BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Kajian Pustaka

Penelitian serupa mengenai sistem penjadwalan pernah dilakukan oleh peneliti yang sebelumnya dengan metode yang berbeda-beda. Berikut ini adalah beberapa penelitian terkait dengan sistem penjadwalan.

Menurut Afandi dan Yulianis (2018) dalam jurnalnya yang berjudul "Implementasi *Genetic Algoritms* Untuk Penjadwalan Mata Kuliah Berbasis *Website"* dijelaskan bahwa tujuan dari penelitian ini diarahkan untuk membantu kaprodi mengetahui penjadwalan mata kuliah yang telah ditentukan sejauh ini terdapat ruang kelas yang secara tidak sistematik dalam menentukan ruang kuliah yang telah tersedia. Penelitian ini merancang sebuah aplikasi *genetic algoritms* untuk membantu kaprodi dalam menentukan bentroknya ruang kuliah. Pada penelitian ini, peneliti menggunakan pendekatan *genetic algoritms*. Sehingga pada penelitian ini *website* tempat untuk menampilkan hasil dari inputan jadwal kuliah. Adapun pada penelitian ini menggunakan bahasa pemograman PHP *(Hypertext Processor)* digunakan untuk membangun *website* penjadwalan mata kuliah tersebut.

Menurut Setia (2017) dalam jurnalnya yang berjudul "Implementasi Sistem Penjadwalan Mata Kuliah Berbasis *Web*" dijelaskan bahwa Perguruan Tinggi vokasi melaksanakan proses penjadwalan mata kuliah secara rutin setiap semester. Ada dua penjadwalan yang sering dijumpai pada perguruan tinggi yaitu penjadwalan perkuliahan dan ujian, baik ujian tulis maupun praktikum. Batasan yang menjadi acuan dalam proses penyusunan jadwal kuliah, diantaranya adanya permintaan dosen yang bersangkutan tidak bisa mengajar pada waktu tertentu. Tidak adanya jadwal kuliah yang bersamaan antar dosen, kelas, ruang ataupun waktu perkuliahan. Dengan implementasi aplikasi penjadwalan pembelajaran pada jurusan Komputerisasi Akuntansi Politeknik Negeri Madiun, maka proses penjadwalan mata kuliah menjadi lebih praktis dan efektif.

Menurut Andriani Y (2017) dalam jurnalnya yang berjudul "Sistem Informasi Penjadwalan Proyek dan Performansi Biaya Pada PT. Kelana Buana Sulawesi Selatan" dijelaskan bahwa metode pengendalian proyek yang dipakai oleh PT. Kelana Buana saat ini adalah metode analisis variansi. Metode ini mampu menjawab pertanyaan apakah pelaksanaan proyek sesuai dengan anggaran atau jadwal, namun belum mampu mengungkapkan performansi kegiatan secara terpadu antara biaya dan jadwal. Implementasi metode analisis variansi saat ini dilakukan dengan cara *manual* menggunakan *Microsoft Excel*. Dengan cara ini terdapat banyak kelemahan, selain prosedurnya sulit juga memungkinkan terjadinya kesalahan baik dalam pembuatan formulasi maupun saat proses *update*. Penelitian ini menghasilkan sebuah sistem yang mempermudah pengukuran performansi proyek secara terpadu dengan metode nilai hasil *(earned value)* dan mempercepat poses perhitungan indikator *earned value*, variansi serta indeks kinerja biaya dan jadwal proyek.

2.2 Kerangka Pemikiran

Kerangka pemikiran merupakan alur pikir yang dijadikan sebagai skema pemikiran atau dasar-dasar pemikiran untuk memperkuat indikator yang melatar belakangi penelitian ini. Dalam kerangka pemikiran ini akan dicoba menjelaskan masalah pokok penelitian. Penjelasan yang disusun akan menggabungkan antara teori dengan masalah yang diangkat dalam penelitian ini.

Berikut adalah kerangka pemikiran yang digunakan untuk membangun sistem komputerisasi *survey* perizinan di DPMPTSP Kota Surakarta berbasis *website* yang ditunjukkan pada Gambar 2.1. Isi dari kerangka pemikiran tersebut meliputi :

1. Latar Belakang Masalah

   Sistem pencatatan jadwal *survey* perizinan di DPMPTSP Kota Surakarta pada saat ini masih berjalan dengan *manual* atau konvensional yaitu dengan cara mencatat di buku. Sistem konvensional membuat kinerja Seksi Verifikasi menjadi kurang efektif dan efisien.

