

BAB 2

LANDANSAN TEORI

2.1 Sistem Informasi

Sistem informasi yaitu suatu sistem yang menyediakan informasi untuk manajemen dalam mengambil keputusan dan juga untuk menjalankan operasional perusahaan, di mana sistem tersebut merupakan kombinasi dari orang-orang, teknologi informasi dan prosedur-prosedur yang terorganisasi (Leod & Arbie, n.d.).

Biasanya suatu perusahaan atau badan usaha menyediakan semacam informasi yang berguna bagi manajemen. Sebagai contoh: Perusahaan toko buku mempunyai sistem informasi yang menyediakan informasi penjualan buku-buku setiap harinya, serta stock buku-buku yang tersedia, dengan informasi tersebut, seorang manajer bisa membuat keputusan, stock buku apa yang harus segera mereka sediakan untuk toko buku mereka, manajer juga bisa tahu buku apa yang paling laris dibeli konsumen, sehingga mereka bisa memutuskan buku tersebut jumlah stocknya lebih banyak dari buku lainnya.

2.1.1 Pengertian Sistem

Definisi Sistem menurut O'Brien (2009:24) adalah sekumpulan komponen yang saling berhubungan yang bekerja bersama-sama untuk mencapai suatu tujuan dengan menerima *input* dan menghasilkan *output* melalui proses transformasi yang terorganisir. Definisi sistem menurut Williams dan Sawyer (2007:552) adalah kumpulan dari komponen-komponen yang saling berhubungan

yang saling berinteraksi untuk melakukan suatu tugas untuk mencapai suatu tujuan. Sedangkan definisi sistem informasi menurut Turban (2009:31) memiliki beberapa komponen dasar yaitu perangkat keras, perangkat lunak, *database*, jaringan, prosedur, dan manusia.

Sementara menurut Mc. Leod, (2004:10) mendefinisikan sistem sebagai sekelompok elemen-elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan. Sumber daya mengalir dari elemen *output* dan untuk menjamin prosesnya berjalan dengan baik maka dihubungkan mekanisme kontrol. Mc. Leod menggambarkan bahwa adanya mekanisme proses yang merubah *input* menjadi *output* yang dibutuhkan, namun proses transformasi dari *input* menjadi *ouput* tersebut harus memiliki mekanisme kontrol agar sesuai dengan tujuan dari adanya perubahan *input* menjadi *output* tersebut.

Menurut Kristanto (2003) analisis sistem adalah orang yang mempunyai kemampuan untuk menganalisis sebuah sistem, memilih alternatif pemecahan masalah dan menyelesaikan masalah tersebut dengan menggunakan komputer sistem yaitu sekelompok elemen-elemen yang saling terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan. Informasi adalah data yang telah diproses, atau data yang memiliki manfaat / kegunaan.

2.1.2 Pengertian Informasi

Menurut Jogiyanto (1995) informasi adalah penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan

Informasi adalah data yang diolah menjadi bentuk yang berguna untuk membuat keputusan.

Informasi berguna untuk pembuat keputusan karena informasi menurunkan ketidakpastian (atau meningkatkan pengetahuan). Informasi menjadi penting, karena berdasarkan informasi itu para pengelola dapat mengetahui kondisi obyektif kantor Infordepe Dili Timor-Leste. Informasi tersebut merupakan hasil pengolahan data atau fakta yang dikumpulkan dengan metode ataupun cara-cara tertentu.

2.1.3 Sistem Informasi di Timor-Leste

Sejak kemerdekaan Timor-Leste pada tahun 2002 dan pada tahun 1999 di bawah pemerintahan PBB, infrastruktur dan sistem birokrasi berkembang begitu cepat. Berdasarkan prinsip pemerintahan yang baik (*Good Governance*), Timor-Leste telah memprioritaskan pengelolaan administrasi pemerintahan, oleh karena itu sebagian instansi pemerintahan RDTL sudah memiliki beberapa sistem informasi untuk menunjangnya. Karena terbatasnya sumber daya manusia, maka sistem informasi yang saat ini hanya sebatas informasi yang menunjang proses bisnis. Secara umum sampai saat ini pemerintahan RDTL belum memiliki sistem informasi persuratan yang terpadu jadi proses pencatatan persuratan dilakukan manual dengan menggunakan aplikasi microsoft excel.

Sistem Informasi di IFMISU-Kementrian Keuangan Timor-Leste pada unit IFMISU sudah memiliki beberapa sistem informasi menunjang yang terkait dengan pengelolaan keuangan namun sejauh ini belum memiliki sistem informasi penunjang administrasi persuratan. Sistem informasi dan organisasi memiliki

hubungan yang saling terkait satu sama lain. Sistem informasi harus selalu disesuaikan dengan organisasi. Organisasi harus mampu mengoptimalkan sistem informasi sehingga mendapatkan keuntungan dari teknologi-teknologi yang ada. Interaksi antara teknologi informasi dan organisasi sangat dipengaruhi oleh faktor mediasi, yaitu lingkungan, kultur, struktur, prosedur baku, proses bisnis, politik, keputusan manajemen, dan peluang (Nugroho & TaxSys, n.d.) anajer harus mampu memahami sistem informasi, karena sangat akan mempengaruhi kehidupan organisasi. Manajer perlu memilih sistem apa dan bagaimana yang akan dibangun didalam organisasi.



Gambar 2.1 Faktor-Faktor Mediasi Antara Organisasi dan Teknologi Informasi

2.2 Pengertian Surat

Pengertian surat menurut para ahli. Menurut Atmosudirdjo (2001:139) surat adalah sehelai kertas yang ditulis (pada waktu ini umumnya diketik) atas nama pribadi penulis, atau atas nama kedudukanya dalam organisasi, yang ditujukan pada suatu alamat tertentu dan memuat sesuatu “bahan komunikasi “.

Surat menurut Barthos (2009:36) adalah alat komunikasi tertulis yang berasal dari satu pihak dan ditujukan kepada pihak lain untuk menyampaikan warta.

Sedangkan pengertian surat menurut Silmi (2008:1) adalah sehelai kertas atau lebih yang digunakan untuk mengadakan komunikasi secara tertulis.

Menurut Rahardi (2008: 12) surat adalah pernyataan tertulis dari pihak satu ke pihak lain, atas nama perseorangan ataupun atas nama jabatan.

Dari beberapa pendapat diatas maka dapat disimpulkan mengenai pengertian surat yaitu sarana atau wahana komunikasi tertulis yang ditujukan kepada orang lain atau suatu instansi dengan tujuan untuk menyampaikan suatu hal baik itu berupa informasi, perintah atau sebuah pemberitahuan.

2.2.1 Defenisis Surat Masuk

Surat masuk adalah semua jenis surat yang diterima dari instansi lain maupun perorangan, baik yang diterima melalui pos, maupun yang diterima dari kurir dengan mempergunakan buku pengiriman.

2.2.2 Defenisis Surat Keluar

Surat keluar adalah surat yang lengkap (bertanggal, bernomor, berstempel dan ditandatangani oleh pejabat yang berwenang) yang dibuat oleh suatu instansi atau lembaga lain. Surat keluar biasanya dikirim melalui pos atau kurir.

Dari pengertian-pengertian di atas dapat disimpulkan bahwa prosedur pengelolaan surat masuk dan surat keluar adalah pekerjaan surat-menyurat yang harus dilakukan secara tertata dan berurutan dengan kegiatan utama yaitu

mengelola, mengatur dan mengurus surat-menyurat agar dapat memperlancar administrasi instansi tersebut.

