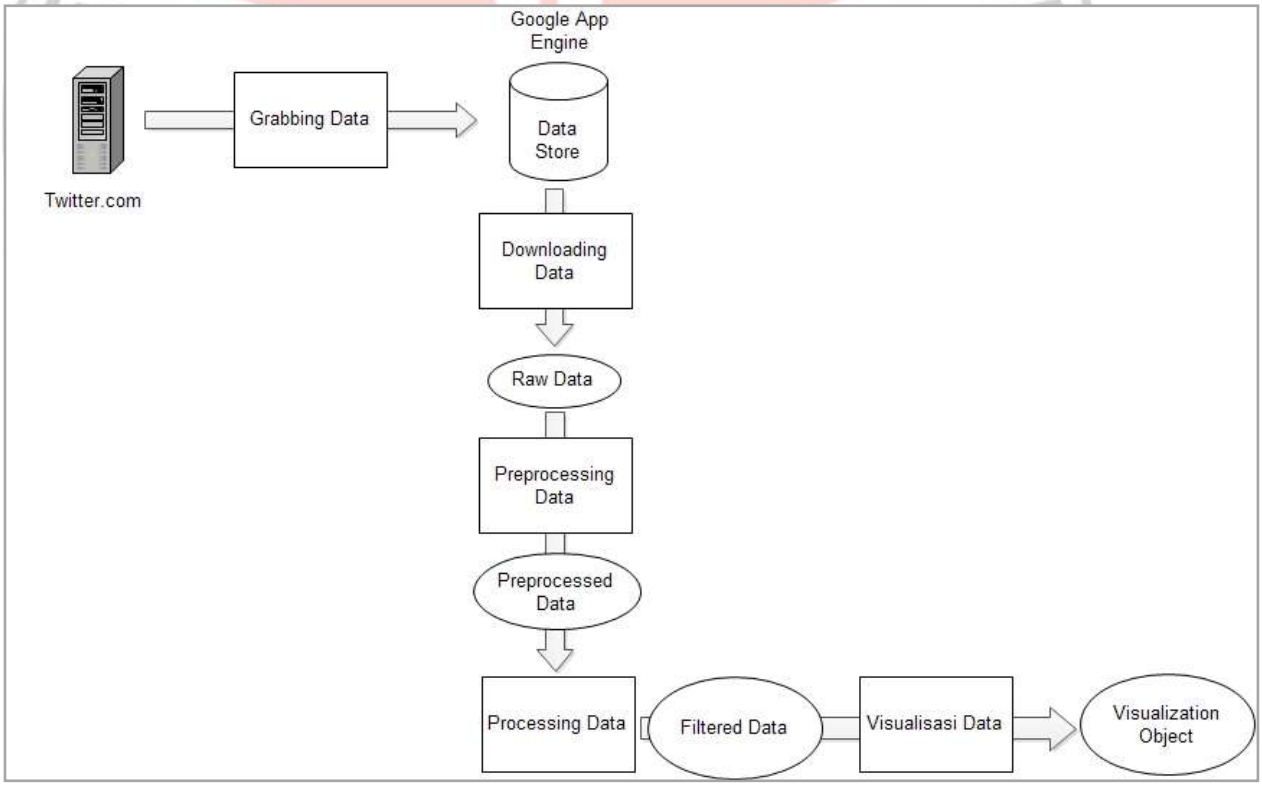
BAB III

METODOLOGI PENELITIAN

3.1. Desain Penelitian

Desain penelitian merupakan langkah-langkah yang akan dilakukan dalam penelitian ini. Penelitian ini dimulai dari pengumpulan data *tweet* yang mengandung "lalinbdg". *Tweet* yang mengandung kata lalinbdg adalah *tweet* yang berisi informasi tentang lalulintas kota Bandung khususnya kemacetan. Penelitian akan dilakukan seperti desain penelitian pada Gambar 3.1 di bawah ini:



Gambar 3. 1 *Desain Penelitian*

Penjelasan dari Gambar 3.1 adalah:

1. Pengambilan dan penyimpanan data, merupakan proses dasar dalam penelitian ini. Data yang diambil harus mengandung “lalinbdg”. Data diambil setiap 30 menit sekali dan disimpan di dalam *Datastore* yang disediakan oleh *Google App Engine*.
2. Data yang telah terkumpul di *Datastore* merupakan data kotor yang akan diolah kembali. Disebut data kotor karena masih mengandung duplikasi data, dan data yang tidak valid. Data tersebut kemudian diunduh menggunakan *appcfg.py*.
3. Preprocessing data adalah tahap untuk membersihkan data dari duplikasi, *stopword*, dan syarat lain yang harus dipenuhi agar data kotor berubah menjadi data yang berkualitas. Misalnya data yang dibutuhkan hanya data yang mengandung kata macet, nama jalan, serta tidak boleh sama. Pada tahap ini juga di *filter keyword* jalan dan *keyword* “ macet”. Nama jalan yang ada di dalam twit diganti dengan nama jalan yang benar. Misalnya “chmpelas” diganti dengan “cihampelas”. Kata “macet” yang tidak valid seperti “maceettttt”, “macccetttossss”, “gagerak”, “mct”, dan lainnya di ubah menjadi “macet”.
4. *Processing* data, merupakan langkah akhir sebelum memasuki visualisasi data. Data yang sudah bersih kemudian diolah kembali. Format data yang telah diolah adalah waktu, isi *tweet*, nama jalan. Jadi pada data terakhir kita

bisa melihat jalan yang sedang macet. Dan format data merupakan xml yang bisa dibaca oleh *actionscript* dan ditampilkan berbentuk animasi.

5. Pembuatan aplikasi untuk membaca informasi dari data dan memvisualisasikannya melalui animasi. Langkah-langkah yang akan dilalui pada tahap ini adalah:
    a. Analisis kebutuhan perangkat lunak
    b. Desain perangkat lunak
    c. *Coding* perangkat lunak
    d. Pengujian perangkat lunak
6. Dokumentasi perangkat lunak merupakan pembuatan dokumen skripsi, dokumen teknis perangkat lunak dan *paper*.

3.2. Fokus Penelitian

Fokus penelitian pada skripsi ini adalah:

1. Tahapan-tahapan yang perlu dilakukan dalam proses ekstraksi informasi sehingga menghasilkan informasi yang tersaring.
2. Sistem visualisasi informasi agar memudahkan pengguna menganalisis dan menyimpulkan informasi yang diberikan.

3.3. Metode Penelitian

3.3.1. Metode Pengumpulan data

3.3.1.1. Explorasi dan Studi literatur

Eksplorasi dan studi literatur dilakukan dengan mempelajari konsep-konsep yang berkaitan dengan tema pada skripsi ini, seperti ekstraksi informasi dengan menggunakan *regular expression* dan teknik visualisasi informasi melalui literatur-literatur seperti buku, *paper*, jurnal ilmiah, dan sumber ilmiah lain seperti situs internet ataupun artikel yang berhubungan.

3.3.1.2. Pengumpulan Data Tweet

Pengumpulan data *tweet* dilakukan dari bulan September 2010 sampai sekarang. Namun data yang digunakan hanya sampai bulan mei 2011. Pengumpulan data menggunakan aplikasi *grabingtweet* yang diatur agar melakukan pengambilan data *tweet* yang mengandung "lalinbdg" setiap 30 menit sekali.

3.3.2. Metode Pengembangan Perangkat Lunak

3.3.2.1. Pendekatan Pengembangan Perangkat Lunak

Dalam melakukan pengembangan perangkat lunak ini, penulis menggunakan pendekatan terstruktur yang merupakan suatu pendekatan berorientasi objek terfokus pada pengolahan data dan proses dari suatu perangkat lunak.

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

2. *Attribute*

   Adalah variabel 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*

   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 proses analisis dan desain berorientasi objek. UML menyediakan standar notasi dan diagram-diagram yang bisa digunakan untuk memodelkan sistem.

Diagram-diagram pada UML terbagi dalam 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

Perangkat Lunak yang akan dikembangkan dalam penelitian ini merupakan perangkat lunak berbasis *web*, sehingga untuk model proses yang akan digunakan adalah *Web Engineering Process* (Roger S. Pressman, 2004).

Web Engineering (WebE) merupakan suatu proses yang digunakan untuk menciptakan aplikasi berbasis web yang berkualitas tinggi. Konsep dasar dan prinsip dari WebE secara umum tidak jauh berbeda dengan *Software Engineering*.