2. Rumusan Masalah

   Berdasarkan latar belakang yang telah diuraikan maka rumusan masalahnya adalah bagaimana membangun sistem komputerisasi *survey* perizinan sebagai sistem pendukung SIPINTER di DPMPTSP Kota Surakarta berbasis *website*.

3. Pengumpulan Data

   Tahap pengumpulan data pada penelitian ini melalui metode wawancara, observasi, pustaka dan dokumentasi. Pengumpulan data bertujuan untuk mengetahui permasalahan dan kebutuhan suatu instansi. Pengumpulan data ini khususnya berasal dari bagian PDTI (Pusat Data dan Teknologi Informasi) dan level manajerial yang berkepentingan secara khususnya dalam penelitian ini.

4. Analisis Sistem Yang Sedang Berjalan

   Analisis sistem yang sedang berjalan bertujuan untuk mengetahui bagaimana alur sistem yang lama dalam pembuatan jadwal *survey* perizinan di DPMPTSP Kota Surakarta dari awal hingga akhir. Tahapan analisis sistem merupakan tahapan yang kritis dan sangat penting karena kesalahan ditahap ini akan menyebabkan kesalahan ditahap selanjutnya.

5. Perancangan Sistem

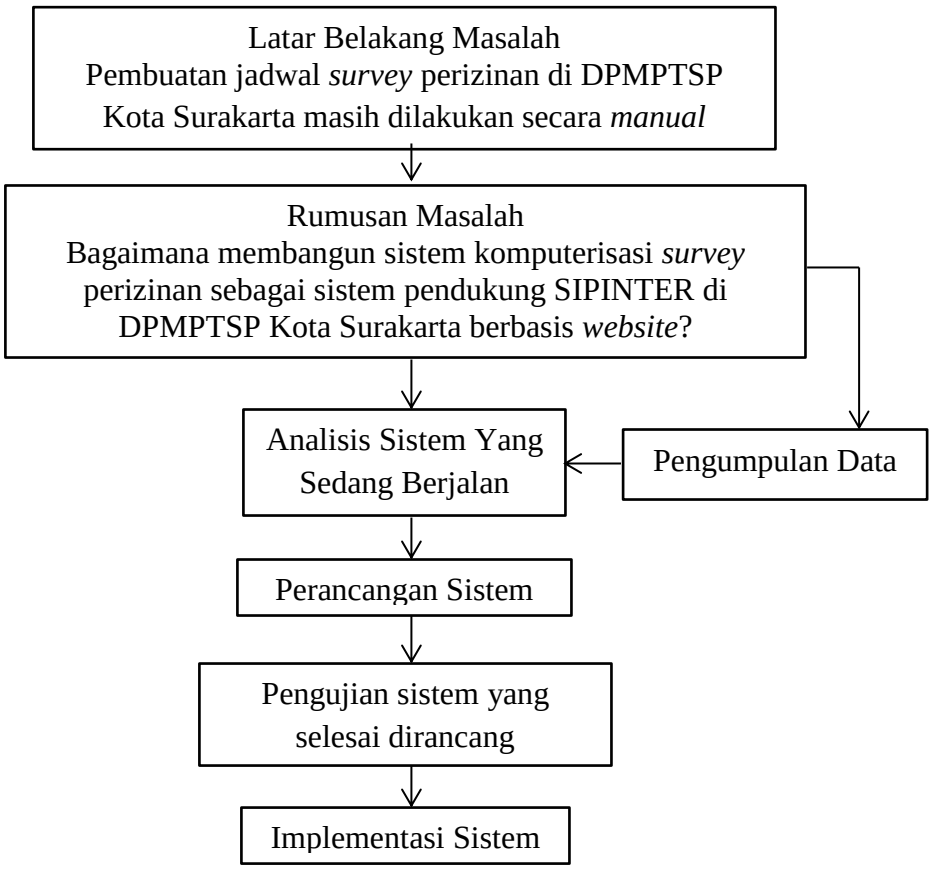
   Perancangan sistem adalah proses pembuatan *diagram* sistem yang akan dirancang dan desain *layout website* yang akan dibuat.

6. Pengujian Sistem

   Pengujian sistem merupakan tahap setelah sistem berhasil dibuat. Pengujian sistem merupakan hal terpenting yang bertujuan untuk menemukan kesalahan-kesalahan atau kekurangan-kekurangan pada perangkat lunak yang diuji.