Dalam pengelolaan surat dapat diselenggarakan dengan menggunakan dua cara yaitu:

1. Buku Agenda

Menurut Agus Sugiarto (2005:27) pencatatan dengan buku agenda dilakukan oleh instansi yang belum menerapkan sistem kartu kendali. Pencatatan surat masuk dan surat keluar dapat dipisahkan dengan menggunakan buku agenda surat masuk dan buku agenda surat keluar, yang biasanya dibedakan tahunnya.

a. Buku Agenda Tunggal

Buku agenda tunggal adalah buku yang digunakan untuk mencatat surat masuk dan surat keluar sekaligus berurutan pada tiap-tiap halaman.

b. Buku agenda berpasangan

Buku agenda berpasangan adalah buku yang dipergunakan untuk mencatat surat masuk dihalaman kiri dan surat keluar pada halaman kanan dengan nomor surat sendiri.

c. Buku agenda kembar

Buku agenda kembar adalah buku yang digunakan untuk mencatat surat masuk dan surat keluar sendiri.

2. Kartu Kendali

Menurut Sudarmayanti (2003:85) pengurusan dengan menggunakan kartu kendali disebut dengan sistem kearsipan pola baru. Kartu kendali adalah helai tipis berukuran 10 x 15 cm berisi kolom-kolom untuk mencatat surat masuk dan surat keluar serta untuk mengendalikan surat tersebut. Kartu kendali berfungsi sebagai pengganti buku agenda, yang mana penggunaannya dapat ditulis rangkap

dua, rangkap tiga atau rangkap empat, sesuai dengan kebutuhan masing-masing kantor.

2.3 Metode Perancangan Model Aplikasi

2.3.1 *Unified Modeling Language (UML)*

Menurut (Nugroho, 2009:4), *Unified Modeling Language (UML)* adalah Metodologi kolaborasi antara metode-metode Booch, *Object Modeling Technique (OMT)*, serta OOSE (*Object Oriented Software Engineering*) dan beberapa metode lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk analisis dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa “Pemrograman Berorientasi Objek” (OOP).



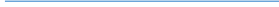
Menurut (Booch, 2005:7) UML adalah bahasa standar untuk membuat rancangan *software*. UML biasanya digunakan untuk menggambarkan dan membangun, dokumen artifak dari *software intensive system*.

Model UML yang dipakai dalam Sistem Informasi Perusahaan dan Kepegawaian antara lain adalah *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, ER Diagram. Bagian dari UML antara lain:

1. *Use Case Diagram*

Use Case Diagram mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem yang dibuat. Dapat dikatakan *Use Case* digunakan untuk mengetahui fungsi yang ada di dalam sistem dan siapa saja yang berhak menggunakan fungsi-fungsi yang dibutuhkan. Simbol-simbol yang digunakan dalam *use case diagram* dapat dilihat pada Tabel 2.1.





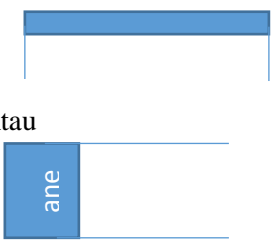
Tabel 2.1 Simbol-Simbol Dalam Use Case Diagram

No	Gambar	Nama Gambar	Keterangan
1		<i>Use Case</i>	Merupakan fungsionalitas yang disediakan sistem sebagai unit yang bertukar pesan dengan actor.
2		<i>Actor</i>	Simbol ini menunjukkan pengguna atau pelaku dalam suatu sistem.
3		<i>Association</i>	Digambarkan dengan garis tanpa panah yang mengindikasikan siapa yang berinteraksi secara langsung dengan sistem.

2. Activity Diagram

Activity Diagram merupakan diagram yang menggambarkan tentang aktifitas yang terjadi pada sistem. Dari pertama sampai akhir, diagram ini menunjukkan langkah-langkah dalam proses kerja sistem yang kita buat. Struktur diagram ini juga mirip dengan *flowchart*. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada Tabel 2.2.






Tabel 2.2 Simbol-Simbol Dalam Activity Diagram

No	Gambar	Nama Gambar	Keterangan
1		<i>Star Point</i>	Merupakan awal dalam aktifitas.
2		<i>End Point</i>	Merupakan akhir dalam aktifitas.
3		<i>Activities</i>	Menggambarkan suatu aktivitas yang dilakukan sistem
4		<i>Decision Point</i>	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
5	 Atau	<i>Swimlane</i>	Digunakan untuk pembagian <i>activity diagram</i> yang menunjukkan siapa yang melakukan aktifitas.

3. *Class Diagram*

Class Diagram yaitu salah satu jenis diagram pada UML yang digunakan untuk menampilkan kelas-kelas maupun paket-paket yang ada pada suatu sistem yang nantinya akan digunakan. Jadi diagram ini dapat memberikan sebuah gambaran mengenai sistem maupun relasi-relasi yang terdapat pada sistem tersebut. *Class Diagram* secara khas meliputi: Nama Kelas (*Class Name*), Atribut (*Attributes*), Operasi (*Operations*), dan Relasi (*Relationships*). Simbol-simbol yang digunakan dalam *activity* diagram dapat dilihat pada Tabel 2.3.

Tabel 2.3 Simbol-Simbol Dalam *Class Diagram*

No	Gambar	Nama Gambar	Keterangan
1		<i>Association</i>	Relasi antara kelas dengan makna umum, <i>Association</i> biasanya juga disertakan dengan <i>multiplicity</i> .
2		<i>Direct Association</i>	Relasi antara kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. <i>Direct Association</i> biasanya juga disertakan dengan <i>multiplicity</i> .
3		<i>Generalisasi</i>	Relasi antara kelas dengan makna generalisasi-spesialis (umum khusus).
4		<i>Dependency</i>	Relasi antara kelas dengan makna kebergantungan antara kelas.
5		<i>Aggregation</i>	Relasi antara kelas dengan makna semua bagian.




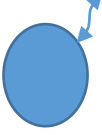






4. *Sequence Diagram*

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah event untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal.

Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya,

Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Simbol-Simbol Dalam *Sequence Diagram*

No	Gambar	Nama Gambar	Keterangan
1		<i>Actor</i>	Menggambarkan orang yang berinteraksi dengan sistem.
2		<i>Entity Class</i>	Menggambarkan hubungan kegiatan yang akan dilakukan
3		<i>Boundary Class</i>	Menggambarkan sebuah penggambaran dari <i>form</i> .
4		<i>Control Class</i>	Menggambarkan penghubung antara <i>boundary</i> dengan tabel.
5		<i>Life Line</i>	Menggambarkan tempat mulai dan berakhirnya sebuah pesan.
6		<i>Line Message</i>	Menggambarkan pengiriman pesan.
7		<i>Object</i>	Menyatakan objek yang berinteraksi pesan.
8		<i>Life Line</i>	Menyatakan kehidupan suatu objek.
9		<i>Active Time</i>	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya aktor tidak memiliki waktu aktif.
10		<i>Stimulus</i>	Menyatakan suatu objek mengirimkan pesan untuk menjalankan operasi yang ada pada objek lain.

