BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Negara Indonesia adalah negara yang memiliki banyak jenis transportasi darat dari transportasi yang menggunakan mesin (bus, angkutan umum, dan kereta api) sampai dengan yang hanya menggunakan tenaga manusia (becak, ojeg, dan sepeda). Kereta api adalah jenis transportasi darat yang paling banyak membawa penumpang. Hal ini dikarenakan ukuran kereta api yang besar dan panjang sehingga dapat memuat ratusan penumpang dalam sekali jalan. Kereta api juga memiliki keunggulan lain yaitu harga tiket yang murah dan juga jangkauannya yang luas sampai ke pelosok daerah. Dengan keunggulan itu kereta api menjadi salah satu primadona transportasi bagi masyarakat Indonesia. Namun, masih ada beberapa kendala yang masih terjadi di dalam penggunaan kereta api, yaitu faktor keterlambatan dan kecelakaan

Keterlambatan kereta api di Indonesia sudah menjadi masalah yang sering terjadi. Berdasarkan informasi dari website PT. Kereta Api diperoleh bahwa data keterlambatan dari kereta penumpang di Indonesia cukup tinggi, yaitu 36,33 menit untuk rata-rata terlambat berangkat dan 59,33 menit untuk rata-rata terlambat datang. Data ini diambil pada tahun 2001-2011[24].

Kecelakaan kereta api itu sendiri dapat didefinisikan berdasarkan jenis peristiwanya, seperti tabrakan kereta api dengan kereta api, tabrakan diantara kereta api kendaraan lain, kereta api anjlok, dan jenis lainnya. Dari klasifikasi di atas, yang menjadi fokus penelitian ini adalah kecelakaan kereta api yang berhubungan dengan tabrakan diantara kereta api dengan kereta api lainya. Data kecelakaan kereta api, dapat dilihat pada gambar 1.1 berikut ini :

KOMITE NASIONAL KESELAMATAN TRANSPORTASI

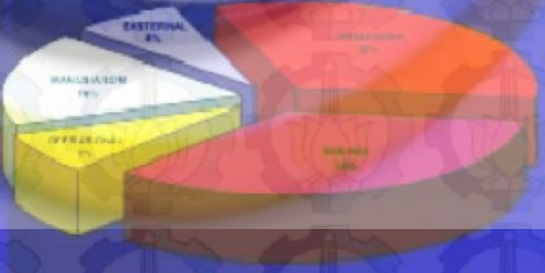
DATA KECELAKAAN KERETA API TAHUN 2007-2010

TAHUN	JUMLAH KECELAKAAN	JENIS KECELAKAAN				KORBAN JIWA	
		TUBRUKAN ANTAR KA	ANJLOKAN/ TERGULING	TUBRUKAN KA DGN ANGKUTAN LAIN	LAIN - LAIN	MENINGGAL	LUKA - LUKA
2007	164	9	117	20	18	94	292
2008	130	5	107	21	6	45	151
2009	82	5	48	21	8	57	186
2010 (21 DESEMBER)	50	3	29	8	10	68	169
TOTAL	427	14	301	70	42	199	804

Sumber : Direktorat Jenderal Perkeretaapian

KOMITE NASIONAL KESELAMATAN TRANSPORTASI

PROSENTASE FAKTOR PENYEBAB KECELAKAAN KERETA API
YANG DIINVESTIGASI KNKT
TAHUN 2007 - 2010



Sumber : Database KNKT

operasi, solusi *deadlock*, dan algoritma *dijkstra.* Sedangkan visualisasi digunakan untuk memperlihatkan pergerakan kereta dalam memvisualisasikan penghindaran *deadlock* dan tabrakan.

Semaphore ini dapat dimanfaatkan dalam menyelesaikan permasalahan tabrakan kereta api karena *semaphore* dapat mencegah dua atau lebih kereta agar tidak menggunakan *track*/rel (*resource*) yang sama pada waktu yang bersamaan. *Semaphore* ini merupakan salah satu teknik yang terdapat dalam *Interprocess Communication* (IPC) yang penerapannya terdapat dalam sistem operasi [23].

