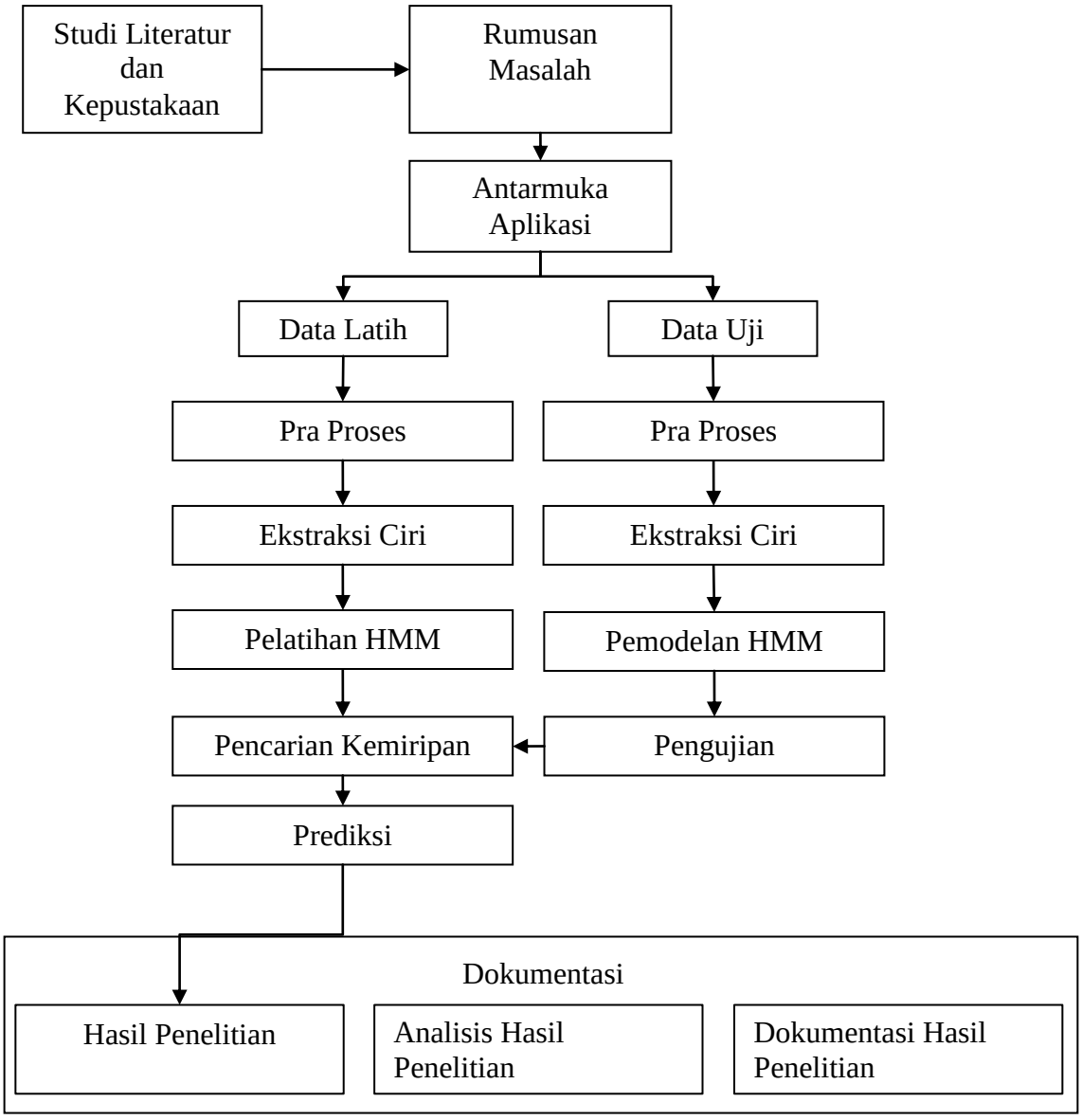
BAB III

METODOLOGI PENELITIAN

3.1 Desain Penelitian

Berikut merupakan desain penelitian yang akan digunakan pada proses penelitian penerapan *Hidden Markov Models* :



Gambar 3.1 *Desain Penelitian*

Penjelasan mengenai gambar desain penelitian adalah sebagai berikut:

1. Rumusan masalah merupakan dasar pemikiran dan merupakan acuan dalam penelitian ini. Dalam penelitian ini, permasalahan yang akan di analisis adalah mengenai pengenalan suara pembicara. untuk lebih jelas

mengenai rumusan masalah dari penelitian ini dapat dilihat pada subbab 1.2 rumusan masalah.

