II. TINJAUAN PUSTAKA

2.1 Jaringan Sensor Nirkabel

Jaringan Sensor Nirkabel (JSN) merupakan sebuah jaringan yang disusun oleh sensor–sensor yang terdistribusi dalam suatu cakupan area tertentu yang dihubungkan melalui kanal komunikasi nirkabel untuk saling bekarja sama melakukan pengukuran dan pemantauan fenomena fisik seperti temperatur, suara, getaran, tekanan, tekanan atau kodisi – kondisi fisik tertentu [4]. Secara umum JSN terdiri dari target atau fenomena fisik yang akan di sensor, nodal sensor yang melakukan sensing fenomena fisik dan nodal koordinator/*gateway* yang bertanggung jawab mengatur jaringan dan mengumpulkan data dari nodal sensor [4,5]. Nodal sebuah JSN bersifat individu untuk melakukan *sensing*, *controlling* dan *communication* terhadap parameter–parameter fisik. Nodal sensor dikomposisikan dari beberapa modul seperti *embeded processor,* memori, sensor dan perangkat *Radio Frequency* (RF) *transceiver* dengan konsumsi energi yang minimum[4].

Terdapat berbagai aplikasi JSN yang merupakan pemantauan, pencarian jejak (*tracking*), dan pengendalian (*controlling*) [4]. Pada bidang militer JSN digunakan untuk pengawasan dan pengintaian di medan perang, pada pabrik-pabrik JSN digunakan untuk melakukan pemeliharaan perangkat, pada bangunan-bangunan

JSN digunakan untuk melakukan pemantauan keadaan infrastruktur, pada rumah tinggal JSN digunakan untuk menciptakan sebuah rumah cerdas (*smart home*) serta pada tubuh manusia JSN digunakan dalam melakukan pemantauan tubuh pasien. Pemantauan suatu area merupakan salah satu aplikasi JSN. Pada aplikasi ini, sensor-sensor disebarkan pada suatu area untuk memantau suatu fenomena fisik tertentu. Sebagai contoh, sejumlah nodal sensor disebarkan pada medan perang untuk mendeteksi posisi musuh. Ketika sensor-sensor tersebut mendeteksi timbulnya gejala fisik yang menjadi objek pantau (panas, tekanan, suara, cahaya, medan elektromagnetik, getaran, dan sebagainya), hasil deteksi ini dilaporkan ke sebuah *gateway* yang menjadi titik pengumpulan data dari JSN. Selanjutnya, data pemantauan pada *gateway* akan digunakan oleh administrator jaringan.

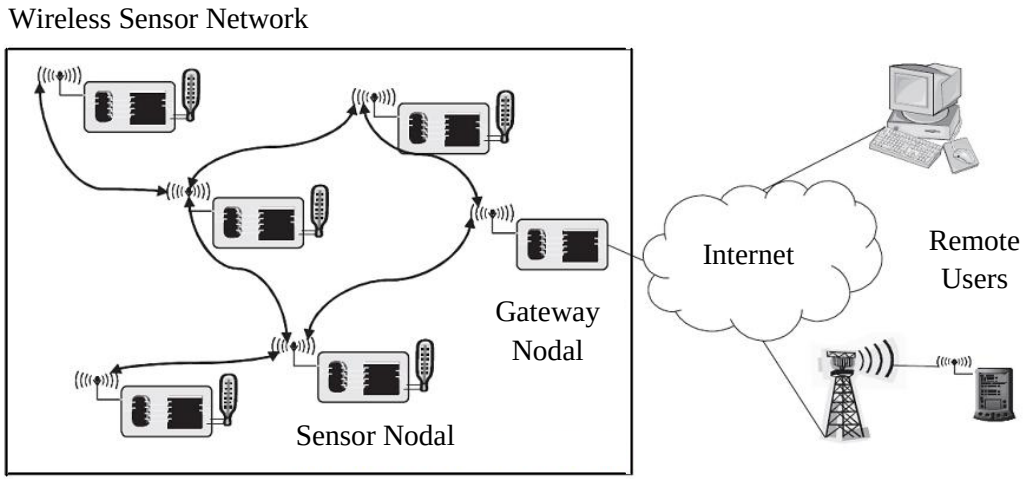
Untuk penelitian ini, simulasi dilakukan dengan batasan parameter luas bidang 500 m x 500 m. Dimana posisi nodal sensor akan diatur secara teratur berdasarkan banyaknya nodal sensor yang disimulasikan. Banyaknya nodal sensor yang digunakan adalah 4, 16, 25, 49, 64, 100, dan 144. Pada penelitian ini juga mengatur interval penyensoran, yaitu 0,5 detik, 1 detik dan 2 detik. Selain itu, simulasi ini juga mengatur ketinggian nodal sensor secara acak. Ketinggian nodal sensor yang diatur akan dimulai dari 0 meter hingga 10 meter. Dengan dua variasi ketinggian, yaitu 0-5 meter dan 0-10 meter. Hal tersebut dilakukan untuk menggambarkan keadaan secara nyata saat melakukan penyensoran di alam terbuka. Dimana terdapat berbagai kemungkinan keadaan lingkungan yang tidak selalu rata. Pada penelitian ini juga akan mengamati perubahan yang terjadi pada

throughput, delay, jitter dan *packet loss* pada jaringan sensor nirkabel. Serta akan membandingkan hasilnya dengan penelitian sebelumnya.

Pada penelitian sebelumnya, parameter yang digunakan untuk luas bidang adalah sama, yaitu 500 m x 500 m dengan pertambahan nodal sensor, posisi nodal sensor dan interval waktu yang sama. Hal yang membedakan penelitian ini dengan penelitian sebelumnya adalah variasi ketinggian nodal sensor.

2.2 Komponen – komponen Jaringan Sensor Nirkabel

JSN terdiri dari empat komponen yaitu nodal sensor, media nirkabel, nodal koordinator/*gateway* dan PC *server/administrator* seperti pada Gambar 2.1. di bawah ini.