Menurut Roger S. Pressman (2010), ada beberapa karakteristik yang membedakan perangkat lunak berbasis *web (WebApp)* dengan perangkat lunak lainnya, yaitu:

1. *NetworkIntensiveness*

   Sebuah *WebApp* berada pada suatu jaringan dan harus melayani kebutuhan dari klien yang berbeda-beda. *WebApp* ini bisa berada di internet sehingga setiap orang bisa mengaksesnya, atau intranet (hanya satu atau beberapa organisasi yang bisa mengaksesnya).

2. *Concurrency*

   Sejumlah besar pengguna dapat mengakses dan menggunakan *WebApp* pada satu waktu secara bersamaan.

3. *Unpredictable load*

   Jumlah pengguna suatu *WebApp* bisa berbeda-beda setiap harinya, dan itu tidak bisa diprediksi. 100 pengguna datang pada hari senin, tapi mungkin saja waktu hari selasa bisa 100.000 pengguna yang datang.

4. *Performance*

   Jika pengguna *WebApp* harus menunggu terlalu lama (untuk mengakses, pemrosesan di sisi server atau sisi klien) maka pengguna mungkin akan memutuskan untuk meninggalkan *WebApp* tersebut. Maka *WebApp* dituntut harus mempunyai performa yang baik.

5. *Availability*

   Walaupun harapan untuk 100 persen *availability* (ketersediaan) tidak rasional, tetapi pengguna dari *WebApp* populer sering menuntut ketersediaan selama 24/7/365.

6. *Data driven*

   Fungsi utama dari sebagian besar *WebApp* adalah dengan menggunakan *hypermedia* untuk menyajikan teks, grafis, audio, dan video ke pengguna. Selain itu, *WebApp* juga digunakan untuk mengakses informasi yang berada pada *database*.

7. *Content sensitive*

   Kualitas dan estetika dari konten tetap menjadi penentu penting dari kualitas sebuah *WebApp*.

8. *Continous Evolution*

   Tidak seperti aplikasi perangkat lunak konvensional yang berkembang melalui serangkaian perencanaan, *WebApp* berkembang secara terus menerus

9. *Immediacy*

   Meskipun *immediacy* (kebutuhan mendesak untuk perangkat lunak segera dirilis dan disebarkan ke publik) adalah karakteristik dari banyak domain perangkat lunak, tatapi *WebApp* sering menunjukan bahwa waktu untuk segera rilis bisa dalam hitungan hari atau minggu.

10. *Security*

    Karena *WebApp* tersedia dalam suatu jaringan, diperlukan penerapan keamanan yang lebih demi berjalan baiknya suatu *WebApp*.

11. *Aesthetics*

    Salah satu komponen penting dan menjadi daya tarik suatu *WebApp* yaitu desain tampilan dan tata letak konten yang estetis.

Berdasarkan karakteristik khas yang dimiliki oleh suatu perangkat lunak berbasis *web*, maka diperlukan teknik dan metode yang tepat dalam proses pengembangannya.

Menurut Pressman (2004), model proses *Agile* (seperti *Extreme Programming, SCRUM, Adaptive Software Development*) bisa berhasil diaplikasikan sebagai *WebE* dengan beberapa adaptasi, karena pada intinya *Agile*

merupakan metodologi pengembangan perangkat lunak yang berdasar pada pengembangan secara iteratif dan inkremental dengan tempo yang pendek. Hal tersebut juga merupakan hal yang akan ditemui dalam pengembangan suatu *WebApp*, yaitu:

1. Perangkat lunak akan sering dirilis secara bertahap.
2. Perubahan akan sering terjadi secara berkala.
3. Mempunyai timeline yang pendek untuk setiap tahap dalam proses.

Secara umum, tahap-tahap proses pengembangan perangkat lunak berbasis web dengan model proses *WebE* terdiri dari 5 tahap (Roger S. Pressman, 2004), yaitu:

1. *Formulation/Bussiness Analysis*

   Formulation merupakan tahap mencari seluruh kebutuhan dari *WebApp* dan dengan melibatkan seluruh *stakeholders*, bertujuan untuk menjelaskan permasalahan yang harus diselesaikan oleh *WebApp* dengan informasi kebutuhan yang tersedia.

