

BAB II

TINJAUAN PUSTAKA

2.1 Sistem Pendukung Keputusan (*Decision Support System*)

Sistem pendukung keputusan ialah proses pengambilan keputusan dibantu menggunakan komputer untuk membantu pengambil keputusan dengan menggunakan beberapa data dan model tertentu untuk menyelesaikan beberapa masalah yang tidak terstruktur. Keberadaan SPK pada perusahaan atau organisasi bukan untuk menggantikan tugas-tugas pengambil keputusan, tetapi merupakan sarana yang membantu bagi mereka dalam pengambilan keputusan. Dengan menggunakan data-data yang diolah menjadi informasi untuk mengambil keputusan dari masalah-masalah semi-terstruktur. Dalam implementasi SPK, hasil dari keputusan-keputusan dari sistem bukanlah hal yang menjadi patokan, pengambilan keputusan tetap berada pada pengambil keputusan. Sistem hanya menghasilkan keluaran yang mengkalkulasi data-data sebagaimana pertimbangan seorang pengambil keputusan. Sehingga kerja pengambil keputusan dalam mempertimbangkan keputusan dapat dimudahkan (Wibowo, 2011).

Sistem Pendukung Keputusan dirancang untuk mendukung seluruh tahap pengambilan keputusan mulai dari mengidentifikasi masalah, memilih data yang relevan, dan menentukan pendekatan yang digunakan dalam proses

pengambilan keputusan sampai mengevaluasi pemilihan alternatif-alternatif yang ada (Fitriani, 2012).

Karakteristik sistem pendukung keputusan menurut Wibowo (Wibowo, 2011):

1. Sistem Pendukung Keputusan dirancang untuk membantu pengambil keputusan dalam memecahkan masalah yang sifatnya semi terstruktur ataupun tidak terstruktur dengan menambahkan kebijaksanaan manusia dan informasi komputerisasi.
2. Dalam proses pengolahannya, sistem pendukung keputusan mengkombinasikan penggunaan model-model analisis dengan teknik pemasukan data konvensional serta fungsi-fungsi pencari/interogasi informasi.
3. Sistem Pendukung Keputusan, dirancang sedemikian rupa sehingga dapat digunakan/dioperasikan dengan mudah.
4. Sistem Pendukung Keputusan dirancang dengan menekankan pada aspek fleksibilitas serta kemampuan adaptasi yang tinggi.

Dengan berbagai karakter khusus di atas, SPK dapat memberikan berbagai manfaat dan keuntungan. Manfaat yang dapat diambil dari SPK menurut Kadarsah dalam tulisan Utami (Utami, 2012) :

1. SPK memperluas kemampuan pengambil keputusan dalam memproses data/informasi bagi pemakainya.
2. SPK membantu pengambil keputusan untuk memecahkan masalah terutama berbagai masalah yang sangat kompleks dan tidak terstruktur.
3. SPK dapat menghasilkan solusi dengan lebih cepat serta hasilnya dapat diandalkan.

4. Walaupun suatu SPK, mungkin saja tidak mampu memecahkan masalah yang dihadapi oleh pengambil keputusan, namun SPK dapat menjadi stimulan bagi pengambil keputusan dalam memahami persoalannya, karena mampu menyajikan berbagai alternatif pemecahan.

2.2 Kecerdasan Buatan (*Artificial Intelegent*)

Secara harfiah *artificial intelegent* adalah kecerdasan buatan, kecerdasan artifisial, inteligensia artifisial, atau inteliginsia buatan. Menurut Sutojo dkk (Sutojo dkk., 2011) cerdas adalah memiliki pengetahuan, pengalaman, dan penalaran untuk membuat keputusan dan mengambil tindakan.

Menurut Suyanto (Suyanto, 2011) para ilmuwan memiliki dua cara pandang berbeda tentang AI (*Artificial Intelegen*/kecerdasan buatan). Yang pertama adalah memandang AI sebagai bidang ilmu yang hanya fokus pada proses berfikir. Sedangkan yang kedua adalah memandang AI sebagai bidang ilmu yang fokus pada tingkah laku. Pada cara pandang kedua memandang AI secara lebih luas karena suatu tingkah laku selalu didahului dengan proses berfikir.

Suyanto juga menambahkan definisi AI yang paling tepat saat ini adalah *acting rationally* dengan pendekatan *relational agent*. Hal ini berdasarkan pemikiran bahwa komputer bisa melakukan penalaran secara logis dan juga bisa melakukan aksi secara relasional berdasarkan hasil penalaran tersebut (Suyanto, 2011).

Menurut Desiani, A dan Arhami, M (Desiani dan Arhami, 2006) dari beberapa definisi AI oleh para ahli maka dapat disimpulkan bahwa AI dapat dibagi dalam empat kategori yaitu:

1. Sistem yang dapat berpikir seperti manusia (*“Thinking humanly”*)
2. Sistem yang dapat bertindak laku seperti manusia (*“Acting humanly”*)
3. Sistem yang dapat berpikir secara rasional (*“Thinking rationally”*)
4. Sistem yang dapat bertindak laku secara rasional (*“Acting rationally”*)

Sejak pertama kali dikemukakan istilah AI pada tahun 1956 di konferensi Dartmouth, AI terus dikembangkan melalui berbagai penelitian mengenai teori-teori dan prinsip-prinsipnya. Perkembangan AI mengalami pasang surut mengikuti antusias para peneliti dan dana penelitian yang tersedia. Tetapi pada periode 1966 sampai 1974, perkembangan AI melambat. Tetapi sejak tahun 1980, AI menjadi sebuah industri yang besar dengan perkembangan yang sangat pesat. Banyak industri skala besar yang melakukan investasi besar-besaran dalam bidang AI (Suyanto, 2011).

Menurut Suyanto (Suyanto, 2011) dalam membangun produk-produk berbasis AI, dikelompokkan kedalam empat teknik yang setiap teknik memiliki karakteristik sendiri untuk menyelesaikan suatu masalah. Teknik-teknik tersebut yaitu:

1. Teknik *searching*

Teknik *searching* (pencarian) harus mendefinisikan ruang masalah untuk suatu masalah yang dihadapi dan mendefinisikan aturan produksi yang digunakan untuk mengubah suatu keadaan (*state*) ke keadaan (*state*) lainnya, selanjutnya memilih strategi untuk menemukan solusi.

