TEKNOLOGI *LOCATION BASED SERVICE* (*GLOBAL POSITIONING SYSTEM*) PADA PERANGKAT *MOBILE*

Budi Yulianto

Jurusan Teknik Informatika, Fakultas Ilmu Komputer, Universitas Bina Nusantara
Jln. K.H. Syahdan No. 9, Palmerah, Jakarta Barat 11480
laboratory@binus.ac.id

ABSTRACT

Article presents analysis and design of software using Location Based Service (LBS) that is part of communication technology based on geographic position. The goal of the research is designing LBS application to be implemented on mobile device that has GPS (Global Positioning System) technology and uses GPRS (General Packet Radio Service) to connect to server for generating shortest path by Dijkstra algorithm method Fibonacci Heap. Software development method used is LBS application implemented on mobile device. Conclusion of the research has shown that shortest path generated using Dijkstra algorithm method Fibonacci Heap as single source shortest path is faster than Dijkstra algorithm and Bellman Ford.

Keywords: *location based service, mobile device, global positioning system*

ABSTRAK

Artikel menjelaskan analisis dan perancangan perangkat lunak menggunakan teknologi Location Based Service (LBS) yang merupakan bagian dari teknologi komunikasi berbasiskan pada posisi lokasi geografi. Tujuan penelitian adalah merancang aplikasi LBS untuk diimplementasikan pada perangkat mobile berteknologi GPS (Global Positioning System) serta menggunakan GPRS (General Packet Radio Service) sebagai penghubung dengan server untuk menghasilkan rute terpendek dengan menggunakan algoritma Dijkstra metode Fibonacci Heap. Metode pengembangan piranti lunak yang digunakan adalah Rational Unified Process. Hasil penelitian merupakan perancangan aplikasi LBS yang dapat diimplementasikan pada perangkat mobile. Simpulan dari penelitian menunjukkan pencarian rute terpendek dengan algoritma Dijkstra metode Fibonacci Heap sebagai algoritma single source shortest path yang lebih cepat daripada algoritma Dijkstra biasa dan Bellman Ford.

Kata kunci: *location based service, perangkat mobile, global positioning system*

PENDAHULUAN

Perkembangan teknologi sekarang ini telah membawa manusia pada tahap di mana kebutuhan hidup telah berubah ke arah digital sehingga lebih mudah, menarik, dan beragam. Salah satu kebutuhan manusia yang dari hari ke hari menjadi semakin vital adalah kebutuhan komunikasi sebagai suatu aliran informasi. Layanan yang memanfaatkan kemajuan teknologi komunikasi atau *Location Based Service* (LBS) berguna untuk menemukan posisi geografi atau posisi seseorang dari peralatan bergerak (perangkat *mobile*) pada suatu waktu. Informasi berupa posisi ini dapat berkembang menjadi layanan pencarian rute terpendek dari suatu posisi ke posisi lain menggunakan suatu algoritma *shortest path*.

Perpaduan antara posisi dan rute terpendek ini akan menjadi suatu aplikasi yang berguna dalam memenuhi kebutuhan akan informasi lokasi (tempat) bagi masyarakat, khususnya bagi mereka yang banyak melakukan pekerjaan atau aktivitas dan menghabiskan waktu di luar bangunan. Perangkat pencarian tersebut menyediakan data dalam bentuk visual (grafik) yang dapat diakses secara cepat dan murah. Selain tempat wisata, informasi lokasi tersebut dapat pula berupa hotel, restoran, dan tempat berbelanja.

Perangkat pencarian tersebut memerlukan *Global Positioning System* (GPS) yang pada dasarnya dikembangkan untuk memberikan informasi posisi geografi. GPS telah digunakan secara luas pada berbagai wilayah dan aktivitas oleh pengguna seperti pengendara kendaraan, penjelajah alam, pelayar, dan pengguna lainnya yang membutuhkan informasi posisi geografi sebagai panduan. Aplikasi GPS pada umumnya terintegrasi pada kendaraan dan beberapa dirancang pada *Personal Digital Assistant* (PDA).

Tujuan penelitian adalah menganalisis, merancang dan menghasilkan sebuah aplikasi LBS yang dapat menampilkan rute terpendek dari suatu lokasi ke lokasi lainnya dengan menggunakan algoritma Dijkstra; Menyediakan peta lokasi wisata dan tempat umum yang berskala kecil ke pengguna. Manfaat penelitian adalah agar pengguna dapat mengetahui lokasi tempat dia berada; Membantu pengguna mencari lokasi umum yang dibutuhkan; Memberikan jalur terpendek dari lokasi pengguna yang terlacak dengan lokasi umum yang dituju.

METODE PENELITIAN

Metode penelitian yang digunakan meliputi studi pustaka dan studi lapangan. Pada metode studi pustaka, dilakukan pengumpulan bahan-bahan pustaka baik yang dilakukan di perpustakaan maupun pencarian melalui internet. Metode ini berguna dalam membantu memperdalam pembahasan materi, pembuatan program aplikasi, dan penyusunan laporan penelitian. Pada metode studi lapangan, dilakukan penambahan data dari pengguna perangkat mobile melalui kuesioner. Metode ini berguna untuk mengetahui kebutuhan pengguna, model tampilan yang disukai, dan kemudahan pengoperasian.

Metode Pengembangan

Metode pengembangan perangkat lunak yang digunakan adalah *Rational Unified Process* (RUP) yang meliputi empat tahap, yaitu Tahap Permulaan (*Inception Phase*) mencakup arsitektural proyek meliputi *Model*, *View*, *Controller* (MVC) *design pattern*, dan pencarian data meliputi studi pustaka dan studi lapangan; Tahap Perluasan (*Elaboration Phase*) mencakup penentuan perangkat keras, *database*, perangkat *mobile*, dan algoritma pencarian rute terpendek; Tahap Pembuatan (*Construction Phase*) mencakup perancangan dan dokumentasi piranti lunak; Tahap Peralihan (*Transition Phase*) mencakup implementasi dan pengujian piranti lunak.

