BAB 2
LANDASAN TEORI

2.1 Perangkat Lunak

Perangkat lunak (*software*) adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaan (*user manual*). Sebuah program komputer tanpa terasosiasi dengan dokumentasinya maka belum dapat disebut perangkat lunak (*software*). Sebuah perangkat lunak juga sering disebut dengan sistem perangkat lunak. Sistem berarti kumpulan komponen yang saling terkait dan mempunyai satu tujuan yang ingin dicapai.

Sistem perangkat lunak berarti sebuah sistem yang memiliki komponen berupa perangkat lunak yang memiliki hubungan satu sama lain untuk memenuhi kebutuhan pelanggan (*customer*). Pelanggan (*customer*) adalah orang atau organisasi yang memesan atau membeli perangkat lunak (*software*) dari pengembang perangkat lunak atau bisa dianggap bahwa pelanggan (*customer*) adalah orang atau organisasi yang dengan sukarela mengeluarkan untuk memesan atau membeli perangkat lunak. *User* atau pemakai perangkat lunak adalah orang yang memiliki kepentingan untuk memakai atau menggunakan perangkat lunak untuk memudahkan pekerjaannya [13].

Secara umum perangkat lunak ini dapat diklasifikasikan menjadi 2 [1], yaitu:

1. Perangkat Lunak Sistem Operasi

   Merupakan perangkat lunak yang berfungsi untuk mengkonfigurasikan komputer agar dapat menerima berbagai perintah dasar yang diberikan sebagai masukan. Perangkat lunak sistem juga merupakan sekumpulan program yang ditulis untuk melayani program-program yang lain. Contoh: *MS-DOS, LINUX, UNIX, FREE BSD, OS/2, SUN OS (JAVA), WINDOWS, MACINTOSH, NOVELL,* dan lain sebagainya.
2. Perangkat Lunak Aplikasi

   Merupakan program siap pakai yang digunakan untuk aplikasi dibidang tertentu. Perangkat lunak aplikasi yang membantu *user* sehingga dapat

bekerja lebih efektif dan efisien. Aplikasi ini dibagi atas beberapa bidang, antara lain:

1) *Business & Office Application*

   Aplikasi perkantoran dipergunakan untuk menyelesaikan pekerjaan kantor seperti mengetik (*Word processing, document management*) membuat tabel kerja (*Worksheet*), membuat Database sederhana (*MS. Access*) ataupun mengolah *image*/citra sederhana. Contoh aplikasi perkantoran yang terpaket komplit: *Mocrosoft Office, Koffice, Star Office, Abiword, Open Office, MS_Office for MAC*, dan lain sebagainya.

   Aplikasi bisnis lainnya juga meliputi aplikasi pengolahan data keuangan, misal (*Personal Finance Application & Business Acounting Application*). Contoh: Aplikasi perhitungan neraca, rugi laba perusahaan, Aplikasi perhitungan pajak personal & perusahaan, Aplikasi manajemen proyek, Aplikasi persediaan stok barang dan penjualan, dan lain sebagainya.

2) *Database Application*

   Aplikasi yang digunakan dalam pengolahan data baik yang berukuran kecil maupun besar bisa digunakan secara *stand-alone* (tunggal) maupun dalam sistem berbasis jaringan lokal *Client server* maupun *webbase* (intranet maupun internet). Contoh: *MS SQL, FoxBase, Oracle, MySQL, PostgresSQL, SQL Server*, dan lain sebagainya.

3) *Graphics Design Application*

   Aplikasi yang digunakan untuk membuat sketsa dua dimensi untuk desain ruangan, perabotan, mesin-mesin model industri, juga model tubuh manusia, hewan, dan lain sebagainya. Contoh: *AutoCad, Pro Design, CorelDraw, Adobe Ilistrator,* dan lain sebagainya.

   Ada juga yang memasukan dalam golongan ini aplikasi *image, audio, video manipulator*, yaitu aplikasi yang digunakan untuk mengolah citra, suara hingga gambar 3D untuk pembuatan animasi, pengolahan *image, photo, audio, video editing*, dan lain sebagainya. Contoh:

Adobe (Photoshop, After Effect, Premire, & Audition), Corel (Draw, RAVE, Photo Paint), Ulead Media studio, GIMP (In Linux), Cool Edit Pro, Sound Force, *Macromedia (Flash, Director), 3D Studio Max*, dan lain sebagainya.

4) *Antivirus & Utility Tools*

Software ini digunakan untuk operasi minor (*Background*) yang bekerja di belakang sistem, berguna untuk meningkatkan kinerja sistem operasi atau aplikasi dan juga performa *hardware*. Terdapat 2 jenis utilitas di dalam sistem operasi:

a) *Utilitas Internal*, sudah termasuk dalam paket yang di instalasi dalam sistem operasi seperti *scandisk* di *Windows* atau perintah *fsck* (*file system check*) di OS Linux.

b) *Utilitas eksternal*, yang harus diinstalasi kemudian pada OS, seperti *System Optimizer, PC Tools, Data/File Recovery, PC Maintenance, Internet Utilities, Norton System Work, Registry Clean & Repair, Hardware Repair*.

Sementara untuk proteksi virus dan antivirus, contohnya adalah antivirus produk *Mc Affee, Grisoft, Norton, PandaSoft,* dan lain sebagainya.

5) *Development Tools*

Merupakan sistem bahasa pemrograman komputer yang dilengkapi dengan *compiler* tertentu sebagai sumber media penyusunan program aplikasi. Contoh: *QuikBasic, Clipper, Visual Basic*, *Turbo Pascal*, *Visual C++, GCC, Turbo Assembler, Phyton, KDEvelopt, Gdekstop,* dan lain sebagainya.

Termasuk dalam jenis aplikasi ini adalah *web development tools*, yaitu aplikasi yang digunakan untuk membuat/mendesain tampilan website (*homepage*) di internet, seperti *FrontPage*, *Macromedi Dreamweaver*, *PHPedit, WebPage Editor*, dan lain sebagainya.

6) *Communication Application*

Aplikasi yang digunakan untuk membangun komunikasi antara satu komputer dengan komputer lain dalam jaringan komputer local maupun global. Contoh: *E-mail & Fax Application, Remote Access, Facebook chat, YM, GTalk, MSN, Terminal Emulation, Instant Messaging, Voice Recognition, Web Browser, Internet Telephony,* dan lain sebagainya.

7) Jenis Aplikasi Lain

a) *Multimedia Player,* seperti *WinAmp, XMMS, Power DVD, JetAudio, Windows Media Player,* dan lain sebagainya.

b) *Hacker Tools,* seperti *Find Password, Keygen/Crack Software License, Scanning Port,* dan lain sebagainya.

c) *Network & Security Application,* seperti *Firewall, Intruder Detections, Network Monitoring*, dan lain sebagainya.

d) *PC Games Apllication*, dan masih banyak lagi.

2.2 Jaringan Komputer

Jaringan komputer adalah himpunan interkoneski antara 2 komputer *autonomus* atau lebih yang terhubung dengan media transmisi kabel atau tanpa kabel (*wireless*). Gabungan teknlogi ini melahirkan pengolahan data yang dapat didistribusikan, mencakup pemakaian *database*, *software* dan *hardware*. Bila sebuah komputer dapat membuat komputer lainnya *restart, shutdown,* atau melakukan kontrol lainnya, maka komputer-komputer tersebut bukan *autonomus* (tidak melakukan kontrol terhadap komputer lain dengan akses penuh) [3].

Berdasarkan jangkauan area atau lokasi, jaringan dibedakan menjadi 3 jenis yaitu [8]:

1. Local Area Network (LAN):

   Merupakan jaringan yang menghubungkan sejumlah komputer yang ada dalam suatu lokasi dengan area yang terbatas seperti ruang atau gedung. LAN dapat menggunakan media komunikasi seperti kabel dan *wireless.*

2. Wide Area Network (WAN):

   Merupakan jaringan antar LAN satu dengan LAN lain yang dipisahkan oleh lokasi yang cukup jauh. Contoh penggunaan WAN antar kantor pusat dengan kantor cabang yang ada di daerah-daerah.