2. Teknik *reasoning*

Berbeda dengan teknik *searching* teknik *reasoning* (penalaran) mempresentasikan masalah kedalam basis pengetahuan dan melakukan proses penalaran untuk menemukan solusi.

3. Teknik *planning*

Pada teknik *Planning* (perencanaan) masalah dipecah ke sub-sub masalah yang lebih kecil, menyelesaikan sub-sub masalah satu demi satu, kemudian menggabungkan solusi-solusi dari sub-sub masalah tersebut menjadi solusi lengkap dan tetap mengingat dan menangani interaksi yang terdapat pada sub-sub masalah tersebut.

4. Teknik *learning*

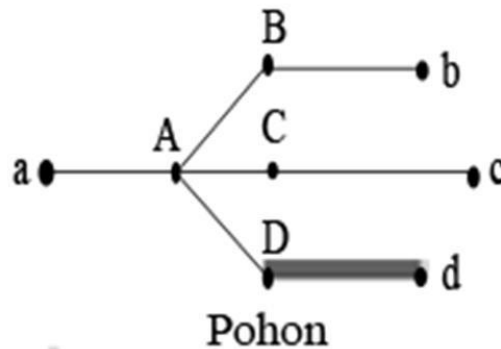
Teknik *learning* berbeda dengan teknik yang lain pada AI. Pada teknik *searching*, *reasoning*, dan *planning* kita harus mengetahui aturan yang berlaku untuk sistem yang akan dibangun. Tetapi, pada masalah tertentu terkadang kita tidak bisa mendefinisikan aturan secara benar dan lengkap untuk sistem yang akan kita bangun.

2.3 Pohon (*Tree*)

Sub Bab 2.3 seluruhnya diambil dari tulisan Wahyudin (Wahyudin, 2011). Pohon merupakan sebuah graf terhubung yang tidak mengandung sirkuit. konsep pohon (tree) dalam teori graf merupakan konsep yang sangat penting, karena terapannya diberbagai bidang ilmu. Oleh karenanya antara pohon (tree) sangat erat hubungannya dengan teori graf.

Definisi pohon adalah graf tak berarah terhubung yang tidak mengandung sirkuit, menurut definisi tersebut, ada dua sifat penting pada pohon yaitu terhubung dan tidak mengandung sirkuit. Pohon (tree) merupakan graf dimana dua simpul memiliki paling banyak satu lintasan yang menghubungkannya. Pohon seringkali memiliki akar, karena setiap simpul pada pohon hanya memiliki satu lintasan akses

dari setiap simpul lainnya, maka tidak mungkin bagi sebuah lintasan untuk membentuk simpul (loop) atau siklus (cycle) yang secara berkesinambungan melalui serangkaian simpul.



Gambar 2.1 Pohon (*tree*) (Wahyudin, 2011)

2.4 Decision Tree Learning

Decision tree learning adalah suatu metode belajar yang sangat populer dan banyak digunakan secara praktis. Metode ini merupakan metode yang berusaha menemukan fungsi-fungsi pendekatan yang bernilai diskrit dan tahan terhadap data-data yang terdapat kesalahan (*noise data*) serta mampu mempelajari ekspresi-ekspresi *disjunctive* (ekspresi *OR*). *Iterative Dichotomiser 3* (ID3), ASSISTANT, dan C4.5 merupakan jenis dari *decision tree learning*. Dalam membangun *decision tree learning* dibutuhkan evaluasi semua atribut yang ada menggunakan suatu ukuran statistik untuk mengukur efektifitas suatu atribut dalam mengklasifikasikan kumpulan sampel data. Dalam hal ini *information gain* adalah yang paling banyak digunakan (Suyanto, 2011).

Berikut adalah cara untuk mencari suatu *informasi gain* dari suatu atribut.

1. *Entropy*

Dalam menghitung *information gain* terlebih dahulu harus memahami suatu ukuran lain yang disebut *entropy*. Di dalam bidang *information theory*, *entropy* sering digunakan sebagai suatu parameter untuk mengukur heterogenitas (keberagaman) dari suatu sampel data. Menurut Slocum (Slocum, 2012) *entropy* adalah ukuran ketidakpastian dimana semakin tinggi *entropy*, maka semakin tinggi ketidakpastian. Dengan pohon keputusan, pengukuran ini digunakan untuk menentukan seberapa informatif sebuah simpul. Jika kumpulan sampel semakin heterogen, maka nilai *entropy*-nya semakin besar. Secara matematis *entropy* dirumuskan sebagai berikut:

$$entropy(S) = \sum_i^c -p_i \log_2 p_i$$

Ket:

c : Jumlah nilai yang ada pada atribut target (jumlah kelas klasifikasi)
 p_i : Menyatakan jumlah sampel untuk kelas i .

2. *Information Gain*

Setelah mendapatkan nilai *entropy* untuk suatu kumpulan sampel data, maka kita dapat mengukur efektivitas suatu atribut dalam mengklasifikasikan data. Menurut Slocum (Slocum, 2012) *Information Gain* (juga dikenal sebagai hanya Gain) menggunakan *entropy* untuk menentukan atribut yang terbaik yang digunakan untuk menciptakan perbedaan. Dengan menghitung Gain, kita menentukan tingkatan *entropy* dengan menggunakan atribut tersebut. Jadi, kolom dengan Gain yang lebih tinggi akan digunakan sebagai node dari pohon

keputusan. Ukuran efektifitas ini disebut sebagai *information gain*. Secara matematis, information gain dari suatu atribut A, dituliskan sebagai berikut:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Ket:

A	: atribut
V	: menyatakan nilai yang mungkin untuk atribut A
Values(A)	: himpunan nilai-nilai yang mungkin untuk atribut A
S _v	: jumlah sampel untuk nilai v
S	: jumlah sampel data
Entropy(S _v)	: <i>entropy</i> untuk sampel-sampel yang memiliki nilai v

2.5 Iterative Dichotomiser 3 (ID3)

ID3 adalah algoritma *decision tree learning* (algoritma pembelajaran pohon keputusan) yang menggunakan strategu pencarian *hill-climbing*, yaitu dimulai dari pohon kosong, kemudian secara progresif berusaha menemukan sebuah pohon keputusan yang mengklasifikasikan sampel-sampel data secara akurat tanpa kesalahan. Pertumbuhan cabang-cabang pohon keputusan pada lgoritma ID3 dilakukan sampai pohon tersebut mampu mengklasisifikasikan sampel data secara akurat dengan tingkat kebenaran 100 % sesuai dengan data latih (Suyanto, 2011). Adapun sample data yang digunakan oleh ID3 memiliki beberapa syarat menurut Setiawan (Setiawan, 2010), yaitu:

1. Deskripsi atribut-nilai. Atribut yang sama harus mendeskripsikan tiap contoh dan memiliki jumlah nilai yang sudah ditentukan.
2. Kelas yang sudah didefinisikan sebelumnya. Suatu atribut contoh harus sudah didefinisikan, karena mereka tidak dipelajari oleh ID3.

