ISSN : 1978 - 1105

ARIKA

Media Ilmuan dan Praktisi Teknik Industri

Vol. 07, Nomor 2 Agustus 2013

JURNAL TEKNIK INDUSTRI

Program Studi Teknik Industri Fakultas Teknik Universitas Pattimura

ARIKA, Vol. 07, No. 2
ISSN: 1978-1105

Agustus 2013

PERANCANGAN PROTOTIPE SOFTWARE TOOLS UNTUK PENGEMBANGANGAN SITUS KULIAH SECARA ELEKTRONIK

Nasir Suruali
Dosen Program Studi Teknik Mesin, Fakultas Teknik, Universitas Pattimura Ambon
e_mail : nsuruli@yahoo.com

ABSTRAK

Idealnya para dosen akan lebih baik jika memberikan kuliah kepada mahasiswa dengan cara tatap muka di depan kelas. Sistem memberi kuliah secara langsung ini mempunyai kendala seperti transportasi, lokasi geografis, waktu dan lain lain, sedangkan pemberian diktat tidak memungkinkan komunikasi dua arah. Sistem Internet adalah merupakan jawaban bagi semua permasalahan perkuliahan adapun kendala yang dihadapi adalah kesulitan untuk membuat suatu website dinamis yang menggunakan database yang lengkap. Untuk tujuan itulah sehingga diperlukan dukungan perangkat lunak yang dapat membantu merubah materi perkuliahan ke dalam format yang dapat bertahan di website. Dalam artikel ini dengan pendekatan object oriented (OO) telah dirancang suatu prototipe perangkat lunak aplikasi yang menghasikan suatu web database untuk sistem kuliah secara elektronik. Perankat lunak sistem perkuliahan ini dapat melakukan hal-hal sebagi berikut yaitu memasukan data mahasiswa, dosen, administrator dan "content" materi kulaih ke dalam suatu database, database ini kemudian dapat diakses oleh website kuliah elektronik. Berdasarkan hasil pengujian perangkat lunak yang dkembangkan ini telah dapat diperlihatkan fungsi-fungsi sperti yang telah dijelaskan di atas. Program yang dirancang dalam artikel ini telah diuji dan dapat menghasilkan "content" situs kuliah secara elektronik dan otomatis.

***Kata kunci**: pendekatan Object Oriented (OO), website dinamis.*

ABSTRACT

Ideally it is better if a lecturer can give lecture face with the student in front of the class. This face to face lecture system has many contraints such as transportation. Geographic location, time etc. Internet system is the answer to all the obove problems, and the constraints encountered are the difficulties of making a dynamic website with fully built database. To meet the purpose it is necessary to use software that can change lecture material into format that can be kept in the website. In this article a software has been prototype designed with object oriented (OO) approach for the application of website database for the electronic lecture system. This electronic lecture system can accomplish the following; input the data of the student, lecture and administrator, and content of lecture material into a certain database, this database can then be accessed by electronic lecture website. The designed program has been article and can automatically give the content of the electronic lecture website.

***Keywords** : object oriented (OO) approach, dynamic website.*

PENDAHULUAN

Secara ideal para pengajar sebaiknya dapat memberikan pengajaran dengan cara tatap muka di depan kelas atau ruangan dengan para peserta belajar. Pada keadaan tertentu kegiatan belajar mengajar tatap muka tersebut mungkin tidak dapat dilakukan karena berbagai hambatan. Hal tersebut dapat disebabkan oleh beberapa kendala, seperti letak geografis yang berjauhan antara pengajar dan peserta (seperti Maluku yang terdiri banyak pulau) atau tempat yang sangat terpencil (belum adanya transportasi yang memadai), pengajar yang tidak mempunyai waktu yang khusus untuk memberikan kuliah tatap muka, para peserta yang hadir tidak secara bersamaan dan berbagai kemungkinan lainnya.

Salah satu usaha pemecahan masalah di atas dapat dilakukan dengan cara memberikan diktat, tetapi hal ini tidak cukup karena diktat tidak memungkinkan komunikasi dua arah. Berkembangnya internet dengan berbagai aplikasi didalamnya telah menjadi alternatif baru untuk menyajikan materi belajar, selain memungkinkan interaktif, Internet juga dapat menyajikan materi dalam berbagai alternatif media, termasuk media audio visual. Adapun kendala yang dihadapi adalah sulitnya mengembangkan suatu situs secara manual untuk orang yang belum berpengalaman. Untuk memenuhi tujuan tersebut maka diperlukan dukungan perangkat lunak yang dapat membantu merubah materi kuliah tersebut ke

dalam format yang dapat disimpan dalam situs *web*. Berdasarkan fakta-fakta tersebut di atas maka pada artikel ini membahas perancangan perangkat lunak untuk membantu membuat sistem kuliah secara elektronik. Perangkat lunak yang akan dibuat memiliki beberapa karakteristik yaitu: tidak terbatas untuk satu mata kuliah tertentu, materi akan disajikan dalam bentuk format HTML, materi *audio/video* dibatasi hanya dalam format dat, avid an wav dan perangkat lunk yang dibuat bukan *web based*, tetapi output dari perangkat lunak yang dibuat bersifat *web based*.

KAJIAN PUSTAKA

Kuliah Secara Elektronik

Jenis pendidikan berbasis teknologi informasi sekarang ini dapat diklasifikasikan ke dalam tiga kelompok, yaitu sebagai berikut:

a. *Offline* CBT (*Computer Based Training*). Kelompok ini terutama bermaksud hanya menggantikan penggunaan kertas dan buku dengan menggunakan CD-ROM, DVD-ROM.
b. *Video Conference System* (VCS). Kelompok ini menggunakan interaksi penuh antara pendidik dan mahasiswanya dengan menggunakan komunikasi dua arah antara keduanya dengan menggunakan komunikasi dua arah *audio visual* melalui Internet.
c. *On-Line Web Base Learning* (WBL) *System*. Kelompok ini menggunakan *browser* yang diimplementasikan kedalam bentuk halaman *web* atau portal yang dicantumkan pada Internet dengan interaksi umumnya dengan menggunakan fasilitas *chating* atau e_*mail*.

