BAB 2

LANDASAN TEORI

2.1 Definisi ITSM

Information Technology Service Management (ITSM) adalah suatu metode pengelolaan sistem IT yang secara filosofis terpusat pada perspektif konsumen layanan IT terhadap bisnis perusahaan. ITSM mengubah paradigma pemikiran dari hanya mengelola IT sebagai komponen individual menjadi penyediaan layanan *end to end* dengan menggunakan *Framework ITIL.*

Gambar 2. 1. Komponen ITSM

ITSM umumnya menangani masalah operasional manajemen teknologi informasi (kadang disebut *operations architecture*, arsitektur operasi) dan bukan pada pengembangan teknolog*in*ya sendiri. Contohnya, proses pembuatan

perangkat lunak komputer untuk dijual bukanlah fokus dari disiplin ini, melainkan sistem komputer yang digunakan oleh bagian pemasaran dan pengembangan bisnis di perusahaan perangkat lunak-lah yang merupakan fokus perhatiannya. Banyak pula perusahaan non-teknologi, seperti pada industri keuangan, ritel, dan pariwisata, yang memiliki sistem TI yang berperan penting, walaupun tidak terpapar langsung kepada konsumennya.

Sesuai dengan fungsi ini, ITSM sering dianggap sebagai analogi disiplin ERP pada TI, walaupun sejarahnya yang berakar pada operasi TI dapat membatasi penerapannya pada aktivitas utama TI lainnya seperti manajemen portfolio TI dan rekayasa perangkat lunak. Ketika kita menjalani sebuah ITSM yang sebenarnya, menurut Rob England (2008), ITSM adalah framework operasional yang benar-benar mengenal bagaimana IT di terapkan di lingkungan sebenarnya. ITSM mengadaptasi lingkungan IT serta dampak-dampaknya sehingga mengurangi kerugian dalam penggunaan dan memaksimalkan nilai positif kepada organisasi yang menerapkan

2.2 ITIL

Information Technology Infrastructure Library (ITIL) merupakan sekumpulan best practice pada ITSM. *The United Kingdom's Central Computer and Telecommunication Agenc*y (CCTA) menciptakan ITIL sebagai tanggapan terhadap pertumbuhan teknologi informasi untuk kebutuhan bisnis.ITIL menyediakan bisnis dengan kerangka disesuaikan penerapan terbaik untuk mencapai kualitas layanan dan mengatasi kesulitan yang berhubungan dengan pertumbuhan sistem TI.Hewlett-Packard Co dan Microsoft adalah dua perusahaan

yang menggunakan ITIL sebagai bagian dari kerangka kerja praktek terbaik mereka s*end*iri.

ITIL diatur dalam beberapa "set" buku yang didefinisikan oleh fungsi terkait, *service* strategy, *service* design, managerial,*service* transition, *service* operation dan continual *service* improvement. Selain buku, yang dapat dibeli secara online, layanan ITIL dan produk meliputi pelatihan, kualifikasi, perangkat lunak dan kelompok pengguna seperti *IT Service Management Forum* (itSMF).

Penerapan *support desk*dengan pendekatan ITIL menurut Alex D Paul (2008), ITIL merupakan best practice untuk memastikan layanan teknologi informasi berjalan sesuai dengan sebagaimana mestinya, yang meliputi manajemen insiden (*incident management*), manajemen masalah (problem 8*management*), dan manajemen perubahan (*change management*). Sedangkan menurut Peter Gilbert, Roger Morse and Monica Lee (2007), dengan menerapkan *support desk* akan menciptakan manajemen pengetahuan (knowledge *management*) yang tercipta dari sebuah dokumentasi resolusi manajemen insiden(*incident management*) yang disimpan pada knowledge base sehingga dapatmempersingkat waktu penyelesaian dari sebuah insiden dan masalah (*incident* andproblem) dan memungkinkan *customer* dapat menyelesaikan s*end*iri masalah yang dihadapi (self *service*) dengan memanfaatkan knowledge base. Untuk ituPenerapan*support desk*akan menggunakan pendekatan ITIL V3 sebagai kerangkaacuan (framework) yang akan di pakai dan diterapkan di Organisasi untukmeningkatkan pelayanan, memonitor dan memastikan layanan teknologi informasi dapat berjalan dan tercapainya tujuan sesuai dengan visi dan misi perusahaan. ITIL saat ini dipelihara dan dikembangkan oleh United Kingdom’s

Office of Government Commerce Edwards Deming dengan plan-do-*check*-act(PDCA) cycle

Pada penelitian ini, ITIL framework, khususnya *support desk*, akan diterapkan pada perusahaan yang bergerak dibidang IT (ISP dan Multimedia). ITIL V2 mencakup delapan kumpulan (ITSMF, 2007) yaitu:

1. *Service Support*
2. *Service Delivery*
3. *Planning* to *Implement Service Management*
4. ICT *Infrastructure Management*
5. *Application Management*
6. *Business Perspective*
7. *Security Management*
8. *Software Asset Management*

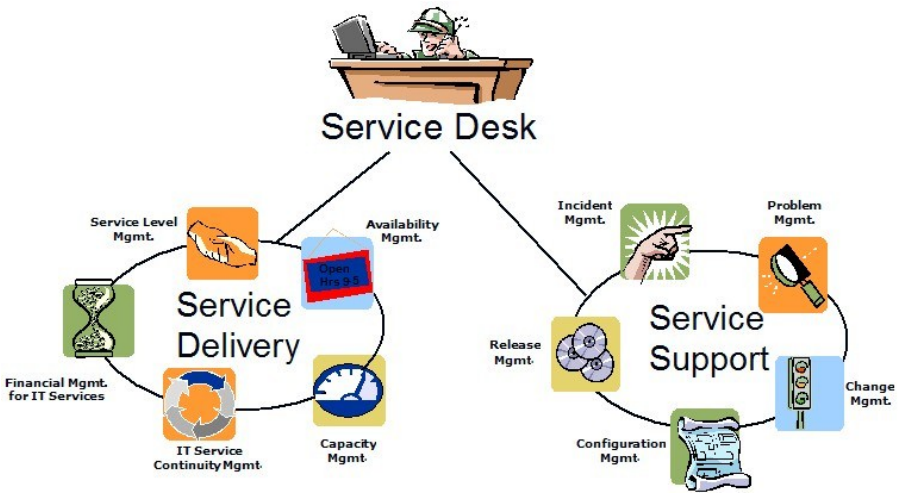
Dua diantaranya, yaitu *Service Support* dan *Service* Delivery merupakan area utama, yang disebut juga *IT Service Management* (ITSM). ITSM merupakan definisi dan penyampaian layanan IT dan pengelolaan organisasi yang menyediakan layanan (Chen Zhao & Fei Gao, 2008)

2.2.1 ITIL *Service* Delivery and *Support*

Service Delivery adalah pelaksanaan dari disiplin-disiplin dalam *framework* ITIL, sehingga layanan bisnis-IT dapat berjalan (OGC, 2002). Elemen dalam *Service* Delivery yaitu:

1. Avalilability *Management* (Manajemen Ketersediaan)
2. Capacity *Management* (Manajemen Kapasitas)
3. Financial *Management* (Manajemen Keuangan)

4. IT *Service* Continuity (Manajemen Kesinambungan Layanan IT)
5. *Service* Level *Management*(Manajemen Tingkat Layanan)



Gambar 2. 2. Komponen *Service Desk*

Service Support adalah suatu penerapan disiplin sebagai p*end*ukung yang memungkinkan tersedianya pelayanan IT (OGC, 2002). Tanpa disiplin ini, organisasi/perusahaan hampir tidak mungkin menyediakan pelayanan IT yang baik dan tidak dapat dikelola dengan sempurna. Elemen dalam *Service Support* antara lain :

1. Configuration *Management*

   Manajemen konfigurasi adalah bagaimana sebuah organisasi dalam mengelola konfigurasi dari infrastruktur yang ada. Apakah ada S.O.P dalam mengelolanya dan lain sebagainya.

2. *Change Management*

Manajemen perubahan adalah bagaimana sebuah organisasi dalam mengelolasegala perubahan yang ada.Apakah setiap perubahan dicatat dalam *change* log, dan dilaporkan ke manajemen, dsb.

3. *Incident Management*

   Manajemen insiden adalah bagaimana sebuah organisasi dalam mengelola insiden dalam pelayanan.

4. Problem *Management*

   Manajemen masalah adalah bagaimana sebuah organisasi dalam mengelola permasalahan ketika melakukan *service support*. Dalam ITIL, insiden dan masalah merupakan bagian yang terpisah. Insiden merupakan kejadian yang wajar terjadi dalam organisasi sehari-hari. Sedangkan klasifikasi '*Problem'* lebih merupakan kejadian yang khusus dan ada pengelola/manajer nya sendiri

5. Release *Management*

   Release *Management* adalah pengelolaan dalam hal release produk, dsb. Apakah ada standar dalam release produk. Misalnya penamaan versi, subversi, dll.