Studi Pustaka

Sistem koordinat adalah suatu sistem yang menentukan sekumpulan bilangan ke masing-masing titik pada suatu ruang berdimensi *n*. Sistem koordinat ini merupakan aturan yang dapat membantu dalam pengukuran jarak antara titik-titik yang ada.

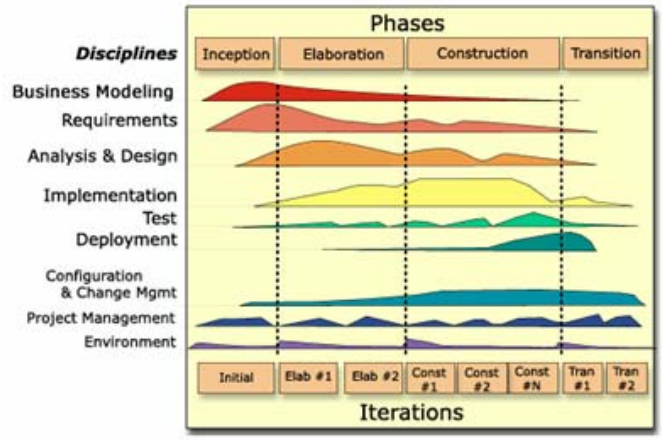
Shortest path adalah jalur terpendek yang dihasilkan melalui pencarian rute atau *path* terpendek antara node sumber dengan node tujuan yang ada pada *graph*. Biaya (*cost*) yang dihasilkan adalah yang paling minimum berdasarkan perhitungan pada *edge*-nya. Pada penelitian ini, nilai pada masing-masing *edge* merupakan panjang jalan yang harus ditempuh. Pencarian *shortest path* bukan berarti langsung menemukan jarak dari *node* awal ke *node* tujuan tetapi ada kalanya usaha itu harus dilakukan dengan melewati *node-node* tertentu sehingga tujuan akhir *node* dapat tercapai.

Single source shortest path adalah jarak terpendek (*shortest path*) dari setiap verteks tunggal pada *graph* berarah yang berbobot menuju verteks tujuan tertentu. Disebut *single source* karena membutuhkan dua titik sebagai awal pencarian. Yang termasuk dalam algoritma *single source shortest path* ini adalah Algoritma Dijkstra dan Bellman Ford.

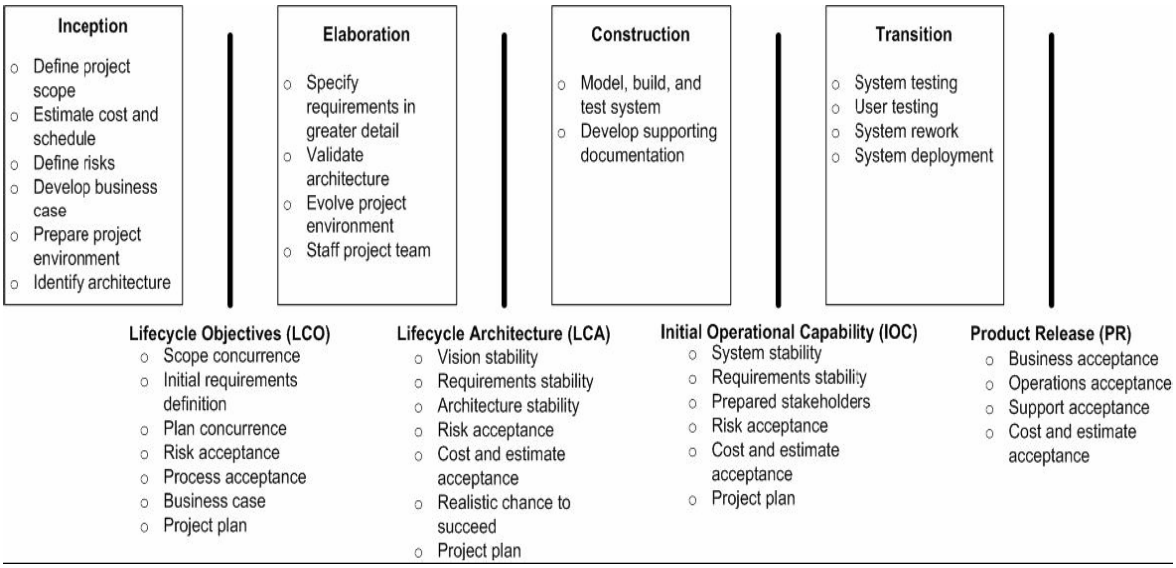
Algoritma Dijkstra dibuat oleh Edsgar Wybe Dijkstra untuk menemukan *path* bernilai terkecil (rute terpendek) dari verteks awal tunggal ke semua verteks pada *graph*. Algoritma ini akan melihat verteks yang terdeteksi dengan verteks awal, melihat *successor* dari verteks tersebut kemudian memperbarui jaraknya dari awal, demikian seluruhnya sampai verteks tujuan berhasil ditemukan. Hasilnya pasti berupa *path* terpendek.

Fibonacci-Heap berasal dari bilangan Fibonacci yang digunakan untuk analisis waktu proses. Fibonacci-Heap diterapkan dalam menghasilkan waktu proses yang terbaik pada beberapa algoritma *graph*, termasuk algoritma Dijkstra dan Prim. Fibonacci-Heap dapat digunakan untuk meningkatkan waktu proses pada algoritma Dijkstra dalam menghitung jarak terpendek pada suatu *graph* dan pada algoritma Prim untuk mendapatkan *minimum spanning tree* pada suatu *graph*.

Rational Unified Process adalah metode pengembangan piranti lunak hasil dari proses pengembangan model spiral. *Rational Unified Process* memiliki karakteristik seperti halnya produk piranti lunak, dirancang dan didokumentasikan menggunakan *Unified Modeling Language* (UML). Bentuk prosesnya memiliki dua struktur atau dimensi meliputi Dimensi Horizontal yang merepresentasikan waktu dan menampilkan aspek daur hidup dari proses; Dimensi Vertikal yang merepresentasikan proses inti (atau aliran kerja) untuk menggabungkan aktivitas rekayasa piranti lunak berdasarkan sifatnya.



