BAB II

LANDASAN TEORI

2.1 Transportasi

Menurut Nasution (2004:15), transportasi diartikan sebagai pemindahan barang dan manusia dari tempat asal ke tempat tujuan. Proses pengangkutan merupakan gerakan dari tempat asal, dari mana kegiatan angkutan dimulai, ke tempat tujuan, ke mana kegiatan pengangkutan diakhiri.

Dan menurut Rustian Kamaluddin (2003:3), transportasi adalah kegiatan pemindahan barang (muatan) dan penumpang dari suatu tempat ke tempat lain.

Unsur – unsur transportasi meliputi :

1. Manusia yang membutuhkan
2. Barang yang dibutuhkan
3. Kendaraan sebagai alat/sarana
4. Jalan dan terminal sebagai prasarana transportasi
5. Organisasi (pengelola transportasi)

Transportasi sebagai dasar untuk pembangunan ekonomi dan perkembangan masyarakat serta pertumbuhan industrilisasi. Dengan adanya transportasi menyebabkan adanya speialisasi atau pembagian pekerjaan menurut keahlian sesuai dengan budaya, adat istiadat dan budaya suatu bangsa dan daerah kebutuhan akan angkutan tergantung fungsi bagi kegunaan seseorang (*personal place utility*).

2.1.1 Pengertian Angkutan Umum

Angkutan pada dasarnya adalah sarana untuk memindahkan orang dan atau barang dari satu tempat ke tempat lain. Tujuannya membantu orang atau

kelompok orang menjangkau berbagai tempat yang dikehendaki atau mengirimkan barang dari tempat asalnya ke tempat tujuannya. Prosesnya dapat dilakukan dengan menggunakan sarana angkutan berupa kendaraan. Sementara Angkutan Umum Penumpang adalah angkutan penumpang yang menggunakan kendaraan umum yang dilakukan dengan sistem sewa atau bayar. Termasuk dalam pengertian angkutan umum penumpang adalah angkutan kota (bus, minibus, dsb), kereta api, angkutan air, dan angkutan udara. (Warpani, 1990).

Angkutan Umum Penumpang bersifat massal sehingga biaya angkut dapat dibebankan kepada lebih banyak orang atau penumpang yang menyebabkan biaya per penumpang dapat ditekan serendah mungkin. Karena merupakan angkutan massal, perlu ada kesamaan diantara para penumpang, antara lain kesamaan asal dan tujuan. Kesamaan ini dicapai dengan cara pengumpulan di terminal dan atau tempat perhentian. Kesamaan tujuan tidak selalu berarti kesamaan maksud. Angkutan umum massal atau masstransit memiliki trayek dan jadwal keberangkatan yang tetap. Pelayanan angkutan umum penumpang akan berjalan dengan baik apabila tercipta keseimbangan antara ketersediaan dan permintaan. Oleh karena itu, pemerintah perlu turut campur tangan dalam hal ini. (Warpani, 1990).

2.1.2 Peranan Angkutan Umum

Angkutan Umum berperan dalam memenuhi kebutuhan manusia akan pergerakan ataupun mobilitas yang semakin meningkat, untuk berpindah dari suatu tempat ke tempat lain yang berjarak dekat, menengah ataupun jauh. Angkutan umum juga berperan dalam pengendalian lalu lintas, penghematan

bahan bakar atau energi, dan juga perencanaan & pengembangan wilayah. (Warpani, 1990).

Esensi dari operasional angkutan umum adalah memberikan layanan angkutan yang baik dan layak bagi masyarakat dalam menjalankan kegiatannya, baik untuk masyarakat yang mampu memiliki kendaraan pribadi sekalipun (*Choice*), dan terutama bagi masyarakat yang terpaksa harus menggunakan angkutan umum (*Captive*). Ukuran pelayanan angkutan umum yang baik adalah pelayanan yang aman, cepat, murah, dan nyaman. (Warpani, 1990).

Beberapa fungsi transportasi yaitu :

a) melancarkan arus barang dan manusia

b) menunjang perkembangan dan pembangunan *(the promoting sector)* penunjang dan perangsang pemberian jasa bagi perkembangan perekonomian *(the service sector*).

2.1.3 Pengelompokan Pelaku Perjalanan & Moda

Masyarakat pelaku perjalanan (konsumen jasa transportasi), dapat kita kelompokkan ke dalam 2 kelompok yaitu :

1. Golongan pertama merupakan jumlah terbesar di negara berkembang, yaitu golongan masyarakat yang terpaksa menggunakan angkutan umum karena ketiadaan kendaraan pribadi. Mereka secara ekonomi adalah golongan masyarakat lapisan menengah ke bawah.
2. Golongan kedua merupakan jumlah terbanyak di negara-negara maju, yaitu golongan masyarakat yang mempunyai kemudahan (akses) ke kendaraan

pribadi dan dapat memilih untuk menggunakan angkutan umum atau angkutan pribadi. Mereka secara ekonomi adalah golongan masyarakat lapisan menengah ke atas (kaya atau ekonomi kuat).

Dan untuk moda transportasi, secara umum ada 2 (dua) kelompok besar moda transportasi yaitu :

1. Kendaraan Pribadi (*Private Transportation*)

Moda transportasi yang dikhususkan buat pribadi seseorang dan seseorang itu bebas memakainya ke mana saja, di mana saja dan kapan saja dia mau, bahkan mungkin juga dia tidak memakainya sama sekali (misal : mobilnya disimpan digarasi). Contoh kendaraan pribadi seperti :

a. Sepeda untuk pribadi
b. Sepeda motor untuk pribadi
c. Mobil pribadi
d. Pesawat pribadi

2. Kendaraan Umum (*Public Transportation*)

Moda transportasi yang diperuntukkan buat bersama (orang banyak), kepentingan bersama, menerima pelayanan bersama, mempunyai arah dan titik tujuan yang sama, serta terikat dengan peraturan trayek yang sudah ditentukan dan jadwal yang sudah ditetapkan dan para pelaku perjalanan harus wajib menyesuaikan diri dengan ketentuan-ketentuan tersebut apabila angkutan umum ini sudah mereka pilih. Contoh kendaraan umum seperti :

a. Ojek sepeda, sepeda motor
b. Becak, bajaj, bemo
c. Mikrolet
d. Bus umum (kota dan antar kota)
e. Kereta api (kota dan antar kota)
f. Kapal Feri
g. Pesawat yang digunakan secara bersama.