3. Kelas-kelas yang diskrit. Kelas harus digambarkan dengan jelas. Kelas yang kontinu dipecah-pecah menjadi kategori-kategori yang relatif, misalnya saja metal dikategorikan menjadi “hard, quite hard, flexible, soft, quite soft”.
4. Jumlah contoh (example) yang cukup. Karena pembangkitan induktif digunakan, maka dibutuhkan test case yang cukup untuk membedakan pola yang valid dari peluang suatu kejadian.

Gambar 2.2 adalah algoritma ID3 untuk membangun suatu pohon keputusan (Suyanto, 2011):

```

Function ID3(KumpulanSample, AtributTarget, KumpulanAtribut)
1. Buat simpul root
2. If semua sample adalah kelas i, maka Return pohon satu simpul Root dengan label=i
3. If KumpulanAtribut kosong return pohon satu simpul Root dengan label=nilai atribut target yang paling umum
   Else
     • A<-atribut the best classifier
     • Atribut keputusan untuk Root <- A
     • For  $v_i$  (setiap nilai pada A)
       • Tambahkan suatu cabang dibawah root dengan nilai  $v_i$ 
       • Buat suatu variable, misal  $Sample_{v_i}$ , sebagai himpunan bagian (subset) dari kumpulanSample bernilai  $v_i$  pada atribut A
       • If  $sample_{v_i}$  kosong
         • Then dibawah cabang ini tambahkan simpul daun dengan label nilai yang paling sering muncul
         • Else dibawah cabang ini tambahkan subtree dengan memanggil fungsi ID3( $Sample_{v_i}$ , AtributTarget, Atribut- $\{A\}$ )
       End
     End
4. Return root

```

Gambar 2.2 Algoritma ID3

2. 5. 1 ID3 : *Example*

Sub Bab 2.5.1 seluruhnya diambil dari buku tulisan Suyanto (Suyanto, 2011).

Perhatikan data penerimaan pegawai pada Tabel 2.1 di bawah ini. Terdapat 11 orang pelamar kerja dengan 3 parameter/atribut penilaian: IPK (Index Prestasi

Kumulatif), hasil test psikologi, dan hasil test wawancara. IPK dikelompokkan menjadi tiga kategori (bagus, cukup, kurang). Hasil test psikologi dikelompokkan menjadi tiga (Tinggi, Sedang, Rendah). Hasil test wawancara dikelompokkan menjadi dua kategori (Baik dan Buruk). Untuk data lengkap, seharusnya terdapat $3 \times 3 \times 2 = 18$ kombinasi sampel data. Tetapi pada Tabel 2.1 terdapat 11 data. Artinya, masih ada 7 sampel data lainnya yang belum diketahui.

Tabel 2.1 Data penerimaan pegawai (Suyanto, 2011)

Pelamar	IPK	Psikologi	Wawancara	Diterima
P1	Bagus	Tinggi	Baik	Ya
P2	Bagus	Sedang	Baik	Ya
P3	Bagus	Sedang	Buruk	Ya
P4	Bagus	Rendah	Buruk	Tidak
P5	Cukup	Tinggi	Baik	Ya
P6	Cukup	Sedang	Baik	Ya
P7	Cukup	Sedang	Buruk	Ya
P8	Cukup	Rendah	Buruk	Tidak
P9	Kurang	Tinggi	Baik	Ya
P10	Kurang	Sedang	Buruk	Tidak
P11	Kurang	Rendah	Baik	Ya

Dengan menerapkan algoritma ID3 maka didapat proses-proses sebagai berikut:

1. Rekursi level 0 iterasi ke-1

Memanggil fungsi ID3 dengan sampel *training* semua sampel data yaitu [8+, 3-] Untuk label *training* 'Diterima' dan atributnya adalah {IPK, Psikologi, Wawancara}. Tanda positif (+) untuk data yang keputusan diterima bernilai 'Ya' sebaliknya tanda negatif (-) untuk data yang keputusan diterima bernilai 'Tidak'. Selanjutnya membuat simpul akar untuk pohon yang akan dibuat dengan mencari *information gain* terbesar dari setiap atribut.

- IPK:

$$S = [8+, 3-], |S| = 11, \text{Entropy}(S) = 0,8454$$

$$S_{\text{bagus}} = [3+, 1-], |S_{\text{bagus}}| = 4, \text{Entropy}(S_{\text{bagus}}) = 0,8113$$

$$S_{\text{cukup}} = [3+, 1-], |S_{\text{cukup}}| = 4, \text{Entropy}(S_{\text{cukup}}) = 0,8113$$

$$S_{\text{kurang}} = [2+, 1-], |S_{\text{kurang}}| = 3, \text{Entropy}(S_{\text{kurang}}) = 0,9183$$

$$\begin{aligned} \text{Gain}(S, \text{IPK}) &= \text{entropy}(S) - (4/11)\text{entropy}(S_{\text{bagus}}) - \\ &\quad (4/11)\text{entropy}(S_{\text{cukup}}) - (3/11)\text{entropy}(S_{\text{kurang}}) \\ &= 0,8454 - (4/11) 0,8113 - (4/11) 0,8113 - (3/11) 0,9183 \\ &= 0,0049 \end{aligned}$$