Gambar 1 Dua Dimensi pada RUP
(http://en.wikipedia.org/wiki/Image:RationalUnifiedProcess.png)



Gambar 2 Tahapan pada RUP
(http://www.ambysoft.com/downloads/managersIntroToRUP.pdf)

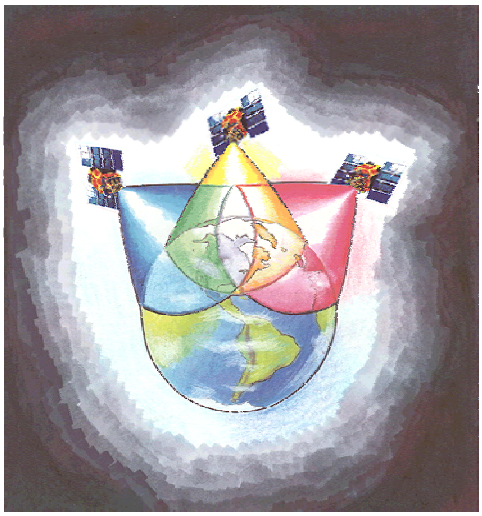
Location Based Service (LBS) adalah suatu layanan yang berbasiskan pada posisi lokasi geografinya yang disediakan kepada pelanggan. Dengan demikian teknologi LBS memungkinkan pengguna untuk mencari tempat-tempat yang diinginkan seperti hotel terdekat, ATM terdekat, SPBU terdekat, alamat tertentu dan sebagainya. Pada dasarnya pencarian lokasi dengan LBS sama dengan pencarian lokasi menggunakan peta biasa, hanya saja LBS dapat memberikan informasi tambahan mengenai posisi benda bergerak.

Magon dan Shukla, pada artikelnya yang berjudul "LBS, the Ingredients and the Alternatives", menyatakan komponen-komponen yang dibutuhkan untuk LBS meliputi lima hal, yaitu Lokasi atau Posisi yang mencakup *latitude* (posisi utara atau selatan), *longitude* (posisi timur atau barat), dan *altitude* (ketinggian dari permukaan laut); Data Geografi menyediakan data komponen lokasi seperti jaringan jalan, alamat pengguna, bangunan, dan daerah seperti pegunungan dan sungai, termasuk juga lokasi gas, restoran, hotel, dan sebagainya; Pusat Kendali yang bertindak sebagai pusat pengelolaan data dan distribusi layanan lokasi; Data Lokasi atau Data Peta dikelola dan dimanipulasi untuk menyediakan beberapa layanan yang berdasarkan pada lokasi pengguna; Sistem Komunikasi untuk menyampaikan lokasi ke pusat kendali dan menyediakan layanan yang diperlukan.

Global Positioning System (GPS) adalah suatu sistem navigasi satelit yang berfungsi untuk menentukan lokasi, kecepatan, dan arah, melalui signal yang diterima lebih dari 24-32 satelit yang berorbit 20000 km (11000 mil laut) di atas bumi. Prinsip dasar di balik GPS adalah pengukuran jarak (*distance*/*range*) antara satelit dan *receiver* dari transmisi signal radio. Sebuah GPS *receiver* memerlukan empat atau lebih satelit untuk menghasilkan jarak mereka, dan menggunakan informasi ini untuk menyimpulkan lokasi mereka. Operasi ini disebut *triangulation*.

Satelit-satelit GPS mentransmisikan signal radio ke bumi yang mengandung informasi tentang satelit, contohnya lokasi dan waktu saat itu, menginformasikan pengguna secara tepat di mana satelit berada dalam orbit mereka sementara GPS *receivers* secara pasif menerima signal satelit tersebut. Semua satelit-satelit GPS menyinkronisasi operasi-operasi agar signal-signal yang berulang ini ditransmisikan pada waktu yang bersamaan.

Gambar 3 Area Pertama, Kedua dan Ketiga
(http://www.sfu.ca/gis/bguide/icons/Fig3_1_GPS.gif)



Gambar 4 Triangulation
(http://ww2.it.nuigalway.ie/staff/h_melvin/threesat.gif)

Rumus yang digunakan untuk menghitung jarak adalah rumus *Great Circle Distance*, yaitu *Great Circle Distance* = RadiusBumi * ACOS (COS(RADIAN(90-(Lat1*24))) * COS(RADIAN(90-(Lat2*24))) + SIN(RADIAN(90-(Lat1*24))) * SIN(RADIAN(90-(Lat2*24))) * COS(RADIAN(24*(Long1-Long2))))

Lat1 adalah *latitude* nilai 1, dalam DD:MM:SS;
Long1 adalah *longitude* nilai 1, dalam DD:MM:SS;
Lat2 adalah *latitude* nilai 2, dalam DD:MM:SS;
Long2 adalah *longitude* nilai 2, dalam DD:MM:SS;
RadiusBumi adalah radius bumi (3963 mil atau 6377 kilometer).

NMEA (atau NMEA 0183) adalah spesifikasi data untuk komunikasi antara mesin *marine* (laut) dan lainnya dengan GPS *receivers* dalam menghasilkan informasi posisi saat itu. Komunikasi GPS *receiver* menggunakan spesifikasi NMEA pada transmisi data mereka. Data ini termasuk solusi lengkap PVT (*position*, *velocity*, *time*) yang dihitung oleh GPS *receiver*. Gagasan NMEA adalah

mengirimkan sebarisan data yang disebut kalimat yang secara keseluruhan termasuk dirinya dan bebas dari kalimat lainnya. Jenis kalimat yang paling sering digunakan adalah RMA, RMB, RMC, GGA, GSA, dan VTG.

General Packet Radio Service (GPRS) adalah suatu layanan transimisi data pada perangkat *mobile* yang disediakan pada pengguna. Biaya penggunaan untuk transmisi data menggunakan GPRS dihitung per *megabyte*, sementara yang lain pada umumnya dihitung per menit waktu koneksi. GPRS dapat digunakan untuk layanan seperti WAP, SMS, MMS, layanan komunikasi internet. Di masa berikutnya, GPRS diharapkan dapat mengurangi biaya komunikasi antar telepon melalui VoIP (*Voice over IP*).