   Business Analysis mendefinisikan konteks dari business untuk *WebApp*, yaitu pengidentifikasian *stakeholder*, memprediksi kebutuhan *WebApp*, *database*, pendefinisan fungsi.

2. *Planning*

   Tahap perencanaan untuk *WebApp* dimana perencanaan berisi definisi dari tiap pekerjaan, lalu jadwal dari jangka waktu yang telah diproyeksikan untuk pengembangan *WebApp* secara inkremental.

3. *Modeling*

   Desain dan analisis rekayasa perangkat lunak secara konvensional diadaptasi pada proses pengembangan *WebApp*, digabungkan dan lalu menyatu pada tahap *modeling* ini. Tahap ini bermaksud untuk menghasilkan suatu analisis dan desain model yang telah didefinisikan pada *requirement*.

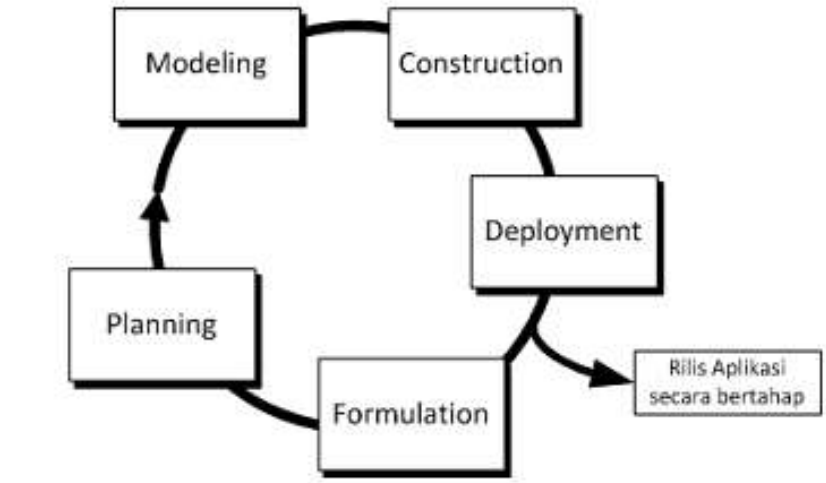
4. *Construction*

   Tahap implementasi, dimana *tools* dan teknologi diterapkan untuk membangun *WebApp* yang telah sebelumnya dimodelkan. Setelah itu *WebApp* yang telah dibangun secara bertahap ditest untuk menemukan kesalahan yang terjadai pada tahap desain (isi, arsitektur, antarmuka).

5. *Deployment*

   WebApp dikonfigurasikan, dirilis terhadap pengguna, dan tahap evaluasi dilakukan. Pada tahap ini, *feedback* dari pengguna dijadikan acuan untuk pengembangan tahap berikutnya.

Kelima alur proses *WebE* tadi diaplikasikan secara berulang dan bertahap seperti pada ilustrasi dibawah ini.



Gambar 3. 2 *Model Proses WebE (Roger S. Pressman, 2004)*

3.4. Alat dan Bahan Peneltian

3.4.1. Alat Penelitian

Alat-alat yang digunakan dalam penelitian ini terbagi kedalam dua kelompok, yaitu perangkat lunak dan perangkat keras. Untuk perangkat keras yang digunakan dalam pengembangan sistem ekstraksi informasi ini, penulis menggunakan perangkat keras berupa laptop dengan spesifikasi sebagai berikut:

1. Processor AMD Athlon™64 Processor 3500+ 2.20GHz
2. Memori 2.00 GB
3. Harddisk 1 TB
4. VGA 128MB
5. LED 20”

6. Mouse
7. Keyboard
8. Perangkat keras penyimpan data berupa flashdisk, CD.

Sedangkan kebutuhan perangkat lunak yang digunakan untuk pengembangan aplikasi ini, penulis menggunakan perangkat lunak dengan spesifikasi sebagai berikut:

1. Sistem Operasi Microsoft Windows 7 Profesional
2. JDK 1.6 Update 18
3. Netbeans 6.8 dan Eclipse Hellios yang telah terintregasi dengan appengine
4. Appengine-java-sdk-1.3.5
5. Phyton 2.7

3.4.2. Bahan Penelitian

Bahan penelitian yang digunakan dalam penelitian ini adalah *tweet* yang mengandung *keyword* "lalinbdg" yang telah dikumpulkan dari http://twitter.com.