Selain dapat memenuhi kebutuhan penduduk di kota, atau pedalaman, keberhasilan pembangunan di sektor transportasi dapat memenuhi peekembangan wilayah. Seiring dengan meningkatnya jumlah habitat, dan semakin majunya peradaban komunitas manusia, selanjutnya wilayah-wilayah pusat kegiatannya berkembang mengekspansi ke pinggiran-pinggiran wilayah, sedangkan kawasan-kawasan terisolir semakin berkurang, dan harak antar kota semakin pendek dalam hal waktu. Lebih dari itu kuantitas dan kualitas baik perkotaan besar maupun perkotaan kecil tumbuh, dimana kota kecil ditumbuh kembangkan sementara kota besar semakin berkembang, sehingga area perkotaan semakin meluas.

2.2 Android

Android adalah sistem operasi untuk selular atau *tablet pc* yang berbasis linux. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam-macam piranti bergerak (*mobile*) (Yopa, 2011).

Awalnya, *Google Inc*. membeli *android Inc*. pendatang baru yang membuat peranti lunak untuk perangkat ponsel. Kemudian untuk

mengembangkan *android*, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, piranti lunak, dan telekomunikasi, termasuk *Goolge, HTC, Intel, Motorola, Qualcom, T-Mobile dan Nvidia* (Yopa, 2011).

2.2.1 Jenis-Jenis dan varian *Android*

Ada beberapa jenis varian *Android*, yaitu :

a) *Android* versi 1.1 pada 9 maret 2009, *Google* Merilis *android* versi 1.1. *Android* versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam, *Voice Search* (pencarian suara), pengiriman pesan dengan *Gmail*, dan pemberitahuan email.

b) *Android* versi 1.5 (*cupcake*) pada pertengahan maret 2009. Terdapat beberapa pemberharuan termasuk juga penambahan beberapa fitur dalam ponsel versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke youtube dan gambar picasa langsung dari telepon, dukungan Bluetooth A2DP, kemampuan terhubung secara otomatis ke *headset Bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

c) *Android* versi 1.6 (*Donut*) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol *applet VPN*. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, kamera, *camcorder* dan galeri yang diintegrasikan; *CDMA/EVDO, 802.1x, VPN, Gestures dan text to speech engine*; kemampuan dial kontak; teknologi pengadaan resolusi VWGA; *teknologi text to change speech.*

d) *Android* versi 2.0/2.1 (*Eclair*) dirilis pada 3 desember 2009. Perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan *google maps 3.1.2*, perubahan UI dengan browser baru dan dukungan HTML5. Daftar kontak yang baru, dukungan flash untuk kamera 3.2 MP, digital zoom, dan mengadakan kompetisi aplikasi terbaik(*killer apps*).

e) *Android* versi 2.2 (*Froyo: Frozen yoghurt*) dirilis pada mei 2010. Perubahan yang dilakukan berupa penambahan dukungan terhadap *adobe flash* 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, integrasi V8 *engine script* yang dipakai *google chrome* yang digunakan untuk mempercepat kemampuan rendering pada *browser*, pemasangan apliaksi dalam *SDCard*, kemampuan *wifi hostspot portable*, dan kemampuan *auto update* pada aplikasi *android market*.

f) *Android* versi 2.3 (*Gingerbread*) dirilis pada 6 desember 2010. Perubahan umum yang dilakukan adalah meningkatkan kemampuan permainan(*gaming*). Peningkatan fungsi *copy paste*, layar antar muka di desain ulang, dukungan format video VP8 dan WebM, efek audio baru(*reverb,equalization,bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

g) *Android* versi 3.0 (*honeycomb*) *android* versi ini dirancang khusus untuk *tablet*. *Android* versi ini mendukung ukuran layar yang lebih besar. Antar muka juga berbeda karena didesain untuk *tablet*. *Honeycomb* juga mendukung untuk multi processor dan juga akselerasi perangkat keras untuk grafis.

2.3 Google Maps

Google Maps merupakan layanan dari google yang mempermudah pengunanya untuk melakukan kemampuan pemetaan untuk aplikasi yang dibuat. Sedangkan Google Maps API memungkinkan pengembangan untuk mengintegrasikan Google Maps ke dalam situs web. Dengan menggunakan Google Maps API memungkinkan untuk menanamkan situs Google Maps ke dalam situs eksternal, di mana situs data tertentu dapat dilakukan *overlay*.

Meskipun pada awalnya hanya JavaScript API, API Maps sejak diperluas untuk menyertakan sebuah API untuk Adobe Flash aplikasi, layanan untuk mengambil gambar peta statis, dan layanan web untuk melakukan geocoding, menghasilkan petunjuk arah mengemudi, dan mendapatkan profil elevasi.

Kelas kunci dalam perpustakaan Maps adalah MapView , sebuah subclass dari ViewGroup dalam standar perpustakaan Android. Sebuah MapView menampilkan peta dengan data yang diperoleh dari layanan Google Maps. Bila MapView memiliki fokus, dapat menangkap tombol yang ditekan dan gerakan sentuh untuk pan dan zoom peta secara otomatis, termasuk penanganan permintaan jaringan untuk ubin peta tambahan. Ini juga menyediakan semua elemen UI yang diperlukan bagi pengguna untuk mengendalikan peta. Aplikasi tersebut juga dapat menggunakan metode MapView kelas untuk mengontrol MapView secara terprogram dan menarik sejumlah jenis Tampilan di atas peta.

Secara umum, kelas MapView menyediakan pembungkus di Google Maps API yang memungkinkan aplikasi tersebut memanipulasi data Google Maps melalui metode kelas, dan itu memungkinkan dikerjakan dengan data Maps seperti jenis lain Views. Perpustakaan Maps eksternal bukan bagian dari

perpustakaan Android standar, sehingga tidak mungkin ada pada beberapa perangkat Android biasa. Demikian pula, perpustakaan Maps eksternal tidak termasuk dalam perpustakaan Android standar yang disediakan dalam SDK. Google API pengaya menyediakan perpustakaan Maps untuk sehingga dapat mengembangkan, membangun, dan menjalankan aplikasi berbasis peta di SDK Android, dengan akses penuh ke data Google Maps.

