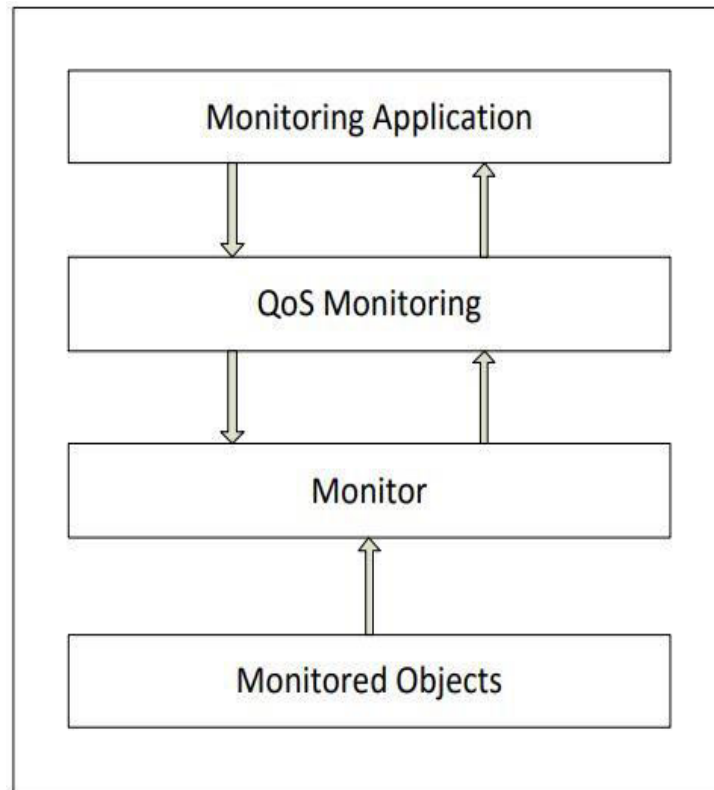


BAB II LANDASAN TEORI

2.1. *QoS (Quality of Service)*

Quality of Service (QoS) merupakan metode pengukuran tentang seberapa baik jaringan dan merupakan suatu usaha untuk mendefinisikan karakteristik dan sifat dari satu servis. QoS digunakan untuk mengukur sekumpulan atribut kinerja yang telah dispesifikasikan dan diasosiasikan dengan suatu servis. Pada penelitian ini menggunakan Model Monitoring QoS pada Gambar 1.



Gambar 2.1 Model Monitoring Quality of Service

Model Monitoring QoS terdiri dari komponen *monitoring application*, *QoS monitoring*, *monitor*, dan *monitored objects*.

1. Monitoring Application

Merupakan sebuah antarmuka bagi administrator jaringan. Komponen ini berfungsi mengambil informasi lalu lintas paket data dari monitor,

menganalisanya dan mengirimkan hasil analisis kepada pengguna. Berdasarkan hasil analisis tersebut, seorang administrator jaringan dapat melakukan operasi-operasi yang lain.

2. QoS Monitoring

Menyediakan mekanisme monitoring QoS dengan mengambil informasi nilai-nilai parameter QoS dari lalu lintas paket data.

3. Monitor

Mengumpulkan dan merekam informasi lalu lintas paket data yang selanjutnya akan dikirimkan kepada monitoring application. Monitor melakukan pengukuran aliran paket data secara waktu nyata dan melaporkan hasilnya kepada monitoring application.

4. Monitored Objects

Merupakan informasi seperti atribut dan aktifitas yang dimonitor di dalam jaringan. Di dalam konteks QoS monitoring, informasi-informasi tersebut merupakan aliran-aliran paket data yang dimonitor secara waktu nyata. Tipe aliran paket data tersebut dapat diketahui dari alamat sumber (source) dan tujuan (destination) di layer-layer IP, port yang dipergunakan misalnya UDP atau TCP, dan parameter di dalam paket RTP

2.2. Parameter-parameter *QoS (Quality of Service)*

Parameter Quality of Service berdasarkan TIPHON (*Telecommunications and Internet Protocol Harmonization Over Network*) terdiri dari:

- *Throughput*

Throughput adalah kecepatan (rate) transfer data efektif, yang diukur dalam bps (bit per second). *Throughput* merupakan jumlah total kedatangan paket yang sukses diterima yang diamati pada tujuan selama satu interval waktu tertentu dibagi dengan durasi interval waktu tersebut [4]. Kategori *Throughput* ditunjukkan pada tabel 2.1.