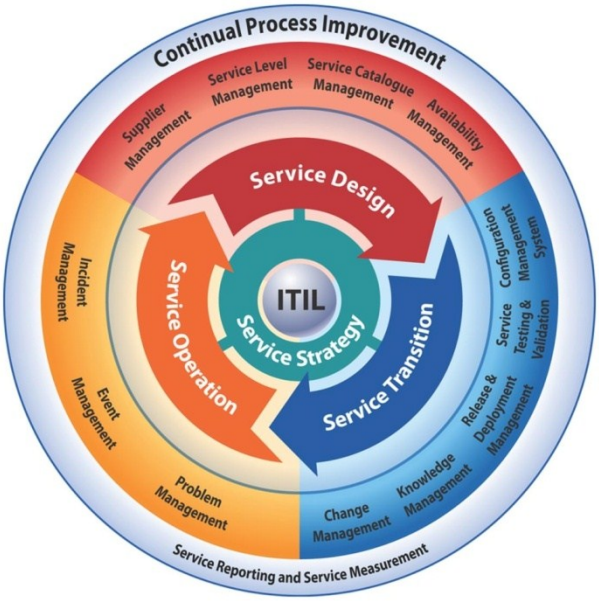
6. *Support Desk*

   Support Desk merupakan hal yang terpenting dalam *service management*. Karena merupakan pintu gerbang utama *service management*, *Support Desk* juga menjadi bagian dalam framework yang bersentuhan langsung dengan *customer*. Bagian ini merupakan tempat bagi *customer* untuk melaporkan segala insiden dalam layanan,

Request For Change, dsb. Pengelolaan di bagian ini harus lebih diperhatikan.

2.2.2 ITIL V2 dan ITIL V3

ITILv3 berbeda dengan ITILv2. Pendekatan ITILv3 melalui sudut pandang layanan yang berkesinambungan. Bila pada ITILv2 komponen utama adalah *Service Support* dan *Service* Delivery, berbeda dengan ITILv3 yang memiliki lima komponen *Service Lifecycle* yaitu *Service Strategy*, *Service Design*, *Service Transition*, *Service Operation* dan *Continual Service* Improvement



Gambar 2. 3. Komponen ITIL V3

Komponen ITIL V3 adalah sebagai berikut:

1. *Service* Strategy, yang terdiri dari:

a. Strategy generation

b. Financial *management*

c. *Service* portfolio *management*

d. Demand *management*

2. *Service* Design, yang terdiri dari

   a. Capacity, Availability, Info Security *Management*

   b. *Service* level & Supplier *Management*

3. *Service* Transition, yang terdiri dari:

   a. *Planning & Support*

   b. *Release & Deployment*

   c. *Asset & Config management*

   d. *Change management*

   e. *Knowledge Management*

4. *Service* Operation, yang terdiri dari:

   a. *Problem & Incident management*

   b. *Request fulfilment*

   c. *Event & Access management*

5. *Continual Service Improvement*, yang terdiri dari:

   a. *Service* measurement & reporting

   b. 7-step improvement process

2.2.3 *Support Desk*

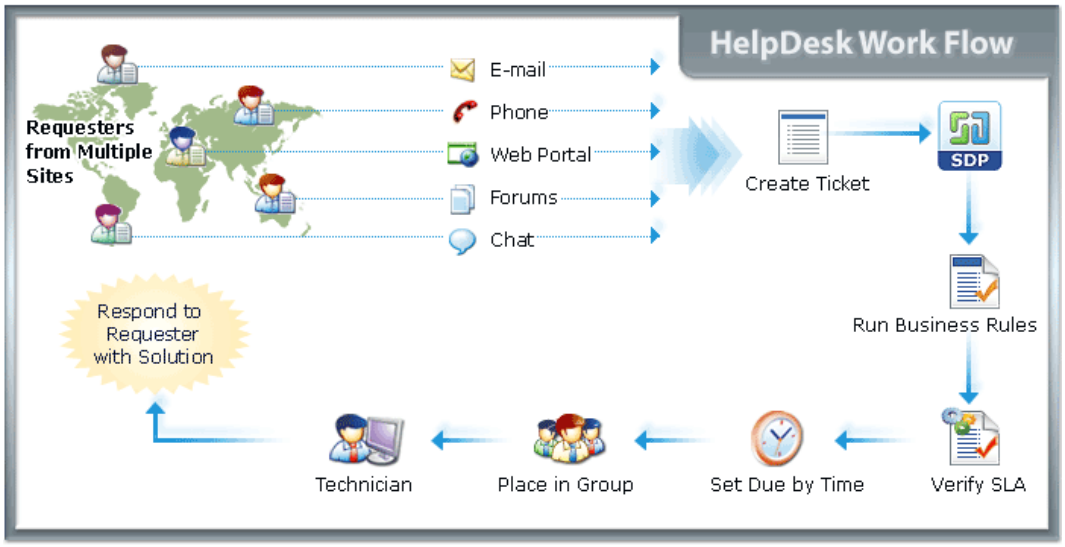
Support desk adalah suatu unit fungsional yang terdiri dari sejumlah staf yang bertanggung jawab dalam hubungan dengan berbagai

insiden layanan yang disediakan perusahaan atau organisasi (N*end*ar Amirulloh Permana, 2010).

Insiden ini masuk melalui panggilan telepon, lapor langsung melalui *help desk*, *interface web*, atau insiden infrastruktur yang secara otomatis dilaporkan melalui *ticket*. *Support desk* amat sangat penting yang merupakan bagian dari suatu*department* organisasi IT dan merupakan titik kontak tunggal untuk para pemakai IT dan akan menangani semua insiden serta permintaan layanan.

Penerapan *support desk* dengan pendekatan ITIL menurut Alex D Paul, ITIL merupakan best practice untuk memastikan layanan teknologi informasi berjalan sesuai dengan sebagaimana mestinya, yang meliputi manajemen insiden (*incident management*), manajemen masalah (problem *management*), dan manajemen perubahan (*change management*). Sedangkan menurut Peter Gilbert, Roger Morse and Monica Lee, dengan menerapkan *support desk* akan menciptakan manajemen pengetahuan (knowledge *management*) yang tercipta dari sebuah dokumentasi resolusi manajemen insiden (*incident management*) yang disimpan pada knowledge base sehingga dapat mempersingkat waktu penyelesaian dari sebuah *incident* dan problem dan memungkinkan *customer* dapat menyelesaikan s*end*iri masalah yang dihadapi (self *service*) dengan memanfaatkan knowledge base. Untuk itu penerapan *support desk* akan menggunakan pendekatan ITIL V3 sebagai kerangka acuan (framework) yang akan di pakai dan diterapkan untuk meningkatkan pelayanan,

memonitor dan memastikan layanan teknologi informasi dapat berjalan dan tercapainya tujuan sesuai dengan visi dan misi perusahaan.



Gambar 2. 4. Proses Self *Service Ticket*ing

Dalam penerapan *support desk* terdiri dari proses *Requirements*, *Analysis*, *Design*, *Implementation* dan *Deployment* yang akan dituangkan dalam bentuk *desk*riptif atau kualitatif :

1. *Requirements*

   Merupakan tahap analisis kebutuhan berupa identifikasi masalah yang ada, identifikasi kebutuhan sistem yang di inginkan,survey lapangan terhadap existing *system support desk* yang sedang berjalan, melakukan studi literature dan pengumpulan bahan yang akan menjadi rancangan sistem *support desk* menggunakan framework ITIL V3

2. *Analysis*

   Merupakan tahap identifikasi sistem yang akan dirancang, m*end*efinisikan sistem yang akan dibuat sesuai dengan requirements dengan framework ITIL v3, analisis sistem *support desk* yang sedang berjalan (existing),

analisis kebutuhan organisasi sesuai dengan visi, misi, dan tujuan penerapan teknologi informasi dan kebutuhan akan layanan,analisis sistem dengan analisis proses bisnis dan use case diagram

3. *Design*

   Merupakan tahap perancangan sistem yang akan di buat sesuai dengan requirement dan analysis, tahap ini lebih menjelaskan sistem secara detail sebagai penjelasan dari tahap analisis yang dituangkan sebagai activity diagram, class diagram, sequence diagram dan collaboration diagram. Tahap design ini akan menjadi *prototype* atau rancangan perangkat lunak yang akan di implementasikan.

4. *Implementation*

   Tahap ini termasuk juga tahapan testing atau pengujian terhadap rancangan perangkat lunak *support desk* yang akan di implementasikan di organisasi atau perusahaan yang merupakan hasil dari tahap desain, implementasi hasil rancangan dengan coding program, monitoring terhadap sistem yang di implementasikan, monitoring terhadap dampak atau resiko yang terjadi setelah sistem *support desk* di implementasikan baik dari sisi pelayanan, *customer*, *help desk*, organisasi dan *management* yang terlibat

5. *Deployment*

   Tahap ini dapat di jelaskan dalam deployment diagram pada desain UML (Unified Modeling Language), tahap ini juga akan mengidentifikasi kebutuhan hardware, software dan SDM (Sumber DayaManusia) yang terlibat.