Gambar 2.1. Komponen dasar JSN [4]

2.2.1 Nodal sensor

Nodal sensor merupakan seperangkat *device* pada JSN yang bertugas melakukan pemantauan atau penyensoran terhadap suatu fenomena fisik tertentu, melakukan

pemrosesan terhadap data yang diperoleh dari fenomena fisik tersebut, dan mengirimkan data yang diperoleh tersebut kepada koordinator/*gateway*. Sebuah nodal sensor harus memenuhi kebutuhan minimum agar dapat berfungsi dengan baik pada JSN. Kebutuhan yang harus dipenuhi setiap nodal sensor ini berdasarkan kebutuhan masing-masing dari tiap aplikasi JSN dimana nodal tersebut diimplementasikan. Secara umum nodal sensor harus berukuran kecil, murah, efisien dalam mengkomsumsi energi, dilengkapi dengan perangkat sensor yang tepat untuk suatu pemantauan, mampu melakukan komputasi yang dibutuhkan dan memiliki sumber daya penyimpanan (memori), serta memiliki fasilitas komunikasi yang memadai.

Sebuah nodal sonsor tersusun dari lima komponen sebagai berikut [4] :

a. Controller

Controller meruapkan sebuah pengendali untuk memproses seluruh data.

b. Memory

Memori merupakan media penyimpanan program-program dan data hasil pemrosesan. Pada umumnya tipe memori yang berbeda digunakan untuk menyimpan program dan data.

c. Sensor atau aktuator

Sensor merupakan antarmuka antara nodal sensor dengan lingkungan fisik, perangkat ini dapat mengamati parameter fisik dari lingkungan.

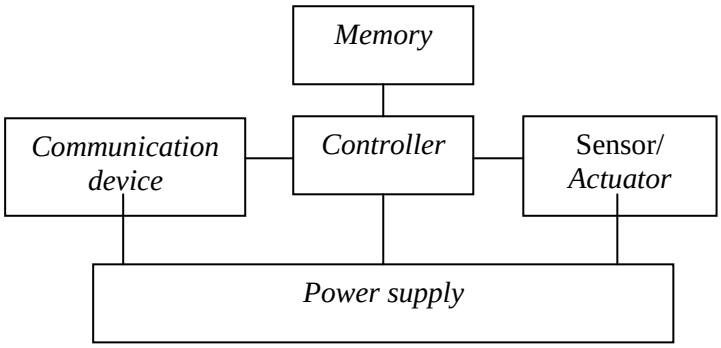
d. Communication Device

Komponen ini merupakan media untuk mengirimkan dan menerima informasi, yaitu melalui kanal nirkabel.

e. *Power Supply*

Komponen ini merupakan perangkat yang menyediakan sumber daya untuk kebutuhan elektrifikasi nodal sensor, sebagian besar nodal sensor menngunakan baterai sebagai sumber daya.

Seperti ditunjukkan pada Gambar 2.2.



Gambar 2.2. Komponen perangkat keras nodal sensor [4]

2.2.2 Media nirkabel

Media nirkabel merupakan *unguided medium* yang berarti bahwa perambatan sinyal tidak terbatas pada lokasi atau saluran yang sudah ditentukan sebagaimana ditentukan pada transmisi menggunakan kabel berpelindung tertentu. Pada media nirkabel gelombang elektromagnetik merambat dalam ruang bebas antara pemancar dan penerima, dimana sinyal merambat bukan pada medium terbatas sebagaimana pada media transmisi kabel. Jaringan nirkabel menggunakan radio sebagai media transmisi informasinya. Radio adalah teknologi untuk pengiriman sinyal dengan cara modulasi dan radiasi elektromagnetik (gelombang elektromagnetik). Media nirkabel memiliki tingkat atenuasi sinyal yang lebih tinggi dibandingkan dengan media kabel. Kualitas kanal aktual tergantung pada beberapa faktor termasuk frekuensi, jarak antara pengirim dan penerima,

kecepatan relatifnya, lingkungan propagasi (jumlah saluran atau path yang dilewati dan pelemahannya), dan lain-lain [6,7].

2.2.3 Nodal koordinator/*gateway*

Pada implentasinya JSN tidak cukup jika hanya dapat berinteraksi dengan dirinya sendiri. JSN harus mampu berinteraksi dengan perangkat informasi lainnya seperti PDA maupun perangkat komputer pengguna yang memberikan permintaan data dari JSN. Selain dapat melakukan pertukaran data pada perangkat tersebut, JSN juga harus mampu melakukan pertukaran data dengan perangkat internet.

2.2.4 PC *server/administrator*

PC *server/administrator* merupakan perangkat pengguna yang melakukan permintaan terhadap data hasil pemantauan oleh JSN, perangkat ini dapat berupa komputer yang terhubung secara langsung dengan koordinator/*gateway*.

2.3 *Quality of Service* (QoS) Jaringan Sensor Nirkabel (JSN)

Quality of Service (QoS) menunjukkan kemampuan sebuah jaringan untuk menyediakan layanan yang lebih baik lagi bagi layanan trafik yang melewatinya. *Quality of Services* (QoS) suatu jaringan merujuk ke tingkat kecepatan dan keandalan penyampaian berbagai jenis beban data di dalam suatu komunikasi. Kemampuan JSN dalam menyediakan suatu penanganan terhadap permintaan data

hasil pemantauannya menentukan tingkat performansi JSN, tingkat performansi JSN bergantung pada bidang implementasinya. Mayoritas pengembangan JSN ditujukan untuk melakukan pemantauan terhadap fenomena fisik seperti temperatur, tekanan, kelambaban, atau lokasi dari suatu objek. Untuk aplikasi tersebut, mayoritas JSN didesain hanya untuk mendapatkan hasil berupa data (nilai hasil perhitungan dari pemantauan) dengan *delay* yang dapat ditoleransi dan menggunakan ukuran *bandwidth* minimal atau kecil. Tingkat performansi JSN pada implementasinya tersebut akan berbeda dengan tingkat performansi yang dibutuhkan untuk aplikasi JSN pada implementasi teknologi yang digunakan untuk mendapatkan tipe data yang berbeda. Untuk mengetahui *Quality of Services* (QoS) suatu JSN, perlu dilakukan kuantifikasi terhadap beberapa metrik parameter yang dapat mewakili QoS JSN. Beberapa parameter tersebut yaitu *throughput*, *delay*, *jitter* dan *packet loss* [1,3].