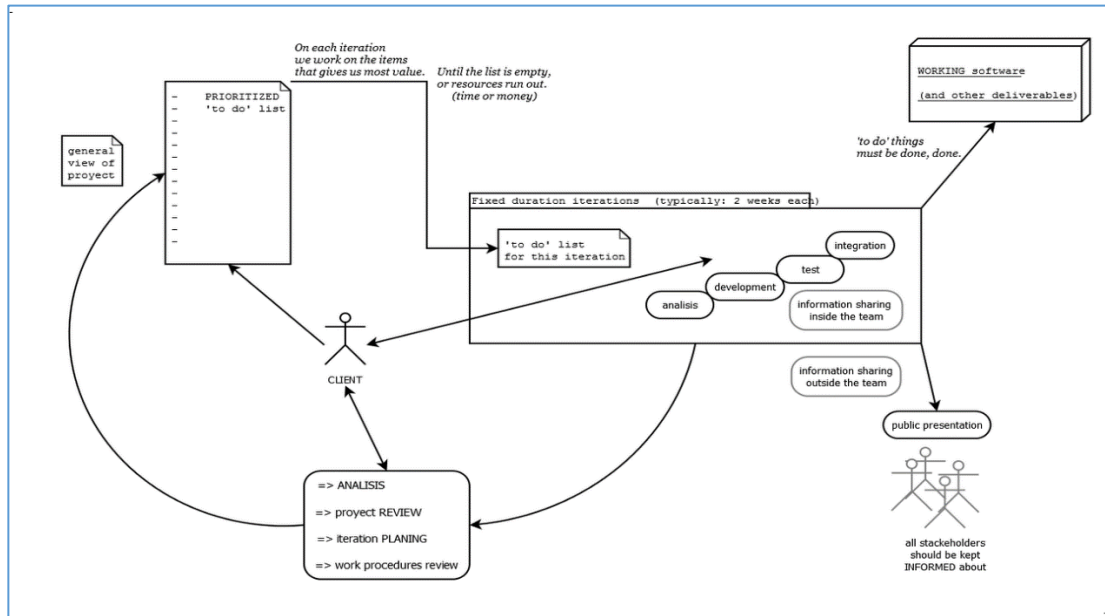
2.4 Metode Pengembangan *Software*

Software untuk manajemen proyek masuk yang ada disekolah, sehingga pemilik (*owner*) bisa mengetahui dengan mudah apa-apa saja proyek yang sedang ditangani, proyek yang belum selesai, dan proyek yang sudah selesai. dan bisa melihat profit perusahaan tiap bulannya. Menurut Pressman (2010:3), ruang lingkup rekayasa perangkat lunak:

1. *Software Requirements*, berhubungan dengan spesifikasi kebutuhan dan persyaratan perangkat lunak.
2. *Software* desain, mencakup proses penampilan arsitektur, komponen, antar muka, dan karakteristik lain dari perangkat lunak.
3. *Software construction*, berhubungan dengan detail pengembangan perangkat lunak, termasuk. algoritma, pengkodean, pengujian dan pencarian kesalahan.
4. *Software testing*, meliputi pengujian pada keseluruhan perilaku perangkat lunak.
5. *Software maintenance*, mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan.

2.4.1 Metode Agile Secara Umum

Metode Agile merupakan salah satu dari beberapa metode yang digunakan dalam pengembangan software. Metode Agile adalah jenis pengembangan sistem jangka pendek yang memerlukan adaptasi cepat dan pengembang terhadap perubahan dalam bentuk apapun (wikipedia, n.d.).



Gambar 2.2 Diagram dari Agile Development Methods (wikipedia, n.d.)

Agile development methods terdefinisi dalam empat nilai, biasa di sebut Agile Alliance's Manifesto, (wikipedia, 2018) diantaranya :

- Interaksi dan personel lebih penting daripada proses dan alat
- Perangkat lunak yang berfungsi lebih penting daripada dokumentasi yang lengkap
- Kolaborasi dengan klien lebih penting daripada negosiasi kontrak
- Respon terhadap perubahan lebih penting daripada mengikuti rencana

Pengertian dari Agile Alliance's Manifesto dijelaskan di bawah ini:

1. Interaksi dan personel lebih penting daripada proses dan alat, di dalam agile interaksi antar anggota tim sangatlah penting, karena tanpa adanya interaksi yang baik maka proses pembuatan perangkat lunak tidak akan berjalan sesuai rencana.
2. Perangkat lunak yang berfungsi lebih penting daripada dokumentasi yang lengkap, saat melakukan proses demonstrasi kepada klien, perangkat lunak

yang berfungsi dengan baik akan lebih berguna daripada dokumentasi yang lengkap.

3. Kolaborasi dengan klien lebih penting daripada negosiasi kontrak, salah satu ciri dari agile adalah klien menjadi bagian dari tim pengembangan perangkat lunak. Kolaborasi yang baik dengan klien saat proses pembuatan perangkat lunak sangatlah penting ketika menggunakan agile. Karena fungsi-fungsi dari perangkat lunak yang dikembangkan harus terus menerus dibicarakan dan diimprovisasi disesuaikan dengan keinginan klien.
4. Respon terhadap perubahan lebih penting daripada mengikuti rencana, agile development methods berfokus terhadap kecepatan respon tim ketika klien menginginkan perubahan saat proses pembuatan perangkat lunak.

Agar suatu tim berhasil dalam menerapkan agile development methods, maka tim tersebut harus mengikuti dua belas prinsip yang ditetapkan oleh Agile Alliance- (wikipedia, 2018) yaitu :

1. Prioritas utama proses agile adalah memuaskan klien dengan menghasilkan perangkat lunak yang bernilai dengan cepat dan rutin.
2. Menyambut perubahan kebutuhan, walaupun terlambat dalam pengembangan perangkat lunak. Proses Agile memanfaatkan perubahan untuk keuntungan kompetitif klien.
3. Menghasilkan perangkat lunak yang bekerja secara rutin, dari jangka waktu beberapa minggu sampai beberapa bulan, dengan preferensi kepada jangka waktu yang lebih pendek.

4. Rekan bisnis dan pengembang perangkat lunak harus bekerja sama tiap hari sepanjang proyek.
5. Kembangkan proyek di sekitar individual yang termotivasi. Berikan mereka lingkungan dan dukungan yang mereka butuhkan, dan percayai mereka untuk menyelesaikan pekerjaan dengan baik.
6. Metode yang paling efisien dan efektif untuk menyampaikan informasi dari dan dalam tim pengembang perangkat lunak adalah dengan komunikasi secara langsung.
7. Perangkat lunak yang bekerja adalah ukuran utama kemajuan.
8. Proses agile menggalakkan pengembangan berkelanjutan. Sponsor-sponsor, pengembang-pengembang, dan pengguna-pengguna dapat mempertahankan kecepatan tetap secara berkelanjutan.
9. Perhatian yang berkesinambungan terhadap keunggulan teknis dan rancangan yang baik meningkatkan Agility.
10. Kesederhanaan (memaksimalkan sumber daya yang tersedia) adalah hal yang amat penting.
11. Arsitektur, kebutuhan, dan rancangan perangkat lunak terbaik muncul dari tim yang dapat mengorganisir diri sendiri.
12. Secara berkala, tim pengembang berefleksi tentang bagaimana untuk menjadi lebih efektif, kemudian menyesuaikan dan menyelaraskan kebiasaan bekerja mereka.

Dua belas prinsip tersebut menjadi suatu dasar bagi tim agar sukses menerapkan agile development methods. Dengan prinsip-prinsip tersebut agile

berusaha untuk menyasati tiga masalah yang biasanya dihadapi saat proses pembuatan perangkat lunak, yaitu:

- Kebutuhan perangkat lunak sulit diprediksi dari awal dan selalu akan berubah. Selain itu, prioritas klien juga sering berubah seiring berjalannya proyek.
- Desain dan pembangunan sering tumpang tindih. Sulit diperkirakan seberapa jauh desain yang diperlukan sebelum pembangunan.
- Analisis, desain, pembangunan dan testing tidak dapat diperkirakan seperti yang diinginkan.

2.4.2 Model Proses Agile

Ada beberapa model proses agile yang sangat populer seperti Extreme Programming dan Scrum. Berikut penjelasan singkat dari metode tersebut:

A. *Extreme Programming*

Menurut Kent Back, definisi XP adalah sebagai berikut: “*Extreme Programming (XP) is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop software* “. Model ini cenderung menggunakan pendekatan Object-Oriented. Tahapan-tahapan yang harus dilalui antara lain: *Planning, Design, Coding, dan Testing*. Sasaran Extreme Programming adalah tim yang dibentuk berukuran antara kecil sampai medium saja, tidak perlu menggunakan sebuah tim yang besar. Hal ini dimaksudkan untuk menghadapi *requirements* yang tidak jelas maupun terjadinya perubahan-perubahan *requirements* yang sangat cepat. Extreme Programming merupakan *agile methods*

yang paling banyak digunakan dan menjadi sebuah pendekatan yang sangat terkenal.