2.2.4 *Incident Management*

Menurut framework ITIL, pengertian insiden adalah sebuah interupsi atau pengurangan kualitas dari layanan TI. Selain itu sebuah kesalahan konfigurasi pada sistem dapat dikatakan sebagai insiden walaupun belum menimbulkan masalah yang berarti pada sistem tersebut. Manajemen insiden (*incident management*) adalah proses yang dilakukan untuk menyelesaikan suatu insiden. Proses manajemen insiden (*incident management*) dilakukan berdasarkan input dari *user* melalui *service desk*, laporan teknisi, dan juga deteksi otomatis dari sebuah tool *event management*. Manajemen insiden (*incident management*) pada frmework ITIL v3 berada pada cycle *Service* Operation.

Tujuan utama dari manajemen insiden (*incident management*) adalah untuk mengembalikan kondisi layanan TI ke keadaan normal secepat mungkin, dan meminimalkan dampak negatif yang ditimbulkan terhadap kegiatan bisnis utama organisasi. Keadaan normal layanan TI adalah keadaan yang telah didefenisikan sebelumnya dalam sebuah SLA (*Service* Level Agreement).

Berikut adalah aktifitas-aktifitas dalam manajemen insiden (*incident management*) menurut framework ITIL v3 :

1. Identifikasi Insiden (*Incident* Identification)

   Proses manajemen insiden (*incident management*) dimulai dengan identifikasi. Identifikasi yang paling umum dilakukan adalah melalui layanan *service desk* dan laporan dari staf teknisi. Selain itu

identifikasi insiden dapat dilakukan secara otomatis oleh tool *event management* yang dipasang pada perangkat-perangkat utama. Kondisi ideal dari langkah identifikasi adalah insiden dapat teridentifikasi sebelum terjadi implikasi terhadap *user*.

2. Pencatatan insiden (*Incident* Logging)

   Langkah ini wajib dilakukan untuk setiap jenis insiden baik yang berskala besar maupun kecil. Beberapa informasi yang harus dicatat terkait suatu insiden adalah ID, kategori insiden, waktu terjadi, *desk*ripsi insiden, nama orang/grup yang bertanggungjawab atas penanganan, implikasi insiden, dan waktu penutupan kasus.

3. Pengkategorisasian insiden (*Incident* Categorization)

   Dalam membuat kategori insiden dibutuhkan sebuah proses khusus antara pengelola TI dan pihak manajemen organisasi. Hal ini bertujuan untuk menghasilkan kategori insiden dan prioritas penanganannya sejalan dengan proses bisnis organisasi. Kategori insiden dapat dibuat berdasarkan perkiraan lamanya penanganan, implikasi terhadap proses bisnis organisasi, dan jumlah staf teknis terkait.

4. Prioritas insiden (*incident* priorization)

   Langkah prioritas insiden dilakukan berdasarkan kategorisasi yang telah dibuat sebelumnya. Prioritas penanganan insiden dapat dilakukan berdasarkan besarnya implikasi insiden terhadap kegiatan bisnis utama organisasi, ataupun berdasarkan lamanya penanganan insiden.

5. Diagnosa awal (*Initial Diagnosis*)

Diagnosa awal terhadap insiden wajib dilakukan oleh setiap pihak yang pertama kali berhubungan dengan insiden baik itu *service desk*, staf teknis, maupun perangkat otomatis seperti *event management*. Jika insiden ditemukan oleh *service desk* melalui telepon dari *user*, maka diusahakan *service desk* tersebut yang menyelesaikan insiden selama *user* masih berhubungan telepon.

6. Eskalasi insiden (*Incident Escalation*)

   Eskalasi insiden adalah tindakan menaikkan level penanganan insiden. Hal ini berkaitan erat dengan hasil diagnosa awal terhadap insiden. Jika dari diagnosa ditemukan insiden yang tidak dapat ditangani, maka wajib dilakukan eskalasi insiden. Eskalasi insiden ada 2 macam, yaitu eskalasi fungsi dan eskalasi hierarki. Eskalasi fungsi adalah tindakan menaikkan level penanganan kepada satu level diatasnya. Sedangkan eskalasi hierarki adalah tindakan menaikkan level penanganan melintasi hirarki organisasi misalnya kepada manajer IT atau manajer bisnis yang terkait.

7. Investigasi (*Investigation* and *Diagnosis*)

   Tindakan investigasi dilakukan untuk menemukan sumber masalah dari insiden. Dalam melakukan investigasi, setiap tindakan wajib dilaporkan juga ke dalam formulir insiden. Hal ini berguna sebagai data historis tindakan penanganan suatu insiden.

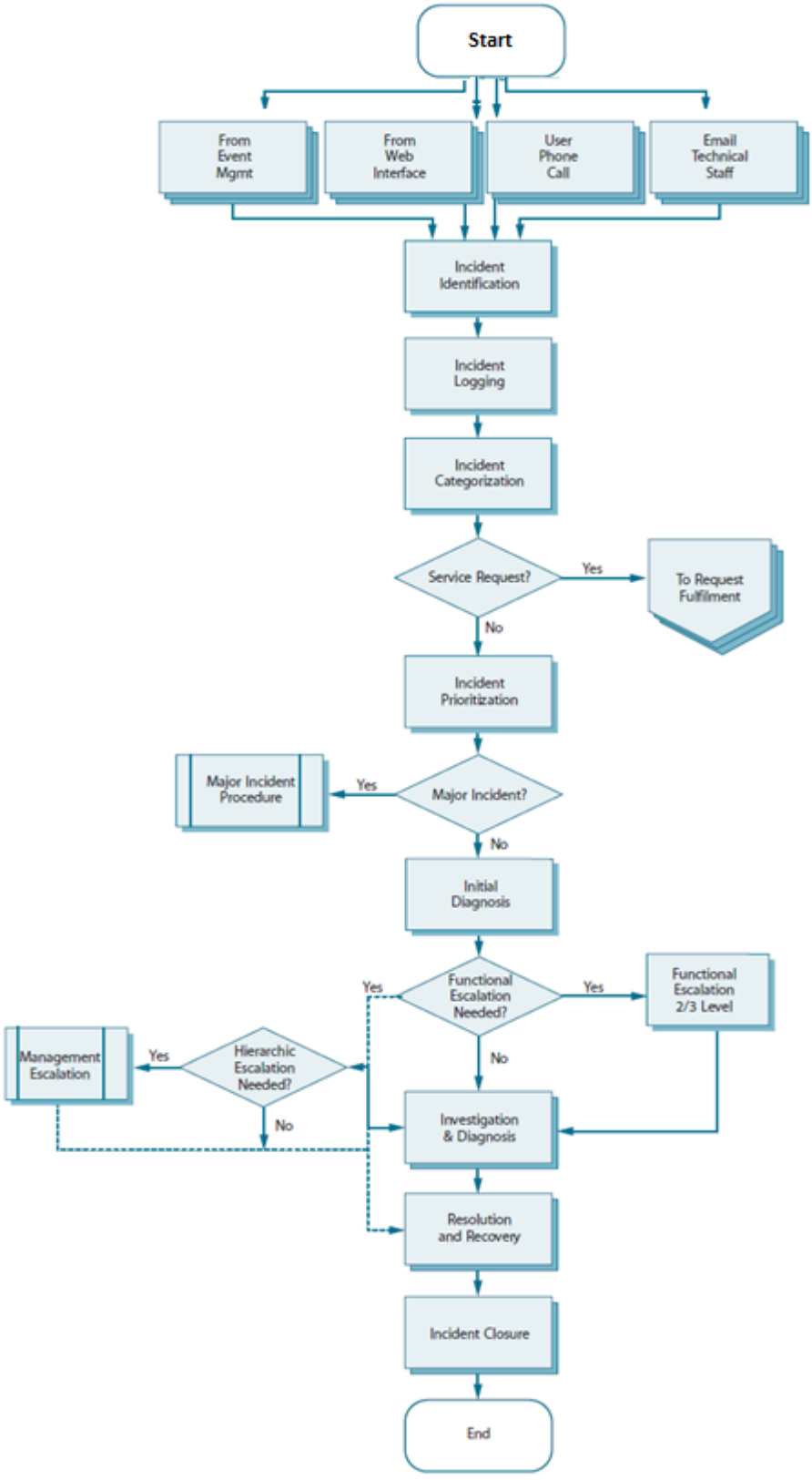
8. Resolusi (*resolution and recovery*)

   Langkah ini merupakan tindakan yang diambil untuk menyelesaikan suatu insiden. Langkah resolusi dapat dilakukan oleh *service desk*

sebagai pihak yang pertama menemukan insiden dari *user*, staf teknisi yang sedang mengerjakan kegiatan konfigurasi, maupun oleh supplier terhadap perangkat yang masih dalam garansi.