3.5. Tahap Penelitian

Tahapan-tahapan yang dilakukan dalam penelitian ini adalah sebagai berikut:

3.5.1. Langkah Awal Penelitian

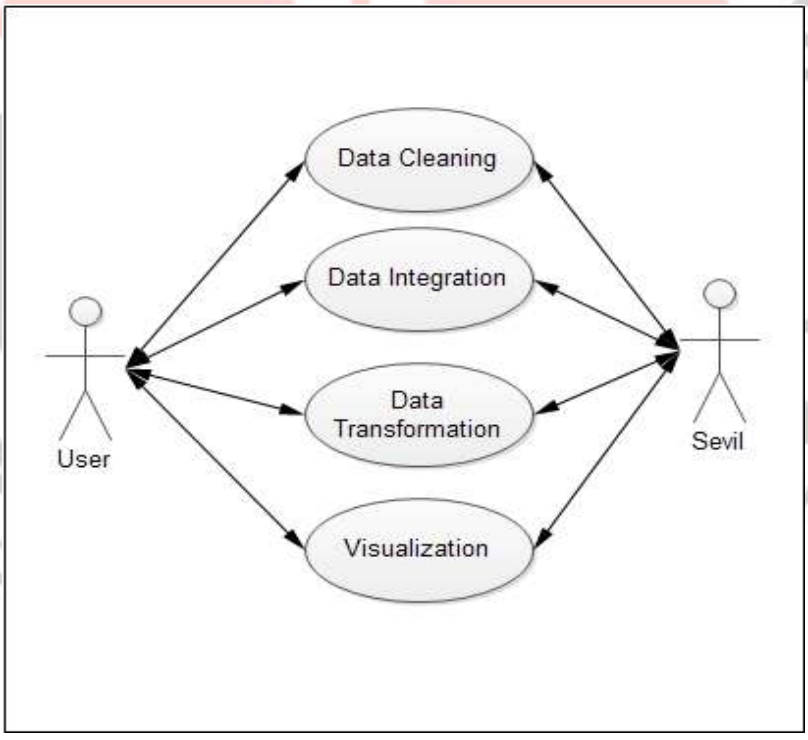
Perumusan masalah dimana perumusan masalah penelitian ini adalah system visualisasi kemacetan lalu lintas bandung. Untuk memahami permasalahan, dilakukan studi literatur mengenai metode ekstraksi dan visualisasi informasi serta

penerapan metode tersebut. Selanjutnya pengumpulan data lalu lintas kota Bandung yang akan digunakan dalam sistem.

3.5.2. Sketsa Sistem

Tahap ini adalah tahap menggambarkan struktur sistem. Tahap ini merupakan tahap desain sistem. Pada tahap ini dikembangkan *usecase diagram, class-diagram, activity diagram* dan *sequence diagram.*

Berikut penulis sertakan *use-case diagram* sistem ekstraksi dan visualisasi lalulintas kota Bandung.



Gambar 3. 3 *Use Case Diagram Sistem Ekstraksi dan Visualisasi Lalulintas Kota Bandung*

3.5.3. *Data Cleaning*

Tahap ini adalah tahap membuang duplikasi data, memeriksa data yang tidak konsisten, dan memperbaiki kesalahan pada data (seperti kesalahan dalam penulisan). Tahap ini memiliki *subtask* seperti *tokenizing*, *preprocessing, stopword removal* dan *synonym replacing*.

3.5.4. *Data Integration*

Pada tahap ini ditambahkan informasi ke dalam data yang sudah dibersihkan. Seperti menambahkan nama jalan dan keadaan kedalam setiap *tuple* di dalam data..

3.5.5. *Data Transformation*

Pada tahap ini bentuk data akan diubah ke bentuk tertentu sehingga sesuai untuk proses visualisasi.

3.5.6. Pengembangan Perangkat Lunak Visualisasi Kemacetan Kota Bandung.

Setelah memperoleh data yang telah tertransformasi, selanjutnya dilakukan proses pengembangan perangkat lunak visualisasi kemacetan kota Bandung untuk menampilkan informasi yang telah diolah dari data yang diperoleh.