2.3.1 *Delay* [3]

Delay adalah waktu tunda seluruh paket yang berhasil sampai kepada tujuan pengirimannya, dimana waktu tunda tersebut merupakan selisih dari waktu sampainya paket dan waktu pengiriman paket. Rata-rata *delay* diperoleh dengan Persamaan 2.1 :

$$Rata - rata\ Delay = \frac{Jumla\ h\ Dela\ y\ Keseluru\ han}{Jumla\ h\ Paket\ \ yang\ \ Diterima} \qquad (2.1)$$

2.3.2 *Throughput* [3]

Throughput adalah jumlah total paket data yang berhasil sampai kepada tujuan pengiriman data dalam suatu satuan waktu. Pada umumnya pengukuran *throughput* dilakukan dalam satuan bits per *second* (bps). Perhitungan *throughput* tidak mengikutsertakan *frame header* namun hanya memperhitungkan *payload* yang dikirimkan. Hal ini dikarenakan *throughput* merupakan nilai dari data aktual (*payload*) yang sampai pada penerima tanpa mengikutsertakan nilai besarnya paket *header* yang disisipkan selama pengiriman data. Hal ini dapat dirumuskan dengan Persamaan 2.2 :

$$Rata - rata\ Throughput = \frac{Jumla\ h\ Paket\ \ yang\ \ Dikirim}{Waktu\ \ Total} \qquad (2.2)$$

2.3.3 *Packet Loss* [3]

Packet loss merupakan suatu nilai yang menyatakan jumlah paket yang gagal disampaikan kepada tujuannya melalui media transmisi tertentu. *Packet loss* dapat disebabkan oleh berbagai faktor termasuk degradasi sinyal pada kanal jaringan, paket yang rusak (*corrupt*) menyebabkan ditolaknya paket pada transit, kegagalan pada perangkat jaringan, kegagalan dalam *routing* jaringan. Persentase *Packet loss* diperoleh dari Persamaan 2.3.

$$Persentase\ Packet\ Loss\ = \frac{Jumlah\ paket\ dikirim - Jumlah\ paket\ sampai}{Jumlah\ paket\ dikirim} \times 100\,\% \qquad (2.3)$$

2.3.4. Jitter [2]

Jitter adalah variasi dari *delay* yang terjadi pada suatu pengiriman paket terhadap *delay* yang terjadi untuk paket yang sebelumnya dari keseluruhan paket yang diterima pada penerima. Perhitungan terhadap *jitter* diperoleh dari selisih waktu tempuh suatu paket dengan paket sebelumnya dari sumber ke tujuan pengiriman paket tersebut.

Waktu tempuh suatu paket dari sumber ke tujuan pengiriman paket tersebut disebut juga dengan *latency*.

$$Jitter_n = \left| Latency_n - Latency_{n-1} \right| \tag{2.4}$$

Di mana *n* adalah paket pada suatu waktu tertentu. Jika *latency* diekspresikan dalam,

$$Latency_n = Arrival_n - Departure_n \tag{2.5}$$

Di mana $Arrival_n$ menunjukkan waktu kedatangan paket (paket sampai ke tujuan) dan $Departure_n$ menunjukkan waktu keberangkatan paket (paket dikirim dari sumber), maka *jitter* dapat ditulis sebagai,

$$Jitter_n = \left| \left(Arrival_n - Departure_n \right) - \left(Arrival_{n-1} - Departure_{n-1} \right) \right| \tag{2.6}$$

Persamaan tersebut dapat disusun kembali sehingga diperoleh,

$$Jitter_n = \left| \left(Arrival_n - Arrival_{n-1} \right) - \left(Departure_n - Departure_{n-1} \right) \right| \tag{2.7}$$

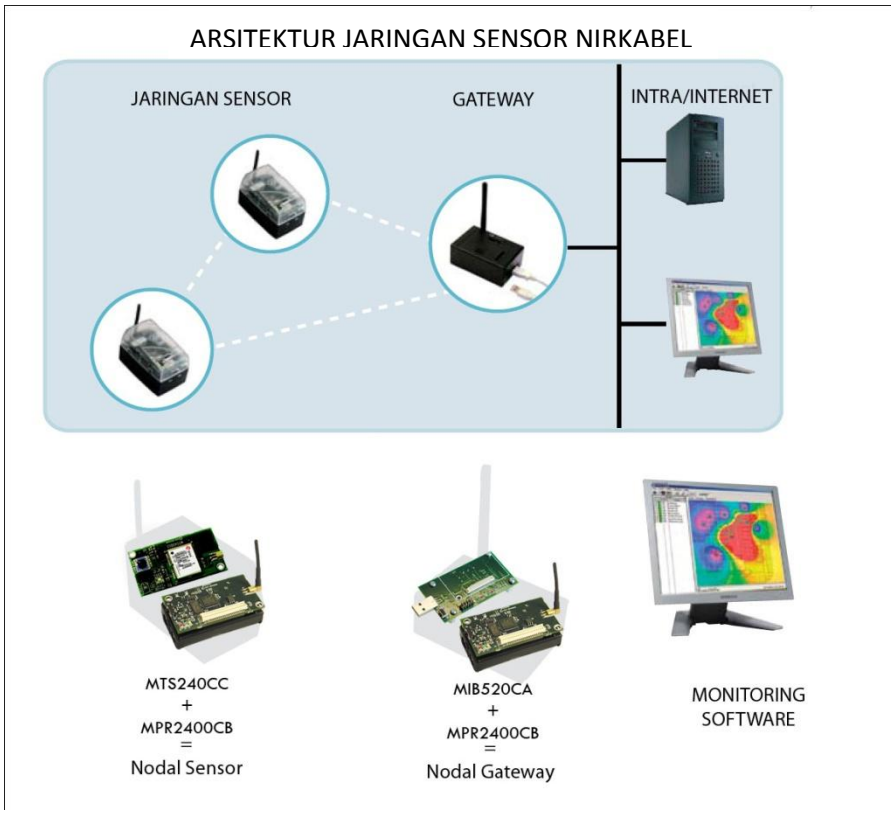
Jitter rata-rata dari sejumlah *i* paket yang dikirimkan dapat diperoleh dengan