9. Penutupan (*incident closure*)

   Langkah penutupan adalah langkah yang dilakukan oleh *service desk* maupun staf teknisi terkait untuk memastikan apakah insiden telah benar selesai ditangani. Yang harus diperhatikan dalam langkah penutupan ini adalah dokumentasi proses penanganan insiden, perkiraan terhadap perulangan insiden, dan survei kepuasan *user* atas penanganan insiden.



Gambar 2. 5. ITIL *Incident Management*

Proses *incident management* s*end*iri perlu untuk didukung dengan peran yang disarankan oleh ITIL berikut ini: (Office, 2011, hal. 194 - 195)

Tabel 2. 1. Penerapan untuk *Incident Management*

Peranan	**Penjelasan**
Incident management process owner	Memastikan proses *incident management* yang dijalankan sudah sesuai dengan yang ditentukan sebelumnya.
Incident management process manager	Mengelola jalannya proses *incident management*.
First-line analyst	Memberikan *support* pertama kali pada sebuah insiden dengan mengikuti proses *incident management*.
Second-line analyst	M*end*edikasikan hampir seluruh waktunya untuk m*end*iagnosa suatu insiden dan mencari tahu solusinya.
Third-line analyst	Grup teknis internal atau eksternal yang m*end*iagnosa dan mencari tahu solusi atas insiden yang dieskalasi.

Teknologi untuk proses *incident management* disarankan untuk menampung informasi berikut ini pada setiap *ticket* insiden: (Office, 2011, hal. 76 & 85)

Tabel 2. 2. Informasi pada *Ticket* Insiden

Nomor referensi yang unik	Status insiden (*active, waiting, closed*, dll)
Nama dan identitas pelaku yang merekam dan memperbaharui data insiden	*Configuration item* yang memiliki hubungan
Kategori insiden	*Incident/Problem/Change/Known error*

	yang memiliki hubungan
Urgency insiden	*Support group*/pelaku yang dialokasikan untuk insiden terkait
Impact insiden	Detil dari kegiatan yang dilakukan untuk menyelesaikan insiden, beserta waktunya
Tanggal dan jam merekam, termasuk semua aktifitas didalamnya	Identitas pelaku yang menutup *ticket*
Metode notifikasi (telepon, email otomatis, tatap muka langsung)	Tanggal dan waktu resolusi
Nama/departemen/telepon/lokasi *user*	Kategori *closure*
Metode *callback* (telepon/email/yang lain)	Tanggal dan waktu *closure*
Penjelasan dari gejala-gejala insiden yang terjadi	*Service catalouge*

Sedangkan untuk fitur yang disaranakan terdapat pada sistem *incident management* adalah: (Office, 2011, hal. 219 - 220)

Table 2. 3. Rekom*end*asi Fitur Teknologi untuk Sistem *Incident Management*

Fungsi	***Desk*ripsi**
Incident logging	*Entry* data *ticket*, menentukan kategori, menentukan prioritas dan pelacakan *ticket*.
Integral CMS	o Menjaga hubungan antara insiden, *service request, problem, known error, workaround* dan *configuration item*. o CMS dapat digunakan untuk menentukan prioritas dan membantu investigasi dan diagnosa.
Process Flow Engine	Mengkontrol proses *incident management* yang sudah ditentukan sebelumnya secara otomatis, misalnya untuk

Fungsi	*Desk*ripsi
Incident logging	*Entry* data *ticket*, menentukan kategori, menentukan prioritas dan pelacakan *ticket*.
	masalah *email* akan ditujukan kepada *support group* tertentu.
Automated alerting and escalation	Untuk menghindari sebuah insiden yang diacuhkan atau mengalami keterlambatan.
Open interfacing to event management tools	Membentuk *ticket* insiden secara otomatis apabila suatu *failure* terjadi dan dideteksi oleh salah satu tools dari proses *event management*.
A web interface	Memiliki modul *self-help* dan dapat digunakan *user* untuk meregister *ticket* insiden.
An integrated KEDB	Untuk menyimpan insiden / *problem* yang sudah diresolusi, agar dapat digunakan sebagai referensi pada insiden / *problem* m*end*atang.
Easy to use reporting	o Laporan untuk menunjukan *metrics* insiden yang dapat digunakan untuk analisis *problem management* (*reactive* atau *proactive*) dan *availability management*. o Laporan insiden secara historikal dan menggolongkan insiden per kategori, prioritas, status (*open, close, in progress*, dll) atau *configuration item* yang terkena dampak.

2.2.5 Knowledge *Management*

Dalam implementasi ITIL *knowledge management* memiliki peranan penting terutama untuk *service desk*, dimana *service desk* harus dapat

menemukan solusi atas sebuah permasalahan secepat-cepatnya dengan menggunakan berbagai sumber pengetahuan yang dimilikinya, dan berikut ini adalah beberapa rekom*end*asi yang memiliki keterkaitan dengan *knowledge management* untuk *service desk* (Jäntti & Kalliokoski, Identifying Knowledge Management Challenges in a Service Desk: A Case Study, 2010)

- Membuat kategori insiden yang sederhana, dikarenakan kategori yang terlalu kompleks dapat memperlambat proses pembuatan *ticket*.
- Membuat penamaan dokumen secara seragam, agar mempermudah pencarian dokumen tersebut.
- *User* perlu diberikan pelatihan mengenai bagaimana caranya mengajukan permintaan *support,* terutama dengan menggunakan fasilitas *web-form*.
- *User* perlu diajak untuk lebih aktif menggunakan *web-forms* ketika h*end*ak mengajukan permintaan *support*, dikarenakan ini merupakan cara yang efektif untuk memperoleh informasi mengenai insiden dan *service request*, dan untuk menghindari pekerjaan *extra*.

Rekom*end*asi lain untuk *knowledge management* adalah: (Liang & Baozhang, 2009)

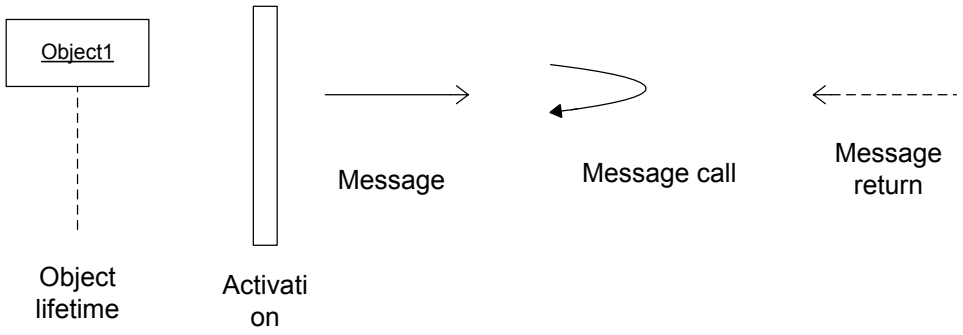
- Adanya *Knowledge Classification Management*, yang berfungsi untuk mengelompokan *knowledge* dan diperbaharui sesuai dengan *service* terbaru.
- Adanya *Knowledge View Management*, dimana akses terhadap *knowledge* dibatasi sesuai dengan *user role* terkait.
- Adanya *Knowledge Maintenance Management*, yang merupakan operasi m*end*asar dari ITSM *knowledge*, meliputi *adding, modifying* dan *deleting*.

2.3 UML (*Unified Modelling Language*)

UML adalah alat untuk menggambarkan gambaran dari sistem yang akan dibuat melalui diagram dan simbol. UML menggunakan konsep Pemrograman Berorientasi Objek (*Object Oriented Programming).* Melalui seperangkat diagram, UML menyediakan standar yang memungkinkan sistem analisis untuk merancang berbagai sudut pandang dari sistem analisis untuk merancang berbagai sudut pandang dari sistem, yang dinamakan model, yang dimengerti oleh *client, programmer,* dan siapapun yang terlibat dalam proses pengembangannya (Schmuller, 1999).

2.3.1 Sequence Diagram

Sequence Diagram menunjukkan dinamika interaksi berbasis waktu yang interaktif antar objek dalam sistem (Schmuller, 1999). Tidak seperti *clasdiagram* yang statis, *sequence diagram ber*sifat dinamis



Gambar 2. 6. Komponen-komponen Sequence Diagram

2.3.2. Use Case Diagram

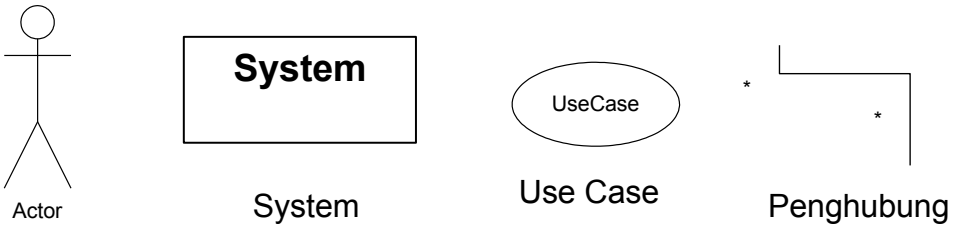
Use Case menggambarkan interaksi antara sistem dengan pelaku yang ada. Diagram ini mendeskripsikan siapa saja yang menggunakan

sistem dan bagaimana cara mereka berinteraksi dengan sistem. *Use case* digunakan untuk menggambarkan bagaimana sistem terlibat pada pengguna (Mathiassen et. al, 2000).

