BAB II

LANDASAN TEORI

II.1. Sistem Pakar (*Expert System*)

Sistem pakar adalah sistem berbasiskomputer yang menggunakan pengetahuan, fakta,dan tehnik penalaran dalam memecahkan masalahyang biasanya hanya dapat dipecahkan oleh seorangpakar dalam bidang tersebut Sistem pakarmemberikan nilai tambah pada teknologi untukmembantu dalam menangani era informasi yang semakin canggih.Konsep dasar suatu sistem pakarmengandung beberapa unsur, diantaranya adalahkeahlian, ahli, pengalihan keahlian, inferensi,aturan dan kemampuan menjelaskan.Keahlianmerupakan salah satu penguasaaan pengetahuan dibidang tertentu yang didapatkan baik secara formalmaupun non formal.Ahli adalah seseorang yangmempunyai pengetahuan tertentu dan mampumenjelaskan suatu tanggapan dan mempunyaikeinginan untuk belajar memperbaharui pengetahuan dalam bidangnya.

Pengalihan keahlianadalah mengalihkan keahlian dari seorang pakardan kemudian dialihkan lagi ke orang yang bukanahli atau orang awam yang membutuhkan.Sedangkan inferensi, merupakan suatu rangkaianproses untuk menghasilkan informasi dari faktayang diketahui atau diasumsikan. Kemampuanmenjelaskan, merupakan salah satu fitur yang harusdimiliki oleh sistem pakar setelah tersedia program di dalam komputer.Tujuan pengembangan sistem pakarsebenarnya tidak untuk menggantikan peran parapakar, namun untuk mengimplementasikanpengetahuan para pakar ke dalam bentuk perangkatlunak,

sehingga dapat digunakan oleh banyak orangdan tanpa biaya yang besar.Untuk membangun sistem yang difungsikanuntuk menirukan seorang pakar manusia harus bisamelakukan hal-hal yang dapat dikerjakan oleh parapakar. Untuk membangun sistem yang seperti itumaka komponen-komponen dasar yang minimalharus dimiliki adalah sebagai berikut:

1. Antar muka *(user interface).*
2. Basis pengetahuan *(knowledge base).*
3. Mesin inferensi *(Inference Engine).*

Kaidah produksi merupakan salah satumodel untuk merepresentasikan pengetahuan.Kaidah produksi menjadi acuan yang sangat seringdigunakan oleh sistem inferensi.Kaidah produksidituliskan dalam bentuk pernyataan ***IF-THEN*(Jika-Maka).**Pernyataan ini menghubungkanbagian premis ***(IF)*** dan bagian kesimpulan ***(THEN)***yang dituliskan dalam bentuk :

IF [premis] ***THEN*** [konklusi]

Kaidah ini dapat dikatakan sebagai suatuimplikasi yang terdiri dari dua bagian, yaitu premisdan bagian konklusi. Apabila bagian premisdipenuhi maka bagian konklusi akan bernilai benar.Bagian premis dalam aturan produksi dapatmemiliki lebih dari satu proposisi. Proposisi-proposisitersebut dihubungkan denganmenggunakan operator logika ***AND*** atau ***OR.***

Sebagai contoh :

IF Darah di dalam air kencing (hematuria)

AND Demam

AND Mudah lelah

AND Nyeri di daerah kandung kemih

AND Penurunan berat badan

AND Tekanan darah tinggi/hipertensi

THEN Kanker ginjal

(Sumber :Aprilia Sulistyohati, Taufiq Hidayat; *Seminar Nasional Aplikasi Teknologi Informasi 2008 (SNATI 2008))*

II.1.1. Kelebihan Sistem Pakar

Adapun banyak manfaat yang dapat diperoleh dengan mengembangkansistem pakar, antara lain (Kusumadewi, 2003):

1. Masyarakat awam non-pakar dapat memanfaatkan keahlian di dalam bidang tertentu tanpa kesadaran langsung seorang pakar
2. Meningkatkan produktivitas kerja, yaitu bertambahnya *efisiensi* pekerjaan tertentu serta hasil solusi kerja
3. Penghematan waktu dalam menyelesaikan masalah yang kompleks
4. Memberikan penyederhanaan solusi untuk kasus-kasus yang kompleks dan berulang-ulang
5. Pengetahuan dari seorang pakar dapat dikombinasikan tanpa ada batas waktu
6. Memungkinkan penggabungan berbagai bidang pengetahuan dari berbagai pakar untuk dikombinasikan. (Feri Fahrur Rohman, Ami Fauzijah, Media Informatika, *Vol. 6, No. 1, Juni 2008, 1-23 ISSN: 0854-4743*)