$$AverageJitter = \frac{\sum_{n}^{i} \left| Jitter_n \right|}{i} \tag{2.8}$$

2.4 MICAz Mote

2.4.1 Pemodelan Sistem

Simulasi adalah proses perancangan model dari sistem sesungguhnya dan melakukan beberapa eksperimen terhadap model tersebut untuk tujuan mengetahui kebiasaan sistem dan atau mengevaluasi operasi dari sistem tersebut [9].



Gambar 2.3. Arsitektur JSN MICAz Mote [9]

Gambar 2.3. diatas menunjukkan arsitektur JSN MICAz Mote, dimana beberapa nodal sensor tersebut membentuk suatu jaringan sensor nirkabel dan *gateway* sebagai pusat pengumpulan informasi. Jaringan sensor nirkabel tersebut terhubung oleh *software* pemantau untuk melihat hasil dari simulasi yang dilakukan.

Adapun nodal sensor yang digunakan terdiri dari MTS240CC + MPR2400CB. Sedangkan nodal *gateway* terdiri dari MIB520CA + MPR2400CB.

Berikut merupakan penjelasan dari beberapa sensor diatas.

***Processing Board* (MPR2400CA)**

Gambar 2.4. MPR2400CA [9]

Gambar 2.4. merupakan platform radio dan pemrosesan yang berbasis pada mikrokontroler Atmel ATmega128L yang merupakan mikrokontroler daya rendah yang mampu menjalankan sistem operasi TinyOS dari *internal flash memory* yang dimiliki mikrokontroler ini. Dengan mengimplementasikan TinyOS, sebuah papan prosesor (MPR2400CA) dapat dikonfigurasi untuk menjalankan aplikasi penyensoran/pemrosesan dan kebutuhan jaringan/komunikasi radio secara bersamaan. Pada MPR2400CA terdapat konektor 51-pin yang merupakan penghubung dengan peripheral eksternal. Perangkat radio MPR2400CA memberikan kemampuan transmisi data kecepatan tinggi (250 kbps) [9].

Papan Sensor MTS420CC

Gambar 2.5. MTS420CC [10]

Gambar 2.5. merupakan papan sensor yang memiliki kapabilitas untuk melakukan pemantauan temperatur lingkungan. Jangkauan operasional dalam pemantauan temperatur ini berkisar dari -10^{o}C hingga 60^{o}C. Papan sensor MTS420CC dihubungkan dengan papan pemrosesan MPR2400CA melalui konektor 51-pin [10].

Mote Interface Board/MIB **(MIB520CA)**

Gambar 2.6. MIB520CA [11]

Gambar 2.6. merupakan sebuah komponen MICAz Mote yang berfungsi sebagai *gateway* pada jaringan sensor. MIB520CA dihubungkan dengan papan pemrosesan MPR2400CA melalui konektor 51-pin [11].

2.4.2 Lapisan Fisik MICAz Mote

Lapisan fisik yang digunakan pada JSN yang dirancang dalam tugas akhir ini meliputi lapisan fisik nodal sensor dan lapisan fisik nodal *gateway*. Lapisan fisik untuk kedua komponen JSN tersebut berdasarkan pada lapisan fisik MICAz Mote. MICAz Mote merupakan modul mote 2,4 GHz yang digunakan pada jaringan sensor. Beberapa fitur yang dimiliki MICAz Mote yaitu :

a. Memiliki *RF Transceiver* yang berdasarkan standar IEEE 802.15.4
b. Beroperasi pada pita frekuens 2,4 GHz sampai 2,48 GHz yang termasuk ke dalam kelompok pita frekuensi ISM.
c. Mengimplementasikan teknik radio *Direct Sequence Spread Spectrum* yang memilik ketahanan terhadap interferens RF.
d. Memiliki *data rate* hingga 250 kbps.
e. Mampu melakukan komunikasi melalui medium nirkabel dan setiap nodal berkemampuan berfungsi sebagai *router*.
f. Memiliki konektor ekspansi untuk papan sensor cahaya, temperatur, RH, tekanan barometrik, *accelerator* / *seismic*, akustik, magnetik dan beberapa papan sensor dari *crossbow* lainnya.

2.5 Network Simulator (NS)

Network Simulator versi kedua atau disebut juga NS-2 adalah sebuah *open-source event-driven simulator*. NS-2 merupakan sebuah perangkat lunak yang dikembangkan dengan lisensi *open-source*. Lisensi *open-source* tersebut mengizinkan setiap pengguna dapat mengakses kode sumber kompilasi NS-2. Dengan akses terhadap kode sumber ini, dapat dilakukan pengembangan, perbaikan, maupun modifikasi terhadap NS-2 oleh siapapun [6]. Dengan kontribusi terhadap pengembangan NS-2 dari berbagai pihak ini, membuat NS-2 menjadi perangkat lunak simulasi dengan kehandalan yang baik. Hal ini berlawanan dengan perangkat lunak yang berlisensi terbatas, di mana hanya hasil kompilasi dari kode sumber yang dipublikasikan kepada pengguna dan hanya kalangan terbatas yang dapat melakukan pengembangan dan perbaikan terhadap perangkat lunak tersebut.