✓ **Kelebihan *Extreme Programming***

- Meningkatkan kepuasan kepada klien
- Pembangunan system dibuat lebih cepat
- Menjalinkan komunikasi yang baik dengan client
- Meningkatkan komunikasi dan sifat saling menghargai antar developer

✓ **Kelemahan *Extreme Programming***

- Cerita-cerita yang menunjukkan requirements dari pelanggan kemungkinan besar tidak lengkap sehingga Developer harus selalu siap dengan perubahan karena perubahan akan selalu diterima
- Tidak bisa membuat kode yang detail di awal (prinsip simplicity dan juga anjuran untuk melakukan apa yang diperlukan hari itu juga)
- XP tidak memiliki dokumentasi formal yang dibuat selama pengembangan. Satu-satunya dokumentasi adalah dokumentasi awal yang dilakukan oleh user

B. *Scrum*

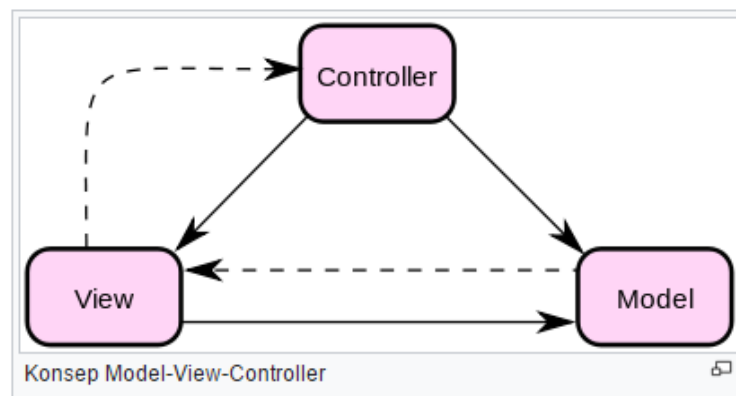
Scrum adalah sebuah metode iteratif yang termasuk dalam metode Agile tentang bagaimana cara Anda mengelola dan menjalankan sebuah proyek. Ini bisa digunakan untuk mengelola segala jenis proyek mulai dari pembuatan software, website, hardware, marketing, event planning, dan sebagainya.

Scrum membantu Anda untuk mengorganisir sebuah tim dan Anda harus memiliki komunikasi yang kuat antar member tim tersebut. Scrum mengatakan bahwa setiap “sprint” dimulai dengan meeting singkat untuk perencanaan dan

diakhiri dengan review. Ini adalah ide fundamental dari Scrum untuk sebuah project management.

2.4.3 Model View Controller (MVC)

Model-View-Controller adalah pola arsitektur yang memisahkan aplikasi dalam tiga komponen utama logis: Model, View dan Controller. Masing-masing komponen ini dibangun untuk menangani aspek-aspek tertentu. MVC adalah salah satu kerangka pembangunan web standar industri paling sering digunakan untuk menciptakan proyek yang terukur besar dan extensible.

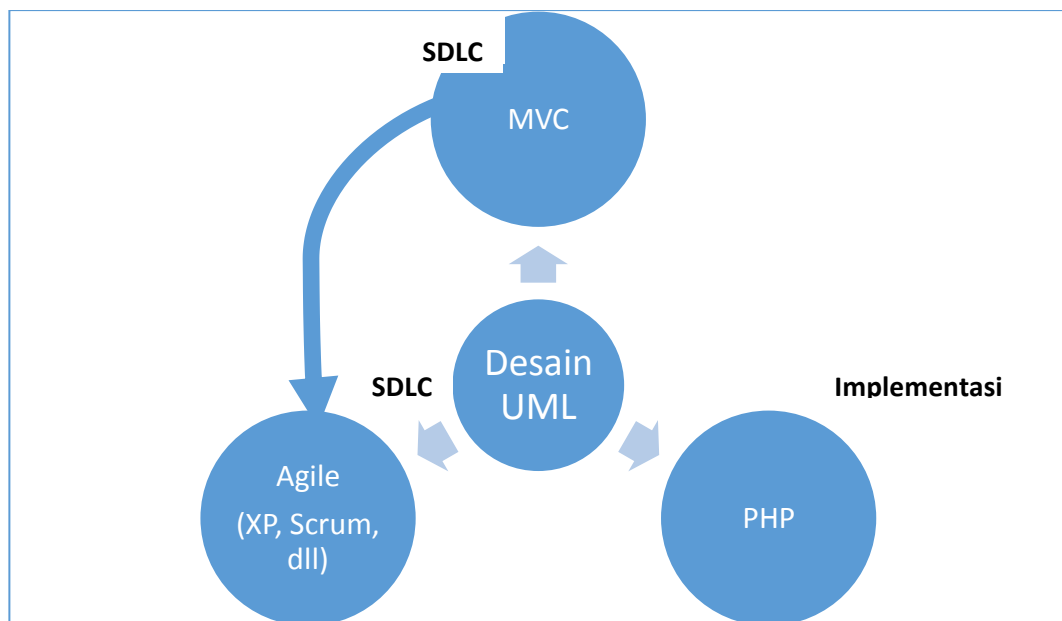


Gambar 2.3 Konsep Model View Controller

Pada gambar 2.3 diatas dijelaskan bahwa Model: Komponen Model yang sesuai dengan semua data yang terkait dengan penggunaan logika dalam berkerja. Ini dapat mewakili baik data yang ditransfer antara View dan Controller komponen atau logika bisnis lain data yang terkait. Sebagai contoh, sebuah objek pelanggan akan mengambil informasi pelanggan dari database, memanipulasi itu dan memperbarui data kembali ke database atau menggunakannya untuk membuat data.

View: Komponen View digunakan untuk semua UI (User Interface) pada logika aplikasi. Misalnya, tampilan pelanggan akan mencakup semua komponen UI seperti kotak teks, dropdown, dll yang digunakan pengguna untuk berinteraksi.

Controller: Controller bertindak sebagai antar muka antara Model dan View komponen proses semua logika bisnis dan permintaan masuk, memanipulasi data menggunakan komponen Model dan berinteraksi dengan View untuk membuat hasil akhir. Sebagai contoh, controller pelanggan akan menangani semua interaksi dan masukan dari View pelanggan dan update database menggunakan Model pelanggan. Controller sama akan digunakan untuk melihat data pelanggan.



Gambar 2.4 relasi Antara metode Agile, Desain, MVC, PHP

Pada gambar diatas dijelaskan bahwa implementasi sistem disusun melalui beberapa tahapan ilmu, pada penelitian ini dimulai dari desain UML yang digunakan dalam proses SDLC perancangan dengan menggunakan kombinasi

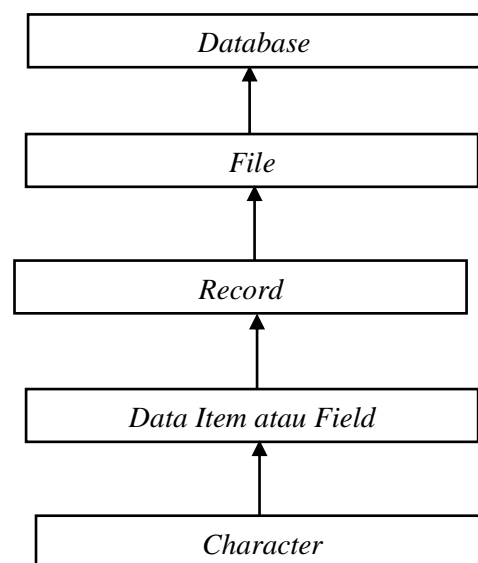
metode agile dan konsep MVC dimana proses codingnya menggunakan bahasa pemrograman PHP yang berorientasi object oriented programming (OOP).

2.5 Konsep Database

2.5.1 Pengertian Basis Data (*Database*)

Menurut James F. Countney Jr. dan David B. paradece (1944) dalam buku “*Database System For Management*”, menjelaskan sistem *database* adalah sekumpulan *database* yang dapat dipakai secara bersama-sama, personal-personal yang merancang dan mengelola *database*, teknik-teknik untuk merancang dan mengelolah *database*, serta komputer untuk mendukungnya (Sutabri, 2005).