2.4 Pengertian Perangkat *Mobile*

Secara bahasa, istilah *mobile* sendiri dapat diartikan sebagai sesuatu yang bergerak, sesuatu yang mudah dibawa kemana-mana. Dan di sini kita akan langsung batasi pengertian dari perangkat *mobile* sebagai alat untuk komunikasi. Jadi, dengan adanya perangkat *mobile* (*mobile device*), dimanapun kita berada, kapan pun waktunya, dan apa pun aktifitasnya, kita akan dapat dengan mudah melakukan hubungan komunikasi dengan siapa pun.

"Dengan perangkat *mobile*, dunia dalam genggaman". Setidaknya kalimat tersebut cukup mewakili pengertian dari perangkat *mobile* secara umum. (Taufik, 2010:2).

2.4.1 Penggunaan Umum Perangkat *Mobile*

Untuk penggunaan perangkat mobile, kita dapat klasifikasikan beberapa macam penggunaan yang umum digunakan oleh pengguna perangkat mobile. Beberapa macam penggunaan tersebut antara lain (Taufik, 2010:2):

1. Telepon dan *Messaging* (SMS)

   Penggunaan ini sebenarnya merupakan fungsi dasar dan awal dari diciptakannya perangkat *mobile*. Karena memang fungsinya adalah sebagai

perangkat telekomunikasi, sehingga cukup dengan penggunaan *voice* dan layanan pesan (*Short Message Service*) kita bisa berkomunikasi jarak jauh dengan orang lain.

2. Multimedia dan *Game*

   Merupakan penggunaan tambahan dari sisi hiburan (*entertainment*) yang ditujukan bagi pengguna perangkat *mobile*. Sekarang, fasilitas multimedia sudah sangat umum dijumpai pada perangkat *mobile*. Fitur-fitur seperti radio, mp3 *player*, sampai *video player* adalah beberapa contoh penggunaan multimedia yang sekarang banyak dijumpai pada perangkat *mobile*. Tak kalah pentingnya, *game* juga menjadi bagian tidak terpisahkan dari perangkat *mobile* saat ini. Rasa-rasanya hampir pada semua perangkat *mobile* sudah ter-*install game* didalamnya.

3. Internet *Browsing*

   Dengan adanya fasilitas koneksi seperti WAP, GPRS, 3G, dan Wifi, sangat memungkinkan bagi kita untuk dapat berselancar di dunia maya seperti yang biasa kita lakukan dengan berinternet menggunakan PC. Hal ini, juga sudah mulai menjadi kebutuhan dasar untuk penggunaan perangkat *mobile*.

4. Kamera

   Ada tidaknya fitur kamera digital nampaknya saat ini telah menjadi salah satu Kriteria *gadgeters* dalam memilih perangkat *mobile*.

5. Pertukaran Data

   Kapasitas memori yang semakin besar dapat memungkinkan pengguna perangkat *mobile* untuk melakukan penyimpanan data dalam jumlah yang cukup besar. Dengan adanya fasilitas pertuakaran data seperti *infrared*,

Bluetooth, MMS (*Multimedia Message Service*), dan sebagainya, memungkinkan bagi pengguna untuk saling bertukar data. Data bisa berupa gambar (*image*), *file* suara (mp3, WAV, dll) dan sebagainya.

6. Transaksi *Mobile*

   Kemudahan bertransaksi via perangkat *mobile* saat ini makin menjadi kepentingan tersendiri bagi para pengguna. Melalui SMS *banking*, atau *mobile banking*, dapat memunginkan untuk melakukan transaksi *online* hanya dengan menggunakan perangkat *mobile* di tangan kita.

7. Penggunaan Lain

   Serta penggunaan-penggunaan lainnya yang menjadikan penggunaan perangkat *mobile* telah begitu membudaya di kalangan masyarakat kita. Sebagai contoh penggunaan perangkat *mobile* untuk memenuhi hasrat *style* hidup juga tak kalah pentingnya bagi sebagian orang.

2.4.2 Jenis-Jenis Perangkat *Mobile*

Perangkat *mobile*, sebagaimana yang telah diuraikan pada bahasan diatas, mempunyai pengertian bahwa perangkat tersebut dapat dengan mudah kita bawa kemana-mana. Itulah mengapa terdapat istilah lain dari perangkat *mobile* ini yang disebut sebagai *handheld*. (Taufik, 2010:4)

Handheld sendiri dibagi lagi menjadi beberapa kelompok berdasarkan karakteristik penggunaannya, yaitu:

1. *Handphone* atau Telepon Seluler (Ponsel)

   Orientasi *handheld* ponsel ini berfungsi dasar sebagai alat telekomunikasi *voice*. Sehingga fitur-fitur umum yang ada biasanya sangat mengutamakan fungsi telekomunikasi itu sendiri. Sebagai contoh, karakteristik utama yang

ada pada tampilan ponsel biasanya selalu terdapat akses cepat untuk *dial* nomor telepon dan SMS.

2. *Smartphone*

   Hampir sama dengan ponsel, *smartphone* juga berorientasi ke arah telekomunikasi. Dari namanya saja kita sudah tahu bahwa *smartphone* adalah perangkat untuk telepon. Hanya saja terdapat sedikit perbedaan dengan ponsel. Selain ditujukan sebagai perangkat telekomunikasi, penciptaan awal dari *smartphone* juga difungsikan sebagai PC atau komputer *desktop* yang bersifat *mobile*.

3. *Personal Digital Assistant* (PDA)

   Agak berbeda dengan ponsel dan *smartphone*, *handheld* jenis PDA mempunyai orientasi yang terbalik. Jika pada dua kelompok *handheld* sebelumnya berfungsi utama sebagai alat telekomunikasi, maka jenis PDA ini memang ditujukan sebagai PC atau komputer *desktop* yang bersifat *mobile*. Sedangkan fungsi telekomunikasinya hanya sebagai fitur tambahan.

4. *Pager* (Radio Panggil)

   Kelompok yang terakhir ini sepertinya sudah tidak lazim digunakan pada masa sekarang. Fungsinya yang terbatas hanya pada layanan pesan saja, membuatnya kalah bersaing dengan kelompok *handheld* lainnya. *Handheld* jenis ini pernah banyak dipakai di Indonesia sekitar tahun 1996-1998, lalu kemudian tidak digunakan lagi.

2.5 Pemrograman Berorientasi Objek

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan / development suatu software dimana dalam struktur software tersebut didasarkan kepada interaksi object dalam penyelesaian suatu proses/tugas.

Merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua datadan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Model data berorientasi objek dikatakan dapat memberi flesibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak dalam sekala besar

2.5.1 Objek

Dalam ilmu komputer, sebuah obyek adalah entitas apapun yang dapat dimanipulasi oleh perintah dari sebuah bahasa pemrograman, seperti nilai (ilmu komputer), variabel, fungsi, atau struktur data. Dengan kemudian pengenalan pemrograman berorientasi objek kata yang sama, “objek”, mengacu pada contoh tertentu dari suatu kelas.

Konsep modern "objek" dan pendekatan berorientasi objek untuk pemrograman diperkenalkan oleh bahasa pemrograman Simula awalnya dirilis pada 1967, yang dipopulerkan oleh Smalltalk merilis dua tahun kemudian pada tahun 1969, dan menjadi alat standar perdagangan dengan penyebaran C + + awalnya dirilis pada tahun 1983.

2.5.2 Kelas

Dalam pemrograman berorientasi obyek, sebuah kelas adalah suatu konstruksi yang digunakan sebagai cetak biru (atau template) untuk membuat

objek dari kelas itu. Cetak biru ini menjelaskan negar dan perilaku bahwa objek dari kelas semua berbagi. Objek dari suatu kelas tertentu disebut sebuah instance dari kelas. Kelas yang mengandung (dan digunakan untuk menciptakan) yang misalnya dapat dianggap sebagai jenis objek, misalnya contoh objek dari "Buah" kelaskan menjadi tipe "Buah".

Bahasa pemrograman yang mendukung halus kelas berbeda dalam dukungan mereka untuk berbagai kelas-fitur terkait. Kebanyakan mendukung berbagai bentuk warisan kelas. Banyak bahasa juga mendukung enkapsulasi menyediakan fitur, seperti akses specifiers. Kelas dapat mempercepat pembangunan dengan mengurangi mubazir kode program, testing dan bug fixing. Jika sebuah kelas telah benar-benar teruji dan dikenal sebagai 'padatkarya', biasanya benar bahwa dengan menggunakan atau memperluas kelas diuji dengan baik akan mengurangi jumlah bug dibandingkan dengan penggunaan baru yang dikembangkan atau ad hoc di final output Selain itu, menggunakan kembali kelas efisien berarti bahwa banyak bug yang perlu diperbaiki dan hanya satu tempat ketika masalah yang ditemukan.

2.5.3 Enkapsulasi

Enkapsulasi adalah proses menyembunikan detil implementasi sebuah objek. Satu-satunya jalan untuk mengakses data objek tersebut adalah melalui interface melindungi internal statesebuah objek dari campur tangan pihak luar. Oleh karena itu objek sering digambarkan sebagai kotak hitam (black box) yang menerima dan mengirim pesan-pesan. Dalam OO programming kotak hitam tersebut berisi kode (himpunan instruksi dengan bahasa yang dipahami komputer).

2.5.4 *Inheritance (*Pewarisan*)*

Dalam Pemrograman Berbasi Objek, inheritance atau pewarisan adalah suatu keadaandimana suatu kelas baru mewarisi seluruh variabel atau data dan method yang dimiliki olehkelas yang menjadi induknya (parents). Suatu kelas bisa dikatakan mewarisi sebuah kelas lainapabila kelas tersebut memliki semua variabel dan method yang dimiliki kelas induknya dandia sendiri mememiliki variabel dan atau method sendiri yang tidak dimiliki olek kelas induk.

2.5.5 *Polimorfisme*

Polimorfisme, yang berarti satu objek dengan banyak bentuk**,** adalah konsep sederhana yang memperbolehkan method memiliki beberapa implementasi yang dipilih berdasarkan tipe objek yang dilewatkan pada pengerjaan metode. Ini dikenal sebagai *overloading* method. Ini akan memungkinkan method yang sama untuk anjing, misalkan memperlihatkan perilaku yang benar-benar berbeda. Jadi secara objek, Polimorfisme adalah suatu bentuk fungsi dalam orientasi objek yang digunakan secara bersama-sama untuk berbagai objek dan berbagai tujuan. Contoh polimorfisme yang digunakan oleh berbagai objek adalah fungsi penjumlahan. Fungsi penjumlahan dapat digunakan oleh objek integer maupun objek real.

2.6 *UML (Unified Modelling Language)*

UML adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi obyek.

UML merupakan standar yang relatif terbuka yang dikontrol oleh OMG (*Object Management Group*), sebuah konsursium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar-standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem berorientasi obyek. OMG mungkin lebih dikenal dengan standar-standar CORBA (*Common Object Request Broker Architecture*).

UML lahir dari penggabungan banyak bahasa pemodelan grafis berorientasi obyek yang berkembang pesat pada akhir 1980-an dan awal 1990. Sejak kehadirannya pada 1997, UML menghancurkan menara Babel tersebut dan menjadi sejarah (Fowler, 2004: 1-2).

Tujuan UML diantaranya adalah :

1. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
2. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
3. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.

Untuk membuat suatu model, UML memiliki diagram grafis sebagai berikut:

a. *Business Use Case Model*.
b. *Activity Diagram.*
c. *Use Case Model.*
d. *Behavior Diagram* antara lain *Sequence Diagram.*

e. *Implementation Diagram*, meliputi *Component Diagram* dan *Deployment Diagram*.

f. *Generate Code*.

Diagram-diagram tersebut diberi nama berdasarkan sudut pandang yang berbeda-beda terhadap sistem dalam proses analis atau rekayasa.

Dibuatnya berbagai jenis diagram diatas karena:

a. Setiap sistem yang kompleks lebih baik jika dilakukan pendekatan melalui himpunan berbagai sudut pandang yang kecil yang satu sama lain hampir saling bebas (*independent*). Sudut pandang tunggal senantiasa tidak mencukupi untuk melihat isi sistem yang lebih besar dan kompleks.

b. Diagram yang berbeda-beda tersebut dapat menyatakan tingkatan yang berbeda-beda dalam proses rekayasa.

c. Tujuan adanya diagram-diagram tersebut adalah agar model yang dibuat semakin mendekati realitas.

Diagram-diagram ini ditambah dengan kemampuan dokumentasi sebagai *artifacts* utama UML. *Data-flow diagram* dan tipe diagram lain yang tidak terdapat dalam UML tidak termasuk dalam paradigma *object-oriented*. *Activity diagram* dan *collaboration diagram* yang terdapat dalam UML menggantikan *data-flow diagram*. *Activity diagram* juga sangat bermanfaat untuk membuat *workflow*.