Deadlock dapat terjadi pada perjalanan kereta api ketika dua atau lebih kereta api tidak bisa berjalan karena saling menunggu kereta lain untuk berjalan. Jika hal ini terjadi maka proses perjalanan kereta api tidak dapat selesai. Masalah *deadlock* pada perjalanan keretini dapat diselesaikan dengan menggunakan *deadlock solution* yang terdiri dari *deadlock prevention, deadlock avoidance,* atau *deadlock detection* (pendeteksian *deadlock*) [23].

Visualisasi pada perangkat lunak ini digambarkan dengan menggunakan *graph*. *Graph* adalah kumpulan *vertex* (titik) yang dihubungkan dengan segmen[5]. Algoritma *dijkstra* adalah algoritma yang diterapkan pada perangkat lunak visualisasi perjalanan kereta ini untuk mencari rute terpendek. Algoritma dijikstra ini merupakan algoritma yang dapat diterapkan pada *graph.* Algoritma *dijkstra* ini memiliki kelebihan yaitu mudah diterapkan dan hasilnya sudah teruji[6].

Perangkat lunak ini diimplementasikan dengan menggunakan bahasa pemograman JavaFX dan database ORACLE 11G. JavaFX dipilih sebagai bahasa pemograman pada perangkat lunak ini karena javaFX memiliki fitur untuk merepresentasikan *graph* dan memiliki fungsi animasi yang dapat membantu menggambarkan pergerakan kereta. Sedangkan database ORACLE 11G dipilih karena database ini memiliki skala data yang besar dan akses yang cepat.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, rumusan masalah pada sistem ini adalah :

1. Bagaimana merancang perangkat lunak visualisasi perjalanan kereta api untuk memperlihatkan dua atau lebih kereta agar tidak terjadi tabrakan dan *deadlock* dengan cara/pendekatan *deadlock solution* (*deadlock detection*, *deadlock avoidance* atau *deadlock prevention*) dan *semaphore* pada sistem operasi ?.
2. Bagaimana mengimplementasikan algoritma *dijkstra* untuk mencari rute terpendek dalam perjalanan kereta api ?.

1.3 Tujuan

Tujuan dari pembuatan aplikasi ini adalah sebagai berikut :

1. Merancang dan mengimplementasikan sebuah perangkat lunak yang terkait dengan proses perjalanan kereta api dengan cara/pendekatan *semaphore* (pada sistem operasi) dan *deadlock solution*.
2. Mengimplementasikan algoritma *dijkstra* untuk mencari rute terpendek pada sistem perjalanan kereta api.
3. Merancang dan membuat visualisasi proses perjalanan kereta api untuk memperlihatkan penghindaran tabrakan dan *deadlock* oleh dua atau lebih kereta api.

1.4 Ruang Lingkup

Ruang lingkup dalam pembuatan visualisasi perjalanan kereta api ini adalah sebagai berikut:

a. Menentukan lokasi awal keberangkatan dan tujuan dari kereta api.

   Pada lingkup ini akan dibuat sub sistem untuk membuat lokasi keberangkatan dan tujuan dari kereta api yang akan beroperasi. Lokasi dan tujuan kereta adalah berupa stasiun-stasiun yang ditentukan oleh petugas kereta api.

b. Penentuan rute kereta api.

Setelah lokasi/stasiun keberangkatan dan stasiun tujuan ditentukan oleh petugas kereta api maka selanjutnya adalah penentuan rute perjalanan kereta api oleh sistem. Sistem akan membuat rute/jalur terpendek yang akan dilalui kereta api dengan algoritma dijkstra. Agar algoritma dijkstra bisa bekerja dengan baik maka *track*/rel dan stasiun yang akan menjadi dari bagian dari perjalanan kereta api akan direpresentasikan dalam bentuk *graph*.

c. Proses perjalanan kereta api.

Setelah rute perjalanan ditentukan, maka selanjutnya adalah proses perjalanan kereta api untuk sampai ke stasiun tujuan. Pada perjalanannya disini sistem akan berusaha untuk mengindari segala resiko yang akan terjadi seperti terjadinya tabrakan, *deadlock*, dan keterlambatan.

d. Visualisasi perjalanan kereta.

Visualisasi perjalanan kereta ini berupa tampilan pergerakan kereta api seperti maju, mundur, belok, dan berhenti pada posisi tertentu.