2. Studi literatur dan kepustakaan dilakukan dengan mempelajari dan memahami teori-teori yang berkaitan dengan penelitian ini seperti masalah pemrosesan sinyal suara, pencarian fitur dari sinyal suara, implementasi *Hidden Markov Models* untuk pelatihan maupun pengenalan suara pembicara, dan yang berkaitan dengan pengembangan perangkat lunak.
3. Data latih maupun data uji didapatkan melalui antarmuka aplikasi yang dikembangkan. Data latih merupakan data sampel suara dari *Target Speaker*. Sedangkan data uji merupakan data sampel suara dari *Test Speaker*.
4. Data latih maupun uji melalui pra proses terlebih dahulu, dimana praproses tersebut adalah normalisasi sinyal, normalisasi dilakukan agar sinyal suara memiliki rentang nilai yang sama. Selain itu dilakukan juga penghilangan *silence-frame, silence-frame* merupakan sinyal suara yang tak bernilai. Penghilangan *silence-frame* dilakukan menggunakan algoritma *EndPoint Detection*.
5. Setelah melalui tahap praproses, sinyal suara dikelompokan menjadi beberapa blok melalui proses *framing*, setelah di lakukan *framing*, kemudian dilakukan proses *window* pada *frame-frame* yang telah dihasilkan pada tahap sebelumnya. Hal ini bertujuan untuk meminimalkan

diskontinuitas pada bagian awal dan akhir sinyal. Model *window* yang digunakan pada sistem ini adalah *hamming window*.

6. Setelah melalui *framing* dan *windowing*, sinyal suara diproses lebih lanjut menggunakan ekstraksi ciri. Pada penelitian ini ekstraksi ciri yang digunakan adalah MFCC (*Mel-Frequency Cepstral Coefficients*).
7. Pada tahap ini, data latih yang telah di ekstrak cirinya akan dilakukan pelatihan menggunakan HMM untuk mendapatkan *template* model dari data tersebut. Sementara itu data uji dilakukan pembuatan model yang nantinya akan dibandingkan dengan template model.
8. Setelah model dari data uji didapatkan, selanjutnya model tersebut dibandingkan dengan *template* model yang telah dibuat dan disimpan sebelumnya untuk mencari kemiripan antar model. Pada tahap ini dicari model dengan tingkat kemiripan yang tinggi.
9. Aplikasi Pengenal Suara Pembicara atau disebut APSP merupakan nama perangkat lunak yang dikembangkan.
10. Metode pendekatan yang digunakan dalam penelitian ini menggunakan pendekatan *Object Oriented* dengan model proses *prototype*.
11. Dokumentasi berupa dokumen teknis perangkat lunak, paper dan dokumen skripsi sebagai hasil dari penelitian.

3.2 Alat dan Bahan Penelitian

3.2.1 Alat Penelitian

Alat yang digunakan dalam penelitian ini adalah seperangkat komputer dengan spesifikasi yang cukup untuk menjalankan perangkat lunak *Netbeans IDE* 7.3 dengan menggunakan bahasa pemrograman *Java* yang berjalan pada Sistem Operasi *Windows XP SP3* 32bit. Adapun spesifikasi dari computer yang digunakan dalam penelitian adalah:

- Processor dual core 2.1 Ghz
- RAM 2 GB
- *Harddisk* 320 GB
- Monitor dengan kemampuan resolusi 1366 x 768 pixel, dengan kedalaman warna 32 bit
- Perangkat *Mouse* dan *Keyboard*
- *Soundcard* internal
- *Microphone* untuk melakukan perekaman suara

3.2.2 Bahan Penelitian

Bahan penelitian yang digunakan dalam penelitain ini menggunakan sampel-sampel suara yang telah direkam sebelumnya. Sampel tersebut disimpan dalam bentuk file *WAV* dengan format suara satu kanal suara, 22050 Hz *sampling rate*, 16 bit tiap sampel. Sampel tersebut berisi ucapan yang telah ditentukan

sebelumnya, yaitu ucapan “Pendidikan”. setiap naracoba diambil sampelnya untuk pelatihan diambil sebanyak 10 sampel, yang nantinya akan diambil data sampel tersebut untuk membuat pola HMM tiap-tiap pembicara untuk dijadikan acuan pada tahap pengenalan. Algoritma yang digunakan untuk membuat pola HMM tersebut adalah algoritma *forward-backward*.