- Psikologi:

$$S = [8+, 3-], |S| = 11, \text{Entropy}(S) = 0,8454$$

$$S_{\text{tinggi}} = [3+, 0-], |S_{\text{tinggi}}| = 3, \text{Entropy}(S_{\text{tinggi}}) = 0$$

$$S_{\text{sedang}} = [4+, 1-], |S_{\text{sedang}}| = 5, \text{Entropy}(S_{\text{sedang}}) = 0,7219$$

$$S_{\text{rendah}} = [1+, 2-], |S_{\text{rendah}}| = 3, \text{Entropy}(S_{\text{rendah}}) = 0,9183$$

$$\begin{aligned} \text{Gain}(S, \text{Psikologi}) &= \text{entropy}(S) - (4/11)\text{entropy}(S_{\text{tinggi}}) - \\ &\quad (4/11)\text{entropy}(S_{\text{sedang}}) - (3/11)\text{entropy}(S_{\text{rendah}}) \\ &= 0,8454 - (3/11) 0 - (5/11) 0,7219 - (3/11) 0,9183 \\ &= 0,2668 \end{aligned}$$

- Wawancara:

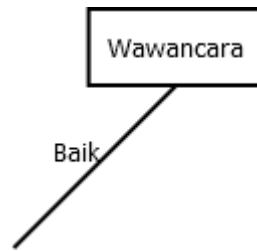
$$S = [8+, 3-], |S| = 11, \text{Entropy}(S) = 0,8454$$

$$S_{\text{baik}} = [6+, 0-], |S_{\text{baik}}| = 6, \text{Entropy}(S_{\text{baik}}) = 0$$

$$S_{\text{buruk}} = [2+, 3-], |S_{\text{buruk}}| = 5, \text{Entropy}(S_{\text{buruk}}) = 0,9710$$

$$\begin{aligned} \text{Gain}(S, \text{Wawancara}) &= \text{entropy}(S) - (4/11)\text{entropy}(S_{\text{baik}}) - \\ &\quad (4/11)\text{entropy}(S_{\text{buruk}}) \\ &= 0,8454 - (6/11) 0 - (5/11) 0,9710 \\ &= 0,4040 \end{aligned}$$

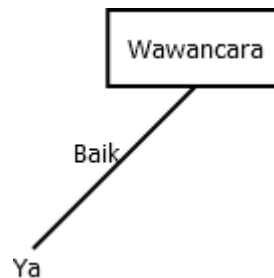
Dari nilai information gain di atas, $\text{gain}(S, \text{Wawancara})$ adalah yang paling besar maka atribut wawancara yang akan menjadi *root*. Untuk nilai baik pada atribut wawancara terdapat 6 sampel, berarti $\text{sample}_{\text{baik}}$ tidak kosong. Sehingga perlu memanggil fungsi ID3 dengan kumpulan sampel berupa $\text{sample}_{\text{baik}} = [6+, 0-]$, atribut target = ‘diterima’ dan kumpulan atribut = {IPK, Psikologi}. Maka pada tahap ini menghasilkan pohon seperti Gambar 2.3.



Gambar 2.3 Pohon keputusan pada rekursi level 0 iterasi ke-1

2. Rekursi level 1 iterasi ke-1

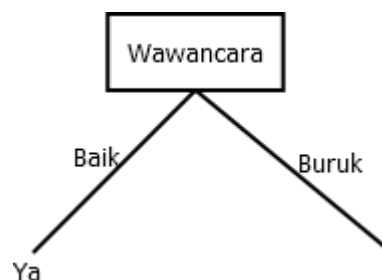
Karena semua sampel termasuk pada kelas 'Ya' pada sampel_{baik} = [6+, 0-] maka fungsi ini akan berhenti dan mengembalikan satu simpul tunggal *root* dengan label 'Ya'. Pada tahap ini menghasilkan pohon seperti Gambar 2.4.



Gambar 2.4 Pohon keputusan pada rekursi level 1 iterasi ke-1

3. Rekursi level 0 iterasi ke-2

Pada rekursi level 0 iterasi ke-1, sudah dilakukan pengecekan atribut wawancara dengan nilai 'baik'. Selanjutnya, dilakukan pengecekan untuk atribut 'wawancara' bernilai 'buruk'. Sehingga memanggil fungsi ID3 dengan sampel_{buruk} = [2+, 3-], target 'diterima', dan kumpulan atribut {IPK, Psikologi}. Sehingga pada tahap ini menghasilkan pohon pada Gambar 2.5.



Gambar 2.5 Pohon keputusan pada rekursi level 0 iterasi ke-2

4. Rekursi level 1 iterasi ke-2

Dengan atribut yang sama pada tahap berikutnya maka dilakukan perhitungan *information gain* untuk atribut IPK dan Psikologi.

- IPK:

$$S = \text{Sample}_{\text{buruk}} = [2+, 3-], |S| = 5, \text{Entropy}(S) = 0,9710$$

$$S_{\text{bagus}} = [1+, 1-], |S_{\text{bagus}}| = 2, \text{Entropy}(S_{\text{bagus}}) = 1$$

$$S_{\text{cukup}} = [1+, 1-], |S_{\text{cukup}}| = 2, \text{Entropy}(S_{\text{cukup}}) = 1$$

$$S_{\text{kurang}} = [0+, 1-], |S_{\text{kurang}}| = 1, \text{Entropy}(S_{\text{kurang}}) = 0$$

$$\begin{aligned} \text{Gain}(S, \text{IPK}) &= \text{entropy}(S) - (2/5)\text{entropy}(S_{\text{bagus}}) - (2/5)\text{entropy}(S_{\text{cukup}}) - \\ &\quad (1/5)\text{entropy}(S_{\text{kurang}}) \\ &= 0,9710 - (2/5) 1 - (2/5) 1 - (1/5) 0 \\ &= 0,1710 \end{aligned}$$

- Psikologi:

$$S = \text{Sample}_{\text{buruk}} = [2+, 3-], |S| = 5, \text{Entropy}(S) = 0,9710$$