HASIL DAN PEMBAHASAN

Analisis Algoritma

Data-data yang akurat diperlukan dalam penelitian agar dapat menghasilkan informasi yang berguna. Data-data tersebut adalah data posisi pengguna, data nama tempat, dan data jalan. Dalam representasi *graph* untuk pencarian rute terpendek, jalan diibaratkan sebagai *edge* dan diberi simpul (*node*) sebagai batas jalan.

Analisis algoritma dilakukan untuk mengetahui efektifitas dari beberapa algoritma *single source shortest path*, algoritma Bellman Ford dan Dijkstra sehingga dapat diketahui algoritma yang lebih baik untuk pencarian rute terpendek pada aplikasi yang akan dibuat. Adapun analisis yang akan dilakukan adalah *time complexity*, yaitu jumlah langkah yang dilakukan pada suatu program untuk mengeksekusi fungsi yang ditulis.

Tabel 1 *Time Complexity* pada Algoritma Bellman Ford

No	Node Awal	Node Akhir	Bellman Ford Waktu (micro-second)										
			Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Rata-rata
1	1	500	12073.08	11532.49	12196.31	11649.83	12696.14	12220.40	10489.24	12392.03	13511.18	10260.99	11902.17
2	10	485	12085.80	11636.34	11910.09	12004.86	11377.16	11594.24	11596.58	11596.44	11872.34	11449.30	11712.31
3	17	360	11845.81	12022.06	11853.92	11858.06	11631.32	11434.15	11001.78	11962.99	11667.97	11865.38	11714.34
4	21	277	12238.19	11934.40	11809.33	12227.05	11931.49	11771.39	11721.44	12271.49	11784.56	11845.52	11953.49
5	66	490	10725.59	10884.36	10128.35	10791.59	11058.92	14271.81	14313.32	11144.94	10917.06	10526.99	11476.29
6	128	394	11283.03	11592.02	11654.58	11497.44	11972.88	11398.35	11565.35	11915.97	11542.91	11470.34	11589.29
7	370	72	11742.49	11601.79	11579.49	11711.57	11767.41	11582.05	11872.26	11684.49	11566.33	11792.69	11690.06
8	396	36	11233.48	11906.77	12046.00	11503.27	11694.22	11526.25	11971.15	11938.93	11507.11	11462.89	11679.01
9	456	20	11644.37	11328.04	11835.98	11927.35	11873.53	11972.22	11652.02	11536.67	11393.92	11315.12	11647.92
10	482	14	11870.07	11600.73	11925.44	11630.33	11664.77	12205.12	11886.69	12286.03	12091.56	12023.33	11918.41

Tabel 2 *Time Complexity* pada Algoritma Dijkstra

No	Node Awal	Node Akhir	Dijkstra Waktu (micro-second)										
			Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Rata-rata
1	1	500	23.814100	22.943100	24.067800	22.861500	24.849500	24.006100	20.824800	24.228400	26.842700	20.607200	23.504520
2	10	485	24.189000	23.418800	23.645100	23.966400	23.519400	23.474100	23.905800	23.708800	24.307800	23.398700	23.753390
3	17	360	23.740600	23.811600	23.721900	23.714400	23.691200	23.119100	22.400500	23.717500	23.741200	23.732500	23.539050
4	21	277	23.994400	23.748000	23.759600	24.063100	23.812000	23.852500	23.281600	23.747200	23.784600	24.039200	23.808220
5	66	490	21.193000	22.100600	20.040300	21.702800	21.261700	28.986000	29.256200	22.131100	21.236000	21.514000	22.942170
6	128	394	22.785900	22.761600	23.221300	22.835800	23.108200	22.837600	23.236200	23.333200	23.056700	22.928100	23.010460
7	370	72	23.284100	23.338200	23.431700	23.361300	23.235200	23.160200	23.750400	23.096000	23.061400	23.602400	23.332090
8	396	36	22.885600	23.271800	23.399200	22.715200	23.296100	23.183600	23.781700	23.459100	23.044700	23.253700	23.229070
9	456	20	22.984800	23.107200	23.703600	23.575100	23.369200	23.644100	22.880500	23.474900	23.088800	23.413500	23.324170
10	482	14	23.729400	23.742200	23.655000	23.078200	23.652400	23.989200	23.635700	23.632600	23.776200	23.778300	23.666920

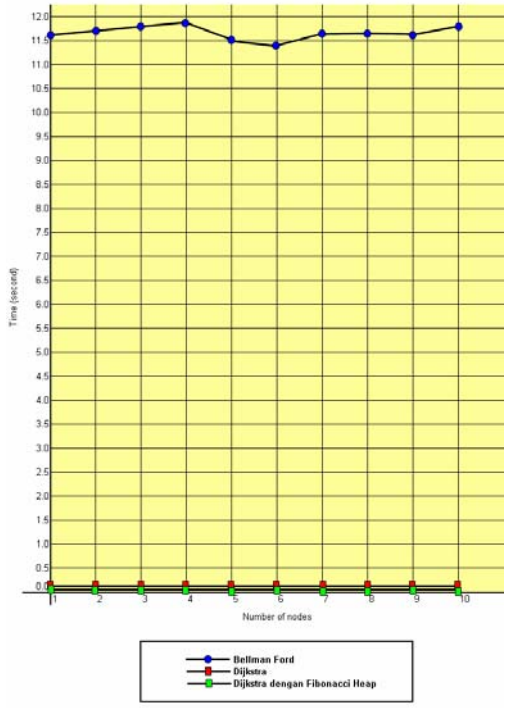
Tabel 3 *Time Complexity* pada Algoritma Dijkstra Metode Fibonacci Heap