2.5.1 Sejarah Perkembangan NS-2 [1]

NS-2 diciptakan pada tahun 1989 sebagai perangkat lunak untuk mendukung penelitian terhadap jaringan komunikasi. NS-2 terus mendapatkan revisi dan perbaikan dari beberapa kontributor yang berperan dalam pengembangan perangkat lunak simulator ini. Beberapa kontributor tersebut di antaranya adalah Universitas California dan Universitas Cornell yang mengembangkan REAL *Network Simulator*. REAL *Network Simulator* pada awalnya diimplementasikan sebagai perangkat lunak untuk mempelajari karakteristik dinamik dari skema kendali aliran dan kongesti (kemacetan) pada *packet-switched data network*. NS-2 diciptakan berdasarkan pada REAL *Network Simulator*. Sejak tahun 1995,

Defence Advance Research Project Agency (DARPA) mendukung pengembangan dari *Network Simulator* (NS) melalui proyek *Virtual InterNetwork Testbed* (VINT). Proyek VINT yang didanai oleh DARPA tersebut bertujuan untuk menciptakan sebuah simulator jaringan untuk mempelajari berbagai protokol yang berbeda untuk jaringan komunikasi. Saat ini, *National Science Foundation* (NSF) telah bergabung dalam usaha pengembangan perangkat lunak NS-2 ini. Selain itu, para peneliti dan pengembang tetap bekerja untuk menjaga dan meningkatkan kehandalan serta versatilitas perangkat lunak NS-2 ini.

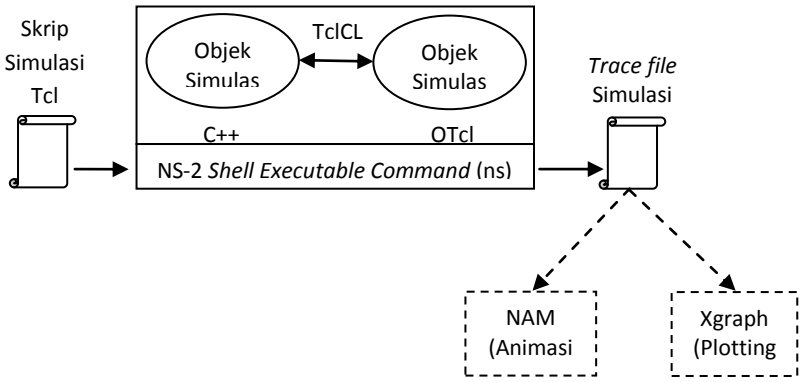
2.5.2 Kelebihan NS-2

Kelebihan NS-2 adalah sebagai perangkat lunak simulasi pembantu analisis dalam riset atau penelitian. NS-2 dilengkapi dengan tool validasi. Tool validasi digunakan untuk menguji validitas pemodelan yang ada pada NS-2. Penggunaan simulasi pada NS-2 jauh lebih mudah, karena pada *software* ini *user* hanya membuat topologi dan skenario simulasi yang sesuai dengan riset.

Keuntungan lain dari NS-2 adalah perangkat lunak ini dapat didistribusikan kembali tanpa berbayar sehingga akan sangat memudahkan kalangan akademia maupun kalangan umum untuk menggunakannya sebagai perangkat lunak pendukung pembelajaran. NS-2 dapat digunakan untuk simulasi fungsi dan protokol jaringan kabel maupun jaringan nirkabel.

2.5.3 Arsitektur Dasar NS-2

NS-2 menyediakan sebuah perintah ns, yang merupakan sebuah perintah yang dapat dieksekusi (*executable*) oleh penggunanya. Dalam menjalankan simulasi pada NS-2, perintah ns tersebut membutuhkan argumen masukan. Argumen masukan yang dibutuhkan tersebut berupa nama dari skrip simulasi Tcl yang telah dipersiapkan untuk simulasi terlebih dahulu. NS-2 akan menjalankan simulasi berdasarkan skenario yang terdapat pada file skrip simulasi Tcl tersebut. Simulasi tersebut akan menghasilkan sebuah *trace file* yang berisikan data hasil simulasi. File tersebut akan digunakan sebagai dasar dalam menampilkan grafik hasil simulasi dan/atau menampilkan animasi simulasi. Gambar 2.7. menunjukkan arsitektur dari NS.



Gambar 2.7. Arsitektur dasar dari NS [10]

NS-2 dapat diperoleh dari situs resmi pendistribusian perangkat lunak simulator jaringan [10]. Perangkat lunak NS-2 dijalankan dengan dukungan dari komponen-komponen utama pendukung NS-2, yaitu Tcl/Tk, OTcl, dan TclCL. Tcl/Tk merupakan komponen bahasa pemrograman Tcl (*Tool Command Language*) yang

dilengkapi dengan perangkat Tk, sebagai ekstensi bagi Tcl dalam menyediakan pustaka antarmuka grafis pengguna (*graphical user interface*) untuk berbagai sistem operasi. OTcl merupakan komponen utama untuk NS-2 untuk bahasa pemrograman OTcl (*Object-oriented Tool Command Language*).