Dari definisi diatas, penulis menyimpulkan bahwa sistem *database* mempunyai beberapa elemen penting, yaitu *database* sebagai inti sistem *database*, perangkat lunak untuk mengelola *database*, perangkat keras sebagai pendukung operasi pengolahan data, serta manusia mempunyai peran penting dalam sistem tersebut. Sampai dengan membentuk *database*, data mempunyai jenjang yang dapat pada gambar dibawah ini.



Gambar 2.4 Jenjang *DataBase*

1. *Characters*

Characters adalah bagian data yang terkecil, dapat berupa karakter *numeric*, huruf ataupun karakter-karakter khusus yang membentuk suatu *item* data atau *field*.

2. *Field*

Field menggambarkan suatu atribut dari *record* yang menunjukkan suatu item dari data, seperti nama, jenis kelamin, dan lain-lain. Kumpulan dari *field* membentuk suatu *record*.

✓ Nama *field* (*field name*)

Field harus diberi nama untuk membedakan *field* yang satu dengan *field* yang lain.

✓ Representasi dari *field* (*field representation*)

Representasi dari *field* menunjukkan tipe dari *field* (*field type*) Dapat berupa tipe *numeric*, karakter, tanggal, dan lain-lain. Serta lebar dari *field* menunjukkan ruang maksimum dari *field* yang dapat diisi dengan karakter-karakter data.

✓ Nilai dari *field* (*field value*)

Nilai dari *field* menunjukkan isi dari *field* untuk masing-masing *record*.

3. *Record*

Record adalah kumpulan dari *field* yang membentuk suatu *record*. Kumpulan dari *record* membentuk *file*. Misalnya *file* pegawai, tiap-tiap *record* dapat mewakili data tiap-tiap pegawai.

4. *File*

File terdiri dari *record-record* yang menggambarkan satu kesatuan data yang sejenis. Misalnya *file* pangkat berisi tentang semua pangkat yang ada.

2.5.2 SQL

SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. DML dari SQL, DML (*data manipulation language*) adalah perintah yang digunakan untuk mengoperasikan atau memanipulasi isi *database*.

Empat perintah DML tersebut diantaranya:

- a. *Select*: digunakan untuk mengambil data dari *database*.
- b. *Delete*: digunakan untuk menghapus data pada *database*.
- c. *Insert*: menambahkan data pada *database*.
- d. *Update*: memodifikasi data pada *database*.

2.6 Bahasa Pemrograman

2.6.1 XAMPP

Xampp merupakan hasil kerja seorang bernama Rasmus Lerdorf pada tahun 1995. Namun kemudian PHP berkembang dan tidak hanya merupakan proyek pribadi Rasmus, PHP ditulis ulang dan dengan banyak menambahkan fungsi-fungsi baru oleh Zeev Suraski Dan Andi Gotmants (disingkat Zend) dan lahirlah 3 pada tahun 1998 (Astamal, 2006). PHP adalah bahasa *server-side scripting* yang didesain khusus untuk *web*. Pada halaman HTML dapat

ditempelkan (*embed*) kode PHP. Kode PHP dieksekusi disisi server bukan dikomputer klien. Dan hasil yang ditampilkan adalah kode HTML (Astamal, 2006).

Maksud dari *server-side scripting* adalah sintaks dan perintah-perintah yang di berikan akan sepenuhnya dijalankan di server tetapi disertakan pada dokumen HTML biasa. Pembuatan web ini merupakan kombinasi antara PHP sendiri sebagai bahasa pemrograman dan HTML sebagai pembangun halaman *web*, PHP dikenal sebagai bahasa *scripting* yang menyatu dengan *tag* HTML, dieksekusi di server dan digunakan untuk membuat halaman *web* yang dinamis, PHP adalah merupakan *software* yang *open source* dan mampu lintas *platform*.

2.6.2 XAMPP MySQL

Pengertian MySQL menurut MySQL *manual* adalah sebuah *open source software database SQL (Search Query Language)* yang menangani sistem manajemen *database* dan sistem manajemen *database relational*. MySQL didistribusikan secara gratis dibawah lisensi GPL (*General Public License*) MySQL mempunyai fitur-fitur yang sangat mudah dipelajari bagi para penggunanya dan dikembangkan untuk menangani *database* yang besar dengan waktu yang lebih singkat. Kecepatan, konektivitas dan keamanannya yang lebih baik membuat MySQL sangat dibutuhkan untuk administrasi basisdata MySQL berbasis *web* yang sangat populer yaitu phpMyAdmin.

2.6.3 PHP

PHP adalah singkatan dari "Hypertext Preprocessor", yaitu bahasa pemrograman yang digunakan secara luas untuk penanganan pembuatan dan

pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML. PHP diciptakan oleh Rasmus Lerdorf pertama kali tahun 1994. PHP merupakan bahasa pemrograman web yang di gunakan untuk mengenerate atau menghasilkan kode HTML, dengan demikian kode program yang ditulis dalam PHP tidak akan terlihat oleh user sehingga keamanan halaman web lebih terjamin. PHP dirancang untuk membuat halaman web yang dinamis, yaitu halaman web yang dapat membentuk suatu tampilan berdasarkan permintaan terkini, seperti menampilkan isi basis data ke halaman web. Salah satu framework php yang powerful adalah framework php codeigniter.

2.6.4 Framework

Framework PHP Codeigniter adalah kerangka kerja yang juga dapat diartikan sebagai kumpulan script (terutama class dan function) yang dapat membantu developer/programmer dalam menangani berbagai masalah-masalah dalam pemrograman seperti koneksi ke database, pemanggilan variabel, file, dan lain-lain, sehingga developer lebih fokus dan lebih cepat membangun aplikasi. Bisa juga dikatakan Framework adalah komponen pemrograman yang siap re-use kapan saja, sehingga programmer tidak harus membuat script yang sama untuk tugas yang sama dari awal.

Codeigniter adalah kerangka kerja pemrograman umum web yang bisa dipakai untuk mengembangkan semua jenis aplikasi web. Oleh karena Codeigniter sangat ringan dan dilengkapi dengan solusi caching yang memuaskan, ia sangat cocok untuk pengembangan aplikasi dengan lalu lintas-tinggi, seperti

portal, forum, sistem manajemen konten (CMS), sistem e-commerce, dan lain-lain (Wikipedia, 2018).

Fitur-fitur yang ada pada codeigniter:

- Menggunakan pola MVC, pola standart pengembangan aplikasi yang memisahkan tampilan, logic program, dan modelnya
- Menggunakan database abstraction layer seperti Data Access Objects (DAO) dan Active Record untuk memudahkan interaksi antar database.
- Terintegrasi dengan JQuery Javascript Framework. Walaupun menggunakan JQuery sebagai internal javascript library, namun Codeigniter dapat menggunakan librari lain tanpa terjadi bentrok.
- Mendukung Internationalization (I18N) and localization (L10N) untuk memudahkan pengembangan aplikasi dalam multi bahasa dan lokasi seperti penggunaan waktu dan tanggal.
- Memiliki layer cache untuk cache data, halaman, sebagian, dan keseluruhan aplikasi sehingga dapat meningkatkan performance dengan beragam pilihan media cache. Penggunaan media cache seperti database, APC, memcache, dan sebagainya mudah diatur tanpa melakukan perubahan besar pada kode.
- Fitur penanganan error dan logging, sehingga memudahkan pengembangan dalam melakukan debugging aplikasi dalam masa pengembangan aplikasi.
- Penggunaan theme, memudahkan pengembangan aplikasi dalam merancang tampilan aplikasi.