No	Node Awal	Node Akhir	Dijkstra dengan metode Fibonacci-Heap Waktu (micro-second)									
			Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Rata-rata
1	1	500	2.273232	2.273232	2.295914	2.270268	2.325160	2.290566	1.672970	2.302104	2.362976	1.669147	2.173557
2	10	485	1.916032	1.972583	1.902516	2.084577	2.018182	1.908533	2.035057	1.806458	2.013443	1.949371	1.960675
3	17	360	1.878805	2.088599	1.880876	1.906090	1.936864	2.253547	2.274952	1.865180	2.051300	2.024099	2.016031
4	21	277	2.429607	2.320770	2.432343	2.441347	2.426622	2.195812	2.161799	2.445227	2.318735	2.539468	2.371173
5	66	490	1.765507	1.804355	1.663220	1.795017	1.724065	1.519617	1.604771	1.780853	1.707157	1.746500	1.711106
6	128	394	2.451384	2.355468	2.556714	2.386625	2.452834	2.359853	2.465634	2.530954	2.463489	2.435351	2.445831
7	370	72	2.533474	2.393412	2.549747	2.497319	2.532659	2.401432	2.429682	2.569652	2.575474	2.594230	2.507708
8	396	36	2.637984	2.529495	2.700256	2.660760	2.668067	2.540852	2.610436	2.720478	2.658370	2.634041	2.636074
9	456	20	2.682159	2.565818	2.646644	2.650289	2.598350	2.659010	2.717483	2.678725	2.661142	2.678581	2.653820
10	482	14	2.666178	2.585220	2.686924	2.689975	2.594812	2.680862	2.680553	2.698504	2.692031	2.676921	2.665198

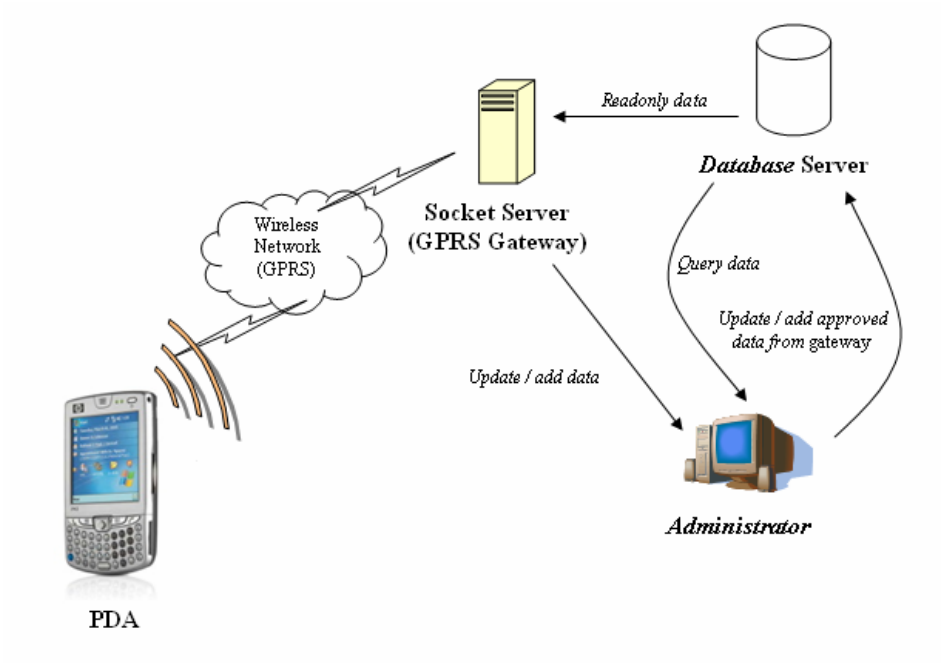
Kesimpulan:
Pencarian rute terpendek menggunakan algoritma Dijkstra dengan metode Fibonacci Heap ternyata menghasilkan nilai waktu yang lebih kecil jika dibandingkan dengan algoritma Dijkstra biasa dan Bellman Ford. Ini berarti algoritma Dijkstra dengan metode Fibonacci Heap lebih cepat dibandingkan dengan kedua algoritma tersebut.

Perancangan Sistem

Perancangan sistem merupakan pengembangan sistem dan prosedur baru agar diketahui konsep dari sistem itu bekerja sehingga kebutuhan dan informasi yang dibutuhkan akan terdefinisi secara jelas. Sistem terbagi menjadi 3 bagian besar, yaitu Mobile Device (PDA) sebagai alat untuk meminta *request* rute terpendek dan menerima *output* dari *request* tersebut; Server dan Back End dimana *server* bertugas menerima dan mengolah informasi yang diminta oleh pengguna berdasarkan data-data yang ada di dalam *database*, hasil pengolahan kemudian akan dikirimkan balik oleh *server* ke *mobile device*, sedangkan *back end* berfungsi untuk memasukkan data dan meng-*update* data pada *database*; *Database* berisi semua data-data yang diperlukan pada sistem, data-data dari *database* akan langsung disimpan di dalam memori.