Selain itu proses pengembangan visualisasi perjalanan kereta api ini memiliki batasan-batasan sebagai berikut:

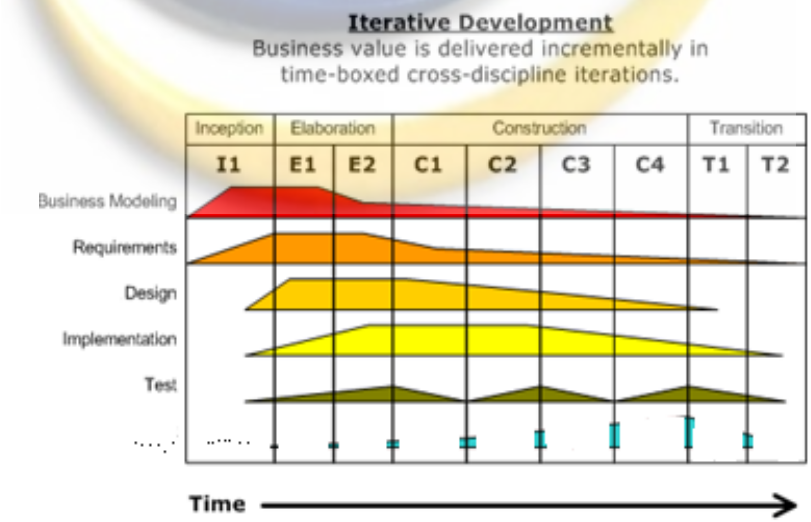
1. Pengembangan sistem ini hanya sebatas visualisasi saja, tidak melibatkan perangkat keras seperti kereta apinya, sensor, rel, dan semacamnya.
2. Visualisasi hanya berupa tampilan pergerakan kereta.
3. Pengembangan sistem hanya melingkupi perjalanan kereta api.
4. Tidak adanya pola/metode tertentu dalam menentukan prioritas kereta mana yang duluan berjalan karena tergantung pada pemanggilan fungsi *pesanVertex()* pada *thread* kereta.
5. Seluruh rute terpendek dari tiap posisi akan disimpan dalam database, sehingga penggunaan algoritma dijkstra hanya dilakukan sekali dan pada awal pemasangan sistem saja.
6. Kereta api yang akan divisualisasikan berjalan dengan kecepatan konstan, sehingga hal-hal seperti tanjakan, turunan, belokan, dan berat barang/punumpang tidak mempengaruhi kecepatan kereta.
7. Tidak ada percepatan maupun perlambatan pada pergerakan kereta

8. Atribut peta pada visualisasi yang terlibat adalah *track*, stasiun, dan kereta api.

1.2 Model Pengembangan Perangkat Lunak

Model pengembangan perangkat lunak yang akan digunakan dalam pengembangan aplikasi ini adalah *Iterative and Incremental Development.* Model *pengembangan Iterative and Incremental Development* dikemukakan dalam buku "Applying UML and Pattern" Third Edition oleh Craig Larman. Pada model *Iterative and Incremental Development* dilakukan pengorganisasian beberapa tahap pengerjaan yang terkait pada waktu. Model ini berbeda dengan model *waterfall*. dimana model *waterfall* mengerjakan tiap *task* satu per satu sedangkan model ini mengerjakan setiap *task* hampir bersamaan. Sebagai contoh pada model *waterfall* pembuatan desain program diselesaikan semuanya terlebih dahulu baru kemudian implementasi, namun pada model *iterative* desain dan implementasi bisa dikerjakan secara bersamaan.

Tahapan pada model ini dimulai dengan perencanaan awal (*planning*) dan diakhiri dengan *deployment* yang didalamnya terdapat perputaran interaksi antara *planning*, analisis, *design*, implementasi, dan *testing* [13].