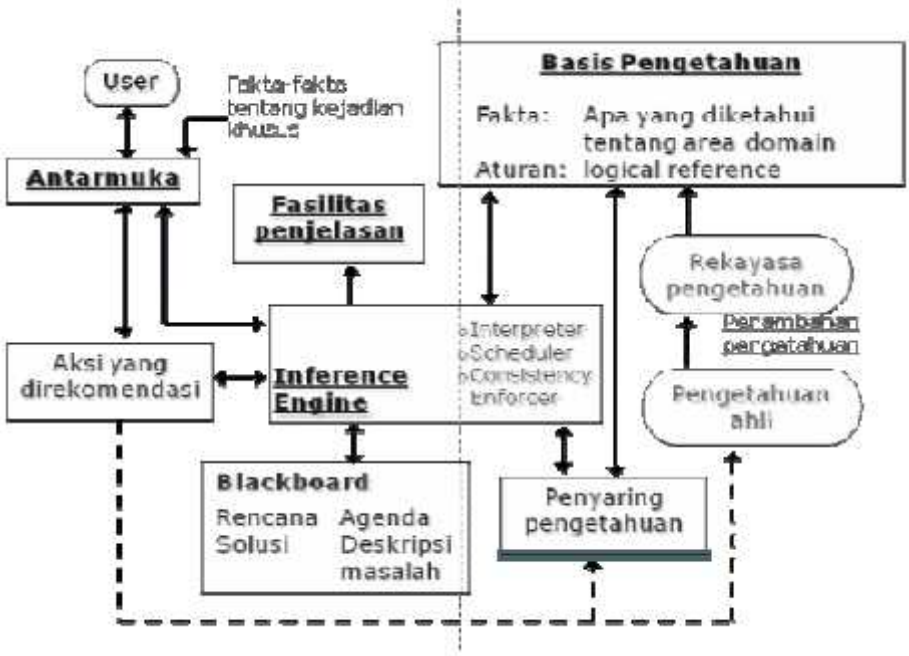
II.1.2. Kelemahan Sistem Pakar

Selain banyak manfaat yang diperoleh, ada juga kelemahan pengembangan sistem pakar, yaitu (Kusumadewi, 2003):

1. Daya kerja dan produktivitas manusia menjadi berkurang karena semuanya dilakukan secara otomatis oleh sistem
2. Pengembangan perangkat lunak sistem pakar lebih sulit dibandingkan dengan perangkat lunak konvensional. (Feri Fahrur Rohman, Ami Fauzijah, Media Informatika*, Vol. 6, No. 1, Juni 2008, 1-23 ISSN: 0854-4743*)

II.1.3Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu lingkunganpengembangan (*development environment*) dan lingkungan konsultasi (*consultationenvironment*) (Turban, 1995).Lingkungan pengembangan sistem pakar digunakanuntuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar,sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakarguna memperoleh pengetahuan pakar.Komponen-komponen sistem pakar dalamdua bagian tersebut dapat dilihat pada Gambar II.1.Komponen-komponen yang terdapat dalam sistem pakar adalah sepertiyang terdapat pada Gambar 1, yaitu *User Interface* (antarmuka pengguna), basispengetahuan, akuisisi pengetahuan, mesin *inference*, *workplace*, fasilitas penjelasan,perbaikan pengetahuan.



Gambar II.1. Komponen-komponen yang terdapat dalam sistem pakar

II.2.Metode Inferensi

Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Metode inferensi adalah program komputer yang memberikan metedologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace*, dan untuk memformulasikan kesimpulan (Turban, 1995).

Kebanyakan sistem pakar berbasis aturan menggunakan strategi inferensi yang dinamakan modus ponen. Berdasarkan strategi ini, jika terdapat aturan “IF A THEN B”, dan jika diketahui bahwa A benar, maka dapat disimpulkan bahwa B juga benar. Strategi inferensi modus ponen dinyatakan dalam bentuk:

$$[A \text{ And } (A \rightarrow B)] \rightarrow B \quad (1)$$

dengan A dan $A \rightarrow B$ adalah proposisi-proposisi dalam basis pengetahuan.

Terdapat dua pendekatan untuk mengontrol inferensi dalam sistem pakar berbasis aturan, yaitu pelacakan ke belakang (*Backward chaining*) dan pelacakan ke depan (*forward chaining*). (Feri Fahrur Rohman, Ami Fauzijah, Media Informatika, *Vol. 6, No. 1, Juni 2008, 1-23 ISSN: 0854-4743*)

II.3. *Unified Modelling Language (UML)*

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai negara dapat mengerti pemodelan perangkat lunak. Seperti yang kita ketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang sama tidaklah mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan oleh banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modelling Language* (UML). UML muncul karena adanya kebutuhan

pemodelan visual untuk menspesifikasi, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak.

"UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung" (Shalahuddin, M. dan Rosa A.S, 2014:137).

UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek. Seperti yang kita ketahui bahwa banyak hal di dunia sistem informasi yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkungan dan konteksnya. Begitu juga dengan perkembangan penggunaan UML bergantung pada *level* abstraksi penggunaannya. Jadi belum tentu pandangan yang berbeda dalam penggunaan UML adalah suatu yang salah, tapi perlu ditelaah dimanakah UML digunakan dan hal apa yang ingin divisualkan.

II.3.1. *Use Case Diagram*

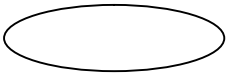
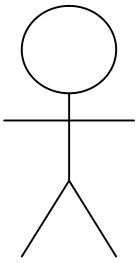
Use case atau diagram use case merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case.*

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel II.1 Simbol-Simbol *Use CaseDiagram*

Simbol	**Nama**	**Keterangan**
	*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
	*Actor*	Orang, proses, atau sistem lain yang berorientasi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
	Association	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan

		aktor.
<<extend>> ----------▶	*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
——————▶	*Generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
<<include>> ----------▶	*Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:156, Rekayasa Perangkat Lunak)

II.3.2. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*, jadi aktivitas yang dapat dilakukan oleh sistem.

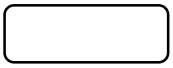
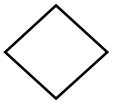
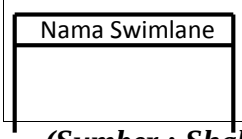
Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.

2. Urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

Tabel II.2 Simbol *ActivityDiagram*

Simbol	Nama	Keterangan
	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/ *Decision*	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
	Penggabungan/ *Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:162, Rekayasa Perangkat Lunak)

II.3.3. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur system dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

a. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas
b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar Antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasu, perancangan kelas yang dibuat tidak sesuaidengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai.

Kelas-kelas yang ada pada struktur system harus dapat melakukan fungsi-fungsi-fungsi sesuai dengan kebutuhan system sehingga pembuat perangkat lunak dapat membuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

a. Kelas main : kelas yang memiliki fungsi awal dieksekusi ketika system dijalankan.
b. Kelas yang menangani tampilan system (*view*) : kelas yang mendefenisikan dan mengatur tampilan ke pemakai.
c. Kelas yang diambil dari pendefenisian *use case* (*controller*) : kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefenisian *use case*, ,kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
d. Kelas yang diambil dari pendefenisian data (*model)* : kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah

kesatuan yang diambil maupun akan disimpan ke basis data. Semua table yang dibuat di basis data dapat dijadikan kelas, namun untuk table dari hasil ralasi atau atribut mutivalue pada ERD dapat dijadikan kelas tersendiri dapat juga tidak asalkan pengaksesannya dapat dipertanggungjawabkan atau tetap ada didalam perancangan kelas.

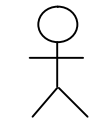
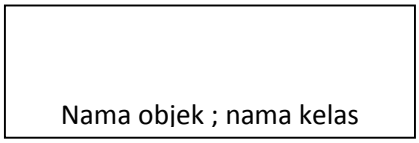
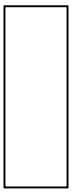
II.3.4. *Sequence Diagram*

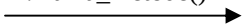
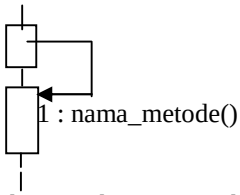
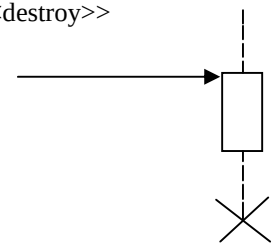
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirmkan dan diterima antar objek. Oleh karean itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimilik kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada *use case*.

Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefenisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefenisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel II.3. Simbol *Sequence Diagram*

Simbol	Deskripsi
 Nama aktor Atau Nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yanng berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
Pesan tipe create <<create>> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.

Pesan tipe call 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yanng dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe send 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe return 1 : keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy <<destroy>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhri, sebaiknya jika ada create maka ada destroy.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:165-167, Rekayasa Perangkat Lunak)

II.4. Teori Penyakit Flu Burung

Penyebab flu burung (Bird Flu/Avian Influenza) adalah virus influenza tipe A. Virus influenza termasuk familiOrthomixoviridae. Virus influenza tipe A dapat berubah-ubah bentuk (Drift, shift) dan dapat menyebabkan epidemi danpandemi. Berdasarkan sub tipenya terdiri dari Hemaglutinin (H) dan Neuramidase (N). Kedua huruf ini digunakansebagai identifikasi kode subtipe flu burung yang banyak jenisnya.Jenis virus yang terdapat pada manusia adalah jenis H1N1, H2N2, H3N3, H5N1, H9N2, H1N2, H7N7.