3. Metropolitant Area Network (MAN):

   Merupakan jaringan yang lebih besar dari jaringan LAN tetapi lebih kecil dari jaringan WAN. Jaringan MAN dan jaringan WAN sama-sama menghubungkan beberapa LAN yang membedakan hanya lingkup areanya yang berbeda.

Pada dasarnya setiap jaringan komputer ada yang berfungsi sebagai *client* dan juga *server*. Tetapi ada jaringan yang memiliki komputer yang khusus didedikasikan sebagai *server* sedangkan yang lain sebagai *client*. Ada juga yang tidak memiliki komputer yang khusus berfungsi sebagai *server* saja. Karena itu berdasarkan fungsinya maka ada dua jenis jaringan komputer :

1. Client-server

   Yaitu jaringan komputer dengan komputer yang didedikasikan khusus sebagai *server*. Sebuah *service*/layanan bisa diberikan oleh sebuah komputer atau lebih. Contohnya adalah sebuah domain seperti www.unikom.ac.id yang dilayani oleh banyak komputer *web server*. Atau bisa juga banyak *service*/layanan yang diberikan oleh satu komputer. Contohnya adalah server if.unikom.ac.id yang merupakan satu komputer dengan multi *service* yaitu *mail server, web server*, file *server*, *database server* dan lainnya.

2. Peer-to-peer

   Yaitu jaringan komputer dimana setiap *host* dapat menjadi *server* dan juga menjadi *client* secara bersamaan, atau dengan kata lain dapat dikatakan dima satu komputer terhubung ke komputer lainnya secara langsung.

Berdasarkan media transmisi datanya, jaringan komputer dapat dibedakan atas beberapa kategori, diantaranya :

1. Jaringan Berkabel (*Wired Network*)

   Pada jaringan ini, untuk menghubungkan satu komputer dengan komputer lain diperlukan penghubung berupa kabel jaringan. Kabel jaringan berfungsi

dalam mengirim informasi dalam bentuk sinyal listrik antar komputer jaringan.

2. Jaringan Nirkabel (*Wireless Network*)

   Merupakan jaringan dengan medium berupa gelombang elektromagnetik. Pada jaringan ini tidak diperlukan kabel untuk menghubungkan antar komputer karena menggunakan gelombang elektromagnetik yang akan mengirimkan sinyal informasi antar komputer jaringan.

2.3 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk penyimpanan data agar dapat diakses dengan mudah dan cepat.

Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa file teks ataupun Database Management System (DBMS) [13]. Kebutuhan basis data dalam sistem informasi meliputi memasukan, menyimpan, dan mengambil data serta membuat laporan berdasarkan data yang telah disimpan.

2.3.1 *Database Management System* (DBMS)

DBMS (Database Management System) atau dalam bahasa indonesia sering disebut sebagai Sistem Manajemen Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal dapat menyediakan fasilitas untuk mengelola akses data, mampu menangani integrasi data, dan mampu menangani backup data.

DBMS sudah mulai berkembang sejak tahun 1960an. Kemudian sekitar tahun 1970an mulai berkembang teknologi Relational DBMS yaitu DMBS berbasis relasional model. Relasional model pertama kali dikembangkan oleh Edgar J. Codd pada tahun 1970. Secara sederhana relasional model dapat dipahami sebagai suatu model yang memandang data sebagai sekumpulan tabel yang saling terkait.

Pada tahun 1980an mulai berkembang Object Oriented DBMS (OODBMS). OODBMS berkembang seiring dengan perkembangan teknologi pemrograman berorientasi objek. Secara umum dapat diartikan bahwa OODBMS merupakan DBMS yang memandang data sebagai suatu objek.

Terdapat beberapa macam DBMS versi komersial yang paling sering digunakan di dunia saat ini, diantaranya Oracle, Microsoft SQL Server, IBM DB2, dan Microsoft Access. Sedangkan DBMS versi open source yang cukup berkembang dan sering digunakan saat ini adalah diantaranya MySQL, PostgreSQL, Firebird, dan SQLite. Hampir semua DBMS mengadopsi SQL sebagai bahasa untuk mengelola data pada DBMS.

2.3.2 *Structured Query Language* (SQL)

SQL (*Structured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada DBMS. Meskipun SQL diadopsi dan diacu sebagai bahasa standar oleh hampir sebagian besar RDBMS yang beredar saat ini, tetapi tidak semua standar yang tercantum dalam QSL diimplementasikan oleh seluruh DBMS tersebut. Sehingga terkadang terdapat perbedaan prilaku (hasil yang ditampilkan) oleh DBMS yang berbeda padahal *query* yang dimasukan sama.

2.4 Pemograman Beroriantasi Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya). Pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dari abstraksi objek, kelas, hubungan antar kelas sampai abstraksi sistem. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses yang dimiliki oleh objek akan dienkapsulasi (dibungkus) menjadi satu kesatuan.

Sistem berorientasi objek merupakan sebuah sistem yang kompunennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen

dalam sistem tersebut dapat mewarisi atribut, sifat, dan kompunen lainnya yang dapat berinteraksi satu sama lain.

Terdapat beberapa konsep utama pada metodologi berorientasi objek, diantaranya [13]:

1. Kelas (*class*), kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dari himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*), dan arti. Suatu kelas dapat diturunkan dari kelas yang lain, dimana atribut dari kelas semula dapat diwariskan ke kelas yang baru.
2. Objek (*object*), abstraksi dari sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.
3. Abstraksi (*abstraction*), prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi suatu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.
4. Enkapsulasi (*encapsulation*), pembungkusan atribut data dan layanan (operasi-operasi) yang dimiliki objek untuk menyembunyikan implementasi dari objek sehingga objek lain tidak mengetahui cara kerjanya.
5. Pewarisan (*inheritance*), mekanisme yang memungkinkan suatu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.
6. Polimorfisme (*polymorphism*), kemampuan seuatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

2.5 *Unified Modeling Language* (UML)

UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (*Object-Oriented*) [5]. UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem software. *Unified Modelling Language* (UML) juga merupakan sebuah bahasa yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Terdapat beberapa diagram yang umum digunakan dalam pemodelan UML diantaranya [13]:

1. *Use Case Diagram,* merupakan pemodelan untuk kelakuan (*behavior*) sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antar satu atau lebih aktor dengan sistem yang akan dibuat. Secara kasar *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi itu.
2. *Class Diagram,* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki atribut dan metode (operasi). Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Metode (operasi) fungsi-fungsi yang dimiliki oleh suatu kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki kelas *main*, kelas *view*, kelas *controller*, serta kelas *model*.
3. *Sequence Diagram,* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui onjek-objek yang terlibat dalam sebuah *use cse* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi

objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*.

4. *Activity Diagram,* menggambarkan aliran kerja (*workflow*) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, melainkan aktivitas yang dapat dilakukan oleh sistem.
5. *Component Diagram,* dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Diagram komonen fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem. Komponen dasar yang biasanya ada dalam suatu sistem adalah komponen *user interface* yang menagani tampilan, komponen *bussiness processing* yang menangani fungsi-fungsi proses bisnis, komponen *data* yang menangani manipulasi data, dan komponen *security* yang menangani keamanan sistem. Komponen lebih terfokus pada penggolongan secara umum fungsi-fungsi yang diperlukan.

2.6 *Near-Field Communication* (NFC)

NFC atau *Near-Field Communication* adalah seperangkat teknologi konektivitas nirkabel berbasis teknologi *Radio Frequency Idetification (RFID)* yang menggunaka induksi medan magnet untuk memungkinkan komunikasi antar perangkat teknologi. Salah satu teknologi komunikasi interaksi dua arah sederhana dan aman antara perangkat eloktronik, yang memungkinkan untuk melakukan transaksi *contactless,* mengakses konten digital, dan menghubungkan perangkat elektronik dengan satu sentuhan.