Gambar 1.3 Fase dan *Discipline* pada *iterative development* [13]

Gambar diatas memperlihatkan model *iterative*, yang terdiri atas beberapa fase yaitu:

1. *Insepsi*

   Pada fase ini dilakukan analisis dari keseluruhan sistem yang akan dikembangkan. Kegiatan yang dilakukan pada fase ini, yaitu ;

   a. Mendeskripsikan *requirements* dari pengembangan;
   b. Mengidentifikasi aturan-aturan sistem;
   c. Mendeskripsikan lingkup sistem dari pengembangan aplikasi;
   d. Mendefinisikan langkah–langkah yang harus dilakukan pada tahap *elaborasi* serta mendefinisikan tools yang akan digunakan dalam pengembangan aplikasi.

2. *Elaborasi*

   Pada fase *elaborasi* dapat dilaksanakan dalam dua atau lebih iterasi. Kegiatan yang akan dilakukan pada pada fase ini, yaitu:

   a. Menyempurnakan tujuan, *requirements*, dan batasan dari pengembangan aplikasi;
   b. Mendefinisikan arsitektur utama dari aplikasi yang dikembangkan;
   c. Mendefinisikan fitur-fitur dari aplikasi yang dikembangkan;
   d. Menentukan *interface* aplikasi;
   e. Memodelkan perilaku aplikasi;
   f. Mengimplementasikan arsitektur utama dari aplikasi secara *iterative*.

3. Konstruksi

   Pada fase konstruksi dapat dilaksanakan dalam dua atau lebih iterasi. Kegiatan yang akan dilakukan pada fase ini, yaitu:

   a. Memperbaiki arsitektur dari aplikasi;
   b. Penyempurnaan aplikasi berdasarkan arsitektur;
   c. Pengintegrasian seluruh bagian aplikasi serta memperbaiki *interface;*
   d. Melakukan pengujian aplikasi.

4. Transisi

   Pada fase ini, proses pengembangan sistem difokuskan pada penyelesaian akhir produk dengan kemampuan fungsional dan non-fungsional yang terpenuhi dan telah siap digunakan.

Selain itu pada gambar diatas terdapat *disciplines* yaitu kumpulan aktivitas dalam subjek area, yang terdiri dari :

1. *Business Modelling :* merupakan pemodelan bisnis, digambarkan dengan artifak domain model (model relasi antar objek). Artifak domain model ini akan lebih dijelaskan pada bab 3 analisis sistem.
2. *Requirement :* merupakan hal-hal yang diperlukan dalam membangun sistem. Requirement akan digambarkan dengan artifak *use case* model. Use case model akan dijelaskan pada bab 3 analisis sistem.
3. *Design*: merupakan rancangan yang dibuat untuk membangun sistem. Artifak yang digunakan desain ini adalah data model (diagram ER), *Sequence* diagram, dan *class* diagram. Artifak data model, *sequence* diagram, dan *class* diagram akan dijelaskan pada bab 4 perancangan sistem.
4. *Test :* merupakan kegiatan pengetesan terhadap perangkat lunak yang dibuat. Artifak yang dihasilkan adalah hasil pengujian. Artifak ini akan dituliskan pada bab 5 implementasi sistem.

1.3 Sistematika Penulisan Laporan

Laporan disusun secara sistematis sehingga mudah dibaca, ditelusuri, dievaluasi. Sistematika penulisan laporan tugas akhir ini terbagi menjadi 6 bab sebagai berikut:

Bab satu pendahuluan, membahas mengenai latar belakang masalah, identifikasi masalah, maksud dan tujuan, ruang lingkup, metodologi pengembangan, dan sistematika penulisan laporan.

Bab dua landasan teori, berisi mengenai teori-teori yang menjadi landasan dalam penulisan laporan Tugas Akhir.

Bab tiga analisis sistem, membahas analisis mengenai sistem yang sedang berjalan untuk dijadikan landasan pada tahap perancangan sistem dan menganalisis kebutuhan sistem yang akan dibangun.

Bab empat perancangan sistem, membahas mengenai rancangan sistem yang akan dibuat berdasarkan hasil analisis dan evaluasi sistem.

Bab lima implementasi dan pengujian sistem, merupakan realisasi dari tahap perancangan sistem yang berupa pengimplementasian ke dalam *source code* beserta pengujian terhadap hasil implementasi.

Bab enam kesimpulan dan saran, merupakan kesimpulan yang diperoleh dari pelaksanaan pengerjaan Tugas Akhir dan saran yang diperlukan berkaitan dengan pengembangan sistem.