Perangkat lunak yang akan dikembangkan adalah *software Tool* untuk pengembangan situs kuliah secara elektronik yang berjenis *Online Web Based Learning* (WBL). *On-Line Web Based Learning System* merupakan sistem CBT klasik pada lingkungan berbasis Internet, dengan menawarkan keunggulan baru sebagai berikut: Tingkat update yang tinggi, Hubungan atau *link* dengan sumber informasi atau materi sejenis, Interaksi antar mahasiswa atau dengan instruktur, Fleksibilitas yang tinggi dalam pemilihan materi (untuk test misalnya), Kemungkinan penyesuaian dengan pemakai/*customization*, dan Biaya investasi dan operasi yang relatif murah.

Kebutuhan Pengguna.

Terdapat tiga pengguna utama di dalam sistem pengajaran, yaitu mahasiswa, pengajar dan administrator. Masing-masing mempunyai kebutuhan tersendiri dan harus dipenuhi oleh sistem kuliah secara elektronik ini. Tabel.1 menunjukan daftar kebutuhan dari tiga pengguna yang berbeda.

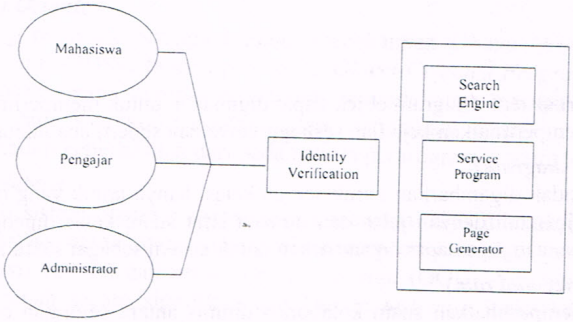
Kebutuhan Pengguna

Kebutuhan Mahasiswa	Kebutuhan Dosen	Kebutuhan Admonistrator
✓ Belajar dari rumah ✓ Mengetahui apakah mereka berada di dalam jalur belajar mereka ✓ Mendapatkan timbal balik mengenai kemajuan mereka ✓ Berhubungan dengan pengajar dan administrator ✓ Mendapatkan bantuan apa saja yang terdapat pada topik tertentu ✓ Mendapatkan bahan yang baru pada topik yang sejenis atau berhubungan ✓ Mendapatkan materi yang berkaitan berdasarkan sdhoc queries	✓ Menilai kemampuan pada mahasiswa tertentu ✓ Menilai kemajuan kelompok mahasiswa didalam menyelesaiakn sekelompok materi ✓ Mengetahui keefektifan materi mereka didalam menyiapkan mahasiswa didalam menghadapi evaluasi ✓ Mendapatkan materi didalam sistem pada topik yang berkaitan ✓ Berhubungan dengan mahasiswa dan administrator	✓ Menilai bagaimana kemampuan suatu kelompok mahasiswa (kelompok dapat berupa kelas tradisional atau sekelompok mahasiswa khusus yang dipilih berdasarkan karakteristik) ✓ Membuka pada kelas yang seharusnya dibuka berdasarkan pada keperluan mahasiswa ✓ Membuat jadwal yang optimal untuk mahasiswa dan materi ✓ Menugaskan pengajar untuk kelas ✓ Berhubungan dengan mahasiswa dan pengajar

Internet Learning Enviroment (ILE)

Untuk memenuhi kebutuhan di atas perlu melangkah lebih jauh dari pemberian informasi yang statis dan mendukung mahasiswa secara langsung di dalam belajar. ILE secara khusus digunakan untuk belajar berdasarkan keingintahuan (belajar yang memotivasi mahasiswa untuk mengajukan pertanyaan dari pada yang hanya diberikan oleh anggota fakultas) bagaimana ILE juga mendukung belajar secara langsung.

Untuk meningkatkan kemampuan pelajar dalam berfikir kreatif, dimungkinkan bila dalam proses pembelajaran Gambar 1 dibawah menunjukan pandangan keseluruhan dari sistem arsitektur.



Internet Learning Enviroment Architecture

Gambar diatas menunjukan pengguna sebagai lingkaran, program sebagai kotak dan *database* sebagai oval. Pengguna dihubungkan dengan sistem belajar melalui sebuah *word wide web* (www). WWW secara langsung dihubungkan dengan modul untuk mengidentifikasi keterangan diri (*Identity Validation*). Ini digunakan untuk memastikan identitas seseorang yang berhubungan melalui www sehingga mereka dapat diidentifikasi dan kegiatan mereka dapat ditelusuri. Setelah diidentifikasi, kontrol menghubungkan mereka dengan bagian program dimana berbagai aktivitas dapat dilakukan berdasarkan permintaan pengguna. Permintaan ini dapat dipenuhi dengan menggunakan *database* ILE atau melalui permintaan yang dikirimkan melalui ILE.

Rekayasa Perangkat Lunak

Rekayasa Perangkat Lunak (*Software Engneering*) adalah ilmu pengetahuan yang menyatukan proses, metode dan peralatan untuk mengembangkan sebuah *software* komputer. Saat ini dikenal dua pendekatan *software engineering*, yaitu pendekatan secara konvesional dan pendekatan berorientasi object (*Object Oriented*). Pendekatan *object oriented* saat ini banyak diimplementasikan menggunakan UML (*Unified Modeling Language*). UML adalah suatu bahasa (notasi dan aturannya) untuk memodelkan rancangan *software* dengan pendekatan *object oriented*.

***Unified Modeling Language* (UML)**

UML adalah bahasa pemodelan yang digunakan untuk mendeskripsikan suatu aktivitas. UML dapat digunakan untuk memodelkan proses bisnis, *software* didalam segala tahapan pengembangan dan untuk semua tipe sistem dan juga untuk memodelkan semua konstruksi yang mempunyai struktur statis dan sifat dinamis.