NFC beroperasi pada frekuensi 13,56 MHz dengan kecepatan transmisi pengiriman mencapai 424 kbit/s. Jarak transmisi NFC sekitar 4-10 cm. Perbedaan antara NFC dan teknologi komunikasi *contactless* lainnya yaitu perangkat NFC dapat bersifat aktif – aktif (*peer to peer*) dan aktif – pasif. Oleh karena itu NFC selalu melibatkan inisiator (*reader*) dan target. Inisiator aktif menghasilkan medan RF (*Radio Frequency*) yang dapat memberikan kekuatan ke target yang pasif

(tidak memiliki sumber daya). Hal ini memungkinkan target NFC untuk memiliki bentuk yang sangat sederhana seperti stiker, gantungan kunci, atau kartu yang tidak memerlukan energy khusus [16].

Teknologi *NFC* ini adalah hasil penelitian dari perusahaan Sony dan NXP pada tahun 2002 yang awalnya digunakan sebagai sistem perantara pembayaran di Amerika Serikat, penelitian ini melibatkan teknologi radio dan chipset yang ditanamkan dalam perangkat. Sebenarnya ide ini berawal dari *RFID (Radio Frequency Identification). RFID* adalah teknologi yang umum untuk melakukan tagging barang, inventarisasi barang di warehouse, dan juga *smart passport*. Hanya saja *NFC* lebih canggih dibandingkan *RFID* karena dilengkapi dengan kemampuan untuk integrasi dengan prosesor dan juga sistem operasi yang canggih seperti contohnya integrasi dengan sistem operasi Android.

Banyak manfaat kegiatan-kegiatan yang diperoleh dengan memanfaatkan teknologi NFC. Seiring perkembangan jaman, invoasi-inovasi terhadap pemanfaatan NFC ini juga ikut berkembang. Perkembangan inovasi-inovasi tersebut terbagi ke dalam 3 kategori:

1. *Service Initiation*

Pada kategori ini NFC berguna untuk melakukan layanan sebagai media untuk mendapatkan informasi tertentu atau dapat juga untuk membuka layanan yang mengharuskan autentifikasi data terlebih dahulu. Hal ini melibatkan NFC *Tag* sebagai media penyimpanan maupun penyampaian sementara. Pada *Tag* tersebut dapat berisi informasi teks, alamat sebuah *website* (*url),* nomor telepon atau data lain yang ukuran yang kecil.

Salah satu contoh penggunaannya yaitu pada poster pintar (*smart poster*). Dengan mendekatkan ponsel ada kepada NFC *Tag* pada poster, maka pengguna akan langsung mendapatkan informasi lebih detail pada yang belum tertera pada poster tersebut tanpa harus menginputkan apapun pada ponselnya.

2. *Peer-to-peer*

Pada kategori ini NFC berperan dalam mengkomunikasikan antara dua perangkat aktif dengan ukuran data yang relatif kecil. Namun untuk kebutuhan

data yang berukuran besar, NFC dapat digunaka sebagai sambungan nirkabel seperti *Bluetooth* dan WiFi.

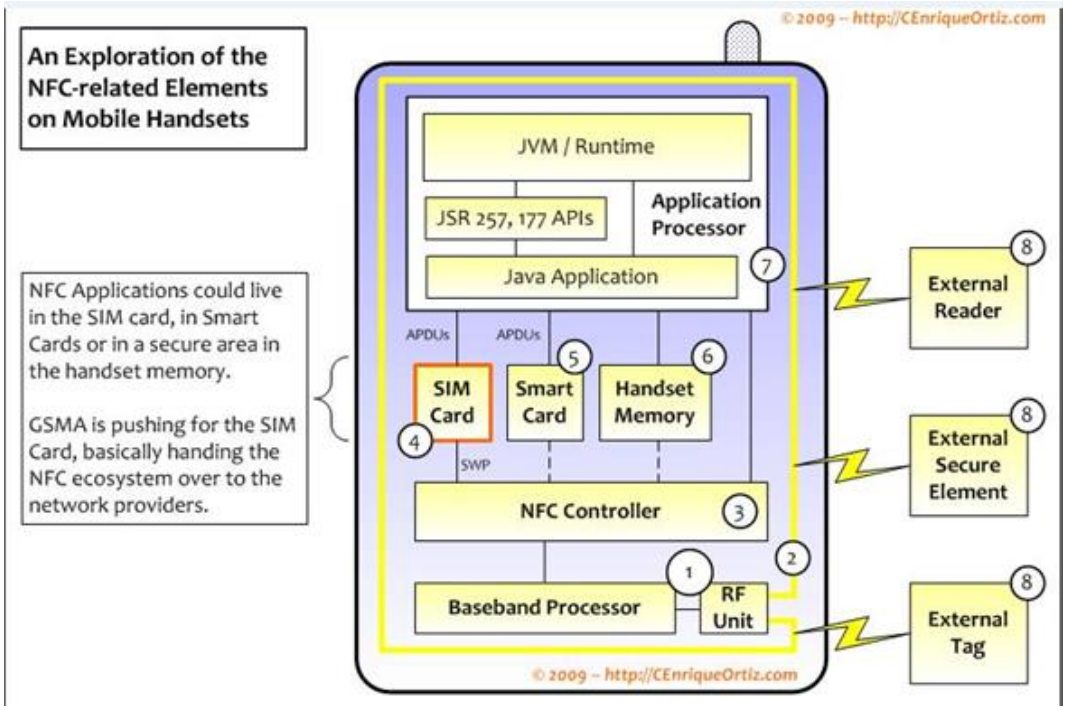
Contoh dari penggunaannya yaitu pada proses pencetakan foto langsung dari ponsel atau kamera. Pengguna tinggal memilih foto yang akan dicetak, kemudia mendekatakn ponselnya kepada printer yang sudah terintegrasi dengan teknologi NFC untuk membuat sambungan *bluetooth* untuk mengirimkan gambar dan mencetaknya.

3. *Payment*

Pada kategori ini NFC berperan dalam proses pembayaran suatu barang yang sudah terintegrasi sebelumnya dengan kartu kredit, bank dan operator jaringan lainnya. Dengan memanfaatkan teknologi NFC dalam pembayaran ini dapat mempermudah dan mempercepat proses transaksi pembayaran karena tidak menggunakan uang tunai maupun metode pembayaran konvensional lainnya.

2.6.1 Struktur *Near Field Communication* (NFC)

NFC akan bekerja ketika dua perangkat yang mendukung NFC bertemu dan salah satunya menjadi inisiator atau sebagai target. Perangkat NFC diantaranya adalah ponsel, kartu (tag) NFC dan NFC Reader. Pada gambar 2.1 menjelaskan tentang struktur NFC pada ponsel [16].



Gambar 2. 1 Struktur Sistem NFC Pada Mobile [16]

Keterangan Gambar :

1. Radio Frequency unit dan Baseband Processor.

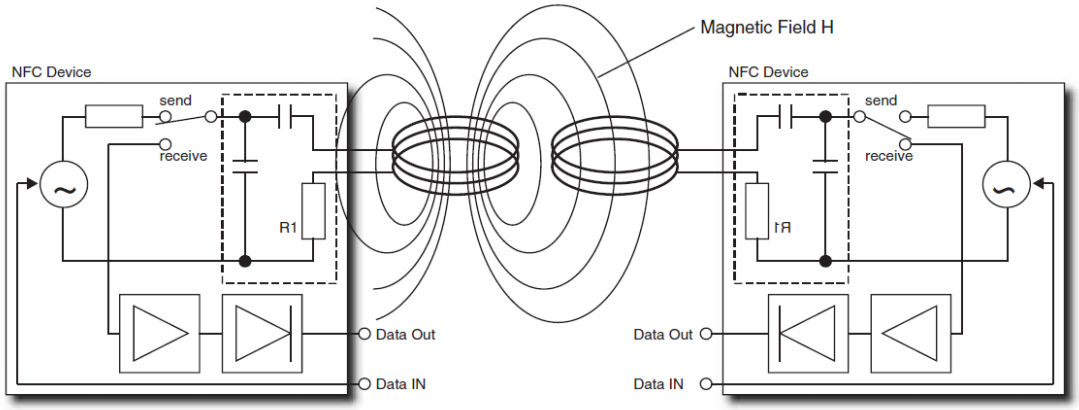
   Baseband Processor merupakan perangkat atau sebuah chip di dalam sebuah smartphone yang mengatur semua fungsi radio. Sedangkan Radio Frequency Unit merupakan sebuah perangkat yang menangkap sinyal masukkan dari baseband dan mengeluarkan sebuah sinyal frekuensi radio dan juga sebagai penghubung antar baseband dan antenna.
2. NFC proximity Antenna
3. NFC processor with security
4. SIM card, terhubung ke NFC processor dengan kabel tunggal.
5. Smart Card, komunikasi dilakukan pada ISO xxx APDUs
6. Memory handset

   Dalam hal ini, aplikasi NFC bisa terdapat pada SIM card, Smart card, maupun memory handset.