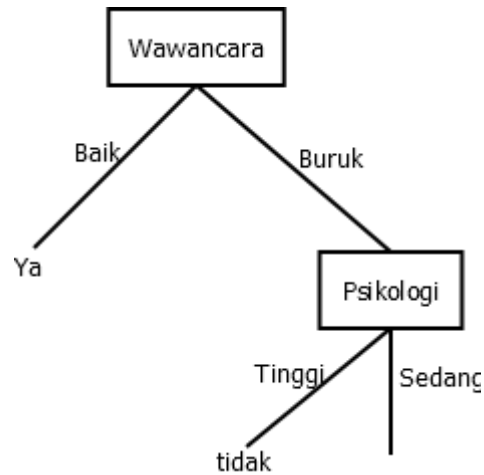
$$S_{\text{tinggi}} = [0+, 0-], |S_{\text{tinggi}}| = 0, \text{Entropy}(S_{\text{tinggi}}) = 0$$

$$S_{\text{sedang}} = [2+, 1-], |S_{\text{sedang}}| = 3, \text{Entropy}(S_{\text{sedang}}) = 0,9183$$

$$S_{\text{rendah}} = [0+, 2-], |S_{\text{rendah}}| = 2, \text{Entropy}(S_{\text{rendah}}) = 0$$

$$\begin{aligned} \text{Gain}(S, \text{Psikologi}) &= \text{entropy}(S) - (0/5)\text{entropy}(S_{\text{tinggi}}) - \\ &\quad (3/5)\text{entropy}(S_{\text{sedang}}) - (2/5)\text{entropy}(S_{\text{rendah}}) \\ &= 0,9170 - (0/5) 0 - (3/5) 0,9183 - (2/5) 0 = 0,4200 \end{aligned}$$

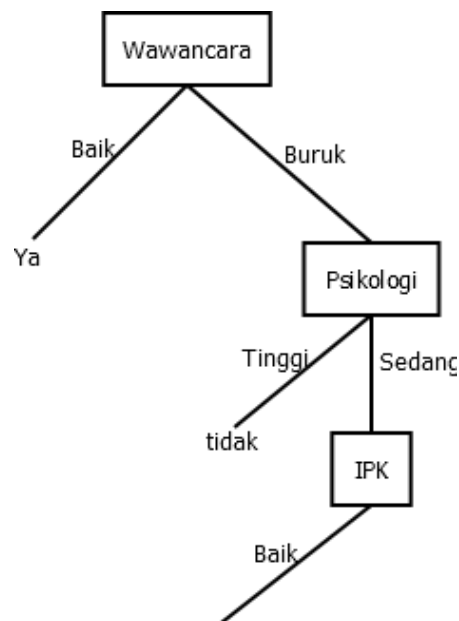
Dari dua *information gain* di atas, $\text{Gain}(S, \text{Psikologi})$ adalah yang paling besar sehingga akan dijadikan simpul berikutnya. Pada atribut Psikologi maka akan dicek nilainya satu persatu. Untuk nilai tinggi terdapat 0 sampel, berarti $\text{Sampel}_{\text{tinggi}}$ kosong. Sehingga dapat dibuat satu simpul daun (*leaf node*, simpul yang tidak mempunyai anak) dengan nilai yang paling sering muncul pada $\text{sampel}_{\text{buruk}}$ yaitu 'tidak'. Kemudian pengecekan dilanjutkan ke nilai sedang terdapat $\text{sample}_{\text{sedang}} = [2+, 1-]$ untuk pemanggilan fungsi ID3 dengan atribut {IPK}. Pada tahap ini menghasilkan pohon pada Gambar 2.6.



Gambar 2.6 Pohon keputusan pada rekursi level 1 iterasi ke-2

5. Rekursi level 2 iterasi ke-1

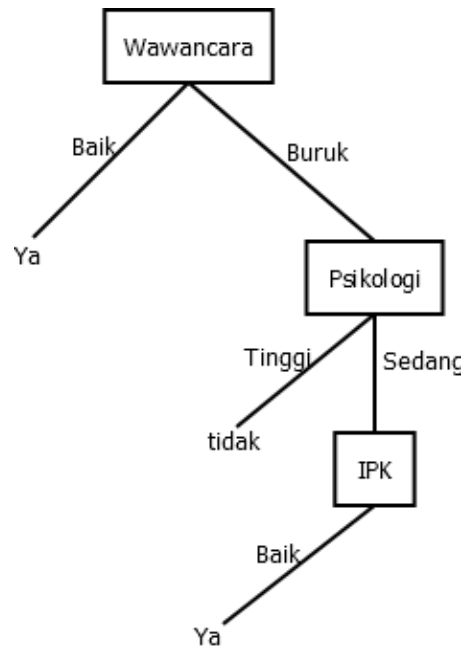
Pada pemanggilan tahap sebelumnya terdapat satu atribut yaitu atribut IPK maka secara otomatis atribut tersebut menjadi simpul berikutnya. Pada IPK bernilai 'bagus' pada $\text{sampe}_{\text{sedang}}=[2+,1-]$, terdapat 1 sampel yaitu $\text{sampel}_{\text{bagus}}=[1+, 0-]$. Maka memanggil fungsi ID3 dengan $\text{sampel}_{\text{bagus}}=[1+, 0-]$ dengan target 'diterima' dan kumpulan atribut $\{\}$. Pada tahap ini menghasilkan pohon pada Gambar 2.7.



Gambar 2.7 Pohon keputusan pada rekursi level 2 iterasi ke-1

6. Rekursi level 3 iterasi ke-1

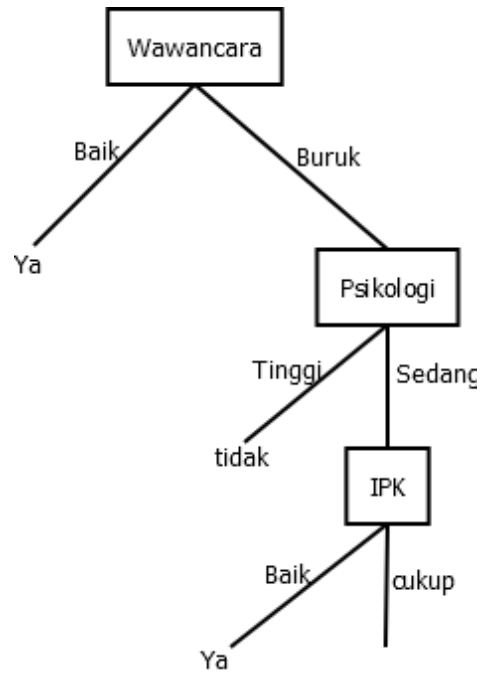
Pada pemanggilan fungsi sebelumnya pemanggilan sampel_{bagus}=[1+,0] maka fungsi akan berhenti dan menghasilkan daun dengan nilai 'Ya'. Sehingga dihasilkan pohon pada Gambar 2.8. selanjutnya proses kembali ke rekursi level 2 untuk iterasi ke-2.