Diagram-diagram di dalam UML

Diagram adalah grafik aktual yang menunjukan simbol elemen model yang diataur untuk menggambarkan bagian tertentu atau aspek suatu sistem. Suatu model sistem umumnya mempunyai beberapa diagram dari tiap-tiap tipe. Diagram adalah bagian dari sudut pandang yang spesifik dan ketika digambarkan biasanya dialokasikan ke suatu suatu sudut pandang. Beberapa tipe diagram dapat menjadi bagian dari beberapa sudut pandang tergantung dari isi diagram.

Bagian ini menjelaskan konsep dasar di balik tiap-tiap diagram. Diagram diambil dari tipe-tipe yang berbeda dari suatu sistem untuk memperlihatkan keberagaman UML.

Diagram *Use case* (*Use case diagram*)

Diagram *uses case* memperlihatkan sejumlah aktor *external* dan hubungannya dengan pengguna yang disediakan sistem. *Use case* digambarkan hanya sebagai pandangan dari luar dari *actor* (aktor yang menggunakan kelakuan sistem), dan tidak menggambarkan bagaimana fungsi-fungsi dijalankan di dalam sistem.

Diagram Kelas (*Class diagram*)

Sebuah diagram kelas menggambarkan struktur statis dari kelas didalam sistem. Kelas-kelas dapat berhubugan satu sama lain didalam beberapa cara: asosiasi (terhubung satu sama lainnya), dependensi (satu kelas tergantung/menggunakan kelas lainnya), specialisasi (satu kelas adalah sebuah spesialisasi dari kelas lainnya). Semua hubungan ini digambarkan didalam sebauh diagram kelas bersama dengan struktur dalam dari kelas didalam arti atribut dan operasi.

Diagram Objek (*Object diagram*)

Sebuah diagram objek adalah berupa variasi dari diagram kelas dan menggunakan notasi yang sama. Sebuah diagram objek adalah sebuah contoh dari diagram kelas yang menunjukan sebuah

kemungkinan eksekusi dari sistem. Diagram objek dapat digunakan untuk memperlihatkan suatu diagram kelas kompleks dengan memperlihatkan kejadian sesungguhnya dan seperti apa hubungannya.

Diagram Keadaan (*State diagram*)

Diagram keadaan tidak digambarkan untuk semua kelas, hanya untuk yang mempunyai sejumlah keadaan yang telah didefinisikan dengan jelas dan dimana sifat suatu kelas dipengaruhi oleh keadaan yang berbeda. Diagram keadaan juga dapat digambarkan untuk sistem sebagai suatu keseluruhan.

Diagram Urutan (*Sequence diagram*)

Diagram urutan memperlihatkan suatu kolaborasi dinmis antara sejumlah objek. Aspek penting dari diagram ini adalah memperlihatkan suatu urutan-urutan pesan yang dikirim antar objek. Ia juga memperlihatkan suatu interaksi antar objek, suatu yang akan terjadi pada suatu titik tertentu dalam pelksanaan sistem.

Diagram Kolaborasi (*Collaboration diagram*)

Diagram kolaborasi memperlihatkan suatu kolaborasi dinamis, seperti diagram urutan. Sering merupakan pilihan antara menunjukkan sebuah kolaborasi sebagai diagram urutan atau sebagai diagram kolaborasi.

Penggunaan diagram kolaborasi dan diagram urutan seringkali dapat ditentukan dengan: jika waktu atau urutan merupakan aspek yang paling penting untuk dipertimbangkan, pilihlah diagram urutan; jika konteks yang penting utnuk dipertimbangkan maka pilihlah diagram kolaborasi.

Diagram Aktivitas (*Activity diagram*)

Diagram ini biasanya digunakan untuk menunjukkan aktivitas yang dijalankan didalam sebuah operasi, meskipun ia dapat digunakan untuk mendeskripsikan aliran aktivitas lainnya, seperti *use case* atau suatu interaksi. Diagram ini terdiri dari keadaan-keadaan aksi yang mengandung spesifikasi suatu aktivitas yang akan dilakukan (aksi).

Diagram Komponen (*Componen diagram*)

Diagram komponen memperlihatkan struktur fisik kode dalam bentuk kompnen-komponen kode. Suatu komponen terdiri informasi mengenai kelas *logic* atau mengkelaskan penggunaannya yang membuat pemetaan dari sudut pandang *logic* ke sudut pandang komponen. Komponen-komponen dapat juga diperlihatkan dengan antarmuka apa saja mereka gunakan, seperti antarmuka OLE/COM dan mereka sapat digrupkan bersama-sama dalam paket.

Diagram Penegmbangang (*Development diagram*)

Diagram pengembangan memperlihatkan *arsitektur* dari *hardware* dan *software* dalam sistem. Kita dapat memperlihatkan komputer dan alat (titik) sesungguhnya, bersama-sama dengan hubungan yang mereka miliki antara yang satu dengan yang lainnya, kita juga dapat memperlihatkan tite-tipe hubungan.

Proses Penegembangan *Software* Berbasis *Object Oriented*

Proses Penegembangan *Software* Berbasis *Object Oriented* dimulai dengan tahapan-tahapan konsep dan prinsip-prinsip *Object Oriented* sebagai berikut:

Object Oriented Analysis

Tujuan dari *Object Oriented Analysis* adalah untuk memodelkan dunia nyata sehingga dapat dimengerti. Langkah-langkah *generic* dari *Object Oriented Analysis* adalah: Penentuan persyaratan pelanggan untuk sistem *Object Oriented Analysis* (OOA), yaitu identifikasi *scenario* atau *use case* dan pembangunan suatu model persyaratan, Pemilihan kelas dan objek dengan menggunakan persyaratan dasar sebagai panduan, Pengidentifikasian atribut dan operasi untuk masing-masing objek sistem, Penentuan struktur dan hirarki yang mengorganisasi kelas, Pembangunan suatu model hubungan objek dan Pengkajian model analisis *object oriented* (OO) terhadap *user case* skenario.