Pelaku dan *use case* adalah dua elemen-elemen yang ada. Pelaku adalah orang-orang atau sistem lain yang berinteraksi dengan sistem. *Use case* adalah suatu bentuk interaksi antara sistem dan pelaku. Pelaku dan *use case* dapat dihubungkan satu sama lain, dengan cara mengindikasikan sebuah pelaku pada sebuah *use case*. *Use case* dapat dikelompokkan dalam hubungannya dengan sistem. Semua *use case* yang didukung oleh sebuah sistem dapat diorganisasikan dalam sebuah kelompok dengan nama dari sistem.

Elemen-elemen yang digunakan dalam *use case diagram* antara lain (Mathiassen et. al, 2000) :

- Sistem, yang digambarkan menggunakan persegi yang di dalamnya terdapat sekumpulan *use case*. *Actor* diletakkan di luar sistem.
- *Use case*, yang digunakan untuk menggambarkan fungsi-fungsi pada sistem digambarkan dengan elips.
- *Actor*, pengguna sistem.
- Penghubung, untuk menghubungkan antara actor dengan use case.



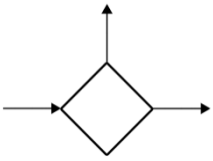
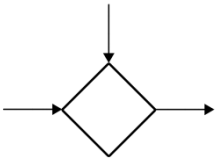
Gambar 2.7 Komponen-komponen *Use Case Diagram*

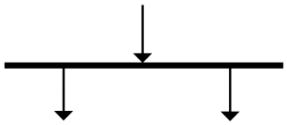
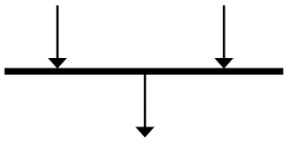
2.3.3. Activity Diagram

Menurut Whitten dan Bentley (2007) *Activity Diagram* adalah diagram yang dapat digunakan untuk menggambarkan secara grafis aliran proses bisnis, langkah- langkah dari sebuah *use case* atau logika dari prilaku objek (*method*).

Berikut ini adalah notasi-notasi yang digunakan dalam *activity diagram*:

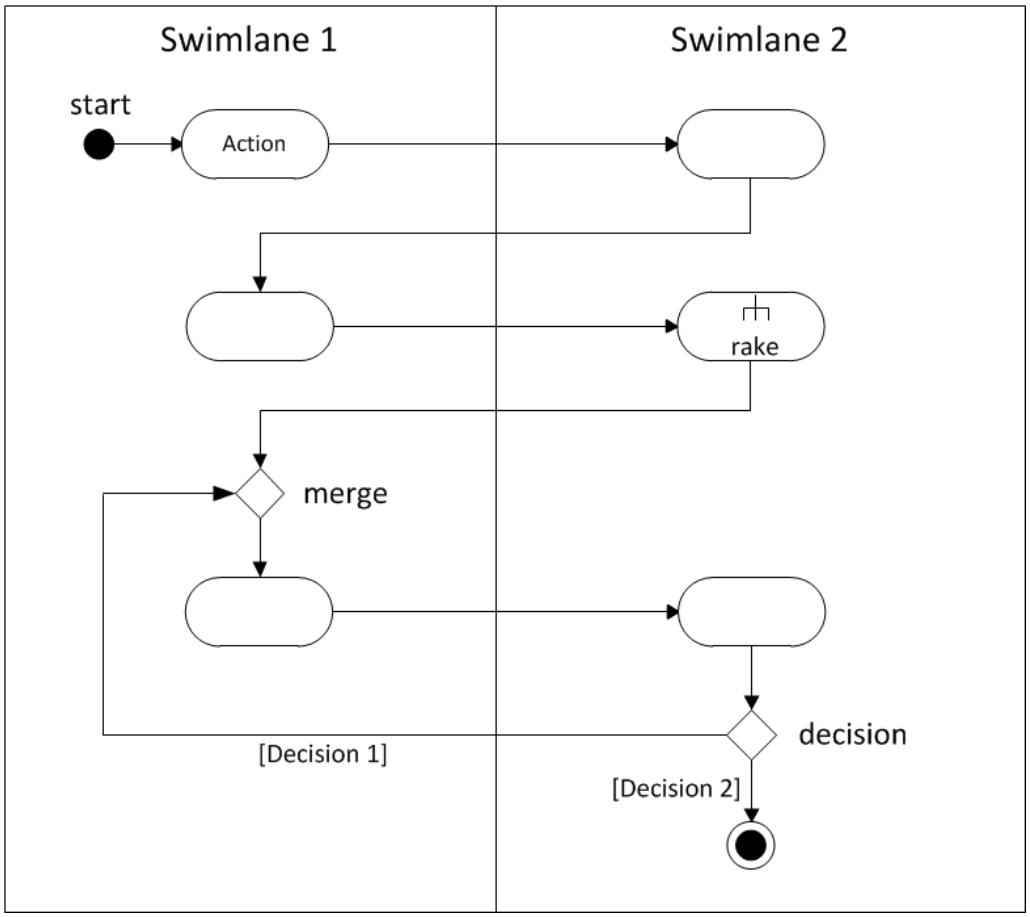
Tabel 2.4 Tabel Notasi dan Keterangan *Activity Diagram*

No	Notasi	Keterangan
1	 *initial node*	Menggambarkan awal dari sebuah proses.
2	 *actions*	Menggambarkan *individual steps*
3	 *flow*	Mengindikasikan *progress* dalam sebuah *action. Flow* tidak perlu diikuti kata identifikasi, kecuali *flow* yang keluar dari *decision*.
4	 *decision*	Berbentuk *diamond* dengan satu aliran masuk dan dua/lebih aliran yang keluar. Aliran yang keluar ditandai untuk mengindikasikan kondisi.
5	 *merge*	Bentuk *diamond* dengan dua/lebih aliran masuk dan satu aliran keluar. Ini mengkombinasikan aliran yang tadinya terpisah oleh *decisions.* Diproses

		sehingga satu aliran.
6	 *subactivity indocator*	Simbol *rake* yang mendentifikasikan bahwa *activity* yang memiliki simbol ini merupakan bagian dari *activity* diagram lainnya.
7	 *fork*	Balok hitam dengan satu aliran masuk dan dua atau lebih aliran keluar. Aksi dalam aliran yang pararel.
8	 *join*	Balok hitam dengan dua atau lebih aliran masuk dan satu aliran keluar. Aksi yang masuk ke dalam *join* harus selesai sebelum proses lainnya dilanjutkan.
9	 *activity final*	Menggambarkan bagian akhir dari proses.

(Sumber: Whitten dan Bentley, 2007)

Berikut ini adalah contoh *Activity Diagram:*

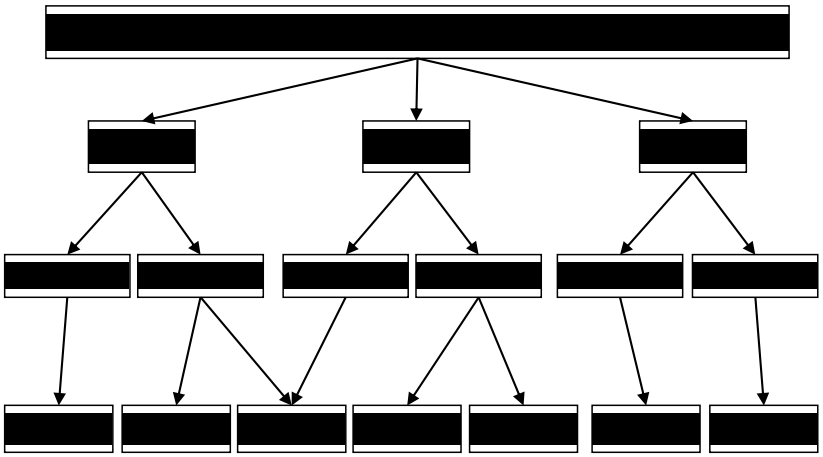


Gambar 2.8 Contoh *Activity Diagram*

2.4 Goal Question Metric

Saat ini penggabungan metode tradisional pengukuran pengembangan software GQM (Goal Question Metric) dan *Six Sigma* sudah semakin banyak digunakan pada industri piranti lunak, dimana metode GQM dapat digunakan pada tahap *Define* dan *Measure* dalam metode DMAIC milik *Six Sigma* (Pradeep & Prabhu, 2009). Ide utama dari GQM adalah pengukuran harus memiliki tujuan yang memiliki arahan jelas, dan model pengukuran seperti berikut ini: (Baldassarre, Caivano, Pino, Piattini, & Visaggio, 2010)

1. *Conceptual Level* (GOAL): *Goal* ditentukan untuk mencapai suatu tujuan yang jelas berdasarkan kebutuhan organisasi.
2. *Operational Level* (QUESTION): Pertanyaan digunakan untuk menunjukan cara yang digunakan untuk mencapai *goal*.
3. *Quantitative Level* (METRIC): Kumpulan data yang memiliki hubungan dengan setiap *question* dengan tujuan menyediakan jawabannya.