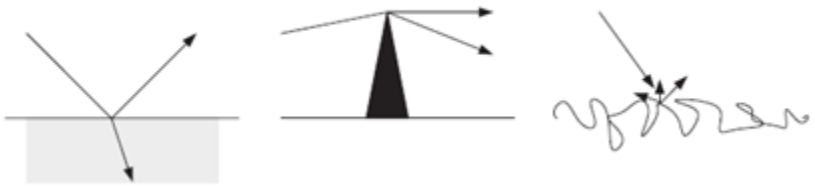
Simulator NS-2 dijalankan dengan menggunakan dua bahasa pemrograman, yaitu C++ dan *Object-orientes Tool Command Language* (OTcl). C++ berfungsi dalam menangani mekanisme internal pada simulasi dengan NS-2. OTcl menangani interaksi langsung antara pengguna dengan simulator serta menangani interaksi antara objek-objek OTcl lainnya. C++ dan OTcl saling terhubung dengan menggunakan komponen TclCL. Variabel-variabel pada domain OTcl dipetakan pada objek C++. Variabel ini cenderung dikenal sebagai sebuah *handle*. Secara konseptual, sebuah *handle* (misal, n sebagai *handle* untuk nodal) hanyalah sebuah kalimat atau karakter biasa dan tidak memiliki fungsional apapun dalam domain OTcl. Fungsionalitas *handle* tersebut (misal, penerimaan paket) didefinisikan pada objek C++ yang dipetakan (misal, pada kelas *connector* pada domain C++). Dalam domain OTcl, sebuah *handle* berfungsi sebagai substansi untuk menangani interaksi simulator dengan pengguna, maupun interaksi dengan objek OTcl lainnya. Dalam melaksanakan fungsi ini, sebuah *handle* dapat mendefinisikan sendiri prosedur (*procedure*) dan variabel (*variable*) untuk memfasilitasi interaksi tersebut. Pada domain OTcl, *procedure* dan *variable* disebut dengan *instance procedure (instproc)* dan *instance variable (instvar)* secara berurutan. Tanpa adanya komponen-komponen tersebut, simulator NS-2 tidak dapat berjalan sebagaimana mestinya.

Selain komponen-komponen utama di atas, terdapat pula komponen-komponen pendukung yang bersifat opsional. Komponen-komponen opsional ini di antaranya adalah NAM (*Network Animator*), Zlib, dan Xgraph. NAM merupakan komponen untuk memvisualisasikan simulasi jaringan dan perjalanan paket pada simulasi. Zlib adalah file pustaka (*library*) yang dibutuhkan oleh NAM. Xgraph adalah komponen untuk memvisualisasikan data yang diperoleh dari simulasi ke dalam bentuk grafik. Kode sumber (*source code*) NS-2 terdistribusi dalam dua bentuk, yaitu bentuk terpadu (*all-in-one*) dan bentuk tidak terpadu. Pada bentuk terpadu, komponen-komponen utama pendukung NS-2 telah dipaketkan dalam sebuah sebuah file. Pada bentuk tidak terpadu, komponen-kompoen NS-2 tidak dipaketkan dalam sebuah file sebagaimana pada bentuk terpadu, sehingga pengguna harus mengunduh atau memperoleh masing-masing komponen dari pengembang penyedia masing-masing komponen tersebut.

Pada simulasi yang dilakukan terhadap JSN, setelah proses simulasi selesai akan diperoleh *trace file* sebagai file keluaran simulasi dimana dapat diperoleh data yang diinginkan. Untuk melakukan visualisasi simulasi, diperlukan pemrosesan tambahan terhadap file tersebut dengan menggunakan perangkat lunak lainnya. Salah satu perangkat lunak visualisator ini adalah iNSpect. Perangkat lunak iNSpect merupakan perangkat lunak visualisator untuk simulasi jaringan *Mobile Ad-hoc Network* (MANET).

2.6 Fenomena Propagasi Gelombang

Gelombang yang dipropagasikan melalui media nirkabel akan mengalami beberapa fenomena fisik yang mendistorsi bentuk gelombang asli yang diperoleh penerima. Distorsi menimbulkan ketidakpastian pada data asli yang dimodulasi dan dikodekan yang menyebabkan *bit error* pada penerima. Gambar 2.8. menunjukkan fenomena propagasi gelombang dasar.



(a) (b) (c)

Gambar 2.8. Ilustrasi fenomena propagasi gelombang pada media nirkabel [4].

Refleksi merupakan fenomena yang terjadi ketika suatu gelombang merambat pada medium A menumbuk perbatasan medium tersebut dengan medium B. Lapisan perbatasan antara kedua medium tersebut sangat halus, hal ini mengakibatkan sebagian dari bentuk gelombang akan dipantulkan kembali ke medium A, sedangkan sebagian yang lain akan dipancarkan ke medium B, sedangkan sisanya akan terserap, sebagaimana ditunjukkan pada Gambar 2.8.(a). Jika suatu gelombang yang merambat pada ruang bebas menumbuk permukaan yang lancip dapat menyebabkan gelombang tersebut mengalami difraksi sebagaimana yang ditunjukkan pada Gambar 2.8.(b). Penghamburan gelombang terjadi ketika suatu gelombang yang merambat pada ruang bebas menumbuk permukaan yang kasar sehingga menyebabkan pantulan sinyal terhambur ke

segala arah sebagaimana ditunjukkan pada Gambar 2.8.(c). Ketika pemancar dan penerima masing-masing bergerak terhadap yang lainnya, bentuk gelombang akan mengalami perubahan pada frekuensinya, berdasarkan efek Doppler. Terlalu banyak perubahan dapat menyebabkan penerima melakukan proses *sampling* sinyal pada frekuensi yang salah. Fenomena ini disebut dengan *doppler fading* [3,4,6].

2.6.1 Model Propagasi

Model propagasi dilakukan untuk memprediksikan besarnya nilai daya sinyal yang diterima pada jarak tertentu dari pemancar. Pada lapisan fisik dari setiap nodal terdapat nilai ambang batas daya minimal penerimaan sinyal. Ketika suatu paket diterima dengan daya penerimaan sinyal di bawah nilai ambang batas tersebut, paket tersebut dianggap sebagai paket yang rusak dan dibuang pada lapisan MAC.

Berikut merupakan berbagai model propagasi.