7. Application processor terdapat runtime, APIs, dan aplikasi NFC
   Dalam kasus ini sebuah runtime java seperti MIDP.
8. Perangkat luar atau perangkat lain seperti smart card, reader, dan tag.

2.6.2 Cara Kerja *Near Field Communication* (NFC)

NFC bekerja pada frekuensi 13.56 MHz dengan jarak komunikasi transmisi data mencapai 20 cm. Jarak yang dekat inilah yang menyebabkannya disebut Near-Field Communication (NFC). Gambar 2.2 menjelaskan prinsip dari



transmisi data pada NFC [17].

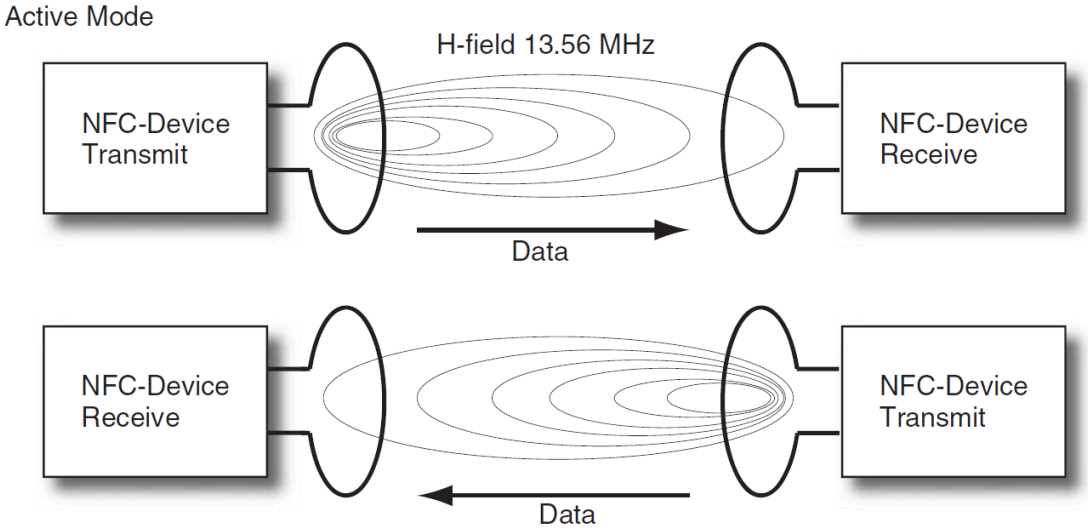
Gambar 2. 2 Transmisi Data NFC [17]

Sebuah interface NFC memiliki sebuah transmitter 13.56 MHz dan sebuah receiver 13.56 MHz yang terhubung ke antena secara bergantian. Dalam komunikasi antara 2 interface NFC, masing-masing interface mengambil fungsi tertentu antara NFC initiator (master device) atau NFC target (slave device) dimana komunikasi selalu dimulai oleh NFC initiator dan perpindahan data dilakukan dari NFC target ke NFC initiator. Komunikasi dapat dilakukan dalam 2 mode yaitu mode aktif dan mode pasif

1. Mode Aktif

   Pada mode aktif, NFC yang berperan sebagai initiator mengaktifkan transmiter yang dimiliki sehingga arus dengan frekuensi tinggi yang mengalir pada antena menimbulkan medan magnetik H yang menyebar ke

sekitarnya seperti terlihat pada Gambar 2.3. Sebagian akan bergerak dan tertangkap oleh antena interface NFC yang terletak pada lokasi berdekatan. Kemudian tegangan U dihasilkan di antena yang dapat dideteksi dan diterima oleh receiver pada NFC target.



Gambar 2. 3 Mode Active [17]

Cara kerja tersebut mirip dengan konsep RFID Reader dan transponder. Namun, terdapat perbedaan dimana pada transponder RFID reader harus memberikan suplai energi kepada transponder untuk mengoperasikan microchip untuk komunikasi data. Sementara pada NFC, perangkat elektronik yang memiliki fitur NFC yang memberikan suplai energi.

Pada mode aktif, pengiriman data dilakukan dari NFC target kepada NFC initiator dengan NFC target mengaktifkan transmitter dan NFC initiator berpindah mengaktifkan receiver. Sehingga pada mode ini kedua interface NFC bergantian menghasilkan medan magnetik.

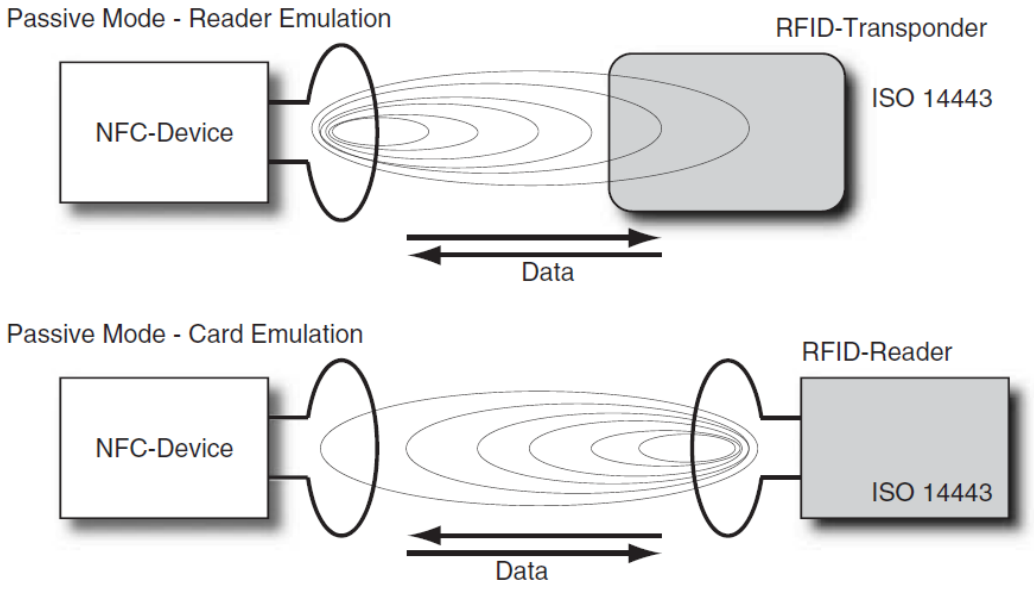
2. Mode Pasif

Pada mode pasif, NFC initiator juga menghasilkan medan magnetik untuk memulai komunikasi dengan NFC target. Namun setelah paket data sampai ke target, NFC initiator tidak mengganti mode menjadi penerima seperti

NFC mode aktif, melainkan tetap mengaktifkan transmitter. Sehingga trasmitter hanya diaktifkan pada NFC initiator.

Kemudian data pada NFC target akan “ditumpangkan” pada gelombang tersebut dengan menghasilkan yang disebut sebagai load modulation. Hal ini memberikan keuntungan dalam pengoperasiannya pada kondisi daya yang dimiliki oleh NFC target sedang lemah.