Gambar 2.9. Contoh struktur *goal question metric*

2.5 ITSM Maturity Assessment

IT *Service Management* memiliki tingkat kematangan fungsi dan proses masing-masing yang dapat di ukur dengan beberapa metode. Dalam ITSM terdapat beberapa tingkat kematangan yang di gunakan tergantung dari maturity model atau *framework* yang di gunakan, beberapa maturity model yang ada pada ITSM adalah *Capability Maturity Model* dengan CMM *Process Maturity Assessment*, ITIL *Process Assessment Framework.*

2.5.1 Capability Maturity Model

CMM pada awalnya di kembangkan sebagai alat pengukuran kemampuan kontraktor pemerintah untuk melakukan proyek perangkat lunak yang objektif. Model ini di bangun berdasarkan *framework process maturity* yang pertama kali di jelaskan pada tahun 1989 dalam buku *Managing the Software Process* oleh Watts Humphrey. Buku itu kemudian di publikasikan dalam sebuah laporan pada tahun 1993 dan sebagai buku dengan pengarang yang sama.

Meskipun berasal dari bidang piranti lunak, model ini juga di gunakan sebagai model umum untuk membantu dalam proses bisnis secara umum dan telah di gunakan secara luas di seluruh dunia di kantor-kantor pemerintahan, perdagangan dan industri piranti lunak

Pada CMM terdapat 5 tingkat yang di tetapkan oleh kontinum model CM. Menurut SEI (Software Engineering Institute): "Prediktabilitas, efektivitas, dan pengendalian proses software dalam sebuah organisasi diyakini dapat meningkat seperti tingkatan pada CMMI. Meskipun tidak pasti namun bukti empiris sampai saat ini mendukung keyakinan ini"

1. Level 1 – Initial (Chaotic)

   Karakteristik dari proses pada level ini biasanya di tandakan dengan tidak terdokumentasinya suatu proses. Sering terjadi Adhoc dan tidak terkontrol.

2. Level 2 – Repeatable

Pada level ini biasanya proses terdokumentasi namun disiplin proses belum terlalu ketat

3. Level 3 – Defined

   Karakteristik pada proses ini adalah proses sudah di definisikan dan di konfirmasi oleh semua stakeholder sebagai sebuah SOP dan telah dibuat leveling 0, 1, 2, 3 sebagai sebuah work instruction.

4. Level 4 – Managed

   Proses pada level 4 dapat di hitung dengan angka (Quantitatively counted) dengan metrics.

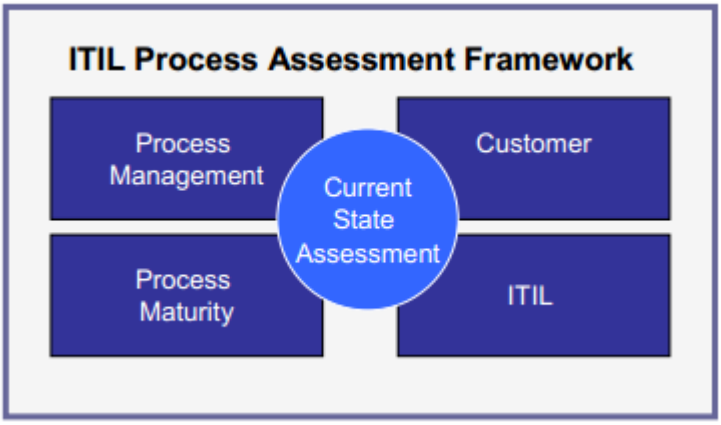
5. Level 5 – Optimizing

   Pada level tertinggi ini, manajemen proses telah meliputi optimisasi dan serta improvement.

2.5.2 . ITIL Maturity Model

Kemampuan untuk melakukan penilaian proses ITIL dan penerapannya untuk menetapkan baseline pengukuran serta benchmark untuk masa depan adalah prinsip kunci dari Continual Service Improvement (CSI) Model. Di dunia IT, terdapat banyak sekali grup konsultan yang memberikan service assessment ITIL proses karena banyak perusahaan lebih memilih untuk menggunakan jasa mereka ketimbang harus melakukan pelatihan kepada karyawannya yang notabene sangat kompleks dan mahal. Namun ternyata kebutuhan untuk training kompetensi sangatlah di perlukan untuk penerapan Continual Service Improvement (CSI) Model, maka dari itu di buatlah penerapan ITIL maturity model assessment technique yang mudah di lakukan.

ITIL Process Assessment Framework (IPAF) saat ini telah di kembangkan untuk memberikan metode adopsi pendekatan modular dalam melakukan ITIL assessment. Hal tersebut memberikan fleksibilitas dalam penjadwalan dan eksekusi dari aktifitas assessment yang di butuhkan.



Gambar 2.10. ITIL Assessment Framework

Kegunaan dari framework ini adalah memastikan bahwa environment proses ITIL seimbang dan dalam keadaan yang saling berkaitan. Kegunaan lainnya adalah agar dapat memberikan stakeholder dari proses IT output yang memiliki *continual service improvement* sehingga kedepannya proses proses tersebut dapat di kembangkan. IPAF menegaskan bahwa dalam penggunaan framework ini telah di perhitungkan dari 4 sudut pandang:

1. Process Management Perspective

   Assessment ini mengindikasikan di level mana Process Management telah di terapkan pada tiap proses ITIL

2. Process Maturity Perspective

Assessment ini menggunakan Self Assessment techniques. Hal ini memberikan indikasi yang berguna dari sisi Praktisi IT dalam hal seberapa proses telah di terapkan.

3. Customer Perspective

   Assessment ini lebih kepada feedback customer tentang seberapa baik ITIL proses di fokuskan kepada kebutuhan dari customer

4. ITIL Perspective

   Assessment ini di lakukan dengan cara membandingkan keadaan yang telah terjadi saat ini dengan publikasi ITIL guidance oleh OGC. Kunci utamanya adalah apakah best practice dari ITIL sudah di terapkan pada ekseskusinya, terminologynya, organisasi, pengukuran serta reporting.

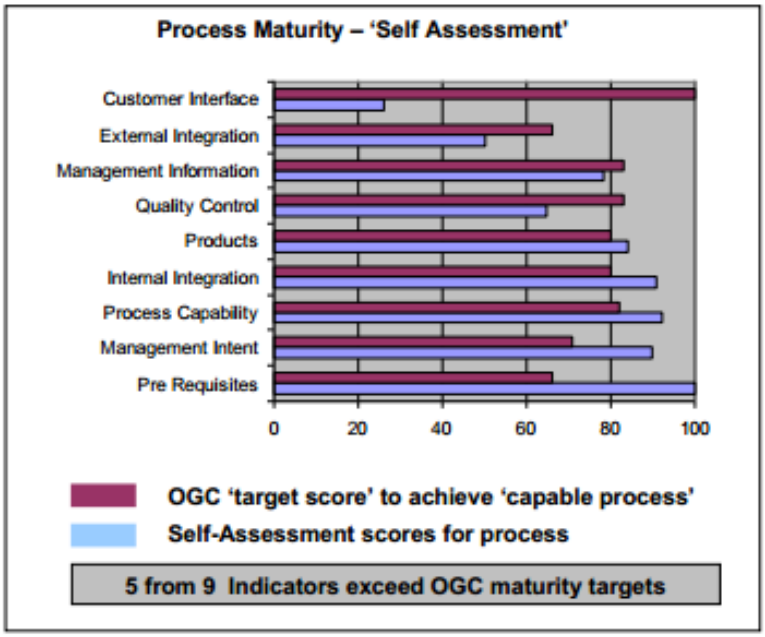
Output dari ke 4 area tersebut adalah sebuah laporan kesimpulan yang bernama *Current State Assessment.* Laporan ini akan berisi detail dari masing masing proses.

Dalam penulisan ini, peneliti melakukan assessment pada sisi proses management, yaitu incident management. Pada perusahan yang ingin menerapkan continual service improvement merupakan hal yang penting untuk memperhatikan proses management agar proses dapat terfedinisikan, tervisualisasikan, terukur, terkontrol, terdokumentasi dengan baik serta proses dapat di improve agar lebih efisien, efektif dan konsisten.