- Console, penggunaan perintah pada console untuk melakukan beragam perintah otomatis seperti mengenerate struktur dasar aplikasi, model, crud, dan sebagainya.
 - Dukungan Authentication dan authorization internal sehingga memudahkan pengembangan aplikasi dengan fitur autentikasi.
 - Widget, semacam kontrol yang memiliki fungsi seperti auto complete, datapicker, table dan lain – lain. Menggunakan JQuery sebagai javascript client sidernya.
 - Form input dan validasi. Memudahkan pengembangan untuk bekerja dengan form pada aplikasi dan melakukan validasi input dari form.
 - Modular dan mudah ditambahkan dengan dukungan extensi dan komponen tambahan sehingga beragam fitur tambahan dengan mudah dimasukkan.
- ✓ **Kelebihan Framework Codeigniter**

Ada beberapa keuntungan yang dapat kita ambil dalam penggunaan framework PHP Codeigniter berikut. Keuntungannya antara lain adalah:

- Codeigniter adalah salah satu framework yang sangat ringan dan dilengkapi dengan solusi caching yang memuaskan.
- Codeigniter sangat cocok untuk pengembangan aplikasi dengan lalu lintas-tinggi, seperti portal, forum, sistem manajemen konten (CMS), sistem e-commerce, dll.
- Codeigniter didokumentasikan dengan jelas, efisien, dan kaya-fitur.

✓ **Kekurangan Framework Codeigniter**

- Tidak banyak kekurangan yang ditemui pada penggunaan Codeigniter. Codeigniter cukup mencakup segala macam fitur-fitur yang diperlukan dalam membangun sebuah web. Namun, bukan berarti Codeigniter tidak mempunyai kekurangan. Salah satu kekurangan Codeigniter yang cukup signifikan adalah tidak kompatibelnya Codeigniter dengan PHP
- Dengan demikian, anda dapat mempertimbangkan MVC Framework sebagai kerangka besar dibangun di atas framework Codeigniter menyediakan set besar menambahkan fungsionalitas dengan fokus pada pembangunan berbasis komponen dan pengujian.

2.7 Kuesioner

Kuesioner atau bahasa aslinya *questionnaire* adalah salah satu bentuk alat atau instrument yang digunakan untuk mencari data, disamping wawancara, observasi, dan studi dokumentasi. Pada umumnya kuesioner berbentuk seperangkat pernyataan atau pernyataan tertulis yang harus dijawab oleh responden tanpa bantuan peneliti.

Sebagai alat pengambilan data maka kuesioner harus dirancang sedemikian rupa agar setiap butir pernyataan atau pernyataan yang ada di dalamnya valid. Valid artinya sesuai, cocok, dengan tujuan yang ingin dicapainya.

Bentuk kuesioner merupakan suatu teknik pengumpulan informasi yang memungkinkan analis mempelajari sikap-sikap, keyakinan. Prilaku dan karakteristik beberapa orang dalam suatu lingkungan yang dapat terpengaruh oleh

sistem yang diajukan atau sudah ada. Penelitian dengan menggunakan kuesioner termasuk penelitian kuantitatif.

Kuesioner terbagi menjadi dua jenis menurut jawabannya:

- a. Tertutup = Jawaban sudah disediakan berupa pilihan ganda yang harus dipilih oleh responden.
- b. Kuesioner terbuka = Kuesioner yang memberikan keleluasaan kepada responden untuk memberikan pendapat mengenai pernyataan yang diajukan.

Dijelaskan ada sembilan langkah dalam mengembangkan sebuah kuesioner:

- a. Informasi yang ingin diteliti dan dibutuhkan
- b. Mendefinisikan responden target
- c. Memilih metode dalam menjangkau responden target
- d. Menentukan isi kuesioner
- e. Mengembangkan kata-kata kuesioner
- f. Menyusun kuesioner dalam format dan susunan yang berarti
- g. ek panjang kuesioner
- h. Uji kuesioner
- i. Mengembangkan form kuesioner final

2.7.1 Populasi dan Sampel

Populasi adalah sekelompok objek yang menjadi sasaran penelitian. Merupakan keseluruhan penelitian. Populasi dibagi menjadi dua macam yaitu populasi *Finit* artinya jumlah individu ditentukan dan populasi *Infinit* ketika individu tak terbatas atau tidak diketahui jumlahnya dengan pasti. Populasi dalam

penelitian ini adalah pengguna sistem informasi sebagai *stakeholder* di IFMISU kementerian keuangan Dili Timor-Leste sebagai responden utama.

Sampel adalah sebagian dari jumlah dan karakteristik yang dimiliki oleh populasi. Sampel merupakan sebagian atau wakil dari populasi yang mempunyai sifat dan karakteristik yang sama serta memenuhi populasi yang diselidiki. Sampel dalam penelitian ini adalah 60 responden sebagai *stakeholder* di IFMISU kementerian keuangan Dili Timor-Leste yaitu Atasan IFMISU, kepala bagian dan IT Administrator dan staf di kementerian , serta para administrasi yang terlibat dalam penggunaan sistem.

2.7.2 Skala Likert

Skala *Likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dalam penelitian fenomena sosial ini telah ditetapkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variabel penelitian. Dengan Skala Likert, maka variabel yang akan diukur dijabarkan menjadi indikator variabel.

Table 2.5 Bobot Penilaian Kuesioner

No.	Bobot Interval	Skor
1	Sangat Setuju/Selalu/Sangat positif	5
2	Setuju/sering/positif	4
3	Ragu – ragu/kadang - kadang / netral	3
4	Tidak setuju/Hampir tidak pernah / negatif	2
5	Sangat Tidak Setuju/Tidak Pernah/Sangat Negatif	1

Rumus persentase untuk mendapatkan nilai persentase dari setiap pertanyaan adalah:

$$\text{Rumus Presentase} = \frac{\text{Frekuensi dari Setiap Jawaban } (f)}{\text{Jumlah Skor Ideal } (n)} \times 100\%$$

Frekuensi dari setiap jawaban dimaksudkan sebagai berikut bobot penilaian dikalikan dengan jumlah responden yang memilihnya, lalu ditotalkan semuanya. Sedangkan jumlah skor ideal (n) seluruh jumlah responden yang telah memilih dikalikan jumlah interval jawaban dari setiap pertanyaan kuisisioner. Rumus rata-rata untuk mendapatkan rata-rata nilai persentase adalah:

$$\text{Rumus Rata - Rata} = \sum \frac{\text{Total Presentase}}{N}$$

Total persentase dimaksudkan sebagai jumlah seluruh persentase yang telah ditotalkan, sedangkan N adalah jumlah seluruh pertanyaan kuesionernya.

2.7.3 Uji Validitas

Validitas merupakan suatu ukuran yang menunjukkan kevalidan atau kesahihan suatu instrumen. Pengujian validitas mengacu pada sejauh mana suatu instrumen dalam menjalankan fungsi. Instrumen dikatakan valid jika instrumen tersebut dapat digunakan untuk mengukur apa yang hendak diukur (Sugiyono,2012). Sebagai contoh, ingin mengukur kemampuan siswa dalam matematika. Kemudian diberikan soal dengan kalimat yang panjang dan yang berbelit-belit sehingga sukar ditangkap maknanya. Akhimya siswa tidak dapat menjawab, akibat tidak memahami pertanyaannya. Contoh lain, peneliti ingin mengukur kemampuan berbicara, tapi ditanya mengenai tata bahasa atau kesusastraan seperti puisi atau sajak. Pengukur tersebut tidak tepat (valid).