Sedangkan padabinatang H1-H5 dan N1-N9. Strain yang sangat Virulen/ganas dan menyebabkan flu burung adalah dari sub tipe AH5N1. Virus tersebut dapat bertahan hidup di air sampai 4 hari pada suhu 220 C dan lebih dari 30 hari pada suhu 00 C.Virus akan mati pada pemanasan 600 C selama 30 menit atau 560 C selama 3 jam dan dengan detergent, desinfektanmisalnya formalin, serta cairan yang mengandung Iodin.(Sumber :Titik Lusiani, Andhika kurniawan Cahyono, Seminar Nasional Sistem dan Informatika 2006; Bali, November 17, 2006, SNSI06-026)

II.4.1 Cara Penularan Flu Burung

Flu burung menular dari unggas ke unggas, dan dari unggas ke manusia.Penularan flu burung pada manusia diantaranyabisa melalui air liur, lendir dari hidung dan *Feces* (tinja) atau debu yang dicemari tinjanya.Penyakit ini dapat menularmelalui udara yang tercemar virus H5N1 yang berasal dari kotoran atau sekreta burung/unggas yang menderita fluburung.

Penularan dari unggas ke manusia juga bisa terjadi jika bersentuhan langsung dengan unggas yang terinfeksi fluburung.Misalnya pekerja di peternakan ayam, atau pekerja pemotong ayam.Memakan daging ayam dan telur matangtidak menyebabkan tertular flu burung.(Sumber : Titik Lusiani, Andhika kurniawan Cahyono, Seminar Nasional Sistem dan Informatika 2006; Bali, November 17, 2006, SNSI06-026)

II.4.2. Cara Mendiagnosa Flu Burung

Flu burung memiliki gejala yang bervariasi.Pada kasus yang sangat ganas (Akut) ditandai dengan kematian tanpa disertaigejala klinis.Hewan tampak sehat tetapi tiba-tiba mati. Namun pada umumnya gejala yang ditimbulkan oleh infeksi virusflu burung akan menunjukkan gejala-gejala, antara lain:

1. Kasus suspek (tersangka)

   Kasus Suspek adalah kategori dari penyakit flu burung yang paling ringan. Biasanya seseorang yang menderita InfeksiSaluran Penafasan Akut (ISPA) dengan gejala: demam (temperatur lebih dari 38 O C), batuk dan atau sakit tenggorokandan hidung beringus.

2. Kasus Probable

   Kasus “Probable” adalah kasus suspek dengan salah satu keadaan sebagai berikut:

   a. 7 hari (seminggu) terakhir sebelum sakit, mengunjungi peternakan yang sedang terjangkit flu burung.

b. 7 hari (seminggu) sebelum sakit, kontak dengan unggas sakit atau mati atau menggunakan produk mentahunggas seperti pupuk kandang dan lain-lain.
c. Kontak dengan kasus konfirmasi flu burung dalam masa penularan.
d. Bekerja pada suatu laboratorium yang sedang memproses spesimen manusia atau binatang yang dicurigaimenderita flu burung.
e. Cluster (kelompok) radang paru berat (pneumonia berat).
f. Pemeriksaan darah : Leukosit jumlah kurang dari 5000, Limfositopenia dan Trombositopenia.
g. Hasil pemeriksaan dengan HI tes positif pada spesimen tunggal atau kenaikan titer sepasang spesimen kurangdari 4 kali.

3. Kasus Konfirmasi

Kasus konfirmasi adalah kasus suspek atau “Probable” disertai oleh salah satu hasil pemeriksaan laboratorium:

a. Kultur virus influenza A/H5N1 positif.
b. RT-PCR influenza (H5) positif.
c. Peningkatan titer antibodi H5 sebesar 4 kali atau lebih pada pemeriksaan spesimen kedua dengan MikroNeutralization tes.
d. IFA tes positif (+) dengan antibodi Monoklonal/influenza A/H5.

4. Gejala Klinis / Observasi

Gejala klinis yang ditemui seperti gejala pada umumnya, yaitu: demam, sakit tenggorokan, batuk, beringus, nyeri otot,sakit kepala, lemas. Dalam waktu singkat penyakit ini dapat menjadi lebih berat yaitu peradangan di paru-paru(pneumonia),

dan apabila tidak cepat ditangani dengan baik dapat menyebabkan kematian. Masa Inkubasi flu burungdapat dibedakan juga pada manusia dan unggas

a. Pada unggas : 1 minggu
b. Pada manusia : 1-3 hari, masa infeksi 1 hari sebelum sampai 3-5 hari sesudah timbul gejala.Pada anak sampai 2 hari. (Titik Lusiani, Andika Kurniawan Cahyono 17 november 2017)