Tabel 2. 1. Kategori Throughput

Kategori <i>Throughput</i>	<i>Throughput</i> (bps)	Indeks
Sangat Bagus	100 bps	4
Bagus	75 bps	3
Sedang	50 bps	2
Jelek	< 25 bps	1

- *Packet Loss*

Packet Loss adalah parameter yang menggambarkan suatu kondisi yang menunjukkan jumlah total paket yang hilang dapat terjadi karena collision dan congestion pada jaringan[4]. Indeks dan kategori packet loss ditunjukkan pada Tabel 2.2.

Tabel 2. 2. Kategori Packet Loss

Kategori <i>Packet Loss</i>	<i>Packet loss</i> (%)	Indeks
Sangat Bagus	0 %	4
Bagus	3 %	3
Sedang	15 %	2
Jelek	25 %	1

- *Latency*

Delay (*Latency*) merupakan waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. Delay dapat dipengaruhi oleh jarak, media fisik, congesti atau juga waktu proses yang lama[4]. Pada Tabel 2.3 diperlihatkan kategori dari delay dan besar delay.

Tabel 2. 3. Kategori Latency

Kategori <i>Latency</i>	<i>Latency</i> (ms)	Indeks
Sangat Bagus	< 150 ms	4
Bagus	150 ms s/d 300 ms	3
Sedang	300 ms s/d 450 ms	2
Jelek	> 450 ms	1

- *Jitter*

Jitter diakibatkan oleh variasi-variasi dalam panjang antrian, dalam waktu pengolahan data, dan juga dalam waktu penghimpunan ulang paket-paket diakhir perjalanan jitter. Jitter lazimnya disebut variasi delay, berhubungan erat dengan latency, yang menunjukkan banyaknya variasi delay pada transmisi data di jaringan yang diperlihatkan pada Tabel 2. 4[4].

Tabel 2. 4. Kategori *Jitter*

Kategori <i>Jitter</i>	<i>Jitter</i> (ms)	Indeks
Sangat Bagus	0 ms	4
Bagus	0 ms s/d 75ms	3
Sedang	75 ms s/d 125 ms	2
Jelek	125 ms s/d 225 ms	1

- Indeks *QoS*

Indeks ini dihasilkan dari rata-rata dalam seluruh parameter seperti *packet loss*, *throughput*, *delay* dan *jitter*. Nilai rata-rata tiap parameter ini akan dijumlahkan dan dibagi sebanyak jumlah parameter sehingga mendapat nilai *QoS* secara keseluruhan. Standar nilai *QoS* diperlihatkan pada Tabel 2. 5[4].

Tabel 2. 5. Index *QoS*

Indeks	Persentase (%)	Nilai
Sangat Memuaskan	95 - 100	3,8 - 4
Memuaskan	75 – 94,75	3 – 3,79
Kurang Memuaskan	50 – 74,75	2 – 2,99
Jelek	25 – 49,75	1 – 1,99

2.3. Topologi Sistem *E-Health Monitoring*

Topologi jaringan yang dapat dipergunakan pada sistem *E-Health Monitoring* menggunakan topologi jaringan komunikasi *Wireless Sensor Network* (WSN) yang terdiri dari beberapa jenis, diantaranya adalah sebagai berikut:

- Topologi Star
- Topologi Cluster / Tree
- Topologi Mesh

Apabila dibandingkan, tiga topologi WSN (star, cluster-tree dan mesh) dalam hal analisis kinerja *Throughput*, end-to-end delay dan sejumlah *hop* dengan membuat dua skenario berbeda adalah subjek dari penelitian [Mihajlov dan Bogdanoski, 2011]. Alasan utama untuk hal ini adalah bahwa dalam jaringan dengan satu koordinator, perangkat akhir hanya berkomunikasi dengan koordinator dan mereka hanya menggunakan satu jalur untuk mencapai tujuan, sehingga menghindari tabrakan antar paket, dan koordinator tidak menerima paket sampai paket yang tertunda tidak sepenuhnya ditransmisikan. *Throughput* tertinggi terjadi pada topologi pohon dan terendah pada topologi Mesh. Bagian kedua dari studi simulasi didedikasikan untuk perbandingan topologi Tree dengan satu *Personal Area Network* (PAN) dan topologi Pohon yang serupa dengan dua PAN (tambahan koordinator ditambahkan). Hasil penelitian menunjukkan bahwa keterlambatan jaringan ujung ke ujung dengan PAN tunggal lebih tinggi daripada penundaan ujung ke ujung PAN pertama atau PAN kedua dalam jaringan dengan dua PAN [5].

Besarnya nilai *hop* mengacu pada jumlah perangkat perantara di mana data harus melewati antara sumber dan tujuan. Para peneliti dalam skenario pertama menggunakan jaringan topologi dengan satu koordinator dan dalam skenario kedua mereka menggunakan dua koordinator. Hasil mereka menunjukkan bahwa keterlambatan *end-to-end* dalam topologi pohon lebih tinggi sebesar 50% dari bintang dan mesh di mana jaringan memiliki satu koordinator. Juga, topologi pohon memiliki *Throughput* maksimum dan mesh dengan *Throughput* terendah[6]. Mereka beralasan bahwa dengan fakta bahwa jaringan pohon menggunakan router selain koordinator di mana beban dibagi antara koordinator dan router yang menghasilkan lebih sedikit tabrakan dan lebih sedikit kehilangan paket.

2.4. Network Simulator 3

NS-3 adalah program simulasi jaringan yang open source, yang umumnya banyak digunakan oleh para peneliti, akademisi, dan praktisi. NS-3 dibuat dengan menggunakan bahasa pemrograman C++ dan juga menggunakan script python [7]. Proses umum pembuatan simulasi pada Network Simulator 3 dapat dibagi menjadi beberapa langkah:

1. Definisi topologi: Untuk memudahkan pembuatan fasilitas dasar dan mendefinisikan keterkaitannya, ns-3 memiliki sistem wadah dan pembantu yang memfasilitasi proses ini.
2. Pengembangan model: Model ditambahkan ke simulasi (misalnya, UDP, IPv4, perangkat dan tautan titik-ke-titik, aplikasi); sebagian besar waktu ini dilakukan dengan bantuan.
3. Konfigurasi node dan link: model menetapkan nilai defaultnya (misalnya, ukuran paket yang dikirim oleh aplikasi atau MTU link point-to-point); sebagian besar waktu ini dilakukan dengan menggunakan sistem atribut.
4. Eksekusi: Fasilitas simulasi menghasilkan peristiwa, data yang diminta oleh pengguna dicatat.
5. Analisis kinerja: Setelah simulasi selesai dan data tersedia sebagai jejak peristiwa yang dicap waktu. Data ini kemudian dapat dianalisis secara statistik dengan alat seperti R untuk menarik kesimpulan. Visualisasi Grafis: Data mentah atau olahan yang dikumpulkan dalam simulasi dapat dibuat grafiknya menggunakan alat seperti Gnuplot, matplotlib, atau XGRAPH.