Validitas tidak berlaku universal sebab bergantung pada situasi dan tujuan penelitian. Instrumen yang telah valid untuk suatu tujuan tertentu belum otomatis akan valid untuk tujuan yang lain. Cara menentukan validitas dari sebuah instrumen dengan menggunakan rumus korelasi *product moment* atau dikenal dengan korelasi *pearson*. Adapun rumusnya adalah sebagai berikut:

$$r_{xy} = \frac{N \sum XY - (\sum X)(\sum Y)}{\sqrt{\{N \sum X^2 - (\sum X)^2\} \{N \sum Y^2 - (\sum Y)^2\}}}$$

Keterangan :

- R_{xy} = Koefisien korelasi
 N = Jumlah responden uji coba
 X = Skor tiap item
 Y = Skor seluruh item responden uji coba

2.7.4 Uji Reabilitas

Reliabilitas dapat diartikan sebagai indeks yang menunjukkan sejauh mana alat ukur yang digunakan dapat dipercaya atau dapat diandalkan, Sugiyono (2012) menambahkan sebuah instrumen dapat mengukur sesuatu yang diukur secara konsisten dari waktu ke waktu. Jadi kata kunci untuk syarat kualifikasi suatu instrumen pengukuran adalah konsistensi, atau tidak berubah-ubah. Keandalan ini dapat berarti berapa kalipun variabel-variabel kuesioner tersebut ditanyakan kepada responden yang berlainan hasilnya tidak akan menyimpang terlalu jauh dari rata-rata jawaban responden untuk variabel tersebut. Ada 3 macam teknik dalam pengukuran reliabilitas, yaitu teknik pengukuran ulang, konsistensi internal, dan teknik paralel. Dengan statistik uji:

- ✓ Membagi item-item yang sudah valid menjadi 2 bagian
- ✓ Skor pada belahan pertama diberi simbol Y1 dan belahan kedua diberi simbol Y2 sehingga akan diperoleh skor total Y1 dan Y2 yang dinotasikan X.
- ✓ Dicari varians antara belahan pertama dan kedua dan skor total Y1 dan Y2
Hitung dengan rumus:

$$r_{11} = \frac{n}{n-1} \left(1 - \frac{\sum_{i=1}^n s_i^2}{s_t^2} \right)$$

Keterangan:

r_{11} = Koefisien Reabilitas

n = Banyaknya Butir Soal

S_i^2 = Varian Skor Soal ke- i

S_t^2 = Varian Skor Total

2.7.5 Metode Analisi Komparatif

Pemilihan jenis uji parametrik atau non parametrik dalam pengujian hipotesis sangat berhubungan dengan distribusi data populasi yang akan diuji. Bila distribusi data populasi yang akan diuji berbentuk normal, maka proses pengujian dapat digunakan dengan pendekatan uji statistik parametrik. Sedangkan bila distribusi data tidak normal atau tidak diketahui distribusinya maka dapat digunakan pendekatan uji statistik non parametrik.

Uji beda rata-rata dilakukan untuk melihat perbandingan sistem informasi yang baru dibuat dengan sistem yang lama. Oleh karena itu sebelum pengujian hipotesis dilakukan, maka terlebih dahulu akan dilakukan pengujian normalitas

data dan homogenitas terlebih dahulu (Sugiyono, 2012:172). Menurut Ghozali (2011:32) untuk mendeteksi normalitas data dapat dilakukan dengan uji Kolmogorov-Smirnov. Tahapan dalam analisis komparatif adalah sebagai berikut:

1. Uji Normalitas Menggunakan Uji *Kolmogorov-Smirnov* Dua Sampel

Test ini digunakan untuk menguji hipotesis komparatif dua sampel independen bila datanya berbentuk ordinal yang telah tersusun pada tabel distribusi frekuensi kumulatif dengan menggunakan kelas-kelas interval (Sugiyono, 2012:326). Analisis hasil uji Kolmogorov-Smirnov dilakukan dengan melihat nilai *Asymp Significant* $> 0,05$, maka data terdistribusi normal (Ghozali, 2011:34).

2. Uji Komparatif *Mann-Whitney*

Uji *Mann-Whitney Test* merupakan pilihan uji non parametris apabila uji independen tidak dapat dilakukan oleh karena asumsi normalitas data tidak dapat terpenuhi. Uji ini menjadi alternatif apabila data tidak berdistribusi normal dan tidak homogen.

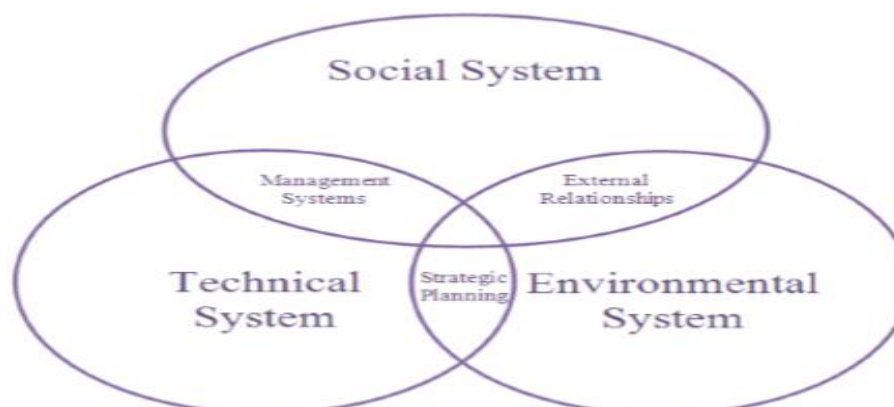
Uji *mann-whitney* sama halnya dengan uji beda independen *sample t-test*, akan tetapi *uji-mann whitney* digunakan untuk mengetahui dua sampel yang tidak berhubungan atau berpasangan satu sama lainnya.

Uji hipotesis menggunakan uji *mann-whitney test* dengan taraf signifikansi $\alpha = 5\%$. Dalam hal ini bila peluang kesalahan $\alpha = 5\%$ maka taraf kepercayaannya 95%. Artinya kesalahan pengambilan keputusan dalam menolak hipotesis yang benar adalah maksimal 5%. Pengambilan keputusan dalam penelitian ini dilakukan dengan kriteria sebagai berikut:

1. Jika nilai *Asymp.Sig (2-Tailed)* $< 0,05$, maka terdapat perbedaan yang signifikan, yang berarti terdapat peningkatan kinerja yang lebih baik dari sistem informasi yang baru dibuat dibandingkan dengan sistem yang sudah ada.
2. Jika nilai *Asymp.Sig (2-Tailed)* $> 0,05$, maka tidak terdapat perbedaan yang signifikan, yang berarti terdapat peningkatan kinerja yang tidak lebih baik dari sistem informasi yang baru dibuat dibandingkan dengan sistem yang sudah ada.

2.8 *Socio Technical System*

Socio-technical system (STS) adalah sistem yang mempertimbangkan persyaratan yang mencakup perangkat keras, perangkat lunak, pribadi, dan aspek komunitas. Ini menerapkan pemahaman tentang struktur sosial, peran dan hak (ilmu sosial) untuk menginformasikan rancangan sistem yang melibatkan komunitas orang dan teknologi.



Gambar 2.5 Socio Technical System

Faktor-faktor yang membuat Socio-Technical System mempermudah kegiatan manusia antara lain :

- 1) **Hardware** yang terdiri dari perangkat penghubung jaringan (kabel, router, hub, dll) sampai pada perangkat komputer pengguna.
- 2) **Software** merupakan bagian internal Socio-Technical dimana terjadi interaksi antara manusia dan komputer untuk menggerakkan sebuah system. Software sendiri menjadi penghubung interaksi para pengguna dan hardware yang digunakan. Namun penggunaan sebuah software dalam beberapa Socio-Technical yang berbeda dapat menyebabkan masalah yang disebabkan ketidakcocokan. Untuk itu sebaiknya setiap sistem Socio - Technical menggunakan software yang diciptakan khusus atau memang didesain hanya untuk sistem tersebut sehingga kesalahan bisa dikurangi.
- 3) **Physical surroundings** atau bangunan fisik juga berpengaruh dalam menentukan aturan-aturan dalam Socio-Technical yang diciptakan.
- 4) **People** di mana sistem juga dirancang menyesuaikan dengan penggunanya, people di sini bukanlah hanya orang perseorangan, namun juga untuk tiap role, maupun kelompok yang berperan dalam penerapan system ini.
- 5) **Procedures** dimana langkah-langkah pekerjaan dari sistem itu sendiri. Prosedur sangat berpengaruh dalam merancang software yang dirancang untuk system tersebut.
- 6) **Laws dan regulations** sedikit mirip dengan procedures, namun hukum dan peraturan lebih menekankan pada tindakan yang terjadi diluar prosedur yang seharusnya
- 7) **Data dan data structures** dimana data yang harus diketahui dalam sebuah sistem tertentu, apakah data-data tersebut yang harus disimpan

menggunakan format tertentu akan mempengaruhi design sistem Socio-Technical.