Untuk mendapatkan proses yang baik tersebut perlu di ketahui level kematangan dari proses ITIL yang terjadi dan yang ingin di dapatkan. Kematangan proses adalah sebuah indikasi seberapa jauh proses telah di kembangkan, di terapkan, dan seberapa capable jika di sandingkan dengan

continuous improvement. Melakukan assessment dari kematangan proses ITIL sama dengan memberikan kewenangan *Process Owner* untuk melihat seberapa lemah dan kuat proses yang terjadi.

Keuntungan dari ITIL Self Assessment adalah dapat memberikan baseline terhadap proses mana yang mau di assess dan di kembangkan. Tidak terlalu luas seperti CMM dan dapat di pilih yang mana yang akan masuk pada assessment dan yang mana yang tidak ingin di assess. ITIL Self Assessment sangat mudah di jadwalkan dan dapat di ulang ketika di butuhkan. Ilustrasi dari ITIL self Assessment dapat di lihat pada bagan di bawah.



Gambar 2.11. 5 From 9 Indicators exceed OGC maturity Targets

2.6 ITSM Self-Assessment

ITSM *self-assessment* dikembangkan oleh OGC sebagai alat bantu untuk mengukur tingkat kematangan fungsi dan proses ITIL, *self-assessment* ini terdiri dari kumpulan pertanyaan yang dapat memperlihatkan area mana saja yang perlu

untuk diperhatikan untuk meningkatkan proses ITIL secara keseluruhan. Penilaian kematangan pada ITSM *self-assessment* menggunakan 9 tingkat kematangan, yaitu:

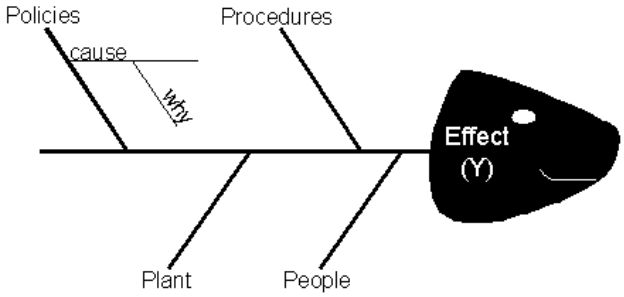
Tabel 2.5. – ITSM *self-assessment maturity* level

Level	Penjelasan
1	**Prequisite**, *ascertains whether the minimum level of prerequisite items are available to support the process activities.*
1.5	**Management Intent**, *establishes whether there are organizational policy statements, business objectives (or similar evidence of intent) providing both purpose and guidance in the transformation or use of the prerequisite items.*
2	**Process Capability**, *examines the activities being carried out. The questions are aimed at identifying whether a minimum set of activities are being performed.*
2.5	**Internal Integration** *seeks to ascertain whether the activities are integrated sufficiently in order to fulfill the process intent.*
3	**Products**, *examines the actual output of the process to enquire whether all the relevant products are being produced.*
3.5	**Quality Control**, *is concerned with the review and verification of the process output to ensure that it is in keeping with the quality intent.*
4	**Management Information**, *is concerned with the governance of the process and ensuring that there is adequate and timely information produced from the process in order to support necessary management decisions.*
4.5	**External Integration**, *examines whether all the external interfaces and relationships between the discrete process and other processes have been established within the organization. At this level, for* IT *service management, use of full* ITIL *terminology may be expected.*
5	**Customer Interface**, is *primarily concerned with the on-going external review and validation of the process to ensure that it remains optimized towards meeting the needs of the customer.*

Salah satu metodologi yang digunakan untuk meningkatkan proses ITIL adalah dengan cara mengukur terlebih dahulu tingkat kematangan proses yang sudah ada saat ini untuk kemudian dikaji apakah perlu untuk ditingkatkan atau tidak. Dengan mengkaji tingkat kematangan proses yang sudah atau belum ada pada saat ini, peneliti dapat mengetahui dengan jelas dimana *process improvement* sebaiknya mulai dilakukan (AlShamy, Elfakharany, & ElAziem, 2012).

2.7 Cause and Effect Diagram

Cause and Effect Diagram atau *fishbone diagram* digunakan untuk menangkap ide dan hasil pemikiran atas penggalian *root causes* dari sebuah masalah (Simon, The Cause and Effect (a.k.a. Fishbone) Diagram, 2010).



Gambar 2.12. Contoh *fishbone* diagram

Fishbone diagram dimulai dengan menuliskan masalah dalam bentuk pertanyaan dan diletakan sebagai "kepala" dalam bagan tulang ikan, menarik garis dan menggambarkan tulang-tulang utama yang melambangkan kategori, lalu setiap *root causes* dilakukan dengan menjawab "*why does that happen?"* dituliskan sebagai cabang pada tiap tulang terkait.

2.8 RPL (Rekayasa Piranti Lunak)

Menurut Pressman (2011) Rekayasa Perangkat Lunak adalah pembuatan dan penggunaan prinsip-prinsip keahlian teknik untuk m*end*apatkan perangkat lunak yang ekonomis yang handal dan bekerja secara efisien pada mesin yang sesungguhnya. Rekayasa Perangkat Lunak m*end*irikan suatu pondasi untuk suatu proses perangkat lunak yang lengkap dengan mengidentifikasi sejumlah aktifitas kerangka kerja yang berlaku untuk semua proyek perangkat lunak, terlepas dari hal ukuran dan kompleksitas.

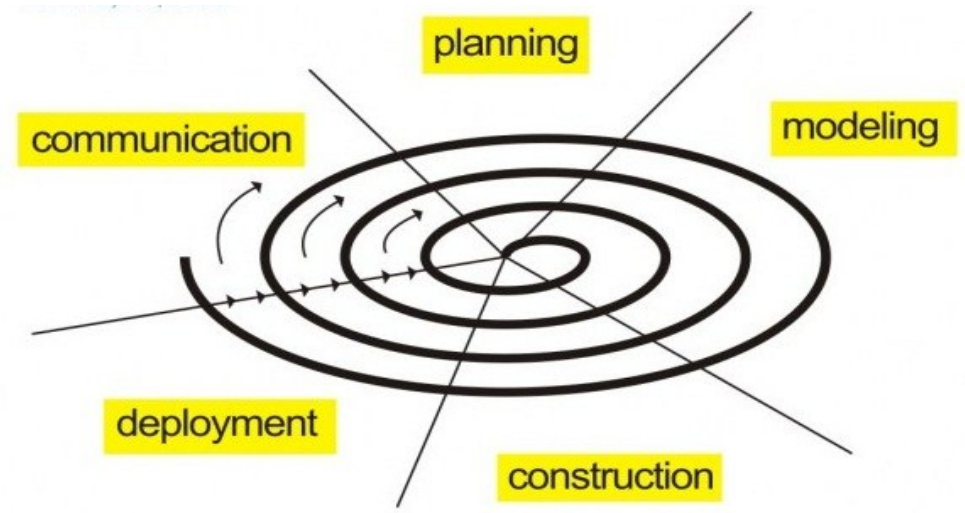
2.8.1 Pengertian Rekayasa Piranti Lunak

Menurut Pressman (2011) Rekayasa Perangkat Lunak adalah pembuatan dan penggunaan prinsip-prinsip keahlian teknik untuk m*end*apatkan perangkat lunak yang ekonomis yang handal dan bekerja secara efisien pada mesin yang sesungguhnya. Rekayasa Perangkat Lunak m*end*irikan suatu pondasi untuk suatu proses perangkat lunak yang lengkap dengan mengidentifikasi sejumlah aktifitas kerangka kerja yang berlaku untuk semua proyek perangkat lunak, terlepas dari hal ukuran dan kompleksitas.

2.8.2 Kerangka Proses

Sebuah kerangka proses menetapkan dasar bagi proses perangkat lunak yang lengkap dengan mengidentifikasi sejumlah kecil aktivitas

kerangka kerja yang berlaku untuk semua proyek perangkat lunak, tanpa memandang ukuran atau kompleksitas. Selain itu, kerangka proses mencakup serangkaian kegiatan yang berlaku di seluruh proses perangkat lunak.



Gambar 2. 13. Spiral Model

Berikut kerangka proses(*proces sframework*) yang berlaku untuk sebagian besar proses perangkat lunak:

- Komunikasi(*Communication*)

  Aktivitas kerangka kerja ini melibatkan komunikasi dan kolaborasi dengan pengguna (dan *stakeholder* lainnya) dan meliputi persyaratan pengumpulan dan kegiatan terkaitlainnya.

- Perencanaan(*Planning*)

  Aktivitas kerangka kerja ini menetapkan suatu rencana untuk rekayasa perangkat lunak yang menggambarkan tugas-tugas yang akan dilakukan, resiko yang mungkin, sumber daya yang akan

dibutuhkan, pekerjaan produk yang harus dihasilkan, dan jadwal kerja.

- Pemodelan(*Modeling*)

  Aktivitas kerangka kerja ini meliputi pembuatan model yang memungkinkan pengembang(*developer*) dan client untuk lebih memahami kebutuhan perangkat lunak(*software requirement*) dan desain untuk pencapaian kebutuhan tersebut.

