BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

Menurut Sutabri (2014:7) “Sistem adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu”.

Menurut Sutabri (2014:9) “mengemukakan bahwa suatu sistem terdiri atas objek-objek atau unsure-unsur atau komponen-komponen yang berkaitan dan behubungan satu sama lain sedemikian rupa sehingga unsur-unsur tersebut merupakan suatu kesatuan pemrosesan atau pengolahan yang tertentu. Sistem dapat terdiri atas kegiatan-kegiatan yang berhubungan guna mencapai tujuan-tujuan perusahaan seperti pengendalian inventaris dan penjadwalan produksi”.

2.1.1. Informasi

Menurut Sutabri (2014:25) “Informasi adalah data yang telah diklasifikasi atau diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan”.

2.1.2. Sistem Informasi

Sutabri (2014:41) “Sistem informasi terdiri dari komponen-komponen yang disebut blok bangunan (*building block*), yang terdiri dari blok masukan, blok model, blok keluaran, blok teknologi, blok basis data dan blok kendali. Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lain membentuk satu-kesatuan untuk mencapai sasaran”.

2.1.3. Pengertian UML

Perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya stadarisasi agar orang di berbagai Negara dapat mengerti pemodelan perangkat lunak. Seperti yang kita ketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang sama tidaklah mudah, oleh karena itu diperlukan bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Banyak orang yang telah membahas bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrogramannya yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan banyak pihal adalah *Data Flow Diagram (DFD)* untuk memodelkan perangkat lunak yang menggunakan pemrograman

proseduran atau structural, kemudian juga ada *State Transition Program (STD)* yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language (UML)*. *UML* muncul karena adanya kebutuhan pemodelan *visual* untuk menspesifikasikan, menggambar, membangun, dan dokumentasi dari sistem perangkat lunak. *UML* merupakan bahasa *visual* untuk pemodelan dan komunikasi sebuah sistem dengan menggunakan diagram dan *text* pendukung.

UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan *UML* tidak terbatas pada metodologi tertentu,meskipun pada kenyataannya *UML* paling banyak digunakan pada metodologi berorientasi objek.

Seperti yang kita ketahui bahwa banyak hal di dunia sistem informai yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkungan dan konteksnya. Begitu juga dengan perkembangan penggunaan *UML* bergantung pada *level* abstraksi penggunaannya. Jadi, belum tentu pendangan yang berbeda dalam penggunaan *UML* adalah suatu yang salah, tapi perlu ditelaah dimanakah *UML* digunakan dan hal apa yang ingin divisualkan.” Secara analogi jika dengan bahasa yang kita gunakan sehari-hari,belum tentu penyampaian bahasa dengan puisi adalah hal yang

salah. Sistem informasi bukanlah ilmu pasti,maka jika ada banyak perbedaan dan interpretasi di dalam bidang sistem informasi merupakan hal yang sangat wajar", Rosa dan Shalahudin, (2013:137).

2.1.4. *Actvity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal sebagai berikut

1. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/user interface di mana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian di mana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

2.1.5. *Use Case Diagram*

Rosa dan Shalahudin (2013:155) menjelaskan:

> *Use Case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebi actor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja

yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu

Syarat penamaan pada use case adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut actor dan use case.

1. Aktor merupakan orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gmbar orang, tapi aktor belum tentu merupakan orang.
2. *use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antarunit atau aktor.

2.1.6. *Sequence Diagram*

Rosa dan Shalahudin (2013:165) menjelaskan:
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuat use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada *use case*.

Banyaknya diagram sekuen yang harus digambar adalah *minimal* sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

2.1.7. Class Diagram

Menurut Rosa dan Shalahudin (2013, 141) memberikan batasan bahwa "Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi".

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Diagram kelas dibuat agar pembuat program atau programmer membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuai dengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak atau programmer dapat membuat kelas-kelas di dalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

1. Kelas main
2. Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

3. Kelas yang menangani tampilan sistem *(view*)
4. Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
5. Kelas yang diambil dari pendefinisian use case (*controller*)
6. Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian use case, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
7. Kelas yang diambil dari pendefinisian data (*mode*l)
8. Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan diambil maupun akan disimpan ke basis data. Semua tabel yang dibuat basis data dapat dijadikan kelas, namun untuk tabel dari hasil relasi atau atribut *multivalue* pada ERD dapat dijadikan kelas tersendiri dapat juga tidak asalkan pengaksesannya dapat dipertanggungjawabkan atau tetap ada di dalam perancangan kelas.

2.1.8. *Basis Data* / *Database*

Rosa dan Shalahudin (2011:43) menjelaskan bahwa:

> Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apa pun bentuknya, entah berupa file teks ataupun *database management system* (DBMS).

Kebutuhan basis data dalam sistem informasi meliputi:

1. Memasukan, menyimpan dan mengambil data.
2. Membuat laporan berdasarkan data yang telah disimpan.

2.1.9. Pengertian ERD

Rosa dan Shalahudin (2013:50) menjelaskan:
pemodelan awal basis data yang paling banyak digunakan adalah mengunakan *Entitiy Relationship Diagram* (ER). ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basis data relasional. Sehingga jika penyimpanan basis data menggunakan OODBMS maka perancangan basis data tidak perlu menggunakan ERD".

2.1.10. *Testing*

Menurut Rosa dan Shalahudin (2013:277) menyimpulkan bahwa:
Black-Box Testing merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. *Black box testing* berfokus pada kebutuhan fungsional pada *software*, berdasarkan pada spesifikasi kebutuhan dari *software*. *Black box testing* bukan teknik alternatif daripada *white box testing*. Lebih daripada itu, ia merupakan pendekatan pelengkap dalam mencakup error dengan kelas yang berbeda dari metode *white box testing*. *Black box testing* melakukan pengujian tanpa pengetahuan detil struktur *internal* dari sistem atau komponen yang dites. juga disebut sebagai *behavioral testing*, *specification-based testing, input/output testing* atau *functional testing*.

2.1.11. *PHP*

Menurut Rahardjo (2015:3) memberikan batasan bahwa:
PHP singkatan rekursif dari PHP : *Hypertext Preprocessor*, adalah bahasa pemrograman yang dapat digunakan untuk tujuan umum, sama seperti bahasa pemrograman lain : C, C++, Pascal, Python, Perl, Ruby dan sebagainya, Meskipun demikian, PHP lebih popular digunakan untuk pengembangan aplikasi web.

2.2. Penelitian Terkait

Menurut Artarti,dkk (150:2011) menjelaskan bahwa:
Persaingan bisnis distro di kota Semarang semakin ketat, Distro Smith adalah salah satu distro yang baru berdiri pada tahun 2010 di Semarang. Mekanisme sistem penjualan pada distro smith sekarang masing menggunakan sistem konvensional, dimana konsumen harus dating langsung ke distro. Dengan adanya aplikasi *e-commerce* pada distro smith diharapkan dapat memberikan kemudahan kepada masyarakat untuk melakukan pembelian produk tanpa harus dating ke tempatnya, serta memperluas pemasaran dan meningkatkan *costumer loyality.*

Menurut Al-Rosyid,dkk (1:2011) menjelaskan bahwa:
Perkembangan teknologi yang sangat pesat secara langsung maupun tidak langsung mempengaruhi sistem perdagangan. Saat ini, hanya dengan mengakses *e-commerce* melalui internet, semua orang dapat memilih barang yang diinginkan, mengetahui spesifikasi produk, harga produk dan informasi lainnya. Dengan adanya website penjualan pada buku *standard book seller*, maka pihak toko dapat terbantu memasarkan produk yang akan dijual.