2.4.1. Prosedur Dasar Simulasi Network Simulator 3



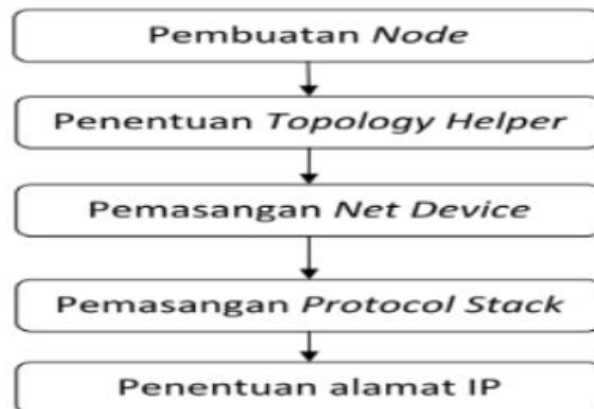
Gambar 2. 2. Dasar Prosedur Simulasi NS-3

Gambar diatas menunjukkan alur dasar prosedur dalam pembuatan simulasi jaringan dengan menggunakan NS-3. Berikut ini deskripsi dari masing-masing tahapnya.

2.4.2. Mengaktifkan Logging

Langkah ini bertujuan untuk mengaktifkan fitur logging yang ada di NS-3. Aktifasi logging file bertujuan agar data hasil simulasi dapat ditampilkan pada konsol Linux. Data ini juga dapat dicatat pada logging file packet capture yang dapat dibaca oleh aplikasi Wireshark untuk melakukan trace pada paket yang mengalir dalam simulasi yang dibuat.

2.4.3. Merancang Topologi Jaringan



Gambar 2. 3. . Dasar Perancangan Topologi Jaringan NS-3

Pembuatan Node pada NS-3 Dev merupakan abstraksi dari end-system atau lebih sering dikenal dengan host pada suatu jaringan komputer. Abstraksi ini diwakili dalam C++ oleh kelas node yang menyediakan metode untuk mengelola representasi perangkat komputasi di simulasi. Node merepresentasikan komputer yang akan ditambahkan sesuatu seperti protokol, aplikasi, dan peripheral card. Topology helper NodeContainer menyediakan sebuah cara yang mudah untuk membuat, mengontrol, dan mengakses Node apapun yang telah dibuat untuk dapat menjalankan sebuah simulasi. Dalam sebuah jaringan simulasi besar akan diperlukan banyak koneksi untuk mengatur node, Net Device serta channel. NS-3 Dev menyediakan apa yang disebut objek topology helpers untuk mengatur simulasi-simulasi jaringan semudah mungkin. Dalam membuat topologi point-to-point, topology helper PointToPointHelper diperlukan untuk membuat jalur untuk menghubungkan kedua node. Istilah yang akan digunakan dalam hal ini adalah Net Device dan Channel. Dalam dunia nyata, istilah tersebut sesuai dengan peripheral card dan kabel jaringan.

Oleh karena itu, pada pembuatan jalur point-to-point akan menggunakan PointToPointHelper untuk mengkonfigurasi dan menghubungkan objek NS-3 PointToPointNetDevice dan PointToPointChannel. Untuk terhubung dengan jaringan, komputer harus memiliki perangkat keras yang disebut dengan peripheral card. Peripheral card tersebut diimplementasikan beberapa fungsi jaringan, sehingga disebut Network Interface Cards (NICs). NIC tidak akan berfungsi tanpa sebuah software driver untuk mengontrol perangkat keras tersebut. Pada Unix (atau Linux), sebuah peripheral hardware disebut sebagai device.