Gambar 2.8 Pohon keputusan pada rekursi level 3 iterasi ke-1

7. Rekursi level 2 iterasi ke-2

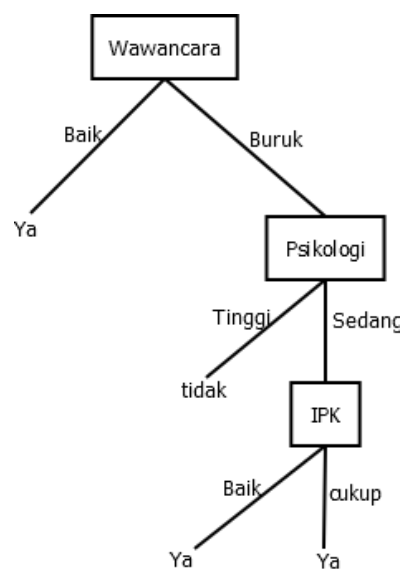
Pada tahap ini melanjutkan kenilai berikutnya yaitu pemanggilan fungsi ID3 dengan sample_{cukup} = [1+, 0-], atribut target = 'diterima' dan kumpulan atribut {}. Sehingga menghasilkan pohon pada Gambar 2.9.



Gambar 2.9 Pohon keputusan pada rekursi level 2 iterasi ke-2

8. Rekursi level 3 iterasi ke-2

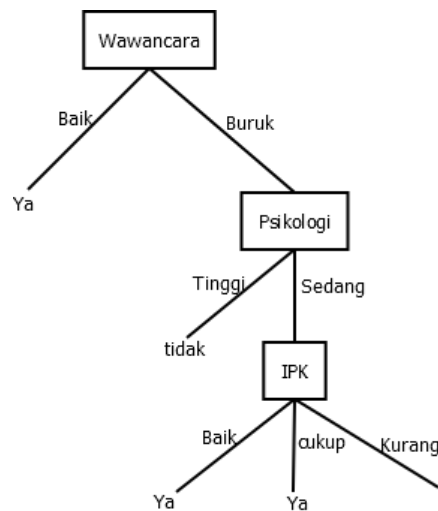
Pada pemanggilan fungsi sebelumnya pemanggilan sampel_{cukup}=[1+,0] maka fungsi akan berhenti dan menghasilkan daun dengan nilai 'Ya'. Sehingga dihasilkan pohon pada Gambar 2.10. Selanjutnya proses kembali ke rekursi level 2 untuk iterasi ke-3.



Gambar 2.10 Pohon keputusan pada rekursi level 2 iterasi ke-2

9. Rekursi level 2 iterasi ke-3

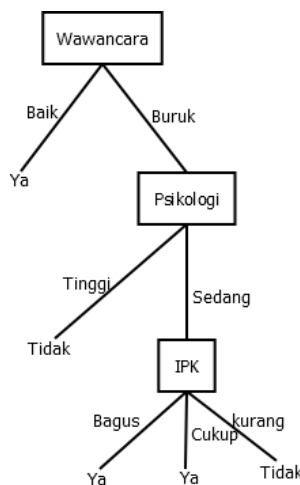
Pada tahap ini melanjutkan kenilai berikutnya yaitu pemanggilan fungsi ID3 dengan $\text{sample}_{\text{kurang}} = [0+, 1-]$, atribut target = 'diterima' dan kumpulan atribut {}. Sehingga menghasilkan pohon pada Gambar 2.11



Gambar 2.11 Pohon keputusan pada rekursi level 2 iterasi ke-3

10. Rekursi level 3 iterasi ke-3

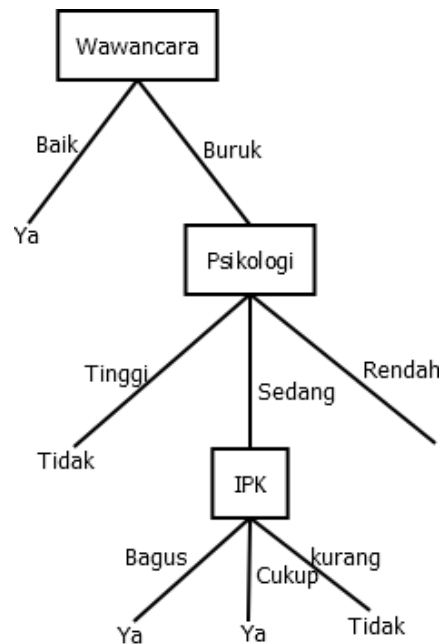
Pada pemanggilan fungsi sebelumnya pemanggilan sampel_{kurang} = [0+, 1-] maka fungsi akan berhenti dan menghasilkan daun dengan nilai 'Tidak'. Sehingga dihasilkan pohon pada Gambar 2.12. Selanjutnya proses kembali ke rekursi level 1 untuk iterasi ke-3.



Gambar 2.12 Pohon keputusan pada rekursi level 3 iterasi ke-3

11. Rekursi level 1 iterasi ke-3

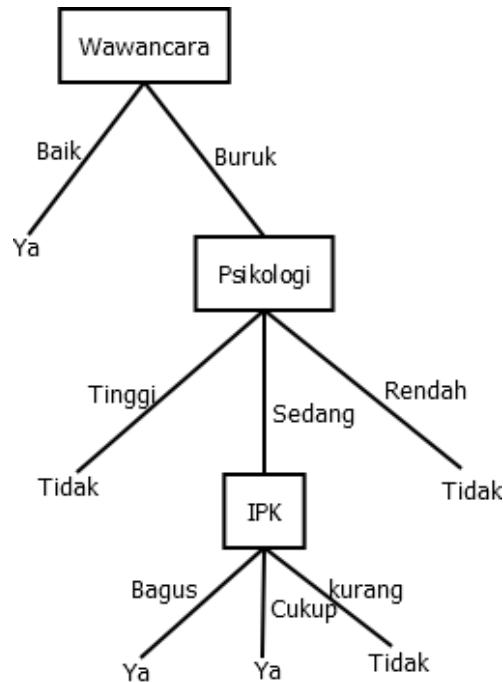
Telah dilakukan atribut psikologi pada nilai ‘tinggi’ dan ‘sedang’. Selanjutnya pada iterasi ini akan dilakukan pada nilai ‘rendah’. Dengan memanggil fungsi ID3 dengan sampel_{rendah} = [0+, 2-], atribut target ‘diterima’ dan kumpulan atribut {IPK}. Pada tahap ini akan menghasilkan pohon pada Gambar 2.13. Selanjutnya ke rekursi level 2 iterasi ke-4.