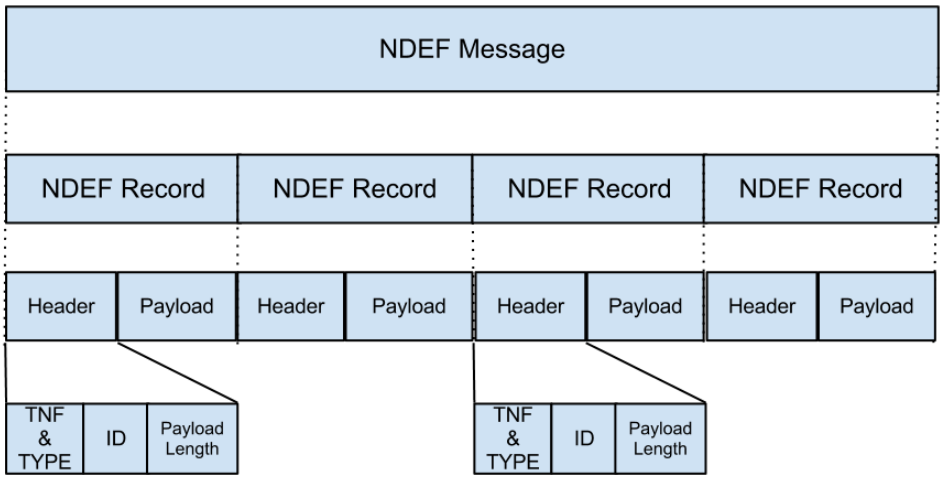
Hal ini juga memungkinkan interface NFC berfungsi seperti RFID Reader untuk berkomunikasi dengan transponder pasif dengan memberikan suplai daya kepada transponder dan menggunakan load modulation data ditransimisikan pada interface NFC. Misalnya : smartphone dengan fitur NFC dapat digunakan untuk membaca dan menulis pada transponder seperti smart label atau e-ticket. Aktivitas interface NFC yang menyerupai RFID Reader disebut juga ‘reader-emulation mode’.



Gambar 2. 4 Mode Pasif [17]

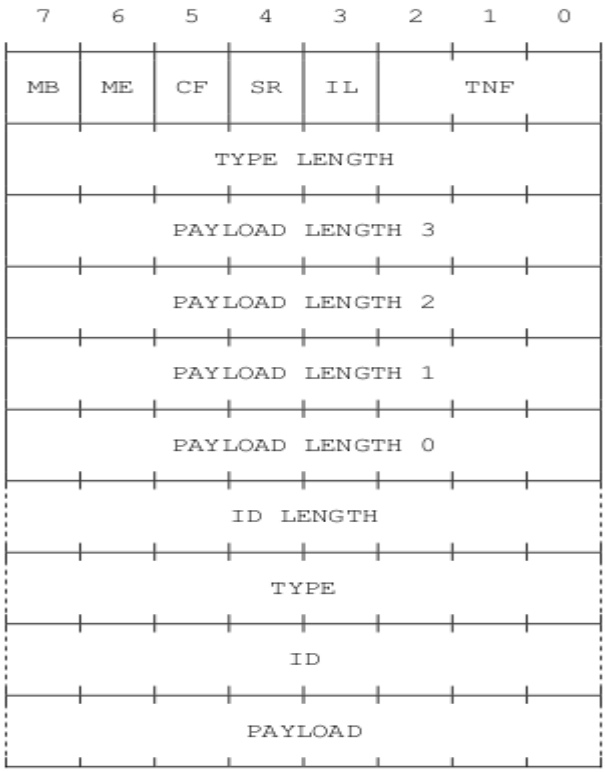
2.6.3 NFC *Data Exchange Format* (NDEF)

NFC Data Exchange Format (NDEF) adalah format pesan standar pada proses pertukaran informasi antara perangkat NFC ke perangkat NFC dan perangkat NFC ke *Tag* NFC. Di dalam NDEF terdapat NDEF *Message* yang terdiri dari beberapa *record* seperti pada gambar 2.5.



Gambar 2. 5 Struktur NDEF Message [16]

NDEF *Record* merupakan format umum dari panjang variabel *record.* Berikut adalah sttuktur NDEF *Record.*



Gambar 2. 6 Struktur NDEF Record [16]

Berikut adalah keterangan dari struktur NDEF *Record* :

a. MB (*Message Begin)* : Area 1 bit, dimana ketika *variabel* ini di set maka akan menunjukan awal NDEF *Message.*

b. ME (*Message End*) : Area 1 bit, dimana ketika variabel ini diset maka akan menunjukan akhir dari NDEF *Message.*

c. CF (*Chunk Flag*) : Area 1 bit, variabel ini digunakan untuk mengidentifikasi apakah potongan *record* pertama atau potongan tengah *record*.

d. SR (*Short Record*) : Area 1 bit, dimana jika diset maka PAYLOAD_LENGTH merupakan oktet tunggal. *Short Record* ini dimaksudkan untuk mengenkapsulasi *Payload* berukuran kecil yang cocok didalam area *Payload,* dengan kisaran ukuran 0 sampai 255.

e. IL (*Identification Length*) : Area 1 bit. Jika diset maka ID_LENGTH akan muncul di bagian *header* sebagai oktet tunggal. Jika ID_LENGTH

bernilai 0, ID_LENGTH dihilangkan dari *header record* dan area ID dihilangkan dari *record.*

f. TNF (*Type Name Format*) : Area bernilai 3 bit. Untuk mengidentifikasi struktur dari *value* TYPE.

Spesifikasi komunikasi data yang di standarisasi oleh NFC

Tabel 2. 1 Komunikasi data standarisasi oleh NFC

Spesifikasi	***Tujuan***
NFC Data Exchange Format (NDEF)	Format yang umum digunakan oleh perangkat NFC dan label.
NFC Record Type Definition (RTD)	Poster yang mengandung fasilitas NFC seperti teks, audio maupun data lainnya.
Text RTD	Untuk record yang hanya mengandung teks.
Uniform Resource Identifier (URI)	Untuk record yang mengacu ke Internet.
Connection Handover	Mendefinisikan bagaimana membangung komunikasi dengan teknologi wireless lainnya.
NFC Tag Types 1–4 Operation	Mendefinisikan tipe label yang diperintahkan oleh NFC Forum.
Logical Link Control Protocol (LLCP)	Mendukung operasi P2P untuk aplikasi berbasis NFC

Tabel 2. 2 Value Type Name Format

Type Name Format	***Value***
Empty	0x00
NFC Forum well-known type [NFC RTD]	0x01
Media-type as defined in RFC 2046 [RFC 2046]	0x02
Absolute URI as defined in RFC 3986 [RFC 3986]	0x03
NFC Forum external type [NFC RTD]	0x04
Unkown	0x05
Unchanged	0x06
Reserved	0x07

Keterangan tabel 2.2 *Value Type Name Format* :

1. 0x00 (Empty)

   Menunjukan bahwa tidak ada jenis atau payload yang terkait dengan *record*. Ketika digunakan TYPE_LENGTH, ID_LENGTH dan bidang PAYLOAD_LENGTH harus bernilai nol dan TYPE_ID dan PAYLOAD dihilangkan dari *record*. Nilai TNF ini dapat digunakan setiap kali *record* kosong diperlukan, misalnya untuk mengakhiri NDEF *message* ketika tidak ada PAYLOAD yang diinputkan oleh pengguna aplikasi.
2. 0x01 (NFC Forum wel-known type [NFC RTD])

   Menunjukan bahwa TYPE berisi nilai yang mengikuti *Type Name Format* RTD sesuai spesifikasi pada NFC Forum.
3. 0x02 (Absolute URI as defined in RFC 3986 [RFC 3986])

   Menunjukan bahwa TYPE berisi nilai yang sesuai dengan yang dibuat oleh BNF dan didefinisikan dalam RFC 2046.
4. 0x03 (Absolute URI as defined in RFC 3986 [RFC 3986])

   Menunjukan bahwa TYPE berisi nilai yang mutlak sesuai dengan URI BNF dan didefinisikan dalam RFC 3986.
5. 0x04 (NFC Forum external type [NFC RTD])

   Menunjukkan bahwa bidang TYPE berisi nilai yang mengikuti *Type Name Format* RTD yang didefinisikan dalam NFC Forum RTD spesifikasi [NFC RTD] untuk nama jenis eksternal.
6. 0x05 (Unkown)