2.7 *Android SDK (Software Developement Kit)*

Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh *Google*. Saat ini disediakan *Android SDK (Software Development Kit)* sebagai alat bantu dan

API diperlukan untuk memulai mengembangkan aplikasi pada platform *Android* menggunakan bahasa *Java* (Bambang, 2011).

Pengembang memiliki akses penuh *framework API* yang sama dengan yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali (*reuse*) dengan mudah. setiap aplikasi dapat memanfaatkan kemampuan ini dan aplikasi yang lain mungkin akan memanfaatkan kemampuan ini (sesuai dengan batasan keamanan yang didefinisikan oleh *framework*). Mekanisme yang sama memungkinkan komponen untuk diganti oleh pengguna. Semua aplikasi yang merupakan rangkaian layanan dan sistem, termasuk:

a. *View Set* kaya dan *extensible* yang dapat digunakan untuk membangun aplikasi, termasuk daftar, grids, kotak teks, tombol, dan bahkan sebuah *embeddable web*.
b. *Content Provider* yang memungkinkan aplikasi untuk mengakses data (seperti dari daftar kontak telp) atau dari data mereka sendiri.
c. *Resource Manager*, yang menyediakan akses ke kode sumber non-lokal seperti string, gambar, dan tata letak file.
d. *Notifikasi Manager* yang memungkinkan semua kustom aplikasi untuk ditampilkan dalam alert status bar.
e. *An Activity Manager* yang mengelola siklus hidup aplikasi dan menyediakan navigasi umum *backstack*.

2.8 PHP

2.8.1 Sejarah Perkembangan PHP

Menurut dokumen resmi PHP, PHP merupakan singkatan dari PHP *Hypertext Preprocessor*. PHP merupakan bahasa berbentuk skrip yang ditempatkan di dalam *server* dan diproses di *server*. Secara khusus, PHP dirancang untuk membentuk aplikasi *web* dinamis. Artinya, PHP dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya, pengguna dapat menampilkan isi suatu *database* pada halaman *web*. Pada prinsipnya PHP mempunyai fungsi yang sama dengan skrip-skrip seperti ASP (*Active Server Page*), *ColdFusion*, ataupun *Perl*. Namun perlu diketahui bahwa PHP sebenarnya dapat dipakai secara *command line*, artinya skrip PHP dapat dijalankan tanpa melibatkan *web server* maupun *web browser*.

Kelahiran PHP bermula saat Rasmus Lerdorf membuat sejumlah skrip *Perl* yang dapat mengamati siapa saja yang melihat-lihat daftar riwayat hidupnya, yakni pada tahun 1994. Skrip-skrip ini selanjutnya dikemas menjadi *tool* yang disebut "*Portable Home Page*". Paket inilah yang menjadi cikal bakal dari PHP. Pada tahun 1995, Rasmus menciptkan PHP/F1 versi 2. Pada versi inilah pemrogram dapat menempelkan kode terstruktur di dalam *tag* HTML. Yang menarik, kode PHP juga dapat berkomunikasi dengan *database* dan melakukan perhitungan-perhitungan yang kompleks.

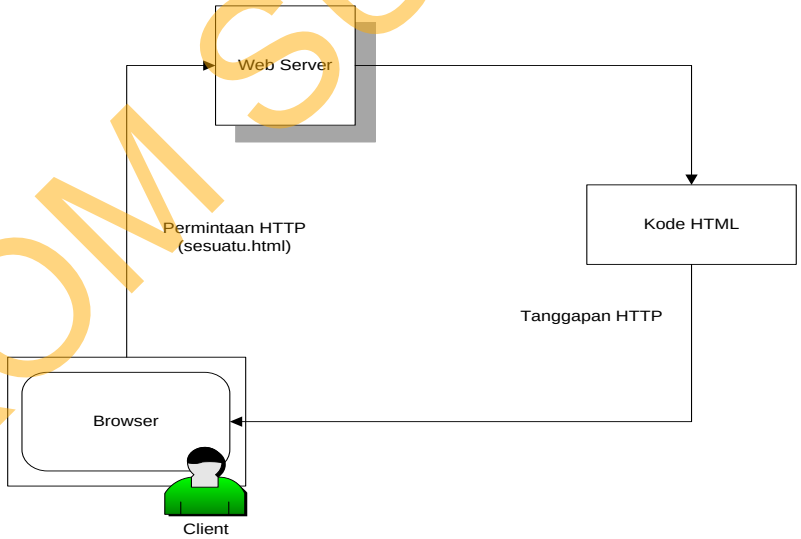
Pada saat ini, PHP cukup populer sebagai peranti pemrograman *web*, terutama di lingkungan *Linux*, walaupun demikian, PHP sebenarnya juga dapat berfungsi pada *server-server* yang berbasis *UNIX*, *Windows*, dan *Macintosh*. Pada mulanya PHP dirancang untuk diintegrasikan dengan *web server Apache*, namun

belakangan PHP juga dapat bekerja dengan *web server* seperti PWS (*Personal Web Server*), IIS (*Internet Information Server*), dan *Xitami*. (Kadir, 2008:2).

2.8.2 Konsep Kerja PHP

Model kerja HTML diawali dengan permintaan suatu halaman *web* oleh *web browser*. Berdasarkan URL (*Uniform Resource Locator*) atau dikenal dengan sebutan alamat internet, *web browser* mendapatkan alamat dari *web server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *web server*.

Selanjutnya *web server* akan mencarikan berkas yang diminta dan memberikan datanya pada *web browser*. *Web browser* yang mendapatkan data dari *web server* segera melakukan proses penerjemahan kode HTML dan menampilkannya ke layar pengguna. (Kadir, 2008:4-5)

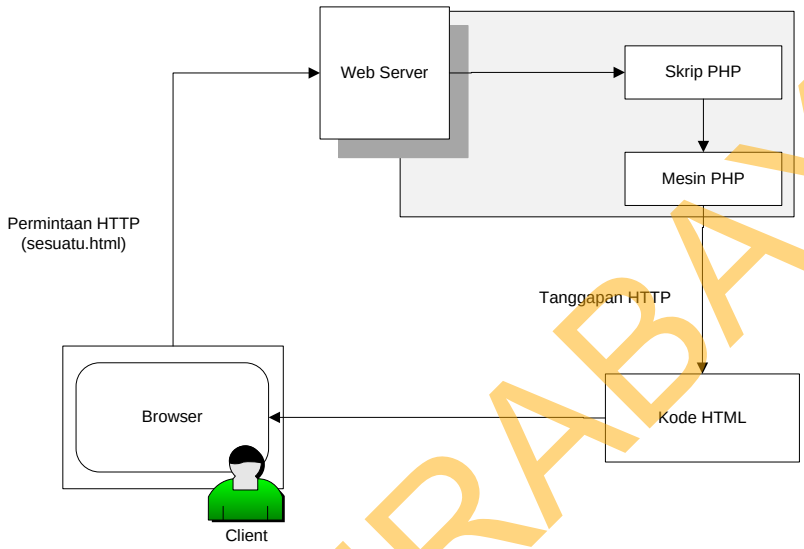


Gambar 2.1 Skema HTML

(Sumber: Kadir, 2008:5)