Gambar 2.13 Pohon keputusan pada rekursi level 1 iterasi ke-3

12. Rekursi level 2 iterasi ke-4

Pada pemanggilan fungsi sebelumnya pemanggilan sampel_{rendah} = [0+, 2-] maka fungsi akan berhenti dan menghasilkan daun dengan nilai ‘Tidak’. Dan semua proses selesai karena sudah tidak ada lagi nilai yang belum di cek. Sehingga dihasilkan pohon keputusan final seperti pada Gambar 2.14.



Gambar 2.14 Pohon Keputusan Final

Setelah diterapkannya algoritma ID3 maka didapatkan pohon keputusan pada Gambar 2.14. Dari pohon keputusan tersebut maka dapat disimpulkan rule-rule dalam menentukan diterima atau tidak diterimanya seorang pelamar. Maka rule-rule tersebut adalah sebagai berikut:

If (wawancara='baik') or ((wawancara='buruk') and (psikologi='sedang') and (IPK='bagus')) or ((wawancara='buruk') and (psikologi='sedang') and (IPK='cukup')) then diterima='Ya'

Else Diterima ='tidak'

Dari *rule-rule* yang diperoleh dapat digunakan untuk menduga data-data yang belum pernah dipelajari. Semakin banyak data *training* maka semakin tinggi tingkat akurasi *rule-rule* yang diperoleh.

2.6 *System Devolepment Life Cycle (SDLC)*

Sub bab 2.8 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasaarkan *best practice* atau cara-cara yang sudah teruji baik).

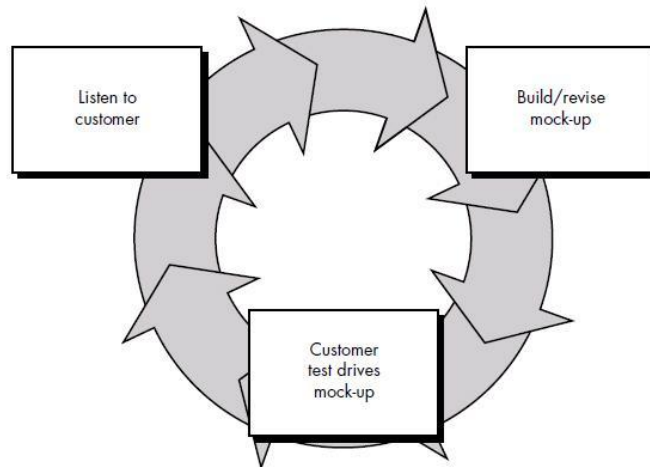
Terdapat banyak model yang dapat digunakan dalam membangun sistem berbasis SDLC yaitu model *Waterfall*, *Prototype*, *Rapid Aplication Development (RAD)*, *Iterative* dan *Spiral*.

2.7 *System Devolepment Live Cycle (SDLC) model prototype*

Sub Bab 2.7 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). Metode System Development life cycle (SDLC) dengan model prototipe (prototype) sangat baik digunakan untuk menyelesaikan masalah kesalahpahaman antara user dan analis yang timbul akibat user tidak mampu mendefinisikan secara jelas kebutuhannya. Prototyping adalah pengembangan yang cepat dan pengujian terhadap model kerja (prototipe) dari aplikasi baru melalui proses interaksi dan berulang-ulang yang biasa digunakan ahli sistem informasi dan ahli bisnis. Prototyping disebut juga desain aplikasi cepat (rapid application design/RAD) karena menyederhanakan dan mempercepat desain sistem.

Sebagian user kesulitan mengungkapkan keinginannya untuk mendapatkan aplikasi yang sesuai dengan kebutuhannya. Kesulitan ini yang perlu diselesaikan oleh analis

dengan memahami kebutuhan user dan menerjemahkannya ke dalam bentuk model (prototipe). Model ini selanjutnya diperbaiki secara terus menerus sampai sesuai dengan kebutuhan user.



Gambar 2.15 SDLC model *prototype* (Rosa dan Salahudin, 2011)

Berikut adalah penjelasan dari setiap tahapan yang dilakukan saat mengembangkan sistem dengan model prototipe berdasarkan Gambar 2.1:

1. *Listen to costomer*

Pada tahap ini pengembang mendengarkan kebutuhan pelanggan sebagai pemakai sistem perangkat lunak (*user*) untuk menganalisis serta mengembangkan kebutuhan *user*.

2. *Build/revise mock-up*

Mengonversi dari kebutuhan *user* pada tahap berikutnya menjadi suatu *mock-up*. Mock-up adalah suatu yang digunakan sebagai model desain yang digunakan untuk mengajar, demonstrasi, evaluasi desain, promosi, atau keperluan lain. Sebuah *mock-up* disebut sebagai prototipe perangkat lunak jika menyediakan atau mampu mendemostrasikan sebagian besar fungsi dari sistem perangkat lunak.

3. *Costomer test drives mock-up*

Customer melakukan pengujian terhadap *mock-up* yang telah dibuat. Jika telah sesuai prototipe akan diselesaikan sepenuhnya jika masih belum sesuai kembali ketahap pertama.

2.8 Pemrograman Berorientasi Objek

Sub Bab 2.8 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metodologi berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, Perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

1. Meningkatkan produktivitas

Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

2. Kecepatan pengembangan

Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.

3. Kemudahan pemeliharaan

Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

4. Adanya konsistensi

Karena sifat perwarisan dan penggunaan notasi yang sama pada saat analisis, perancangan, maupun pengkodean.

5. Meningkatkan kualitas

Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Saat ini sudah banyak bahasa pemrograman berorientasi objek. Banyak orang berpikir bahwa pemrograman berorientasi objek identik dengan bahasa Java. Memang bahasa Java merupakan bahasa yang paling konsisten dalam mengimplementasikan paradigma pemrograman berorientasi objek. Namun sebenarnya bahasa pemrograman yang mendukung pemrograman berorientasi objek tidak hanya bahasa Java.

2.9 *Unified Modeling Language (UML)*

Sub Bab 2.9 seluruhnya diambil dari buku tulisan Siswoutomo (Siswoutomo, 2005). UML dirilis tahun 1987 sebagai sebuah metode untuk menggambarkan desain *software*. Ia didesain oleh untuk konsorsium untuk mendesain dan menganalisa berorientasi objek. UML merupakan metode standar untuk dokumentasi software berorientasi objek.