   Menunjukan jika tipe PAYLOAD tidak diketahui. Hal ini serupa dengan "*application/octet-stream*" jenis media yang didefinisikan oleh MIME RFC 2046. Ketika digunakan TYPE_LENGTH harus bernilai nol. Maka dengan demikian TYPE dihilangkan dari NDEF *record*.
7. 0x06 (Unchanged)

   Hanya untuk *record* tengah dan mengakhiri *record* dari Chunked Payload. TYPE_LENGTH harus bernilai nol.
8. 0x07 (Reserved)

   Milik NFC Forum untuk digunakan di masa depan.

g. TYPE_LENGTH : 8 bit integer yang menentukan panjang oktet dari TYPE.
h. ID_LENGTH : 8 bit integer yang menentukan panjang oktet dalam ID. ID_LENGTH muncul apabilan IL diset menjadi 1 di dalam *record header*. ID_LENGTH boleh bernilai 0 oktet namun ID harus dihilangkan dari NDEF *record*.
i. PAYLOAD_LENGTH : Yang menentukan panjang oktet dari PAYLOAD.
j. TYPE : Identifikator yang menjelaskan jenis nilai PAYLOAD. Nilai dari TYPE harus sesuai dengan struktur, encoding dan format pada TNF.
k. ID : Identifier dalam bentuk resferensi URI yang dijelaskan dalam RFC 3986. Referensi URI dapat bersifat relative dan absolut karena NDEF tidak mendefiniskan basis URI.
l. PAYLOAD : Konten yang dapat berisi apa saja dalam ukuran beberapa bytes.

2.6.4 *Local Link Control Protocol* (LLCP)

LLCP adalah protokol data link untuk mendukung komunikasi peer-to-peer antara dua perangkat yang berkemampuan NFC. LLCP sangat penting untuk setiap aplikasi NFC yang melibatkan komunikasi dua arah. Hal ini juga meningkatkan fungsi dasar yang disediakan oleh protokol NFCIP-1. Protokol NFCIP-1 mempunyai kemampuan *Segmentation and Reassembly* (SAR), serta kontrol aliran data, tergantung pada " *Go and Wait* " prinsip biasa untuk protokol half-duplex [17].

LLCP menyediakan lima layanan penting yaitu, *connectionless transport, connection-oriented transport, link activation, supervision, and deactivation, asynchronous balanced communication,* dan *protocol multiplexing*. Berikut adalah cara kerja dari masing-masing layanan yang dimiliki oleh LLCP [17]:

1. *Connectionless Transport* menyediakan layanan transmisi data yang tidak diakui. Mode transport ini dapat digunakan jika lapisan protokol bagian atas menerapkan mekanisme sendiri pengendali alirannya. Oleh karena itu, lapisan ini tidak perlu bergantung pada mekanisme lapisan data link kontrol aliran lainnya.

2. *Connection-Oriented Transport* Mode ini menyediakan layanan transmisi data dengan pengiriman sequencing dan menjamin unit data pelayanan pengguna. Transmisi data dikendalikan melalui *sliding window protocol*.
3. *Link activation, Supervision, and Deactivation* LLCP menentukan bagaimana dua perangkat NFC berkomunikasi dengan jarak LLCP yang sesuai, membangun link LLCP, mengawasi koneksi ke perangkat yang jauh, dan menonaktifkan link ketika diminta.
4. *Asynchronous Balanced Communication* Tipe NFC MACs (*Medium Access and Control*) beroperasi dalam Mode Respon normal di mana hanya master yang disebut inisiator, diperbolehkan untuk mengirim data ke target dan juga meminta data dari tag.
5. *Protocol Multiplexing* LLCP ini mampu menampung beberapa contoh protokol tingkat yang lebih tinggi pada waktu yang sama.

2.7 *Tag* NFC

Tag NFC merupakan suatu objek yang kecil seperti stiker yang berisi informasi. Informasi ini dapat diambil oleh perangkat ber-NFC lainnya. NFC tag biasanya bersifat read-only. Tetapi pada beberapa tag, informasinya dapat diedit.

Perangkat NFC berkomunikasi satu sama lain dengan membuat frekuensi radio. Agar dapat berinteraksi efektif, pembuatan perangkat NFC harus mengikuti standar yang telah ditentukan oleh forum NFC. Dua spesifikasi utama NFC adalah ISO/IEC 14443 dan ISO/IEC 18000-3. ISO/IEC 14443 adalah kartu identitas yang digunakan untuk menyimpan informasi, sedangkan ISO/IEC 18000-3 adalah standar komunikasi RFID yang digunakan oleh perangkat NFC. Karena menggunakan standar ISO/IEC 18000-3, NFC berkomunikasi secara wireless pada frekuensi 13,56 MHz menggunakan kartu Tipe A atau Tipe B, atau FeliCa (umumnya ada di Jepang), dengan jarak perangkat kurang lebih 4cm. Pengiriman data NFC dapat dilakukan pada kecepatan 106, 212, atau 424 kb per detik.



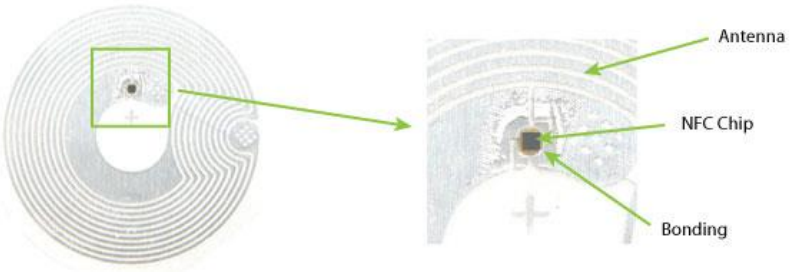
Gambar 2. 7 NFC *Tag*

Ada 4 tipe tag dan 2 set role aktif/pasif pada NFC. Tipe tag mengarah pada kecepatan dan kesesuaian antara NFC tag dan NFC reader. Role adalah untuk mendefinisikan seberapa aktif atau pasif suatu perangkat selama terjadi komunikasi NFC. Berikut merupakan tipe-tipe tag pada nfc [17]

1. Tipe 1 : didasari pada ISO/IEC 14443A. Tags mampu membaca dan menulis ulang. Pengguna dapat mengatur tag menjadi read-only. Memori yang tersedia adalah 96 bytes dan dapat di upgrade hingga 2 Kbyte.
2. Tipe 2 : didasari pada ISO/IEC 14443A. Tags mampu membaca dan menulis ulang. Pengguna dapat mengatur tag menjadi read-only. Memori yang tersedia adalah 48 bytes dan dapat di upgrade hingga 2 Kbyte.
3. Tipe 3 : didasari pada standar industri Jepang atau *Japanese industrial standard* (JIS) X 6319-4, atau disebut Felica. Tag diatur pada pembuatannya untuk mendukung baik dalam *read* dan *re-writeable*, ataupun *read-only*. Memory yang tersedia dapat diubah, secara teoris batas memorinya adalah 1 Mbyte per service.
4. Tipe 4 : kompatibel sepenuhnya dengan ISO/IEC 14443 standard *series*, memiliki memori 32 KB dan lebih cepat daripada tipe tag lainnya, memiliki proteksi terhadap data collision, dapat menggunakan komunikasi NFC-A maupun NFC-B. Tipe 4 dapat ditulis ulang *(rewriteabl*e) atau *read-only*. Tetapi pengaturan tersebut dilakukan saat tipe 4 dalam tahap pembuatan. Saat sudah selesai dibuat, pengaturannya tidak dapat diubah.

2.7.1 Struktur dan Cara Kerja *Tag* NFC

Struktur dari NFC *Tag* terdiri dari *Antenna,* NFC *Chip,* dan Bonding untuk lebih jelas bisa dilihat pada Gambar 2.8.