3.3 Metode Penelitian

Pada penelitian ini, metode yang digunakan meliputi metode pengumpulan data dan metode pengembangan perangkat lunak.

3.3.1 Metode Pengumpulan Data

Metode pengumpulan data dalam skripsi ini adalah studi literatur. Studi literatur dilakukan dengan mengumpulkan dan mempelajari literatur atau kepustakaan yang berkaitan dengan skripsi ini, seperti teori dan konsep *Hidden Markov Models* dan pembahasan mengenai masalah identifikasi suara pembicara melalui literatur-literatur seperti buku (*textbook*), *paper*, dan sumber ilmiah lain seperti situs internet ataupun artikel dokumen teks yang berhubungan dengan penelitian.

3.3.2 Metode Pengembangan Perangkat Lunak

3.3.2.1 Pendekatan Pengembangan Perangkat Lunak

Dalam proses pengembangan perangkat lunak ini, pendekatan yang digunakan adalah pendekatan berorientasi objek, dimana dalam paradigma ini domain permasalahan diabstraksikan sebagai suatu set objek yang mempunyai atribut dan perilaku tertentu.

Pada paradigma berorientasi objek ini, ada beberapa konsep yang harus diketahui, yaitu :

1. *Class* dan *Object*

   Class merupakan model yang berisi kumpulan *attribute* dan *method* dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh *class* manusia memiliki *attribute* berat, tinggi, usia kemudian memiliki *method* makan, minum, tidur. *Method* dalam sebuah *class* dapat merubah *attribute* yang dimiliki oleh *class* tersebut. Sebuah *class* merupakan dasar dari modularitas dan struktur dalam pemrograman berorientasi object.

   Sedangkan *Object* merupakan perwujudan dari *class*, setiap *object* akan mempunyai *attribute* dan *method* yang dimiliki oleh *class*-nya, contohnya: amir, ahmad, yani merupakan *object* dari *class* manusia. Setiap *object* dapat berinteraksi dengan *object* lainnya meskipun berasal dari *class* yang berbeda.

2. *Attribute*

Adalah berbagai variabel yang mengitari *class*, dengan nilai datanya bisa ditentukan di *object*.

3. *Operations, Method,* dan *Services*

   Setiap *object* membungkus data (yang direpresentasikan dalam suatu koleksi *attribute*) dan algoritma yang akan mengolah data tersebut. Algoritma-algoritma tersebutlah yang dimaksud dengan *operations, method* atau *services*.

4. *Messages*

   Suatu *class* harus berinteraksi dengan *class* lainnya untuk mencapai suatu tujuan tertentu. *Messages* ini memungkinkan *object* untuk menstimulasi *object* lainnya untuk melakukan suatu *behavior* terentu.

5. *Encapsulation, Inheritance,* dan *Polymorphism*

   Ketiga hal ini merupakan karakteristik dari paradigma berorientasi objek, *Encapsulation* yaitu merupakan suatu mekanisme untuk menyembunyikan atau memproteksi suatu proses dari kemungkinan interferensi atau penyalahgunaan dari luar sistem dan sekaligus menyederhanakan penggunaan sistem tersebut.

   Inheritance merupakan konsep mewariskan *attribute* dan *method* yang dimiliki oleh sebuah *class* kepada *class* turunannya (*subclass*). Dengan konsep ini *class* yang dibuat cukup mendefinisikan *attribute* dan *method*

yang spesifik didalamnya, sedangkan *attribute* dan *method* yang lebih umum akan didapatkan dari *class* yang menjadi induknya.

Polymorphism merupakan konsep yang memungkinkan digunakannya suatu *interface* yang sama untuk memerintah suatu *object* agar melakukan suatu tindakan yang mungkin secara prinsip sama tetapi secara proses berbeda.