Keuntungan menggunakan UML adalah:

1. *Software* terdesain dan terdokumentasi secara profesional sebelum dibuat.
2. Oleh karena mendesain terlebih dahulu, maka *reusable code* dapat dikode dengan tingkat efisiensi yang tinggi.
3. ‘Lubang’ dapat ditemukan saat penggambaran desain.
4. Dapat melihat gambaran besar dari suatu *software*.

UML menjanjikan akan menghasilkan hasil dengan biaya rendah, *software* lebih efisien, lebih dapat dipercaya, dan hubungan antar bagian yang terlibat menjadi lebih baik.

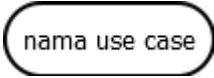
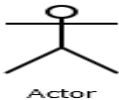

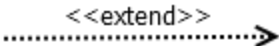

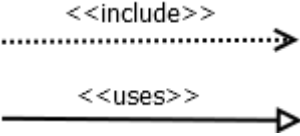
Terdapat banyak diagram yang dapat digunakan pada UML antara lain *object diagram*, *class diagram*, *component diagram*, *composite structure diagram*, *package diagram*, *deployment diagram*, *use case diagram*, *activity diagram*, *state machine diagram*, *sequence diagram*, *communication diagram*, *timing diagram*, dan *interaction overview diagram*.

2.9.1 Use case diagram

Sub Bab 2.9.1 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). *Use case* merupakan pemodelan untuk tingkah laku (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan di buat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin

dan dapat di pahami. Berikut adalah simbol-simbol yang digunakan pada *use case diagram*.

Tabel 2.2 Simbol *use case*


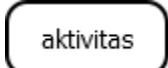



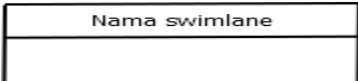
Simbol	Deskripsi
<p><i>Use case</i></p> 	<p>Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja frase nama <i>use case</i></p>
<p>Aktor</p> 	<p>Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat</p>
<p>Asosiasi</p> 	<p>Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>usecase</i> yang memiliki interaksi dengan aktor</p>
<p>Ekstensi</p> 	<p>Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu</p>
<p>Generalisasi</p> 	<p>Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya</p>
<p>Menggunakan / <i>include</i> / <i>uses</i></p> 	<p>Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini</p>

2.9.2 Activity diagram

Sub Bab 2.9.2 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan di sini adalah bahwa diagram aktivitas menggambarkan aktifitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh

sistem. Tabel 2.2 merupakan simbol-simbol yang terdapat pada diagram *activity diagram*.

Tabel 2.3 Simbol-simbol *activity diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan / <i>join</i> 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
<i>Swimlane</i> 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.9.3 Class diagram

Sub bab 2.9.3 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefisian kelas-kelas yang akan dibuat untuk membangun sistem.

Kelas memiliki apa yang disebut atribut dan metode atau operasi.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas sebagai berikut:

1. Kelas *main*

Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

2. Kelas yang menangani tampilan sistem

Kelas yang mendefinisikan dan mengatur tampilan kepemakai

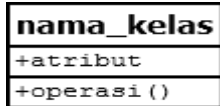






3. Kelas yang diambil dari pendefinisian *use case*

Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*

4. Kelas yang diambil dari pendefinisian data

Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Tabel 2.4 Simbol-simbol *class diagram*

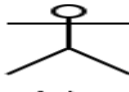
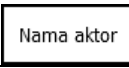

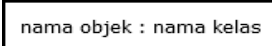

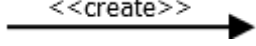
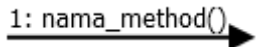
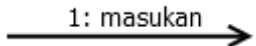
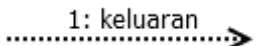
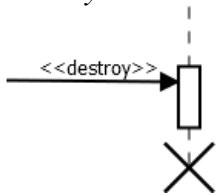
Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Antarmuka / <i>interface</i> 	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
Asosiasi / Association 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Asosiasi berarah / Directed association 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan / <i>dependency</i> 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / aggregation 	Relasi antar kelas dengan makna semua-bagian (<i>whole-part</i>)

2.9.4 *Sequence Diagram*

Sub Bab 2.9.4 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). Diagram sekuen menggambarkan kelakuan objek pada *use case*

dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Banyaknya diagram sekuen yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri. Berikut adalah simbol-simbol yang pada diagram sekuen:

Tabel 2.5 Simbol-simbol *sequence diagram*

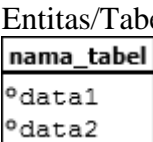
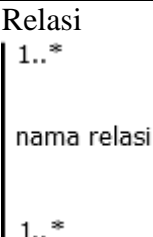
Simbol	Deskripsi
<p>Aktor</p>  <p>Atau</p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem
<p>Garis hidup/<i>lifetime</i></p> 	Menyatakan kehidupan suatu objek
<p>Objek</p> 	Menyatakan objek yang berinteraksi pesan
<p>Waktu aktif</p> 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
<p>Pesan tipe <i>create</i></p> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
<p>Pesan tipe <i>call</i></p> 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri
<p>Pesan tipe <i>send</i></p> 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek kirimi
<p>Pesan tipe <i>return</i></p> 	Menyatakan bahwa suatu objek yang telah menjalankan operasi atau metode menghasilkan suatu pengembalian ke objek tertentu
<p>Pesan tipe <i>destroy</i></p> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri.

2. 10 Conceptual Data Model (CDM)

Sub Bab 2.10 seluruhnya diambil dari buku tulisan Rosa dan Salahudin (Rosa dan Salahudin, 2011). CDM (*conceptual data model*) atau model konsep data merupakan konsep yang berkaitan dengan pandangan pemakai terhadap data yang disimpan dalam basis data. CDM dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi kebasis data. CDM merupakan hasil penjabaran lebih lanjut dari ERD. *Entity Relationship Diagram* (ERD) salah satu bentuk pemodelan basis data yang sering digunakan dalam pengembangan sistem informasi.

Berikut adalah simbol-simbol yang ada pada CDM:

Tabel 2.6 Simbol-simbol pada CDM

Simbol	Deskripsi
	Entitas/tabel yang menyimpan data dalam basis data
	Relasi antar tabel yang terdiri atas nama relasi dan <i>multiplicity</i>