Gambar 2. 8 Struktur *Tag* NFC [1]

Pada *Tag* NFC terdapat perangkat kecil dengan *microchip* yang menempel pada antena. Perangkat *microchip* tersebut dapat melakukan penyimpanan informasi yang akan di transferkan ke perangkat lain. *Tag* NFC sangat kecil ukurannya sehingga cukup untuk dimasukan ke dalam segala macam produk seperti stiker, gantungan kunci, label dan kartu nama. *Tag* NFC bersifat pasif yang artinya tidak terdapat baterai ataupun sumber listrik lain didalamnya, *tag* NFC menarik listrik dari perangkat yang dapat membacanya melalui induksi magnet ketika *tag* NFC tersebut di dekatkan dengan perangkat pembaca [1].

Tag NFC bekerja dengan perangkat aktif salah satu contohnya adalah ponsel yang menghasilkan medan magnet yang menginduksi arus listrik pada antena *tag* NFC yang tertanam pada chip NFC.

Berikut adalah cara kerja pada setiap komponen yang terdapat pada *Tag* NFC [10]

1. Antena

   Antena pada *Tag* NFC terbuat dari alumunium atau tembaga. Antena pada *tag* NFC sebenarnya hanya indukator yang di rancang untuk mengubah medan magnet yang berada didekatnya menjadi energi. Antena ini dirancang untuk bekerja pada frekuensi 13.56Mhz.

Kinerja antena tidak di pengaruhi oleh ketebalan *tag* NFC. Namun dengan medan magnet yang kuat semakin besar ukuran antena semakin baik kinerja *tag* NFC tersebut. Pada setiap ponsel tidak menghasilkan kekuatan yang banyak jadi ada batasan untuk ukuran optimum antena.

2. Bonding

   Bonding ini adalah penghubung antara chip kecil dan antena pada *tag* NFC.

2.8 Aplikasi *Mobile*

Aplikasi *mobile* adalah sebuah aplikasi yang memungkinkan untuk melakukan mobilitas dengan menggunakan perlengkapan seperti telepon genggam (*hanphone*), PDA, atau *smartphone*. Aplikasi *mobile* dapat mengakses dan menggunakan sebuah aplikasi secara *nirkabel* dengan menggunakan perangkat *mobile*, dimana data yang diperoleh hanya berupa teks sehingga tidak perlu membutuhkan *bandwidth* yang terlalu besar. Data dalam bentuk teks tidak semenarik data dalam bentuk teks disertai dengan gambar. Namun penggunaan data teks adalah yang paling banyak digunakan saat ini. Penggunakan aplikasi *mobile* hanya memerlukan *handphone* yang sudah dilengkapi dengan fasilitas GPRS dan koneksinya [2].

Terdapat beberapa aspek yang harus diperhatikan khususnya perangkat keras dalam membangun sebuah aplikasi *mobile*. Selain itu pertimbangan terhadap keterbatasan pada piranti *mobile* pun harus di perhatikan, antara lain:

1. Keterbatasan dari kecepatan *processor* dalam mengeksekusi program
2. Keterbatasan memori penyimpanan dan RAM
3. Perbedaan ukuran layar secara fisik dan resolusi pada masing-masing piranti
4. Keterbatasan input pada masing-masing piranti mobile
5. Ketahanan baterai yang berbeda pada setiap piranti mobile
6. Selain itu dari segi perangkat lunak (*software*), kompatibilitas dan *mobile* pendukung cukup berpengaruh dalam menjalankan sebuah aplikasi *mobile*.

2.9 Android

Android adalah sebuah sistem operasi berbasis *Linux kernel* dan dirancang untuk perangkat *mobile* dengan layar sentuh seperti *smartphone* dan komputer tablet. Awal mulanya dikembangkan oleh Android, Inc., yang didukung oleh Google secara finansial dan kemudian dibeli pada tahun 2005. Android ini diresmikan pada tahun 2007 seiring dengan berdirinya *Open Handset Alliance* yang merupakan konsorsium dari *hardware*, *software* dan perusahaan telekomunikasi yang ditujukan untuk memanjukan standar terbuka untuk perangkat *mobile.*

2.9.1 *Version History*

Version history dari sistem operasi *mobile* Android dimulai dari Android *beta* yang dirilis pada Nopember 2007. Android 1.0 yang merupakan versi komersil pertama, dirilis pada September 2008. Android berada di bawah pengembangan Google dan OHA (*Open Handset Alliance*) dan memiliki banyak perbaharuan pada sistem operasinya sejak pertama kali dirilis. Tabel 2.3 merupakan sejarah versi dari Android berdasarkan API levelnya [11].

Tabel 2. 3 Versi Android Berdasarkan API level

API Level	Versi	Code-Name	Tanggal Rilis
API Level 1	1.0		28 September 2008
API Level 2	1.1		9 Februari 2009
API Level 3	1.5	Cupcake	30 April 2009
API Level 4	1.6	Donut	15 September 2009
API Level 5	2.0	Éclair	26 Oktober 2009
API Level 6	2.0.1		3 Desember 2009
API Level 7	2.1		12 Januari 2010
API Level 8	2.2	Froyo	20 May 2010
	2.2.1		18 Januari 2011
	2.2.2		22 Januari 2011
	2.2.3		21 November 2011
API Level 9	2.3	Gingerbread	6 Desember 2010
	2.3.1		Desember 2010
	2.3.2		Januari 2011
API Level 10	2.3.3		9 Februari 2011

	2.3.4		28 April 2011
	2.3.5		25 Juli 2011
	2.3.6		2September 2011
	2.3.7		21 September 2011
API Level 11	3.0	Honeycomb	22 Februari 2011
API Level 12	3.1		10 Mei 2011
API Level 13	3.2		15 Juli 2011
	3.2.1		20 September 2011
	3.2.2		30 Agustus 2011
	3.2.3		
	3.2.4		Desember 2011
	3.2.5		Januari 2012
	3.2.6		Februari 2012
API Level 14	4.0	Ice Cream Sandwich	19 Oktober 2011
	4.0.1		21 Oktober 2011
	4.0.2		28 November 2011
API Level 15	4.0.3		16 Desember 2011
	4.0.4		29 Maret 2012
API Level 16	4.1	Jelly Bean	9 Juli 2012
	4.1.1		23 July 2012
	4.1.2		9 Oktober 2012
API Level 17	4.2		13 November 2012
	4.2.1		27 November 2012
	4.2.2		11 February 2013
API Level 18	4.3		24 Juli 2013
	4.3.1		3 Oktober 2013
API Level 19	4.4	KitKat	31 Oktober 2013
	4.4.1		5 Desember 2013
	4.4.2		9 Desember 2013
API Level 20	4.4.3		2 Juni 2014
	4.4.4		19 Juni 2014

2.9.2 Fitur Android

Android memiliki banyak sekali fitur. Fitur-fitur pada Android dapat diklasifikasikan sebagai berikut [5].

1. *General*
   1) *Messaging*

      SMS dan MMS tersedia dalam bentuk pesan, termasuk pesan teks *threaded* dan Google *Cloud Messaging* (GCM).

   2) *Web browser*

      Web browser yang tersedia pada Android adalah berbasis Blink *open-source layout engine*, ditambah dengan Chrome's V8 JavaScript Engine.

   3) *Voice Based features*

      Pencarian Google melalui suara telah tersedia sejak awal rilis. Perintah suara untuk perintah lainnya didukung pada Android 2.2 dan seterusnya.

   4) *Multi-touch*

      Android memiliki dukungan asli untuk *multi-touch* yang awalnya tersedia pada *handset* seperti HTC Hero.

   5) *Multitasking*

      Tersedianya *multitasking* bagi aplikasi dengan penanganan alokasi memori yang unik.

   6) *Screen capture*

      Android mendukung pengambilan *screenshot* dengan cara menekan tombol *power* dan *volume-down* secara bersamaan.

   7) *Video calling*

      Android tidak mendukung panggilan video asli, tetapi beberapa perangkat memilki versi kostumisasi dari sistem operasi yang mendukung, baik melalui jaringan UMTS ataupun melalui IP (VoIP). Panggilan video melalui applikasi lainnya telah tersidia saat ini seperti Skype, Hangouts (Google Talk), Vibe, dan aplikasi lainnya.

   8) *Multi language support*

      Android mendukung multi bahasa.