7. Implementasi Sistem

   Tahap implementasi sistem adalah sistem yang telah dibuat telah lolos uji aplikasi dan telah digunakan di DPMPTSP Kota Surakarta untuk membantu dalam pembuatan jadwal *survey* perizinan.



Gambar 2.1 Kerangka Pemikiran

2.3 Teori Pendukung

2.3.1 Sistem

Menurut Romney dan Steinbert (2015), sistem adalah serangkaian dua atau lebih komponen yang saling terkait dan berinteraksi untuk mencapai tujuan, terdiri dari subsistem yang mendukung sistem yang lebih besar.

2.3.2 Komputerisasi

Menurut Teguh Wahyono (2004), komputerisasi merupakan kegiatan pengelolaan data yang dilakukan sebagian besarnya menggunakan komputer sebagai alat bantu.

2.3.3 *Survey*

Menurut Sugiyono (2015), pengertian penelitian *survey* adalah penelitian yang dilakukan pada populasi besar maupun kecil, tetapi data yang dipelajari adalah data dari sampel yang diambil dari populasi tersebut, sehingga ditemukan kejadian-kejadian relatif, distribusi, dan hubungan-hubungan antar variabel sosiologis maupun psikologis.

2.3.4 Perizinan

Menurut Adrian (2015), perizinan dapat diartikan sebagai salah satu bentuk pelaksanaan fungsi pengaturan dan bersifat pengendalian yang dimiliki oleh pemerintah terhadap kegiatan - kegiatan yang dilakukan oleh masyarakat. Bentuk perizinan antara lain: pendaftaran, rekomendasi, sertifikasi, penentuan kuota dan izin untuk melakukan sesuatu usaha yang biasanya harus memiliki atau diperoleh suatu organisasi perusahaan atau seseorang sebelum yang bersangkutan dapat melaksanakan suatu kegiatan atau tindakan.

2.3.5 *Website*

Menurut Bekti (2015), *website* merupakan kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing masing dihubungkan dengan jaringan-jaringan halaman.

2.3.6 *CodeIgniter*

Proses pengembangan *website* dapat dilakukan dengan beragam bahasa pemrograman seperti PHP, *Python, Ruby, Perl,* C++, JAVA dan sebagainya. Saat ini, banyak bermuculan *framework website* yang dirancang untuk bahasa-bahasa pemrograman tersebut. Salah satunya adalah *CodeIgniter*.

Menurut Supono dan Putratama (2016), *CodeIgniter* adalah aplikasi open source berupa *framework* dengan model MVC *(Model, View, Controller)* untuk

membangun *website* dinamis dengan menggunakan PHP. *CodeIgniter* memudahkan pengembang *website* untuk membuat apliaksi *website* dengan cepat dan mudah dibandingkan dengan membuat dari awal.

Kelebihan *CodeIgniter* dibandingkan dengan *framework* lain :

a) Performa yang cepat

   CodeIgniter merupakan *framework* yang paling cepat dibanding *framework* lain.

b) Konfigurasi yang sedikit *(nearly zero configuration)*

   Untuk menggunakan *CodeIgniter* dengan pengaturan yang standar, hanya perlu mengubah sedikit *file* pada *folder config*.

c) Menggunakan konsep MVC

   Pengerjaan antara logika dengan *layout* telah dipisahkan, sehingga *programmer* dan *designer* dapat mengerjakan tugas masing-masing dengan mudah.

d) Banyak komunitas

   Banyaknya komunitas *CodeIgniter* memudahkan *developer* berdiskusi satu sama lain.

e) Dokumentasi lengkap

   Setiap paket instalasi *CodeIgniter* sudah disertai *user guide* yang lengkap dan mudah dipahami.

2.3.7 Basis Data *(Database)*

Menurut Fathansyah (2015), Basis Data terdiri dari 2 kata, yaitu basis dan data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang diwujudkan dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

2.3.8 MySQL

Menurut Hidayatullah dan Kawistara (2015) MySQL adalah salah satu aplikasi DBMS yang sudah banyak oleh para pemogram aplikasi *website*. Contoh DBMS lainnya adalah PostgreSQL *(freeware)*, SQL *Server*, *MS Access* dari

Microsoft, DB2 dari IBM, *Oracle* dan *Oracle Corp*, Dbase, *FoxPro*, dan sebagainya.