Jika halaman yang diminta adalah sebuah halaman PHP maka prinsipnya serupa dengan kode HTML. Hanya saja ketika berkas PHP yang diminta didapatkan oleh *web server*, datanya akan segera dikirimkan kepada mesin PHP

dan mesin inilah yang memproses dan memberikan hasilnya (berupa kode HTML) kepada *web server*. Selanjutnya *web server* menyampaikannya kepada *client*. (Kadir, 2008:6)



Gambar 2.2 Skema PHP
(Sumber: Kadir, 2008:6)

2.8.1 PHP dan *Database*

Salah satu kelebihan dari PHP adalah kemampuan untuk berkomunikasi dengan berbagai *database* terkenal. Dengan demikian, menampilkan data yang bersifat dinamis yang diambil dari *database* merupakan hal yang mudah untuk diimplementasikan. Itulah sebabnya sering dikatakan bahwa PHP sangat cocok untuk membangun halaman-halaman *web* dimanis.

Pada saat ini PHP sudah dapat berkomunikasi dengan berbagai *database* meskipun dengan kelengkapan yang berbeda-beda. Beberapa jenis database yang dapat terhubung dengan PHP di antaranya adalah (Kadir, 2008:6-7):

1. *Base*
2. DBM

3. *FilePro* (*Personic, Inc.*)
4. *Informix*
5. *Ingres*
6. *InterBase*
7. *Microsoft Access*
8. MSSQL
9. MySQL
10. *Oracle*
11. *PostgreeSQL*
12. *Sybase*

2.9 Model Arus Jaringan

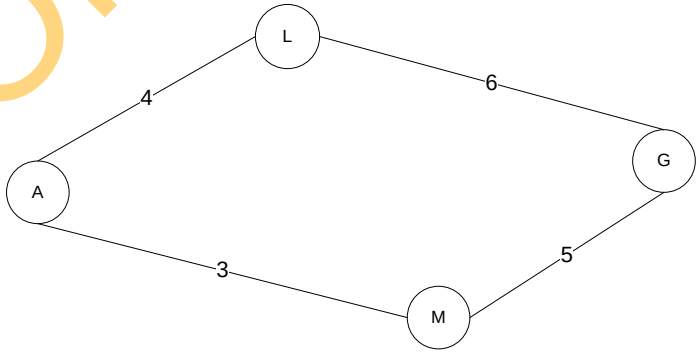
Suatu jaringan (*network*) adalah susunan garis edar (*path*) yang terhubung pada berbagai titik, dimana satu atau beberapa barang bergerak dari satu titik ke titik lain. Setiap orang telah mengenal jaringan seperti sistem jalan tol, jaringan telepon, jaringan rel kereta api, dan jaringan televisi. Misalnya, suatu jaringan rel kereta api terdiri dari sejumlah rute (garis edar) yang dihubungkan oleh stasiun-stasiun pada pertemuan di berbagai rute tersebut.

Dalam beberapa tahun terakhir ini model jaringan telah teknik ilmu manajemen untuk analisis yang sangat populer karena beberapa alasan penting. Pertama, suatu jaringan digambarkan sebagai diagram yang benar-benar memberikan gambaran mengenai suatu sistem yang sedang dianalisis. Hal ini memudahkan manager untuk menginterprestasikan sistem tersebut secara visual, sehingga membantu manager untuk dapat lebih memahami. Kedua, sejumlah

besar sistem dalam kehidupan sehari-hari dapat diperagakan menjadi suatu jaringan, yang relatif mudah untuk dipahami dan dibuat. Model arus jaringan tersebut digunakan untuk menganalisa tiga jenis masalah: masalah rute terpendek, masalah pohon rentang minimal, dan masalah arus maksimal.

2.9.1 Komponen-komponen Jaringan

Jaringan diilustrasikan sebagai diagram yang terdiri dua komponen penting yaitu, simpul (*nodes*) dan cabang (*branches*). Simpul melambangkan titik-titik persimpangan, contoh: persimpangan jalan. Cabang menghubungkan simpul-simpul tersebut dan mencerminkan arus satu titik ke titik lain dalam jaringan tersebut. Simpul-simpul dalam jaringan dilambangkan dengan lingkaran, dan cabang dilambangkan dengan garis yang menghubungkan simpul-simpul tersebut, seperti jalan yang menghubungkan kota dan persimpangan, serta jalan kereta api atau rute udara yang menghubungkan stasiun-stasiun. Sebagai contoh, rute jalan kereta api yang berbeda antara titik A, titik G dan titik L , titik M, serta stasiun-stasiun lainnya. Seperti pada Gambar 2.3



Gambar 2.3 Jaringan Rute

Jaringan yang ditunjukkan dalam Gambar 1.1 memiliki 4 simpul dan 4 cabang. Simpul yang melambangkan titik A disebut titik awal dan 3 simpul

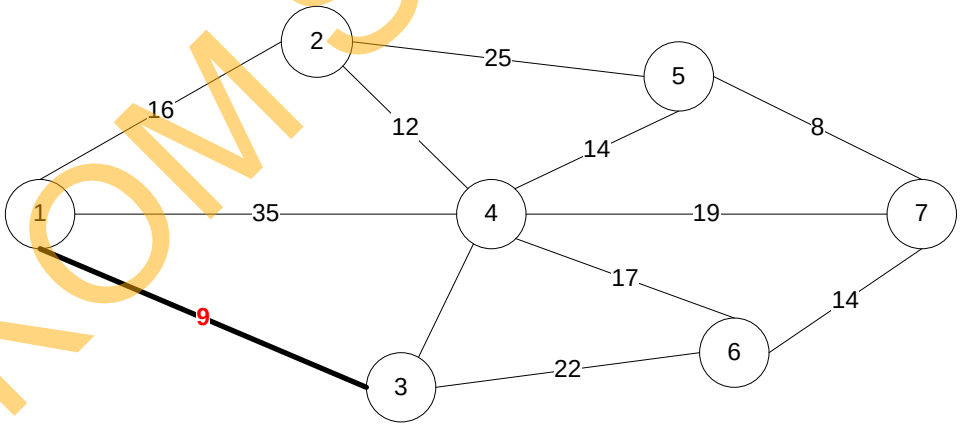
sisannya dapat merupakan tujuan, tergantung dari apa yang ingin kita tentukan dari jaringan tersebut. Perhatikan bahwa pada masing-masing simpul telah diberikan suatu angka. Angka-angka tersebut memberikan sarana yang lebih baik daripada nama untuk mengidentifikasi simpul dan cabang tersebut. Sebagi contoh, kita anggap titik A adalah titik awal sebagai simpul 1 dan cabang dari titik A ke L sebagai cabang 1-2.