1. *Free Space Propagation Model*

Model ini mengasumsikan keadaan propagasi ideal di mana hanya terdapat sebuah saluran *line-of-sight* antara pemancar dan penerima serta tidak terdapat penghalang yang menghalangi saluran transmisi tersebut. Model propagasi ini memprediksikan bahwa tingkat daya sinyal akan menurun dengan bertambahnya jarak pemisah antara pemancar dan penerima. Daya sinyal yang diterima oleh

antena penerima yang terpisah pada jarak *d* dimodelkan dengan persamaan ruang bebas Friss sebagai berikut,

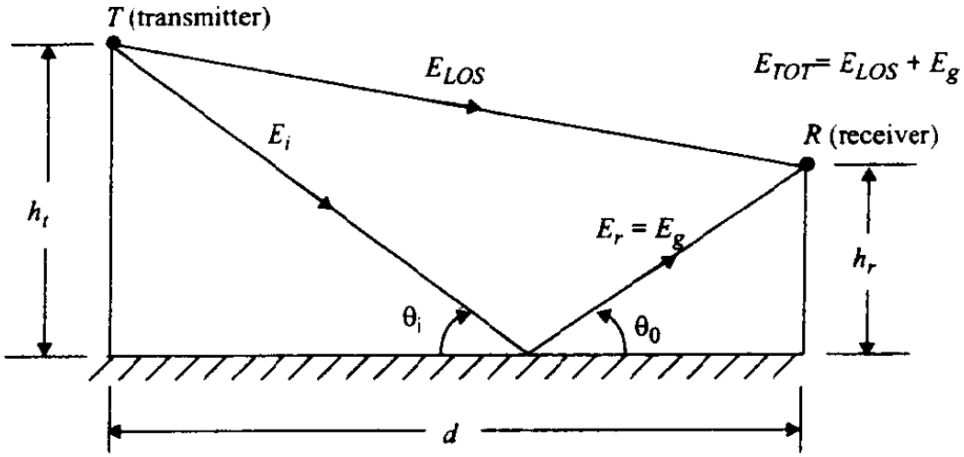
$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.9}$$

Di mana $P_r(d)$ adalah daya yang diterima sebagai fungsi jarak pemisahan pemancar dan penerima, P_t adalah daya yang dipancarkan, G_t adalah gain antena pemancar, G_r adalah gain antena penerima, d adalah jarak pemisahan antena pemancar dan antena penerima dalam meter, λ adalah panjang gelombang dalam meter dan L adalah faktor rugi-rugi sistem. Nilai L di mana $L \geq 1$ merupakan faktor rugi-rugi sistem yang dikarenakan oleh pelemahan sinyal pada saluran transmisi, rugi-rugi filter, dan rugi-rugi antena pada sistem komunikasi. Jika nilai $L = 1$ mengindikasikan tidak terdapat rugi-rugi pada perangkat keras sistem. Model Propagasi *free space* merepresentasikan jangkauan komunikasi dari perangkat komunikasi sebagai suatu lingkaran yang melingkupi pemancar. Jika penerima berada pada cakupan lingkaran tersebut, maka penerima akan menerima seluruh paket-paket. Jika posisi penerima berada di luar cakupan lingkaran tersebut maka penerima akan kehilangan seluruh paket-paket yang dipancarkan.

2. Ground Reflection Propagation (2-ray) Model

Model propasi ini memodelkan perambatan sinyal pada media nirkabel tidak hanya sebagai suatu saluran langsung (LOS) antara pemancar dan penerima, sebagaimana dimodelkan pada Model propagasi *free space*, namun juga mengikutsertakan saluran pantulan permukaan (*ground reflection*) perambatan

sinyal antara pemancar dan penerima. Model propagasi ini digambarkan sebagai berikut.



Gambar 2.9. Model propagasi *Ground Reflection (2-ray)* [4]

Pada Gambar 2.9. di atas, h_t adalah tinggi antena pemancar, h_r adalah tinggi antena penerima dan d adalah jarak pemisah yang memisahkan antena pemancar dan antena penerima. Pada model propagasi ini permukaan bumi dianggap sebagai permukaan yang datar sehingga sinyal yang diterima pada penerima merupakan hasil penjumlahan antara komponen sinyal yang merambat pada saluran langsung dan komponen sinyal yang merambat pada saluran terpantulkan (*ground reflected*). Sinyal yang diterima pada penerima untuk model propagasi ini diperhitungkan dengan persamaan

$$P_r(d) = \frac{P_t G_t G_r h_t^{\,2} h_r^{\,2}}{d^4 L} \tag{2.10}$$

Model propagasi *two-ray ground reflection* ini tidak memberikan hasil yang akurat jika dimodelkan untuk transmisi jarak dekat karena adanya osilasi yang disebabkan oleh kombinasi konstruktif dan destruktif dari gelombang-gelombang yang merambat pada saluran yang berbeda. Ketika jarak pemisahan d bernilai

kecil, maka model propagai *free space* lebih cocok untuk digunakan. Oleh karena itu, sebuah jarak *cross-over, d_c* perlu diperhitungkan pada model ini. Ketika $d < d_c$ maka persamaan model propagasi *free space* digunakan. Ketika $d > dc$ maka persamaan model propagasi *two-ray ground reflection* digunakan. Jika jarak sama dengan jarak *cross-over* maka penggunaan persamaan untuk model propagasi *free space* maupun *two-ray ground reflection* akan memberikan hasil yang sama. Nilai jarak *cross-over* dapat dihitung dengan menggunakan persamaan 2.11 [3].

$$d_c = \frac{4\pi h_t h_r}{\lambda} \tag{2.11}$$

Tabel.2.1. Beberapa *band frequency* ISM [1].