2.3.9 PHP *(Hypertext Preprocessor)*

Menurut Purbadian (2015), mengemukakan bahwa PHP *(Hypertext Preprocessing)* merupakan bahasa pemrograman berbentuk skrip yang ditempatkan disisi *server*, sehingga php disebut juga sebagai bahasa *Server Side Scripting*, artinya bahwa dalam menjalankan php selalu membutuhkan *web server*, dan untuk melihat hasilnya menggunakan *web browser*.

2.3.10 *Sublime Text*

Menurut Supono dan Putratama (2016), *Sublime Text* merupakan perangkat lunak *text editor* yang digunakan untuk membuat atau meng-*edit* suatu aplikasi. *Sublime Text* mempunyai fitur *plugin* tambahan yang memudahkan *programmer*. Selain itu, *Sublime Text* merupakan IDE yang ringan, cepat dalam menyimpan dan membuka file. *Sublime Text* juga memiliki desain terkesan elegan untuk sebuah *syntax editor*.

2.3.11 *Xampp*

Menurut Purbadian (2016), *XAMPP* merupakan suatu *software* yang bersifat *open source* yang merupakan pengembangan dari LAMP *(Linux, Apache, MySQL, PHP* dan *Perl).*

2.3.12 UML

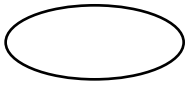
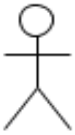
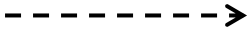
Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung (Sukamto dan Shalahuddin, 2018).

2.3.13 Pengertian *Use Case Diagram*

Use Case Diagram merupakan pemodelan untuk kelakukan *(behavior)* sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi

apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu (Sukamto dan Shalahuddin, 2018). *Use case diagram* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.1.

Tabel 2.1 Simbol-simbol *Use Case Diagram* (Sukamto dan Shalahuddin, 2018)

No	Simbol	Deskripsi
1.	*Use Case* 	Abstraksi dan interaksi antara sistem dan aktor.
2.	Aktor / *Actor* 	Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*.
3.	Asosiasi / *Association*	Abstraksi dari penghubung antara aktor dengan *use case*.
4.	Ekstensi / *Extend* <<extend>> 	Menunjukkan bahwa suatu *use case* merupakan tambahan fungsional dari *use case* lainnya jika suatu kondisi terpenuhi.
5.	Generalisasi / *Generalization* 	Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan *use case*.
6.	Menggunakan / *Include* / *Uses* <<include>> 	Menunjukkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari *use case* lainnya.

2.3.14 Pengertian *Class Diagram*

Menurut Sukamto dan Shalahuddin (2018), 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. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki

oleh suatu kelas. *Class diagram* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.2.

Tabel 2.2 Simbol-simbol *Class Diagram* (Sukamto dan Shalahuddin, 2018)

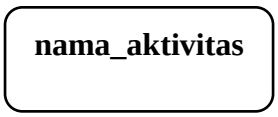
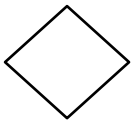
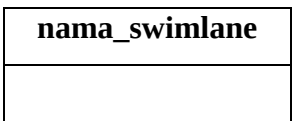
No	Simbol	Deskripsi
1.	Kelas / *Class* **nama_kelas** +atribut +operasi()	Kelas pada struktur sistem.
2.	Antarmuka / *Interface* **nama_interface**	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.	Asosiasi / *Association*	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*
4.	Asosiasi berarah / *Directed association*	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
5.	Generalisasi	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus).
6.	Kebergantungan / *Dependency*	Relasi antarkelas dengan makna kebergantungan antarkelas.
7.	Agregasi / *aggregation*	Relasi antarkelas dengan makna semua-bagian (whole-part).

2.3.15 Pengertian *Activity Diagram*

Menurut Sukamto dan Shalahuddin (2018), diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem

atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. *Activity diagram* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.3.

Tabel 2.3 Simbol-simbol *Activity Diagram* (Sukamto dan Shalahuddin, 2018)

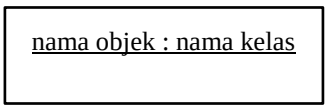
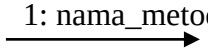
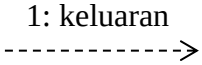
No	Simbol	Deskripsi
1.	Status Awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan / *Decision* 	Asosiasi percabangan dimana jika ada ada pilihan aktivitas lebih dari satu.
4.	Penggabungan / *Join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.	Status Akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	*Swimlane* 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.3.16 Pengertian *Sequence Diagram*

Menurut Sukamto dan Shalahuddin (2018), 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 menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. *Sequence diagram* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.4.