Object Oriented Design

Sistem *design* mengembngkan detail *arsitektur* yang dibutuhkan untuk membangun sebuah sistem atau produk. Proses *design* sistem terdiri dari beberapa aktivitas sebagai berikut: Partisi model analisis ke dalam subsistem, Pengidentifikasian konkurensi yang ditentukan oleh masalah, Pengalokasian subsistem ke prosesor dan tugas-tugas, Pemilihan strategi dasar bagi pengimplementasian manajemen data, Pengidentifikasian sumber-sumber daya global dan mekanisme *control* yang diperlukan untuk mengaksesnya, Pendesainan mekanisme *control* yang sesuai untuk sistem tersebut, Pengkajian dan pertimbangan *trade-offs*

Pemograman *Object Oriented*

Sudut pandang rekayasa perangkat lunak menekankan *Object Oriented Analysis* (OOA) dan *Object Oriented Design* (OOD) serta memperhatikan *Object Oriented Programming* (OOP), pengkodean sebagai hal yang penting tetapi merupakan aktivitas sekunder juga merupakan hasil pertumbuhan dari analisis dan desain. Pada saat kompleksitas sistem bertambah, *arsitektur* desain dari produk akhir berpengaruh kuat terhadap keberhasilan dari pada bahasa pemrograman yang telah digunakan.

Pengujian *Object Oriebted*

Untuk menguji sistem *object oriented* secara memadai arus dilakukan tiga hal berikut: Definisi pengujian harus diperluas untuk mencakup teknik penemuan kesalahan yang diaplikasikan ke model *object oriented analysis* (OOA) dan *object oriented design* (OOD), Strategi untuk pengujian unit dan terintegrasi harus berubah secara signifikan dan Desian *test case* harus bertanggung jawab terhadap karakteristik unti perangkat lunak *object oriented.*

ANALISA DAN PERANCANGAN

Analisis

Pada bagian analisis ini akan digunakan pendekatam *object oriented.*

Object Oriented Analysis

Sasaran analisis *object oriented* adalah mengembangkan sederetan model yang menggambarkan perangkat lunak komputer pada saat perangkat itu bekerja untuk memenuhi serangkaian persyaratan yang ditentukan oleh pihak yang akan menggunakan. Untuk memenuhi sasaran tersebut maka sejumlah langkah generik harus dilakukan sebagai berikut:

Merinci Keperluan Pengguna

Pada sistem kuliah secara elektronik ini terdapat tiga pengguna utama, yaitu mahasiswa, dosen dan administrator. Masing-masing dari mereka mempunyai kebutuhan spesifik yang harus dipenuhi oleh sistem.

Mengidentifikasi Skenario atau *Use Case*

Untuk memenui kebutuhan tersebut maka perlu dipikirkan cara yang mudah untuk menyebarkan dan membuat informasi dan secara langsung membantu mahasiswa di dalam belajar. Dari tabel 2. kebutuhan pengguna dapat dibuat spesifikasi *use case* yang dipakai untuk pembuatan diagram *use* case. Spesifikasi-spesifikasi *use case* yang digunakan di dalam sistem kulai secara elektronik ini adalah sebagai berikut :

1) *Use Case Login*

Actor : Pemakai sistem kuliah: dosen, administrator dan mahasiswa (*user*).
Tujuan : Memperoleh akses menggunakan sistem kuliah secara elektronik.
Overview : Ketika pertama kali masuk ke dalam sistem *user* akan melakukan *login* untuk menda-

patkan hak akses terhadap fungsi-fungsi sistem perkuliahan. Sistem akan meminta data *user* dan *password*. Setelah itu sistem akan melakukan verifikasi untuk menentukan apakah *user* telah memiliki akses ke dalam sistem, seperti terlihat pada tabel 2.

Jalur Alternatif:

Jalur 4: Jika *password* atau data tidak ada maka sistem akan memberikan pesan bahwa data *login* yang dimasukkan salah.

Tabel. 2 *Use Case Login*

Actor Avtion	*System Response*
1. *User* memberikan identitas dan *password* serta menekan tombol *login*	2. Sistem akan melakukan verifikasi terhadap identitas, *password* dan IP *adress* dari *client* yang diberikan oleh *user*
3. *User* menunggu hasil *login*	4. Jika proses *login* berhasil maka sistem akan menampilkan *form* baru berisi menu utama yang siap dipergunakan

2) *Use Case* Memasukkan Materi Kuliah

Actor : Dosen (*user*).
Tujuan : Memasukkan materi kuliah yang digunakan sistem kuliah secara elektronik.
Overview : Dosen memasukkan deskripsi dan mteri kuliah yang akan digunakan oleh sistem kuliah

cara elektronik. Dosen akan menunjukkan *file-file* yang akan yang akan dimasukkan ke dalam sistem. Setelah itu sistem akan membuat *directory* khusu untuk menyimpan

materi kuliah dan meng*copy* materi kuliah tersebut ke dalam *directory* terebut, seperti terlihat pada tabel 3.

Tabel. 3 *Use Case* Memasukkan Materi Kuliah

Actor Avtion	*System Response*
1. Dosen akan memasukkan deskripsi dan materi kuliah yang akan dimasukkan	2. Sistem akan membuat *folder directory untuk meletakkan file-file materi kuliah*
	3. Sistem akan meng-*copy file-file* materi kuliah ke dalam *folder directory* yang telah disediakan
	4. Sistem menyimpan deskripsi dan alamat *file* ke dalam *database*.

3) *Use Case* Mencari Materi Kuliah

Actor : Dosen dan Mahasiswa (*user*).
Tujuan : Mencari materi kuliah yang berhubungan dengan mata kuliah.
Overview : *User* memasukkan kata kunci ke dalam sistem. Sistem kemudian akan mencari *file yang*

berhubungan dengan kata kunci tersebut. Sistem akan memberikan laporan hasil pencarian kepada *user*, seperti terlihat pada tabel 4.

Tabel. 4 *Use Case* Mencari Materi Kuliah

Actor Avtion	*System Response*
1. Dosen akan memasukkan kata kunci dari materi yang di cari	2. Sistem akan mencocokkan kata kunci dengan materi kuliah yang ada di dalam *database*
	3. Sistem memberikan laporan menegenai hasil pencarian *database*.

