berorientasi objek, sehingga pemodelan dilakukan dengan menggunakan diagram UML.

Perancangan arsitektur menghasilkan pemodelan *sequence diagram* dan *class diagram*. Pemodelan *sequence diagram* merupakan gambaran dari interaksi antar objek dan pertukaran pesan antar objek di dalam sistem berdasarkan urutan waktu. Relasi antar objek-objek dalam sistem digambarkan dalam pemodelan *class diagram*. Klas-klas yang terdapat dalam sistem adalah klas *model* yang terdiri dari klas DaftarKebutuhanNonfungsional, Kebutuhan, DaftarKebutuhanFungsional, NilaiPerbandingan, KebutuhanFungsional, KebutuhanNonfungsional, NilaiPerbandinganF, NilaiPerbandinganNF dan klas *controller*, yaitu AnalisController. Pemodelan *class diagram* sistem dalam penelitian ini terdapat pada Gambar 3.

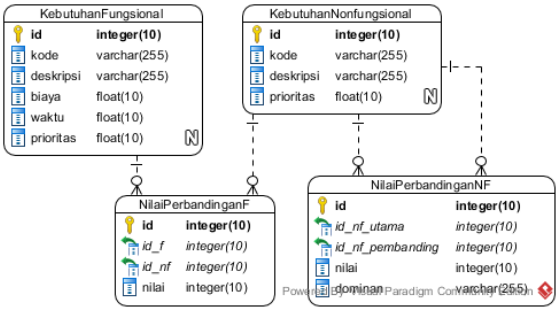


Gambar 3. Pemodelan *Class Diagram*

Perancangan komponen dilakukan dengan membuat *pseudocode* dari proses yang terjadi dalam komponen sistem. Perancangan data menghasilkan CDM dan PDM. CDM terdapat pada Gambar 4 dan PDM terdapat pada Gambar 5. Relasi *many-to-many* pada CDM menghasilkam dua entitas baru pada PDM, yaitu NIlaiPerbandinganF dan NilaiPerbandinganNF yang menyimpan nilai dari relasi *many-to-many* tersebut. Perancangan yang terakhir adalah perancangan antarmuka yang berupa *wireframe* yang menggambarkan tata letak komponen pada antarmuka sistem.



Gambar 4. *Conceptual Data Model*



Gambar 5. *Physical Data Model*

6. IMPLEMENTASI

Implementasi dilakukan berdasarkan perancangan yang sudah dilakukan. Implementasi kode program dilakukan dengan menggunakan *framework* CodeIgniter. Implementasikan antarmuka menggunakan *framework* MaterializeCSS dan bahasa HTML. Implementasi data dilakukan dengan MySQL.

7. PENGUJIAN

Pengujian bertujuan untuk memastikan sistem yang dibangun sudah memenuhi kebutuhan sistem, menghasilkan keluaran yang sesuai, dan tidak terdapat *error*. Pengujian yang dilakukan adalah pengujian *white box* berupa pengujian unit dan pengujian integrasi dengan metode uji *basis path*, serta pengujian *black box*, yaitu pengujian validasi dengan metode uji *scenario based*. Tabel 4 merupakan salah satu kasus uji dan hasil uji dari pengujian validasi.

Tabel 3 Kasus Uji dan Hasil Uji Pengujian Validasi Menambah Kebutuhan Fungsional Kondisi Berhasil

Kode Kebutuhan	RPBN-F-01
Nama Kasus Uji	Menambah kebutuhan fungsional kondisi berhasil
Prosedur	1. Membuka halaman daftar kebutuhan fungsional 2. Klik tombol "Tambah Kebutuhan Fungsional". 3. Isi *form* dengan data sebagai berikut : a. Deskripsi kebutuhan = Sistem menyediakan fungsi registrasi b. Estimasi waktu implementasi = 1 c. Estimasi biaya implementasi = 50.000 4. Klik tombol tambah.
Expected Result	Sistem menampilkan halaman daftar kebutuhan fungsional dan data yang baru ditambahkan ditampilkan pada daftar kebutuhan.
Result	Sistem menampilkan halaman daftar kebutuhan fungsional dan data yang baru ditambahkan ditampilkan pada daftar kebutuhan.
Status	Valid

Pengujian unit dilakukan terhadap klas Kebutuhanfungsional yaitu pada *method* tambahKebutuhan, Klas Kebutuhannonfungsional pada *method* updatePrioritas, serta Klas DaftarKebutuhanNonfungsional pada *method* getAllDesc. Dari masing-masing pengujian dihasilkan nilai *cyclometic complexity* 1, 1, dan 1 dengan hasil dari semua kasus uji bernilai valid.

Pengujian integrasi dilakukan pada integrasi antar klas AnalisController, klas DaftarKebutuhanNonFungsional, dank las Kebutuhannonfungsional. Pengujian ini menghasilkan nilai *cyclometic complexity* 2 dan nilai valid pada semua kasus uji. Pengujian validasi terdiri dari 27 kasus uji yang semuanya memiliki hasil valid.

8. KESIMPULAN

Kesimpulan dapat dibagi menjadi tiga bagian, yaitu hasil rekayasa kebutuhan, perancangan dan implementasi, dan pengujian. Rekayasa kebutuhan menghasilkan 9 kebutuhan fungsional dan satu aktor yang berperan dalam sistem. Perancangan menghasilkan *class diagram* yang terdiri dari 11 klas pembangun sistem, *sequence diagram* yang menggambarkan komunikasi antar objek dalam sistem dalam urutan waktu, *pseudocode*, dan CDM serta PDM. Implementasi dilakukan dengan menggunakan *framework* CodeIgniter dan MaterializeCSS serta basis data MySQL. Pengujian yang dilakukan yaitu, pengujian unit, integrasi, dan validasi menghasilkan hasil valid untuk setiap kasus uji, sehingga bisa dikatakan aplikasi sudah memenuhi semua kebutuhan.

Saran untuk penelitian selanjutnya, aplikasi penentuan prioritas kebutuhan fungsional berdasarkan kebutuhan non-fungsional ini dapat dikembangkan lebih jauh untuk keperluan manajemen kebutuhan perangkat lunak atau manajemen proyek perangkat lunak.

9. DAFTAR PUSTAKA

Garg, U. & Singhal, A., 2017. *Software Requirement Prioritization based on Non-Functional Requirements.* Noida, IEEE.

Liaqat, R. M., Azam, F., Ahmed, M. A. & Mehboob, B., 2016. *A Majority Voting Goal Based Technique for Requirement Prioritization.* Colchester, IEEE.

Pandey, D., Suman, U. & Ramani, A., 2010. *An Effective Requirement Engineering Process Model for Software Development and Requirements Management.* Kottayam, IEEE.

Perini, A., Susi, A. & Avesani, P., 2013. A Machine Learning Approach to Software Requirement Prioritization. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,* 39(4), pp. 445-461.

Satzinger, J. S., Jackson, R. B. & Burd, S. D., 2012. *Systems Analysis and Design in a Changing World.* 6th ed. Boston: Joe Sabatino.

Sommerville, I., 2011. *Software Engineering.* 9th ed. Boston: Addison-Wesley.