Untuk pemodelan perangkat lunak berorientasi objek, digunakan UML (*Unified Modeling Language*) yang merupakan bahasa standar yang digunakan untuk memvisualisasikan dan menjelaskan artifak dari proses analisis dan desain berorientasi objek. UML menyediakan standar notasi dan diagram-diagram yang bisa digunakan untuk memodelkan sistem.

Diagram-diagram pada UML terbagi kedalam 3 klasifikasi, yaitu :

1. *Behavior Diagrams*

   Jenis diagram yang menggambarkan perilaku fitur dari sistem atau proses bisnis. Diagram-diagram yang termasuk dalam klasifikasi ini adalah *activity diagram, state machine diagram, use case diagram,* dan ke 4 subset dari *interaction diagrams.*

2. *Interaction Diagrams*

   Sebuah subset dari diagram perilaku yang menekankan pada interaksi antar objek. Diagram-diagram yang termasuk dalam klasifikasi ini adalah

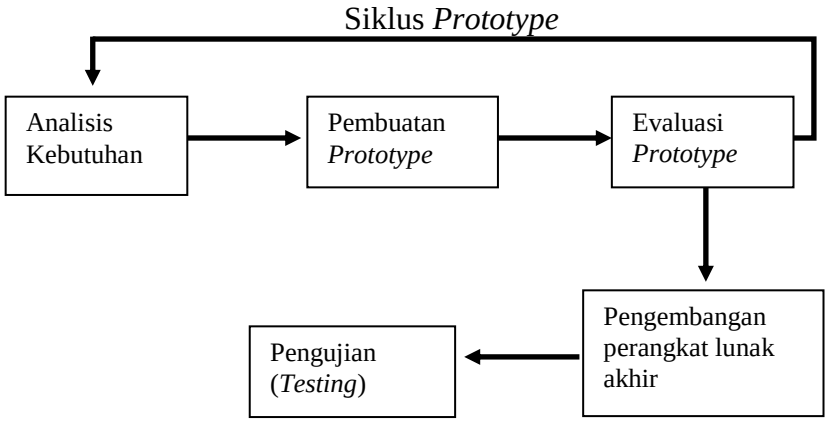
communication diagram, interaction overview diagram, sequence diagram, dan *timing diagrams.*

3. *Structure Diagrams*

   Jenis diagram yang menggambarkan unsur-unsur yang harus ada pada sistem. Diagram-diagram yang termasuk dalam klasifikasi ini adalah *composite structure diagram, component diagram, deployment diagram, object diagram,* dan *package diagrams.*

3.3.2.2 Model Proses

Dalam pengembangan perangkat lunak, penulis menggunakan model *Prototype*. Untuk lebih jelasnya dapat dilihat pada gambar berikut:



Gambar 3.2 *Model Pengembangan Perangkat Lunak*

Adapun aktivitas-aktivitas yang dilalui sebagai berikut:

1. Analisis Kebutuhan

Pada tahap awal dilakukan analisis kebutuhan, proses ini dilakukan untuk mengetahui informasi, model, dan spesifikasi dari sistem yang dibutuhkan.

2. Pembuatan *Prototype*

   Pada tahap ini, akan dilakukan pembuatan *prototype* sesuai dengan kebutuhan.

3. Evaluasi *Prototype*

   Tahap dimana *prototype* dievaluasi apakah sudah cocok atau belum dengan kebutuhan.

4. Pengembangan perangkat lunak akhir

   Melakukan pembuatan perangkat lunak yang telah cocok sesuai dengan kebutuhan sekaligus melakukan penyelesaian pengembangan perangkat lunak.

5. Pengujian (*Testing*)

   Tahapan selanjutnya adalah proses pengujian perangkat lunak, proses pengujian ini dilakukan untuk memastikan perangkat lunak yang telah dibuat telah sesuai dengan kebutuhan. Pengujian menggunakan metode *blackbox*.

   Pengujian terhadap aplikasi yang dibangun dengan mengukur tingkat akurasi terhadap pengenalan indentitas dari pembicara (*speaker*).

3.4 Implementasi

Implementasi yang dilakukan adalah dengan mengimplementasikan *Hidden Markov Models*, tepatnya algoritma *forward-backward* untuk pelatihan dan algoritma *viterbi* untuk pengujian dalam pengenalan suara pembicara (*Speaker Recognition*).