Tabel 2.4 Simbol-simbol *Sequence Diagram* (Sukamto dan Shalahuddin, 2018)

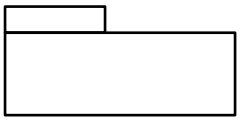
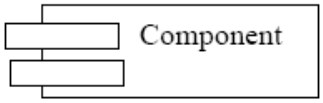
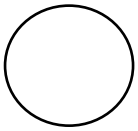
No	Simbol	Deskripsi
1.	Aktor / *actor*  nama_aktor	Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*.
2.	Garis hidup / *Lifeline* 	Objek *entity*, antarmuka yang saling berinteraksi.
3.	Objek  	Menyatakan objek yang berinteraksi pesan.
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
5	Pesan tipe *call*  	Menyatakan bahwa suatu objek memanggil operasi.metode yang ada pada objek lain atau dirinya sendiri.
6	Pesan tipe *return*  	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu.

2.3.17 Pengertian *Component Diagram*

Menurut Sukamto dan Shalahuddin (2018), diagram komponen atau *component diagram* dibuat untuk menunjukan organisasi dan ketergantungan

diantara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem. *Component diagram* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.5.

Tabel 2.5 Simbol-simbol *Component Diagram* (Sukamto dan Shalahuddin, 2018)

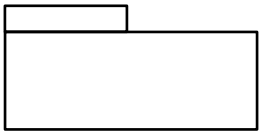
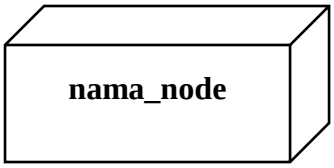
No	Simbol	Deskripsi
1.	*Package* 	*Package* merupakan sebuah bungkusan dari satu atau lebih komponen.
2.	Komponen 	Komponen sistem.
3.	Kebergantungan / *dependency* 	Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai.
4.	Antarmuka / *Interface*  **nama_interface**	Sama dengan konsep *interface* pada pemrograman berorientasi objek, yaitu sebagai antarmuka komponen agar tidak mengakses langsung komponen.
5.	*Link*	Relasi antar komponen.

2.3.18 Pengertian *Deployment Diagram*

Menurut Sukamto dan Shalahuddin (2018), diagram *deployment* atau *deployment diagram* menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. Diagram *deployment* mempunyai simbol-simbol yang ditunjukkan pada Tabel 2.6. Diagram *deployment* juga dapat digunakan untuk memodelkan hal-hal berikut.

1) Sistem tambahan *(embeddded system)* yang menggambarkan rancangan *device*, *node* dan *hardware*.
2) Sistem *client*/*server*.
3) Sistem terdistribusi murni.
4) Rekayasa ulang aplikasi.

Tabel 2.6 Simbol-simbol *Deployment Diagram* (Sukamto dan Shalahuddin, 2018)

No	Simbol	Deskripsi
1.	*Package* 	*Package* merupakan sebuah bungkusan dari satu atau lebih *node*.
2.	*Node* 	*Node* menggambarkan bagian-bagian *hardware* dalam sebuah sistem. Notasi untuk *node* digambarkan sebagai sebuah kubus 3 dimensi.
3.	Kebergantungan / *dependency* 	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.
4.	*Link*	Relasi antar *node*.

2.3.19 *Black Box Testing* (Pengujian Kotak Hitam)

Menurut Sukamto dan Shalahuddin (2018), *Black Box Testing* (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus

salah, misalkan untuk kasus proses *login* maka kasus uji yang dibuat adalah jika *user* memasukkan nama pemakai *(username)* dan kata sandi *(password)* yang benar. Jika *user* memasukkan nama pemakai *(username)* dan kata sandi *(password)* yang salah, misalnya nama pemakai benar tapi kata sandi salah, atau sebaliknya, atau keduanya salah.

2.3.20 Metode Angket / Kuesioner

Menurut Sugiyono (2014), metode angket atau kuesioner merupakan teknik pengumpulan data yang efisien apabila peneliti tahu dengan siapa variabel akan diukur dan tahu apa yang bisa diharapkan dari responden. Kuesioner dapat berupa pertanyaan-pertanyaan tertutup atau terbuka, dapat diberikan kepada responden secara langsung atau dikirim melaui pos atau internet.