- Konstruksi(*Construction*)

  Aktivitas kerangka kerja ini menggabungkan kegiatan *coding* dan pengujian(*testing*) yang diperlukan untuk mengungkapkan kesalahan dalam *code*.

- Penyebaran(*Deployment*)

  Perangkat lunak diberikan kepada *client* dimana client yang akan mengevaluasi dan memberikan umpan balik berdasarkan hasil evaluasi.

2.9 *Web* Application

2.9.1 Apache *Web* Server

Web server menurut Minoli (1998) adalah suatu program yang menawarkan pelayanan yang bisa diperoleh seluruh jaringan. *Web server* merupakan suatu tipe *server* khusus yang dapat berkomunikasi dengan *client* menggunakan HTTP. *Web server* menerima permintaan dari *client* dan meresponsnya, biasanya dengan mengembalikan sebuah dokumen atau gambar.

Proyek *Apache* HTTP *Server* adalah upaya untuk mengembangkan dan memelihara *server* HTTP *open source* untuk sistem operasi modern termasuk UNIX dan Windows NT. Tujuan dari proyek ini adalah untuk menyediakan *server*, aman efisien dan *extensible* yang menyediakan layanan HTTP dalam sinkron dengan HTTP standart saat ini. Httpd *Apache* telah menjadi *web server* yang paling populer di *Internet* sejak April 1996. *Apache* HTTP *Server* ("httpd") adalah sebuah proyek dari *Apache Software Foundation*.

Apache versi pertama dibuat oleh Robert Mc Cool (yang terlibat di NCSA) pada tahun 1996.Ditulis dalam bahasa C, perkembangannya dilakukan bersama rekan-rekan melalui email. Dia mengerjakan proyek itu bersama *Apache* grupnya : Brian Behl*end*orf, Roy T. Fielding, Rob Hartill, David Robinson, Cliff Skolnick, Randy Terbush, Robert S. Thau, Andrew Wilson, Eric Hagberg, Frank Peter dan Nicolas Pioch.

2.10 HTML (Hypertext Markup Language)

HyperText Markup Language (HTML) adalah sebuah bahasa *markup* yang digunakan untuk membuat sebuah halaman *web*, menampilkan berbagai informasi di dalam sebuah penjelajah*web Internet* dan *formating hypertext* sederhana yang ditulis kedalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegerasi. Dengan kata lain, berkas yang dibuat dalam perangkat lunak pengolah kata dan disimpan kedalam format ASCII normal sehingga menjadi *home page* dengan perintah-perintah HTML. Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML *(Standard Generalized Markup Language),* HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman *web*. HTML saat ini merupakan standar *internet* yang di definisikan dan di k*end*alikan penggunaannya oleh *World Wide Web Consortium* (W3C).HTML dibuat oleh kolaborasi Caillau TIM dengan Berners-lee robert ketika mereka bekerja di CERN pada tahun 1989 (CERN adalah lembaga penelitian fisika energi tinggi di Jenewa).

2.11 CSS (Cascading Style Sheet)

Cascading Style Sheet (CSS) merupakan salah satu bahasa pemrograman *web* untuk meng*end*alikan beberapa komponen dalam sebuah *web* sehingga akan lebih terstruktur dan seragam. Sama halnya styles dalam aplikasi pengolahan kata seperti Microsoft Word yang dapat mengatur beberapa style, misalnya heading, subbab, bodytext, footer, images, dan style lainnya untuk dapat digunakan

bersama-sama dalam beberapa file. Pada umumnya CSS dipakai untuk memformat tampilan halaman *web* yang dibuat dengan bahasa HTML dan XHTML.

CSS dapat meng*end*alikan ukuran gambar, warna body teks, warna tabel, ukuran border, warna border, warna hyperlink, warna mouse over, spasi antar paragraf, spasi antar teks, margin kiri/kanan/atas/bawah, dan parameter lainnya.CSS adalah bahasa style sheet yang digunakan untuk mengatur tampilan dokument. Dengan adanya CSS memungkinkan kita untuk menampilkan halaman yang sama dengan format yang berbeda.

2.12 Hyper Text Preprocessor (PHP)

Menurut Janet Valade (2006), PHP adalah bahasa pemprograman yang dirancang khusus untuk membuat suatu *web* dan ini merupakan alat untuk membuat halaman *web* yang dinamis.

PHP merupakan *SoftwareOpen Source* yang disebarkan dan dilisensikan secara gratis serta dapat di*download* secara bebas dari situs resminya http http://www.php.net/. Pengguna dapat mengubah *Source Code* dan m*end*istribusikannya secara bebas diedarkan secara gratis.

2.13 Database

Menurut Connolly dan Begg (2002), *database* adalah kumpulan data, yang terhubung secara logis yang dapat digunakan secara bersama, dan *desk*ripsi dari data ini dirancang untuk memenuhi kebutuhan informasi dari suatu organisasi.

Database juga dapat diartikan sebagai kumpulan data yang berfungsi sebagai penyedia informasi bagi pengguna. Objek-objek yang ada dalam sebuah basis data :

- Tabel, yaitu objek yang berisi tipe-tipe data.
- Kolom, yaitu sebuah tabel berisi kolom untuk menampung data. Kolom mempunyai tipe dan nama yang unik.
- Tipe data, yaitu sebuah kolom mempunyai sebuah tipe data. Tipe data yang dapat dipilih misalnya *character, numeric*, dan sebagainya.
- *Primary key*, yaitu kata kunci utama yang menjamin data agar unik, hingga dapat dibedakan dari data yang lain.
- *Foreign key*, merupakan kolom-kolom yang mengacu pada *primary key* dari tabel yang lain. Dengan kata lain, *primary key* dan *foreign key* digunakan untuk menghubungkan sebuah tabel dengan tabel lain .

Dalam *database* dikenal pula istilah *database relational*, yaitu basis data yang menghubungkan antara satu tabel dengan tabel lain dalam satu basis data. *Database relational* selalu menggunakan *field* kunci untuk m*end*efinisikan relasi antar tabel. Semakin banyak tabel yang ada, semakin banyak relasi yang diperlukan untuk menghubungkan semua tabel. Sebuah tabel tidak harus

langsung berhubungan dengan setiap tabel lain, tetapi setiap tabel dalam basis data terhubung satu sama lain (tidak ada tabel yang berdiri s*end*iri). Jadi tabel dapat berhubungan dengan setiap tabel lain dengan hubungan langsung atau tidak langsung.

2.13.1 Database *Management System* (DBMS)

Menurut Connolly dan Begg (2002), DBMS adalah suatu sistem perangkat lunak yang memungkinkan pengguna untuk menentukan, menciptakan, memelihara dan mengontrol akses ke *database*.

Secara khusus, DBMS menyediakan fasilitas berikut :

- Memungkinkan pengguna untuk menentukan *database*, biasanya melalui *Data Definition Language* (DDL). DDL memungkinkan *user* untuk menentukan tipe *user* dan struktur data m*end*orong data untuk disimpan ke *database*.
- Memungkinkan pengguna untuk melakukan *insert*, *update*, *delete* dan *retrieve* dari *database*, biasanya melalui *Data Manipulation Language* (DML).
- Menyediakan akses terkontrol ke *database*.

2.13.2 SQL dan MySQL

Menurut Connolly dan Begg (2002),, SQL merupakan bahasa yang dirancang untuk menggunakan relasi untuk mengubah masukan menjadi keluaran yang diharapkan.

SQL dimaksudkan untuk memenuhi keputusan berikut :

- Membuat *database* dan struktur relasi;
- Melakukan tugas dasar manajemen data, seperti pemasukan, modifikasi dan penghapusan data dari relasi.
- Melakukan *query* sederhana dan kompleks.

Standar SQL memiliki dua komponen :

- *Data Definition Language* (DDL) untuk menetapkan struktur database dan mengontrol akses ke data.
- *Data Manipulation Language* (DML) untuk m*end*apatkan kembali (*retrieve*) dan memperbahurui data.

Menurut Janet Valade (2006), MySql adalah aplikasi *database* yang cepat dan mudah menggunakan sistem manajemen basisdata relasional (RDBMS) yang diogunakan pada situs *web* pada umumnya. Namun, meskipunMySQL kurang dengan fitur yang lengkap daripesaing komersial, MySQL memiliki semua fitur yang dibutuhkan olehmayoritasdatabase pengembang. Lebih mudah untuk menginstal dan digunakan daripada pesaing komersial, dan selisih harga sangat m*end*ukung.

Keuntungan dari menggunakan MySql menurut Janet Valade(2006) antara lain:

- Cepat.
- Murah.
- Mudah digunakan.
- Bisa dijalankan di berbagai macam *operating system.*

- Dukungan teknis yang tersedia secara luas.
- Aman.
- M*end*ukung *database* besar.

Dapat memodifikasi MySql sehingga cocok dengan lingkungan progammernya