Menurut Sommerville (2010) faktor-faktor di atas dibedakan dalam beberapa layer sebagai berikut:



Gambar 2.6. Socio-Technical System Stack, Referensi Literatur Review

Socio Tecnichal System mempunyai 3 karakteristik di antaranya:

1. Emergent Property

Sesuatu yang muncul sebagai akibat dari seluruh komponen sistem bekerja sama. Emergent properties ini merupakan suatu sinergi, yaitu jika kita menjumlahkan hasil dari masing-masing element atau komponen dalam sistem, maka hasilnya akan lebih kecil daripada hasil keseluruhan komponen sistem saat bekerja sama.

- ✓ Properti dari sistem sebagai satu kesatuan melebihi dibandingkan properti-properti yang dapat diturunkan dari properti komponen-komponen suatu sistem.
- ✓ Emergent properties sebagai konsekuensi hubungan-hubungan antar komponen-komponen suatu sistem.

- ✓ Oleh karenanya mereka hanya dapat dikaji dan diukur ketika komponen-komponen telah diintegrasikan ke dalam suatu sistem.

Table 2.5. Contoh Emergent Property

Property	Descriptio
Volume (Volume)	Volume sistem (total ruang yang dipakai) bervariasi tergantung pada bagaimana komponen-komponen diassembling tersebut diatur dan dihubungkan.
Keandalan (Reliability)	Kendalan sistem tergantung pada keandalan komponen tetapi interaksi yang terduga dapat menyebabkan suatu kegagalan oleh karenanya mempengaruhi keandalan sistem.
Keamanan (Security)	Keamanan suatu sistem (kemampuannya untuk menahan serangan) merupakan properti kompleks yang mudah diukur. Serangan mungkin merupakan penemuan yang tidak diantisipasi sebelumnya oleh pengembang sistem dan mungkin melemahkan sistem perlindungan didalamnya (built-in safeguards).
Kemudahan diperbaiki (Repairability)	Properti tersebut merefleksikan bagaimana mudahnya memperbaiki suatu masalah dalam sistem ketika ditemukan. Hal ini tergantung pada kemampuan mendiagnosa masalah tersebut, mengakses komponen-komponen yang menyebabkan kegagalan, dan memodifikasi atau mengganti komponen-komponen tersebut
Kemudahan digunakan (Usability)	Properti tersebut merefleksikan bagaimana mudahnya menggunakan sistem. Hal ini tergantung pada teknis komponen-komponen sistem, operator-operatornya, dan lingkungan operasinya.

Hal-hal yang harus dilakukan sebelum mengembangkan perangkat lunak di lingkungan tertentu maka harus dicari informasi tentang :

- Pengetahuan lingkungan tentang teknologi informasi dan komputer.
- Pengetahuan sosial (Social Knowledge) atau pengetahuan tentang budaya lokal (Local Knowledge) di lingkungan yang akan dikembangkan perangkat lunak, apakah mungkin mengembangkan perangkat lunak dengan kondisi lingkungan yang ada.
- Pengetahuan tentang apa saja yang bisa dibatasi dan yang tidak, sehingga saat pengembangan perangkat lunak dapat mendefinisikan aturan main dari perangkat lunak.

2. *Non-Deterministic Property*

Non-Deterministic Property konsep memberikan setiap input yang diberikan adalah sama akan mempunyai output yang mungkin berbeda karena faktor human behavior.

Contoh *Non-Deterministic property*

- a. Human error Kesalahan manusia terjadi dalam proses operasional yang mempengaruhi ketergantungan sistem secara keseluruhan.
 - ✓ Pendekatan orang membuat kesalahan tanggung jawab individu dan menyalahkan kesalahan pada operator yang bersangkutan. Tindakan untuk mengurangi kesalahan termasuk ancaman hukuman, pelatihan yang lebih baik, prosedur yang lebih ketat.
 - ✓ Pendekatan sistem mengasumsikan bahwa orang-orang bisa keliru dan akan membuat kesalahan. Sistem ini dirancang untuk mendeteksi kesalahan-kesalahan ini sebelum menyebabkan kegagalan sistem. Ketika kegagalan terjadi, tujuannya bukan untuk menyalahkan

seseorang tetapi untuk memahami mengapa pertahanan sistem tidak menjebak kesalahan.

- b. Pertahanan sistem Untuk meningkatkan keamanan dan ketergantungan, desainer harus memikirkan tentang kesalahan manusia yang harus dimasukkan dalam sistem, namun dengan banyak hambatan, semua harus gagal karena kegagalan sistem terjadi.

3. *Complex Relationship with Organizational Objectives*

Sistem perangkat lunak oleh karena itu bukan sistem yang terisolasi tetapi merupakan komponen penting dari sistem yang lebih luas yang memiliki tujuan manusia, sosial atau organisasi.

Contoh : Sistem cuaca padang gurun adalah bagian dari sistem pencatatan dan peramalan cuaca yang lebih, ini termasuk perangkat keras dan perangkat lunak, proses peramalan, pengguna sistem, organisasi yang bergantung pada prakiraan cuaca.

2.9 Studi Pustaka

Beberapa jurnal internasional banyak yang membahas tentang metode Agile dengan konsep MVC karena dapat mempermudah dan mempercepat proses pembangunan sistem informasi yaitu:

Tabel 2.6 Studi Pustaka

Nama Peneliti dan Tahun	Judul	Hasil
Malik, R. F., Fachrurozi, M., & Prabowo, R. (2017)	Sistem Informasi Manajemen Laboratorium Menggunakan Metode Agile Dengan Konsep Model-View-Controller Data Access Object	Dengan menerapkan metode AGILE dengan konsep Model-View-Controller Data Access Object dalam pembuatan sistem informasi laboratorium. Perbandingan dilakukan dengan metode SDLC. Hasil yang diperoleh adalah kecepatan dalam menyusun dan membuat sistem informasi manajemen laboratorium lebih baik dibandingkan dengan metode SDLC.
Bragge, M. (2013)	Model-View-Controller architectural pattern and its evolution in graphical user interface frameworks	Tujuan dari tesis ini adalah untuk menyajikan asal mula pola MVC dan bagaimana hal itu telah berubah dari waktu ke waktu. Arsitektur perangkat lunak pada umumnya dan evolusi MVC dalam aplikasi web bukanlah fokus utama. Desain fundamen

Nama Peneliti dan Tahun	Judul	Hasil
		diabstraksikan, dan kemudian digunakan untuk memeriksa versi yang lebih baru. Probabilitas dengan subjek dan terminologinya juga disajikan.
Ratnasari, D., & Munir, S (2016)	Analisis dan Perancangan Sistem Informasi Pemetaan Kompetensi Pengajar LP3 STT-NF Berbasis Web Menggunakan Codeigniter Framework	Pembangunan sistem dengan menggunakan framework php Codeigniter yang sudah mendukung konsep MVC memberikan hasil yang baik dan responsive terhadap platform mobile dan desktop serta platform-platform yang lainnya.

2.10 Kesimpulan

Sistem informasi persuratan ini pertama kali dilakukan di IFMISU dikarenakan sampai saat ini, belum memiliki sistem informasi yang efektif dan

efisien khususnya pada pengelolaan surat masuk dan surat keluar. Oleh karena itu dengan adanya implementasi sistem yang berbasis web ini, bisa memberikan solusi bagi permasalahan administrasi dan pengelolaan persuratan pada IFMISU khususnya di Kementerian Keuangan Timor Leste. Dengan harapan mampu meningkatkan kepuasan pengguna dan mempermudah proses persuratan di IFMISU.