Gambar 5 Grafik Perbandingan Perhitungan Waktu dari Ketiga Algoritma

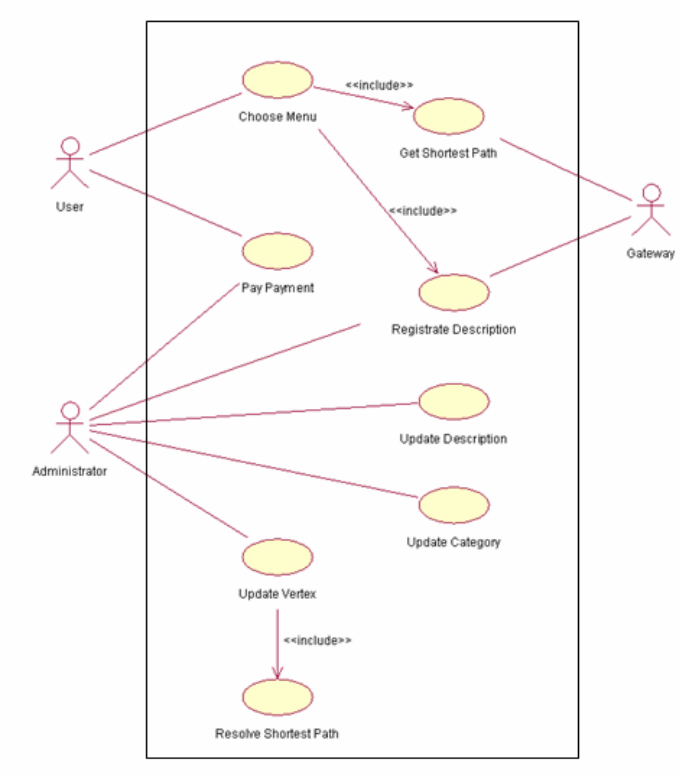


Gambar 6 Rancangan Sistem

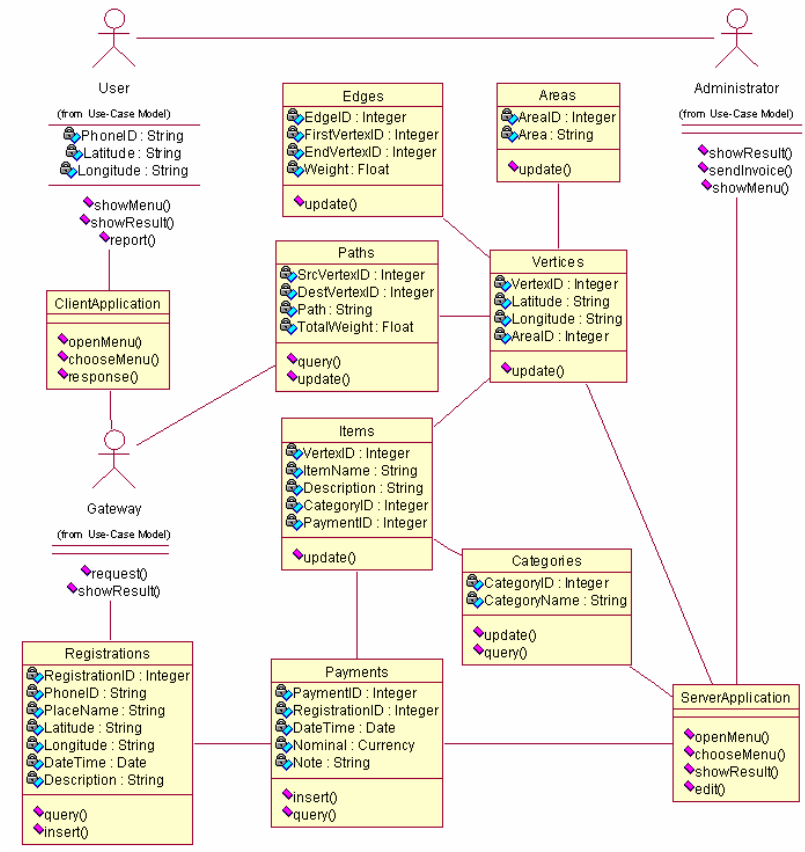
Perancangan UML

Perancangan UML meliputi *Use Case Diagram*, dan *Class Diagram*. *Use Case Diagram* memberikan gambaran mengenai sistem dari sudut pandang pengguna, dimana pengguna dapat memilih kategori pilihan tempat yang hendak dituju sebagai tempat terdekat dari posisi pengguna berada. Pengguna tidak perlu memasukkan posisi dimana dia berada karena koordinat posisi pengguna akan langsung dikirimkan pada saat pengguna memilih tempat yang akan dituju. Agar informasi yang diberikan selalu merupakan informasi yang terbaru, maka informasi mengenai kondisi jalan atau kondisi lalu lintas selalu di-*update* oleh Administrator sedangkan data statik seperti tempat, jalan, dan karakteristik jalan akan disimpan di dalam *database*.

Pengguna menggunakan PDA-nya untuk memilih kategori tempat tujuan, dimana satu atau lebih pengguna hanya dapat memilih satu kategori. Lalu kategori yang dipilih tersebut digunakan untuk mencari data di tabel Vertices, data-data di tabel Vertices sendiri terhubung dengan tabel Edges, dimana satu titik di tabel Vertices menghubungkan satu atau lebih Edges.



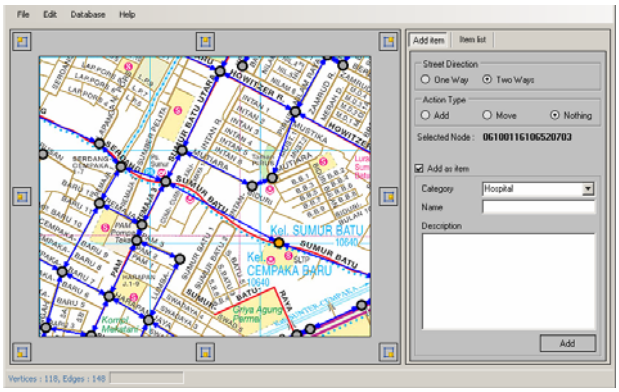
Gambar 7 *Use Case Diagram*



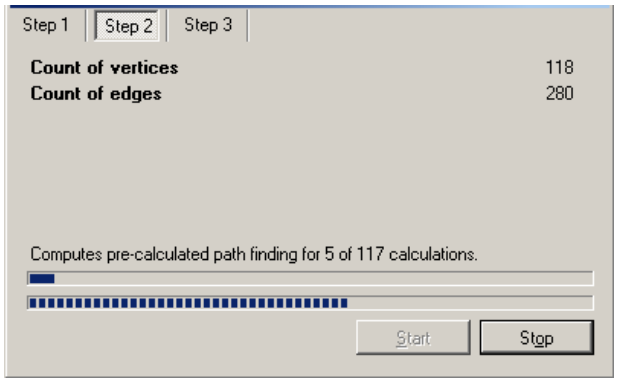
Gambar 8 *Class Diagram*

Implementasi Piranti Lunak

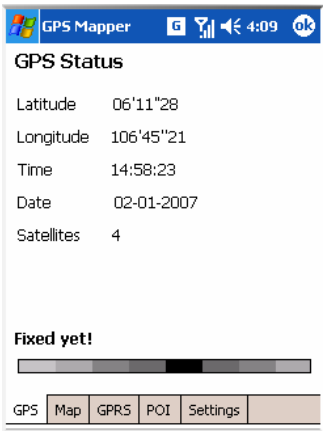
Piranti keras dan piranti lunak dibutuhkan agar sistem yang dibuat dapat berjalan dengan baik. Piranti lunak dirancang untuk administrator dan pengguna (*user*). Administrator dapat menentukan arah jalan, apakah satu arah (*one way*) atau dua arah (*two ways*), menambah titik (tempat), atau menggeser titik. Administrator juga dapat melakukan pencarian rute terpendek dari satu titik ke titik lainnya, dan melakukan uji coba terhadap pencarian dari suatu titik ke titik tujuan. *User* dapat melihat status GPS, menampilkan peta serta menampilkan hasil permintaan rute terpendek yang dicari.