Umumnya, suatu nilai yang melambangkan jarak, lamanya waktu, atau biaya diberikan pada masing-masing cabang. Maka, tujuan dari jaringan adalah untuk menentukan jarak terpendek, waktu tersingkat, atau biaya terendah di antara titik dalam jaringan. Dalam Gambar 2.3, nilai 4,6,3, dan 5 berhubungan dengan empat cabang yang melambangkan lamanya waktu dalam jam di antara simpul-simpul yang terkait. Maka, seorang pelancong dapat melihat bahwa rute ke L melalui G membutuhkan waktu 10 jam dan rute dari L melalui M membutuhkan waktu 8 jam.

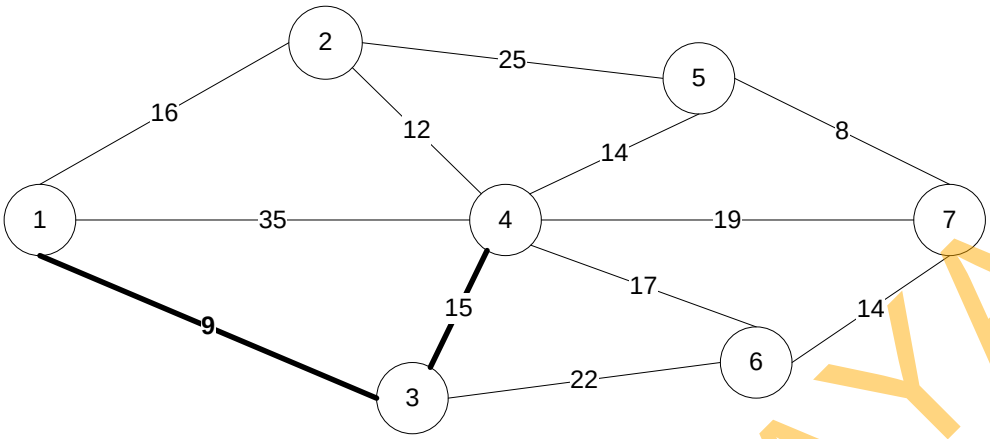
2.9.2 Pohon Rentang Minimal (Spanning Tree)

Minimum spanning tree merupakan teknik yang digunakan untuk mencari jalur terpendek dari sebuah lintasan, dengan kata lain adalah teknik yang digunakan untuk mencari solusi membangun sebuah jaringan agar tidak memakan banyak jalur seperti pemasangan kabel, rute penerbangan serta penugasan. Masalah pohon rentang minimal bertujuan untuk menghubungkan seluruh simpul dalam jaringan sehingga total panjang cabang tersebut diminimisasi. Jaringan yang dihasilkan merentangkan (menghubungkan) semua titik dalam jaringan tersebut pada total jarak (panjang) minimal.

Pendekatan solusi untuk masalah pohon rentang minimal sebenarnya lebih mudah daripada metode solusi rute terpendek. Dalam pendekatan solusi pohon rentang minimal, dapat dimulai dari simpul manapun. Walaupun demikian, pendekatan konvensional memulainya dari simpul 1. Dengan memulai dari simpul 1, kita memilih simpul terdekat (cabang terpendek) untuk bergabung dengan pohon rentang kita. Cabang terpendek dari simpul 1 adalah simpul 3, dengan panjang 9. Cabang ini ditandai dengan garis tebal dalam Gambar 2.4. Sekarang pohon rentang tersebut terdiri dari dua simpul: 1 dan 3. Langkah selanjutnya adalah memilih simpul terdekat yang belum berada dalam pohon rentang. Simpul terdekat ke simpul 1 ataupun simpul 3 (simpul-simpul dalam pohon rentang kita) adalah simpul 4, dengan panjang cabang 15. Tambahan simpul 4 dalam pohon rentang kita ditunjukkan dalam Gambar 2.5.