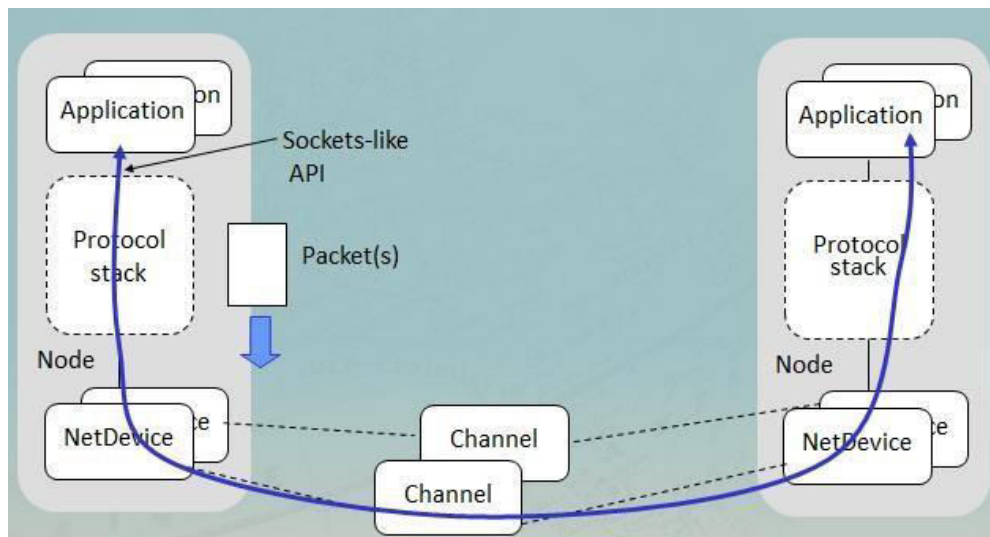
Untuk menampung objek NetDevice yang akan dibuat, digunakanlah NetDeviceContainer, seperti halnya pada NodeContainer untuk menampung node yang telah dibuat. Kode di bawah ini akan menyelesaikan pengkonfigurasi device dan channel. Setelah node dan device terkonfigurasi dengan baik, langkah berikutnya yaitu memasang protokol pada node yang telah dibuat. Kode berikut berfungsi untuk pemasangan protokol tersebut. InternetStackHelper stack; stack.Install (nodes); InternetStackHelper adalah sebuah topology helper yang berfungsi untuk memasangkan protokol Internet pada point-to-point net device. Metode Install memiliki parameter yakni NodeContainer. Ketika dijalankan, akan diinstall stack Internet

seperti TCP (Transmission Control Protocol), UDP (User Datagram Protocol), IP (Internet Protocol), dan sebagainya pada setiap node yang ada dalam NodeContainer.

2.4.4. Penentuan Alamat IP

Berikutnya node yang telah dibuat akan dipasangkan alamat IP untuk dapat berkomunikasi antar node satu sama lain. Topology helper IPv4AddressHelper ini berfungsi untuk mengatur pengalokasian dari alamat IP. Pada proses pendeklarasian IPv4AddressHelper mendeklarasikan sebuah objek address helper yang memberitahukan agar alamat IP yang dialokasikan adalah dengan jaringan 10.1.1.0 dan menggunakan subnet mask 255.255.255.0. Secara default alamat yang dialokasikan akan dimulai dari satu dan akan bertambah secara statik, maka alamat pertama yang dialokasikan dalam hal ini adalah 10.1.1.1, diikuti dengan 10.1.1.2, dan seterusnya.

2.4.5. Dasar Model Simulasi NS-3.



Gambar 2.4. Dasar Model Simulasi NS-3

Gambar 2.4 menunjukkan dasar dari model simulasi dengan menggunakan NS-3. Semua simulasi jaringan yang dibuat dengan menggunakan NS-3 mengikuti alur model tersebut. Dimana paket yang dihasilkan oleh Application akan melewati berbagai susunan protokol sebelum dikirimkan melalui Channel yang merupakan sebuah media yang menjadi tempat mengalirnya data dalam suatu jaringan. Dalam NS-3 Dev abstraksi komunikasi dasar subnetwork disebut channel dan diwakili di C++ oleh kelas channel, oleh NetDevice. Application, Protocol stack, dan 19 NetDevice tersebut terdapat dalam sebuah Node

yang telah dibuat. Lalu paket tersebut akan dikirimkan ke node yang dituju, yang pertama-tama akan diterima oleh NetDevice node yang dituju, kemudian melewati lapisan protokol, dan akan dibaca dan ditampilkan isi dari paket tersebut oleh application juga. Channel berfungsi sebagai media perantara yang menjembatani antara node yang satu dengan yang lain.