4) *Use Case* Memberikan Soal-Soal

Actor : Dosen (*user*).
Tujuan : Memasukan soal-soal dan jawaban ke dalam *database*.
Overview : Dosen memasukkan soal-soal dan jawaban ke dalam sistem. sistem kemudian akan me-

nyimpan soal-soal dan jawaban tersebut ke dalam *database*. Sistem memberikan laporan penyimpanan soal-soal dan jawaban, seperti terlihat pada tabel 5.

Tabel. 5 *Use Case* Memberikan Soal-Soal

Actor Avtion	*System Response*
1. Dosen akan memasukkan soal-soal yang telah dibuat ke dalam *database*	2. Sistem akan menyimpan soal-soal tersebut ke dalam *database*
	3. Dosen memberikan laporan penyimpanan soal-soal dan jawaban.

5) *Use Case* Memberikan Nilai Mahasiswa

Actor : Dosen (*user*).
Tujuan : Memberikan nlai mahasiswa.
Overview : Jika jawaban mahasiswa berupa pilihan ganda maka jawaban akan secara otomatis dibandingkan dengan kunci jawaban dan langsung menghitung nilai yang didapat mahasiswa sedangakan kalau jawaban berupa praktek atau *esai* maka jawaban akan dikirimkan kepada dosen melalui *e_mail*, seperti terlihat pada tabel 6.

Tabel. 6 *Use Case* Memberikan Nilai Mahasiswa

Actor Avtion	*System Response*
1. Mahasiswa akan memberikan jawaban kepada sistem	2. Sistem akan menentukan jenis jawaban berupa pilihan ganda atau *esai*
	3. Sistem akan menyamakan jawaban pilihan gand dengan kunci jawaban dan mengirimkan jawaban *esai* melalui *e_mail*.
4. Dosen akan menilai jawaban *esai* dan mengisinya ke sistem	5. Sistem akan menghitung nilai hasil jawan *esai*
	6. Sistem menghitung dan menampilkan nilai jawaban total

6) *Use Case* Menjawab Soal-Soal

Actor : Mahasiswa (*user*).
Tujuan : Menyimpan jawaban mahasiswa.
Overview : Mahasiswa menjawab soal-soal dan mengirim soal-soal jawaban kepada sistem. Sistem

akan menilai batasan waktu mahasiswa menjawab soal-soal dan berapa kali pengiriman jawaban. Sistem akan menyimpan mahasiswa di dalam *database*, seperti terlihat pada tabel 7.

Tabel. 7 *Use Case* Menjawab Soal-Soal

Actor Avtion	*System Response*
1. Mahasiswa akan memberikan jawaban kepada sistem	2. Sistem akan menentukan jawaban mahasiswa yang tidak melebihi batasan waktu dan yang mengirim hanya satu kali yang diperiksa
	3. Sistem akan menyimpan jawaban mahasiswa dan waktu pengerjaan ke dalam sistem.

7) *Use Case* Berdiskusi

Actor : Mahasiswa, administrator dan dosen (*user*).
Tujuan : Berdiskusi di antara aktor.
Overview : *User login* diskusi dan mengirimkan tujuan diskusi ke dalam sistem. Sistem akan menentukan ketersediaan *user* untuk berdiskusi. Jika tersedia maka sistem akan membuka hubungan diskusi di antara kedua *actor*, seperti terlihat pada tabel 8.

Tabel. 8 *Use Case* Berdiskusi

Actor Avtion	*System Response*
1. *User* memasukkan *login* dan tujuan diskusi	2. Sistem akan memeriksa kesediaan *user* tujuan untuk berdiskusi
3. *User* menunggu hubungan komunikasi diskusi	4. Sistem akan menciptakan hubungan antara kedua *user* untuk berdiskusi.

8) *Use Case* Menugaskan Dosen

Actor : Administrator (*user*).
Tujuan : Memberikan tugas kepada dosen untuk mengajar suatu mata kuliah.
Overview : *Adminstrator* mengisikan kesediaan dosen untuk mengajar dan jenis mata kuliah dosen.

Sistem akan memeriksa kebutuhan mata kuliah dan menyesuaikan dengan kesediaan dan kemampuan dosen. Sistem akan membuat beberapa kombinasi yang dapat diterapkan pada mata kuliah tersebut. Administrator memilih kombinasi tugas dosen yang dipakai oleh sistem *database*, seperti terlihat pada tabel 9.

Tabel. 9 *Use Case* Menugaskan Dosen

Actor Avtion	*System Response*
1. *Administrator* mengisikan kesediaan dosen untuk mengajar dan jenis mata kuliah	2. Sistem akan memeriksa kebutuhan mata kuliah dan menyesuaikan dengan kesediaan dan kemampuan dosen
3. *Administrator* akan menentukan kombinasi tugas mana yang paling baik untuk diterapkan	4. Sistem akan memberikan beberapa pilihan kombinasi tugas yang dapat diterapkan pada mata kuliah tersebut.
	5. Sistem akan menyimpan tugas yang dipilih oleh *administrator*.

9) *Use Case* Membuka Mata Kuliah

Actor : *Administrator* (*user*).
Tujuan : Membuka mata kuliah berdasarkan kebutuhan para mahasiswa.
Overview : *Sistem* menampilkan mata kuliah yang dapat dibuka pada tahun ajaran tersebut. Administrator memutuskan mata kuliah yang dapat dibuka pada saat itu berdasarkan kebutuhan mahasiswa. Sistem menyimpan mata kuliah yang dibuka dalam *database*, seperti terlihat pada tabel 10.

Tabel. 10 *Use Case* Membuka Mata Kuliah

Actor Avtion	*System Response*
1. *Administrator* melihat mata kuliah yang dapat di buka	2. Sistem menampilkan mata kuliah yang dapat dibuka pada tahun ajaran tersebut
3. *Administrator* menentukan mata kuliah yang dibuka pada tahun ajaran tersebut berdasarkan kebutuhan para mahasiswa	4. Sistem akan menyimpan mata kuliah yang di buka dalam *database*.