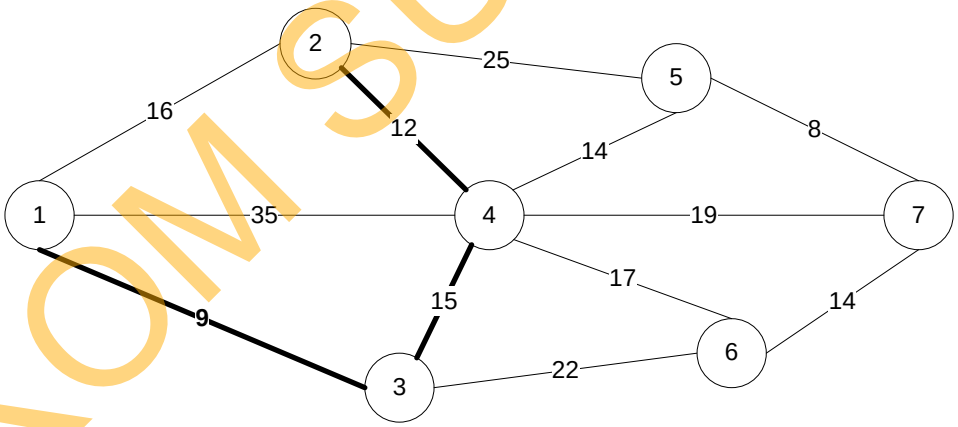


Gambar 2.4 Pohon Rentang dengan simpul 1 dan 3



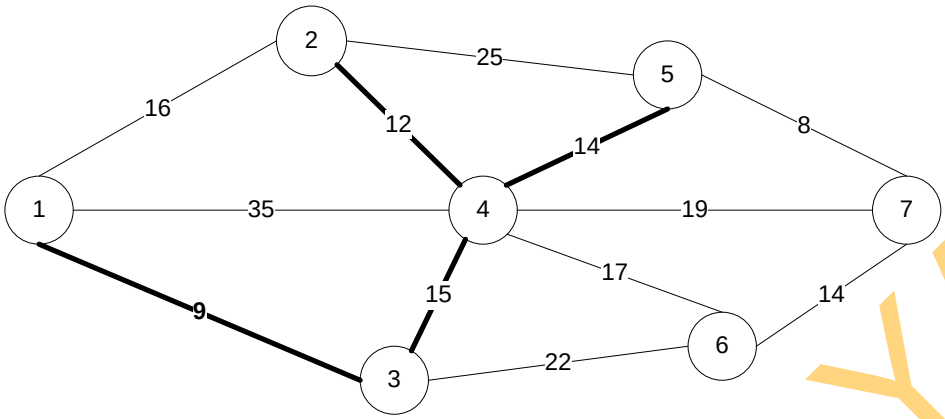
Gambar 2.5 Pohon Rentang dengan simpul 1, 3, dan 4

Kemudian kita ulangi proses pemilihan simpul terdekat dengan pohon rentang kita (simpul 1, 3, dan 4). Simpul terdekat yang belum dihubungkan dengan simpul 4 ke simpul 2 adalah 12. Tambahan simpul 2 pada pohon rentang kita ditunjukkan dalam Gambar 2.6



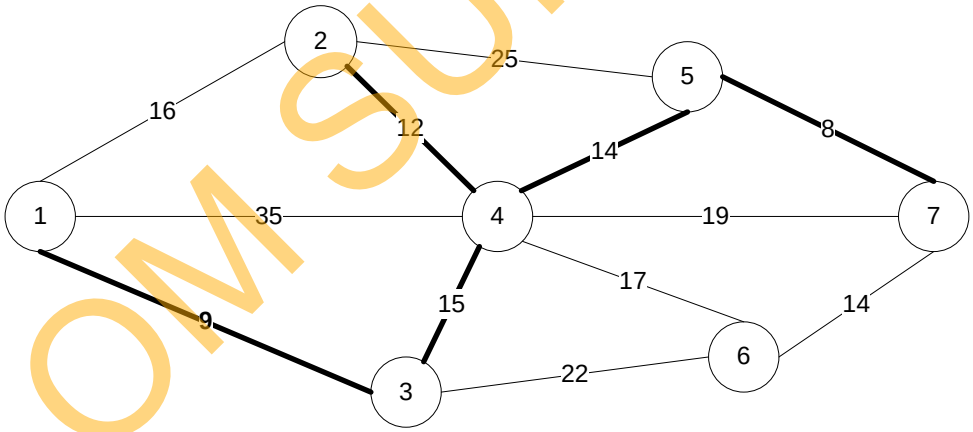
Gambar 2.6 Pohon Rentang dengan Simpul 1,3,4 dan 2

Pohon rentang kita sekarang terdiri dari simpul 1, 2, 3, dan 4. Simpul terdekat dengan pohon rentang ini adalah simpul 5, dengan pangjang cabang 14 ke simpul 4. Maka, simpul 5 bergabung dengan pohon rentang kita, seperti ditunjukkan dalam Gambar 2.7



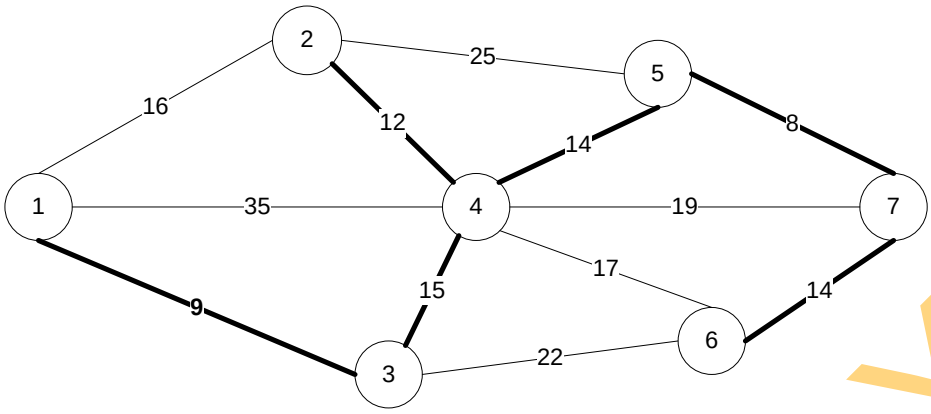
Gambar 2.7 Pohon Rentang dengan Simpul 1,2,3,4 dan 5

Sekarang pohon rentang tersebut terdiri dari simpul 1,2,3,4 dan 5. Simpul terdekat yang belum dihubungkan dengan pohon rentang adalah simpul 7. Cabang yang menghubungkan simpul 7 ke simpul 5 memiliki panjang 8. Gambar 2.8 menunjukkan tambahan simpul 7 ke pohon rentang.



Gambar 2.8 Pohon Rentang dengan Simpul 1,2,3,4, 5 dan 7

Sekarang pohon rentang kita mencakup simpul 1,2,3,4,5 dan 7. Satu-satunya simpul yang tersisa yang belum dihubungkan dengan pohon rentang adalah simpul 6. Simpul dalam pohon rentang yang terdekat dengan simpul 6 adalah simpul 7, dengan panjang cabang 14. Pohon rentang yang lengkap, yang mencakup seluruh tujuh simpul ditunjukkan dalam gambar 2.9



Gambar 2.9 Pohon Rentang Minimal

Pohon rentang yang ditunjukkan dalam Gambar 2.9 membutuhkan jumlah minimal kabel televisi untuk menghubungkan tujuh kota adalah 72. Pohon rentang minimal yang sama juga dapat diperoleh dengan memulai proses ini pada salah satu enam simpul selain simpul 1.

2.9.3 Langkah-langkah metode solusi pohon rentang minimal

Langkah-langkah metode solusi pohon rentang minimal adalah sebagai berikut :

1. Tentukan simpul awal manapun
2. Tentukan simpul yang terdekat dengan simpul awal untuk bergabung dengan pohon rentang.
3. Tentukan simpul terdekat yang belum termasuk pohon rentang.

Ulangi langkah 3 sampai seluruh simpul telah bergabung pohon rentang.