Gambar 9 Tampilan Layar untuk Menambah Status Tempat



Gambar 10 Tampilan Layar untuk Proses Pencarian Rute Terpendek



Gambar 11 Tampilan Layar Pertama atau Tab GPS

Gambar 12 Tampilan Layar Hasil Pencarian Rute Terpendek

Evaluasi

Evaluasi dilakukan untuk mengetahui lama waktu rata-rata yang dibutuhkan untuk mendapatkan rute terpendek pada aplikasi *server* dari 100, 200, 350 dan 500 titik. Evaluasi kestabilan program ini dilakukan dengan menggunakan 3 jenis prosesor, yaitu Intel Centrion 1.73 GHz, Intel Celeron 2.53 GHz, dan Intel Pentium 4 3.0 GHz.

Tabel 4 Spesifikasi Sistem pada Server yang Digunakan untuk Evaluasi

Hardware	Software
Memori 1 GB Hard Disk 60 GB 5400 rpm QWERTY Keyboard Mouse Monitor resolusi 1024 x 768	Windows XP SP2 .NET Framework 2.0 Microsoft SQL Server 2000 SP4

Tabel 5 Hasil Pengujian untuk 100 Verteks

Pengujian ke-	Centrino 1.73 GHz		Celeron 2.53 GHz		Pentium 4 3.0 GHz	
	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)
1	0:9.136	9136	0:6.247	6247	0:5.257	5257
2	0:10.031	10031	0:6.858	6858	0:5.782	5782
3	0:9.237	9237	0:6.322	6322	0:5.328	5328
4	0:9.338	9338	0:6.386	6386	0:5.372	5372
5	0:9.209	9209	0:6.297	6297	0:5.305	5305
6	0:9.99	9990	0:6.833	6833	0:5.757	5757
7	0:9.327	9327	0:6.373	6373	0:5.371	5371
8	0:10.128	10128	0:6.919	6919	0:5.832	5832
9	0:9.102	9102	0:6.221	6221	0:5.238	5238
10	0:10.308	10308	0:7.046	7046	0:5.938	5938
Total Waktu	**1:35.806**	**95806**	**1:5.501**	**65501**	**0:55.178**	**55178**
Rata-rata	**0:9.5806**	**9580.6**	**0:6.55**	**6550**	**0:5.518**	**5518**

Tabel 6 Hasil Pengujian untuk 200 Verteks

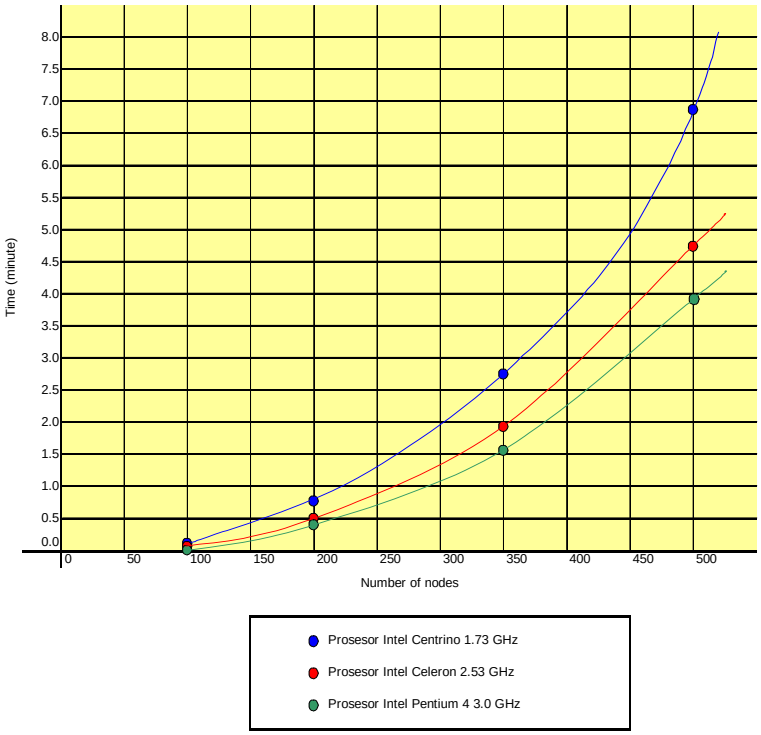
Pengujian ke-	Centrino 1.73 GHz		Celeron 2.53 GHz		Pentium 4 3.0 GHz	
	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)
1	0:43.424	43424	0:29.692	29692	0:25.01	25010
2	0:44.68	44680	0:30.543	30543	0:25.744	25744
3	0:41.362	41362	0:28.279	28279	0:23.825	23825
4	0:41.419	41419	0:28.32	28320	0:23.854	23854
5	0:43.639	43639	0:29.838	29838	0:25.135	25135
6	0:42.243	42243	0:28.881	28881	0:24.328	24328
7	0:42.458	42458	0:29.026	29026	0:24.46	24460
8	0:42.374	42374	0:28.983	28983	0:24.414	24414
9	0:43.385	43385	0:29.667	29667	0:24.996	24996
10	0:42.366	42366	0:28.967	28967	0:24.401	24401
Total Waktu	**7:7.35**	**427350**	**4:52.195**	**292195**	**4:6.168**	**246168**
Rata-rata	**0:42.735**	**42735**	**0:29.219**	**29219**	**0:24.617**	**24617**