10) *Use Case* Membuat Jadwal

Actor : Administrator (*user*).
Tujuan : Membuka jadwal mata kuliah yang optimal untuk orang atau materi.
Overview : *Administrator* melihat mata kuliah yang dibuka, dosen yang tersedia dan ruangan yang dapat di pakai untuk proses belajar mengajar. Sistem menampilkan mata kuliah, dosen dan ruangan yang dapat dipakai, serta membuat beberapa kombinasi yang dapat dipilih. Administrator memilih kombinasi yang tepat. Sistem menyimpan kombinasi itu dalam *database*, seperti terlihat pada tabel 11.

Tabel. 11 *Use Case* Membuka Mata Kuliah

Actor Avtion	*System Response*
1. *Administrator* melihat mata kuliah yang di buka, dosen yang tersedia dan ruangan yang dapat dipakai untuk proses belajar mengajar.	2. Sistem menampilkan mata kuliah, dosen dan ruangan yang dapat dipakai serta membuat beberapa kombinasi yang dapat dipilih.
3. *Administrator* memilih kombinasi yang tepat.	4. Sistem menyimpan kombinasi itu dalam *database*.

11) *Use Case* Membuka Kelompok Belajar

Actor : Administrator (*user*).

Tujuan : Membuka kelompok belajar mahasiswa dan menilai kelompok tersebut.

Overview : *Administrator* melihat data mahasiswa dan memutuskan kelompok mahasiswa diben-

tuk berdasarkan kelas tradisional atau kelompok mahasiswa khusus yang dipilih. Sistem akan membuat beberapa pilihan kombinasi kelompok mahasiswa. Administrator memilih kombinasi dan memberikan tugas ke setiap kelompok tersebut. Sistem akan menyimpan data tugas dan kombinasi ke dalam *database* Administrator menilai kemampuan setiap kelompok dan menyimpannya ke dalam *database*, seperti terlihat pada tabel 11.

Tabel. 11 *Use Case* Membuka Kelompok Belajar

Actor Avtion	*System Response*
1. *Administrator* melihat mata kuliah mahasiswa dan memutuskan kelompok mahasiswa dibentuk berdasarkan kelas tradisional atau kelompok mahasiswa khusus yang dipilih	2. Sistem akan membuat beberapa pilihan kombinasi kelompok mahasiswa.
3. *Administrator* memilih kombinasi dan mem-berikan tugas ke setiap kelompok tersebut	4. Sistem akan menyimpan data tugas dan kombinasi ke dalam *database*.
5. *Adminstrator* menilai kemampuan setiap kelompok dan menyimpannya ke dalam *database*	6. Sistem menyimpan nilai setiap kelompok.

Memilih Kelas Dan Objects Menggunakan Keperluan Dasar Sebagai Tuntutan

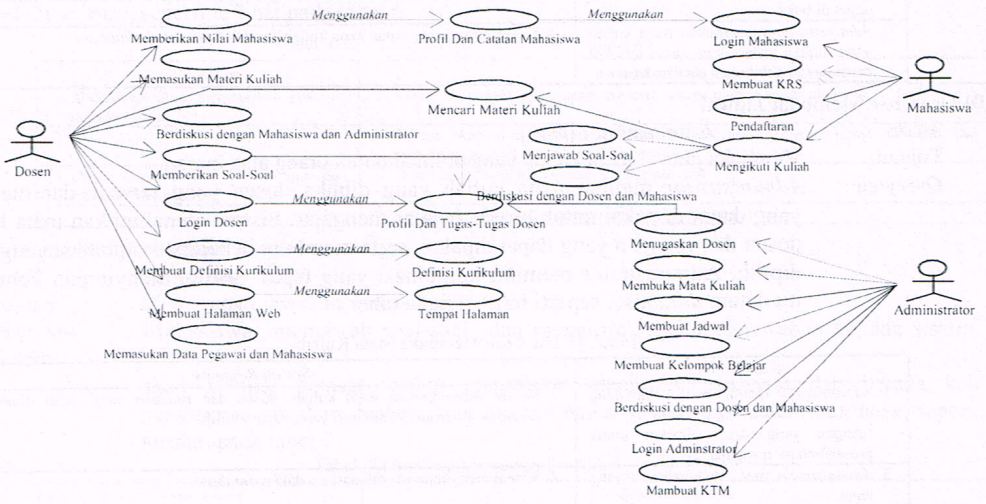
Pencocokan Identitas

ILE untuk dapat melakukan semua tugas-tugasnya harus mengetahui identitas dari penggunanya. Identitas dapat dibagi menjadi dua bagian sebagai berikut: 1) Pengguna harus mempunyai *account* dan *password*, 2)Alamat IP dari komputer yang digunakan *client* harus dapat diketahui.

Sementara metode pertama sudah lama digunakan untuk pencocokan identitas, metode kedua juga sangat berguna ketika mahasiswa berhubungan untuk aktivitas tertentu.

Layer Pemrograman

Arsitektur *layer* pemrograman adalah yang paling rumit dari pada bagian sebelumnya. Sekilas program ini hanya mengumpulkan informasi dari *database* mereka dan mengubah formatnya menjadi halaman Web. Program ini juga melakukan hal lainnya yang sangat berbeda. *Web* dirancang untuk berdiri sendiri sehinga tidak perlu untuk mempunyai informasi dari satu halaman untuk berinteraksi dengan halaman lainnya. Program yang diperlukan untuk ILE akan membutuhkan informasi yang tetap harus disimpan. Sebagai contoh berikut ini dapat dengan mudah menggambarkan hal ini.



Use Case Sistem Kuliah Seacara Elektronik

Peraturan dosen yang mengharuskan mahasiswanya mengambil ujian hanya sekali. Untuk melaksanakan peraturan ini maka diperlukan pengetahuan apakah mahasiswa yang bersangkutan sudah mengambil ujian ini sebelumnya dimana mahasiswa berinteraksi. Hal ini memerlukan rekaman yang dapat digunakan untuk ujian sehingga setiap mahasiswa yang mengirimkan ujian pada detik kedua dapat diblokade.