Frequency	comment
13.553 – 13.567 MHz	
26.957 – 27.283 MHz	
40.66 – 40.70 MHz	
433 – 464 MHz	Europe
902 – 928 MHz	Only in the Americas
2.4 – 2.5 GHz	Used by WLAN/WPAN technologies
5.725 – 5.875 GHz	Used by WLAN technologies
24 – 24.25 GHz	

Tabel 2.1. berisikan daftar beberapa *band* frekuensi *Industrial Scientifik and Medical* (ISM). Mayoritas alokasi *band* frekuensi yang populer digunakan pada aplikasi JSN adalah *band* frekuensi ISM. Band frekuensi ISM merupakan alokasi *band* frekuensi *unlicense*, dimana penggunaan *band* frekuensi tersebut tidak memerlukan perizinan dari pemerintah atau badan yang menangani masalah alokasi frekuensi.

2.7 Protokol Jaringan Sensor Nirkabel

2.7.1 Protokol Transport

- ***User Datagram Protocol*** **(UDP)**

UDP merupakan protokol *transport* sederhana dengan model layanan yang minimalis. Tipe koneksi UDP bersifat *connectionless,* di mana tidak diperlukan suatu proses *handshake* sebelum membangun suatu koneksi, sebagaimana yang dilakukan pada TCP. Dengan model layanan minimalis yang dimilikinya, UDP hanya memberikan suatu layanan transfer data yang bersifat *unreliable*. Layanan ini lebih identik dengan suatu layanan transfer yang data yang kurang handal. Pada protokol *transport* UDP, tidak terdapat jaminan bahwa pesan yang dikirimkan akan sampai kepada tujuannya dan terdapat kemungkinan bahwa pesan yang dikirimkan tersebut sampai dengan urutan paket yang tidak tepat. UDP tidak mengimplementasikan pemeriksaan terhadap kesalahan dan koreksi terhadap pengiriman data serta mekanisme-mekanisme lainnya yang memberikan kehandalan suatu sistem protokol *transport*. Hal ini dilakukan untuk mengurangi beban pemrosesan yang harus ditanggung oleh sistem untuk mentransmisikan data. Aplikasi sensitif-waktu pada umumnya menggunakan protokol *transport* UDP karena pilihan di mana paket harus hilang lebih diutamakan dari pada menunggu paket yang tertunda karena dilakukan pengiriman ulang terhadap paket tersebut.

2.7.2 Protokol Perutean

- ***Ad-hoc On Demand Distance Vector Routing* (AODV)**

Jaringan Ad-hoc merupakan jaringan yang terdiri atas nodal-nodal yang saling bekerja sama secara koperatif dan dalam melakukan fungsi tersebut jaringan ini tidak tergantung pada satu titik akses atau infrastruktur-infrastruktur tetap lainnya. *Ad-hoc On Demand Distance Vector Routing* (AODV) merupakan sebuah algoritma yang dapat mendukung pengoperasian jaringan ad-hoc tersebut. Pada AODV, setiap nodal bertindak sebagai *router*, dan suatu rute hanya akan terbentuk jika dibutuhkan atau terdapat permintaan (*on demand*). Implementasi protokol perutean dilakukan pada perangkat keras MICAz Mote. Perutean AODV memilih nodal hop selanjutnya dengan jumlah nodal paling sedikit yang harus dilalui hingga ke tujuan, di mana tujuannya adalah nodal *gateway*. Protokol AODV menggunakan pesan perutean *Route Request* (RREQ), *Route Reply* (RREP) dan *Route Error* (RERR) untuk menemukan hop selanjutnya. Setiap nodal sensor menyimpan dua buah tabel perutean untuk mencatat pesan-pesan tersebut di mana data maksimum yang dapat disimpan oleh tabel tersebut adalah tujuh masukan. Kedua tabel tersebut adalah :

a. Route Cache table

Aktifitas yang dilakukan pada tabel ini berupa pembaharuan data yang dikandungnya. Pembaharuan data dilakukan saat sebuah pesan RREQ diterima dari nodal pengirim pesan "*originator*" dan saat penyebaran kembali pesan tersebut oleh nodal perantara "*source*". Pembaharuan data tabel dilakukan oleh nodal-nodal disekitar nodal "*originator*" dan "*source*".

Dengan data yang dimiliki oleh tabel ini, dibangun saluran balikan yang menghubungkan nodal tujuan pengiriman pesan ke nodal sumber pengiriman pesan.

b. Route table

Aktifitas yang dilakukan terhadap tabel ini juga berupa pembaharuan data yang dikandungnya. Pembaharuan data dilakukan pada tabel-tabel yang terdapat di nodal "*originator*" dan "*source*". Pembaharuan ini dilakukan saat pesan RREP diperoleh sebagai respon terhadap pesan RREQ. Tabel ini mengandung data mengenai alamat hop selanjutnya untuk mencapai nodal tujuan pengiriman data.

Pada AODV, saat diperlukan rute ke suatu nodal, nodal sumber pengiriman data "*originator*" menyiarkan/menyebarkan pesan-pesan RREQ kepada nodal-nodal di sekitarnya. Nodal-nodal di sekitarnya tersebut "*source*" menerima dan menyimpam pesan RREQ, selanjutnya dilakukan pembaharuan data pada *route cache table* untuk setiap nodal tersebut dan dilakukan penyiaran/penyebaran kembali pesan RREQ tersebut kepada nodal-nodal di sekitarnya. Pada akhirnya, setelah pesan RREQ sampai kepada nodal tujuannya (nodal *gateway*), nodal yang dituju tersebut akan menanggapi dengan pengiriman pesan balasan RREP sebagai respon terhadap pesan RREQ yang datang. Pesan RREP akan dikirimkan ke nodal sumber yang merupakan asal pengiriman pesan RREQ. Perjalanan pesan RREP mengikuti saluran balikan yang dibentuk dari data pada tabel *route cache table*, di mana data pada tabel ini selalu diperbaharui saat pesan RREQ diterima oleh suatu nodal. Ketika nodal awal pengiriman pesan RREQ "*originator*" menerima pesan

RREP maka sebuah saluran utuh yang menghubungkan nodal sumber dan nodal tujuan pengiriman data telah terbentuk [12-14].