Tabel 7 Hasil Pengujian untuk 350 Verteks

Pengujian ke-	Centrino 1.73 GHz		Celeron 2.53 GHz		Pentium 4 3.0 GHz	
	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)
1	2:46.629	166629	1:53.947	113947	1:35.986	95986
2	2:41.516	161516	1:50.44	110440	1:33.032	93032
3	2:43.762	163762	1:51.974	111974	1:34.327	94327
4	2:44.522	164522	1:52.496	112496	1:34.771	94771
5	2:41.126	161126	1:50.176	110176	1:32.808	92808
6	2:41.716	161716	1:50.587	110587	1:33.152	93152
7	2:41.324	161324	1:50.317	110317	1:32.918	92918
8	2:44.321	164321	1:52.365	112365	1:34.648	94648
9	2:49.533	169533	1:55.929	115929	1:37.649	97649
10	2:43.901	163901	1:52.077	112077	1:34.41	94410
Total Waktu	**27:18.35**	**1638350**	**18:40.309**	**1120309**	**15:43.701**	**943701**
Rata-rata	**2:43.835**	**163835**	**1:52.031**	**112031**	**1:34.37**	**94370**

Tabel 8 Hasil Pengujian untuk 500 Verteks

Pengujian ke-	Centrino 1.73 GHz		Celeron 2.53 GHz		Pentium 4 3.0 GHz	
	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)	Waktu (menit:detik)	Waktu (milidetik)
1	9:38.198	578198	6:35.372	395372	5:33.039	333039
2	6:36.767	396767	4:31.312	271312	3:48.541	228541
3	6:34.933	394933	4:30.049	270049	3:47.477	227477
4	6:32.404	392404	4:28.326	268326	3:46.025	226025
5	6:31.968	391968	4:28.034	268034	3:45.772	225772
6	6:32.881	392881	4:28.651	268651	3:46.305	226305
7	6:35.358	395358	4:30.35	270350	3:47.727	227727
8	6:29.957	389957	4:26.652	266652	3:44.623	224623
9	6:32.136	392136	4:28.15	268150	3:45.872	225872
10	6:31.052	391052	4:27.406	267406	3:45.248	225248
Total Waktu	**68:35.654**	**4115654**	**46:54.302**	**2814302**	**39:30.628**	**2370628**
Rata-rata	**6:51.5654**	**411565.4**	**4:41.43**	**281430**	**3:57.063**	**237063**



Gambar 13 Grafik Hasil Pengujian Pencarian Rute Terpendek

SIMPULAN

Simpulan yang diperoleh dari penelitian ini, yaitu algoritma Dijkstra metode Fibonacci Heap memiliki waktu yang lebih cepat dalam mencari rute terpendek dibandingkan dengan algoritma Dijktra metode biasa dan Bellman Ford; Dengan adanya aplikasi seperti ini, memungkinkan pengguna PDA yang dilengkapi GPS mendapatkan informasi rute terpendek; Aplikasi ini mampu menyajikan data GPS berupa lokasi tempat ke dalam bentuk grafikal (peta) yang lebih mudah untuk dipahami dibandingkan data-data dalam bentuk koordinat global; Kecepatan pemrosesan data pada saat *nodes training* sangat dipengaruhi oleh spesifikasi piranti keras yang digunakan. Untuk data yang lebih banyak peningkatan piranti keras yang lebih baik sangat dibutuhkan.

Saran untuk penelitian dan pengembangan lebih lanjut, yaitu peta wilayah diperluas; Informasi yang dihasilkan bukan hanya rute terpendek tapi bisa pula rute tercepat dengan memperhatikan faktor kemacetan; Penggunaan peta *vector* sebagai peta dasar akan meningkatkan kecepatan aplikasi dalam menampilkan peta, dan memperkecil media penyimpanan yang dibutuhkan; Meningkatkan keakuratan posisi yang dihasilkan dengan membangun sistem peta khusus (tipe *vector*) yang didapat dengan melakukan *plotting* atau pemetaan beberapa tempat dengan menggunakan alat GPS langsung; Aplikasi lebih dikembangkan lagi agar pengguna dapat membentuk suatu komunitas sendiri sehingga dapat mengetahui informasi posisi pengguna lain dalam komunitasnya; Aplikasi dilengkapi dengan fasilitas panduan penggunaan.

DAFTAR PUSTAKA

Ambler, S. W. *A manager's introduction to the Rational Unified Process (RUP)*. Retrieved 2007 from http://www.ambysoft.com/downloads/managersIntroToRUP.pdf

Anonymous1. *Fibonacci heap*. Retrieved 2007 from http://en.wikipedia.org/wiki/Fibonacci_heap

Anonymous2. *Coordinate system*. Retrieved 2007 from http://en.wikipedia.org/wiki/Coordinate_system

Anonymous3. *GPS*. Retrieved 2007 from http://en.wikipedia.org/wiki/GPS

Anonymous4. *GPRS*. Retrieved 2007 from http://en.wikipedia.org/wiki/GPRS

Anonymous5. *Location based service*. Retrieved 2007 from http://en.wikipedia.org/wiki/Location_based_service

Anonymous6. *NMEA*. Retrieved 2007 from http://en.wikipedia.org/wiki/NMEA

Anonymous7. *Rational unified process*. Retrieved 2007 from http://www.therationaledge.com/content/jan_01/f_rup_pk.html

Anonymous8. *Rational unified process*. Retrieved 2007 from http://en.wikipedia.org/wiki/Rational_Unified_Process

Booch, Gr., James R., and Ivar J. (1999). *The unified modeling language user guide*, United States of America: Addison Wesley.

Horowitz, E., Sahni. S, and Rajasekaran, S. (1998). *Computer algorithms/C++*, United States of America: Computer Science Press.

Johnsonbaugh, R. (2001). *Discrete mathematics*, 5th ed., United States of America: Prentice Hall.

Magon, A., and R. Shukla. *LBS, the ingredients and the alternatives*. Retrieved 2007 from http://www.gisdevelopment.net/technology/lbs/techlbs006.htm