Seorang mahasiswa mungkin mempunyai batasan waktu pada ujian *online* yang mereka ambil. Ketika ujian dikembalikan perlu diketahui waktu pengerjaan dan menentukan apakah mereka telah melebihi batasan waktu mereka. Hal ini juga memerlukan rekaman untuk menjaga ujian diberikan kepada mahasiswa. Sebagai contoh ketika mahasiswa ingin melakukan tes *online* sangat penting untuk mengetahui di mana mahasiswa itu berada. Sistem perlu menjaga sejumlah daftar alamat IP yang tersedia untuk ujian.

Contoh diatas menggambarkan alam *stateless* dari web yang dibutuhkan *server* dalam menyimpan dan mendistribusikan informasi. Layer pemograman dapat membangun fasilitas untuk memenuhi kebutuhan tersebut, tetapi membutuhkan penggunaan yang sangat luas dari *database*.

Untuk pendaftaran mahasiswa dibutuhkan beberapa kelas sebagai berikut: formulir pendaftaran, kontrol pendaftaran yang berhubungan dengan mata kuliah, catatan mahasisw, jadwal dan mata kuliah. Catatan mahasiswa jug harus berhubungan dengan jadwal untuk melihat tugas-tugas, jadwal berhubungan dengan data mata kuliah.

Proses belajar mengajar memerlukan sistem layanan yang sangat banyak antara lain kontrol mengikuti kuliah, kontrol ruang diskusi, kontrol soal-soal dan kontrol pencarian materi kuliah. Pada kontrol mengikuti kuliah mahasiswa dapat otomatis mengisi absensi, menjawab soal-soal, mengikuti kuliah, berdiskusi dengan dosen dan mahasiswa lain. Pada inti dari sistem kuliah secara elektronik ini adalah proses belajar yang dialami oleh mahasiswa harus mengisi absensi kemudian mengikuti kuliah, berdiskusi dan akhirnya menjawab soal-soal yang diberikan oleh dosen dan hasilnya kalau pilihan ganda dapat langsung diperiksa oleh komputer dan kalau berupa *esai* langsung dikirimkan ke *e_mail* dosen yang bersangkutan. Proses belajar mengajar ini penting sekali untuk dijaga komunikasi dua arah antara dosen dengan mahasiswa, untuk itu diperlukan ruang diskusi untuk menjaga komunikasi antara dosen dengan mahasiswa. Mahasiswa juga harus mengisi kembali KRS mereka sendiri.

Para dosen juga diharapkan memasukkan definisi kurikulum ke dalam *database*. Dosen juga diberikan fasilitas absensi langsung ke *database* dan fasilitas memasuka soal ujian. Kegiatan *administrator* adalah membuat jadwal, kelompok belajar, membuat KTM dan menugaskan dosen.

Database

Database ini akan berisi informasi mengenai partisipasi manusia di dalam lingkungan, kurikulum dan materi pendidikan untuk mendukungnya, jadwal dan penugasan sumber daya dan contoh halaman untuk meletakkan informasi didalamnya.

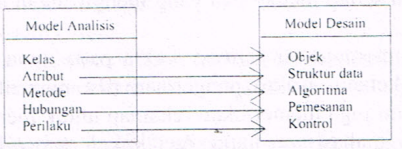
Database akan digunakan oleh program *service* yang menyediakan informasi yang berkaitan. *Database* juga akan digunakan untuk merancang lingkungan pengguna. Seorang mahasiswa yang sedang *log on* ke sistem akan mendapatkan data dari aktivitas yang dapat dimasuki. Berdasarkan *query* sebelumnya, lingkungan akan dapat juga lebih jauh menyaring hasil dari *query* sekarang ini.

Perancangan Sistem Berorientasi Objek

Selama perancangan subsistem, perlu ditentukan empat komponen perancangan yang penting sebagai berikut: Domain masalah, yaitu subsistem yang bertanggung jawab terhadap pengimplementasin persyaratan pelanggan secara langsung. Ineteraksi manusia, yaitu subsistem yang mengimplementasikan *interface* pemakai (menyangkut subsistem GUI *reuseable*). Manajemen tugas, yaitu subsistem yang bertanggun jawab dalam pengontrolan dan pengkoordinasian tugas-tugas konkuren yang dapat dikemas pada subsistem yang berbeda, dan Manajemen data, yaitu subsistem yang bertanggung jawab dalam penyimpanan dan pemanggilan kembali objek-objek.

Masing-masing komponen generik dapat dimodelkan (selama OOA) dengan serangkaian kelas serta hubungan dan tingkah laku yang diharuskan. Komponen desain juga diimplementasikan dengan menentukan *protocol* yang secara formal menggambarkan model pemesanan bagi masing-masing komponen.

Begitu subsistem ditentukan dan desain masing-masing komponen tersebut di mulai, penekanan bergeser ke desain objek. Pada tingkat ini elemen model CRC di terjemahkan ke dalam *realisasi* desain. Secara mendasar penerjemahan OOA ke dalam OOD dapat digambarkan pada gambar 3. berikut.



Translasi model analisis kedalam objek model desain selama desain objek

Proses Desain Sistem

Langkah-langkah desain yang dilakukan dalam perancangan ini adalah sebagai berikut: Penentuan partisi model analisa ke dalam subsistem, Pengidentifiasian konkurensi yang ditentukan oleh masalah, Pengalokasian subsistem ke dalam *processor* dan tugas-tugas, Pemilihan strategis dasar bagi pengimplementasian manajemen data, Pengidentifikasian sumber-sumber daya global dan mekanisme kontrol yang diperlukan untuk mengaksesnya, Pembuatan desain mekanisme kontrol yang sesuai untuk sistem tersebut dan Pengkajian dan pertimbangan *trade-off*.

Penentuan Partisi Model Analisa Ke Dalam Subsitem

Dalam desain sistem *Oject Oriented* dilakukan partisi terhadap model analisis untuk menentukan kumpulan kelas, hubungan dan tingkha laku. Elemen desain ini dikemas sebagai subsistem. Secara umum, semua elemen subsistem berbagi pakai beberapa properti. Properti dapat dilibatkan untuk melakukan fungsi-fungsi yang sama, dapat tinggal dalam perangkat keras produk yang sama. Subsistem dikarakterisasikan oleh tanggung jawab, yaitu bahwa subsistem dapat diidentifikasikan oleh pelayanan yang diberikan.

IMPLEMENTASI PROTOTITE DAN PENGUJIAN SISTEM

Implemetasi Prototipe

Aplikasi sistem kulah secara elektronik yang digunakan pada pengujian prototipe ini berbasis PHP berfungsi sebagai berikut, menempatkan data pengguna ke dalam *database*, menempatkan *'content'* materi kuliah ke dalam *database* dan mengcopy *'content'* ke dalam folder yang telah ditentukan.

Data institusi dipakai untuk *header* pada setiap halaman di dalam *web page* dan berisiskan data institusi, data fakultas dan data jurusan dan data program studi. Pada tampilan pengguna mata kuliah ini pengguna dapat memasukkka data mata kuliah yang mendukung jalannya mata kuliah ke dalam *database*. Aplikasi materi kuliah berisi aplikasi yang sangat berguna utnuk mencari materi kuliah, menjalankan materi kuliah, menaru alamat *file* materi kuliah ke dalam *database*.

Website sistem kuliah secara elektronik berfungsi sebagai tempat penyebaran informasi yang terdapat di dalam *database* dan diterjemahkan kedalam *user interface* yang menarik.

Pengujian Sistem

Pada pengujian sistem ini program aplikasi dan *websaite* dari sistem kuliah secara elektronik dengan metodologi pengujian adalah sebagai berikut :

Aspek-Aspek Yang Diuji

Aspek-aspek pengujian yang dilakukan pada aplikasi sistem kuliah elektronik antara lain: Hubungan antara program aplikasi dengan *database* (menambahkan data, memperbaharui data, mendelete data) dan Hubungan *website* dengan *database* (menambah, memperbaharui dan mendelete data).

Langkah-Langkah Pengujian

Langkah-langkah pengujian dilakukan sebagai berikut: Memasukka data melalui aplikasi sistem kuliah secara elektronik dan mencoba semua fasilitas di dalamnya dan Menggunakan *website* dengan mencoba beberapa *error* yang digunakan.

Konfigurasi Pengujian

Pengujian dilakukan pada sebuah komputer intel P4 1500, 512 MB, HD 80 GB 7200 rpm, dengan operting system Windows XP Profesional.

Hasil dan Analisa Hasil Pengujian

Hasil dari aplikasi sistem kuliah secara elektronik adalah sebagai berikut: Aplikasi dapat memasukkan data ke dalam *database MySQL*, Aplikasi dapat melakukan pencarian *file*, Aplikasi dapat mengcreate *folder* dan memindahkan *file* ke *folder* tersebut, Aplikasi dapat memanggil program lain yang berkaitan dengan ekstensi *file* tersebut dan Aplikasi dapat memutar *file audio visual*.

Hasil dari pengujian *website* kuliah secara elektronik adalah sebagai berikut: *Login website* dapat dijalankan, Semua warna pada *backgrund, text* dan tabel dapat diubah sesuai dengan keinginan pengguna, *Website* dapat menampilkan materi kuliah yang telah dimasukkan, Program *chating* pada *website* dapat berjalan dengan baik dan Program *database guest book* dapat berjalan dengan baik.

KESIMPULAN DAN SARAN

Kesimpulan

Pada artikel ini telah dirancang prototipe perangkat lunak aplikasi sistem kuliah secara elektronik dengan spesifikasi-spesifiksi sebagai berikut:

1. Memasukkan dan mengedit isi kuliah secara elektronik ke dalam suatu *database.*
2. Mengkonversikan isi kuliah ke dalam format *web* yang dapat dijalankan pada suatu *web server.*
3. Kelengkapan-kelengkapan fasilitas dalam format *web* yang dapat dijalankan adalah *login,* materi kuliah, *chating, guest book,* pendaftaran, isi krs dan menjawab soal-soal *testing online.*

Saran

Sistem kulias secara elektronik ini diharapkan dapat dikembangkan untuk sistem kuliah elektronik dengan *database* dan *website* yang lebih lengkap untuk suatu perguruan tinggi.

DAFTAR PUSTAKA

1. Craig Larman, *Applying UML and Patterns, Prentice Hall PTR Upper Saddle,* New Jersey-USA, 1997.
2. Daniel Minola, *Distance Learning, Internacional Journal of Education Telecomunications,* 1996.
3. Ericsson, Hans-Erick and Penker, Magnus, *UML Toolkit,* John Wiley and Sons, Inc, USA, 1998.
4. Harvey M. Dietel, Paul J. Dietel, *Internet and World Wide Web ow to Program,* Prentice Hall, 2000.
5. Locledge, Jeffery, Giester, Donald, Mitchell, Lance and Go, Phill, *An Internet Based Learning.* IEEE Prcedings, 1996.
6. Oetomo Dharma B.S, S.Kom.,MM, *e-Education,* Penerbit ANDI Yogyakarta, 2002.
7. Presman Roger S., *Software Engineering,* McGraw-Gill, New York, 2001.
8. Suhendar A. S.Si,Gunadi H. S.Si., MT, *Visual Modeling Menggunakan UML dan Rational Rose,* Informatika Bandung, 2002.
9. Suruali, Nasir, Artikel, *Perancangan Prototipo Sistem Tes Keberhasilan Pembelajaran Berbasis Komputar,* Jurnal Teknologi, Fakultas Teknik Unpatti Vol. 5, 2008.
10. Tanembaum, Andrew S., *Computer Networks*, Edisi 3, Prentice Hall, 1996.
11. Wazdi, Farid Dr. Ing, *Sistem Pendidikan Berbasis Teknologi Informasi dari Distance-Learning Menuju Electronic-Education*, Seminar Perkembangan Internet dan Teknologi Informasi, Pekan Baru, 24-25 Juni 2000.