9) *Accessibility*

Built in text to speech disediakan melalui TalkBack bagi orang-orang dengan gangguan penglihatan. Penambahan fitur bagi orang-rang dengan gangguan pendengaran tersedia seperti alat bantu lainnya.

2. *Connectivity*

1) *Connectivity*

Android mendukung teknologi konektifitas termasuk GSM/EDGE, Wi-Fi, Bluetooth, LTE, CDMA, EV-DO, UMTS, NFC, IDEN dan WiMax.

2) Bluetooth

Mendukung panggilan suara dan mengirim kontak antar telepon, mengirim file (OOP), mengakses buku telepon (PBAP), A2DP dan AVRCP. Dukungan keyboard, mouse dan joystick (HID) tersedia pada Android versi 3.1 keatas dan pada versi sebelumnya melalui kostumisasi produsen dan aplikasi pihak ketiga.

3) *Tethering*

Android mendukung *tethering*, yang memungkinkan ponsel untuk berbagi konektifitas internet. Dapat dilakukan dengan media kabel ataupun nirkabel sebagai Wi-Fi hotspot. Sebelum Android 2.2, fitur ini didukung oleh aplikasi pihak ketiga atau kostumisasi pabrikan.

3. Media

1) *Streaming* media *support*

RTP / RTSP *streaming* (3GPP PSS, ISMA), progresif *download* HTML (HTML5 <video> tag). Adobe Flash *Streaming* (RTMP) dan *Streaming* HTTP Dinamis didukung oleh plugin Flash. Apple HTTP *Live Streaming* didukung oleh RealPlayer for Android, dan oleh sistem operasi Android 3.0 (Honeycomb) keatas.

2) Media *support*

Android mendukung format media audio/video sebagai berikut: WebM, H.263, H.264, AAC, HE-AAC (in 3GP or MP4 *container*), MPEG-4 SP, AMR, AMR-WB (in 3GP *container*), MP3, MIDI, Ogg Vorbis, FLAC, WAV, JPEG, PNG, GIF, BMP, WebP.

3) *External Storage*

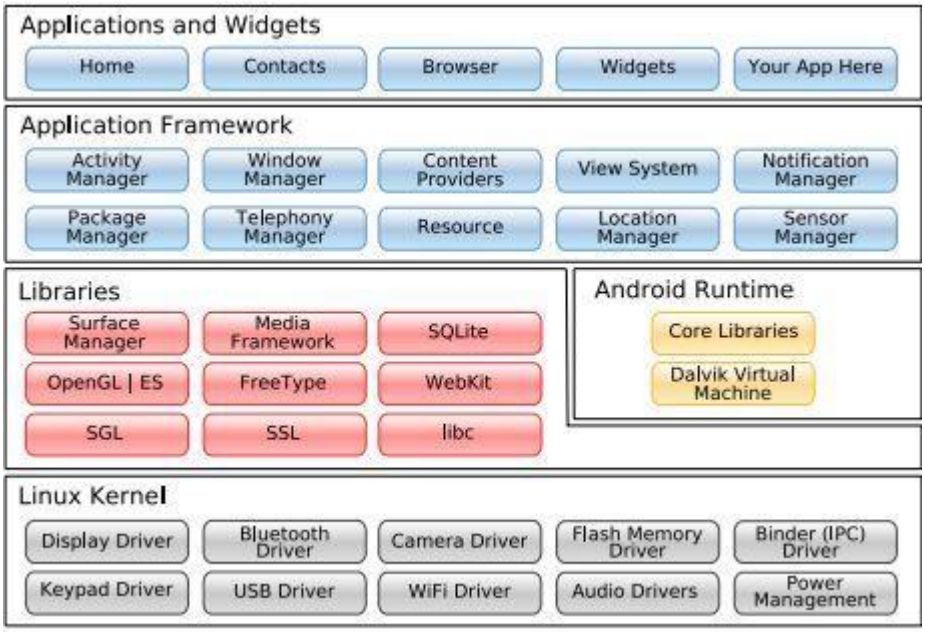
Kebanyakan perangkat Android menyertakan slot microSD dan dapat membaca kartu microSD dengan format Fat32, Ext3 atau sistem file Ext4. Untuk memungkinkan penggunaan media penyimpanan berkapasistas tinggi seperti USB *flash drive* dan HDD USB, banyak tablet Android yang juga menyertakan USB 'A' *receptable* yang biasa disebut USB *On-The-Go* atau USB *host*.

4. *Hardware Support*

Perangkat Android dapat menyertakan kamera video, layar sentuh, GPS, akselerometer, giroskop, barometer, magnetometer, kontrol *game* khusus, sensor kedekatan dan tekanan, thermometer, 2D *bit blits* terakselerasi (dengan orientasi perangkat keras, *scaling*, konversi format piksel) dan grafis 3D terakselerasi.

2.9.3 Arsitektur Android

Secara garis bersar Arsitektur Android dapat dijelaskan dan digambarkan pada gambar 2.9 [5]:



Gambar 2. 9 Arsitektur Android [5]

1. *Applications* dan *Widgets*

   Applications dan *Widgets* ini adalah *layer* dimana pengguna berhubungan dengan aplikasi saja, dimana biasanya pengguna men-*download* aplikasi kemudian melakukan instalasi dan menjalankan aplikasi tersebut. Di *layer* terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Applications Frameworks*

   Android adalah "*Open Development Platform*" yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi *resources*, menjalankan *service background*, mengatur *alarm*, menambahkan status *notificatios*, dan sebagainya. Pengembang memiliki akses penuh menuju API *framework* seperti yang dilakukan oleh aplikasi pada kategori inti. Arsitektur aplikasi dirancang supaya pengembang dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (*reuse*).

   Sehingga pengembang bisa menyimpulkan *Applications Frameworks* ini adalah *layer* dimana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada *layer* inilah aplikasi dapat dirancang, seperti *contont-providers* yang berupa sms dan lain sebagainya.

   Komponen-komponen yang termasuk didalam *Applications Frameworks* diantaranya *Views*, *Content Provider*, *Resouce Manager*, *Notification* Manager, dan Activity *Manager.*

3. *Libraries*

   Libraries ini adalah *layer* dimana fitur-fitur Android berada, biasanya para pembuat aplikasi kebanyakan mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas kernel, *layer* ini meliputi berbagai *library* C/C++ inti seperti sebagai libc dan SSL, serta:

   a) *libraries* media untuk pemutaran media audio dan video

b) *libraries* untuk manajemen tampilan
c) *libraries Graphics* mencakup SGL dan OpenGL untuk grafis 2D dan 3D
d) *libraries* SQLite untuk dukungan *database*
e) *libraries* SSL dan *WebKit* terintegrasi dengan *web browser* dan *security*
f) *libraries* LiveWebcore mencakup *modern web browser* dengan *engine embeded web view*
g) *libraries* 3D yang mencakup implementasi OpenGL ES 1.0 API's.

4. *Android Run Time*

   Layer yang membuat aplikasi Android dapat dijalankan dimana dalam prosesnya menggunakan Implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Didalam Android *Run Time* dibagi menjadi dua bagian yaitu:

   a) *Core Libraries*: Aplikasi Android dibangun dalam bahasa java, sementara Dalvik sebagai virtual mesinya bukan Virtual Machine Java, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menerjemahkan bahasa java/c yang dihandle oleh *Core Libraries.*
   b) *Dalvik Virtual Machine*: Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat linux kernel untuk *thereading* dan manajemen tingkat rendah.

5. Linux *Kernel*

   Linux *kernel* adalah *layer* dimana inti dari *operating sistem* dari Android itu berbeda. Berisi *file-file system* yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem-sistem operating sistem android lainnya. Linux *karnel* yang digunakan android adalah linux *karnel* 2.6.

2.9.4 Android *Software Development Kit* (Android SDK)

Android SDK adalah *tools* API (*Aplication Programming Interface*) yang diperlukan untuk memulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. 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 sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java.

Beberapa fitur Android yang paling penting diantaranya:

1. *Framework* Aplikasi yang mendukung penggantian komponen dan *reusable*.
2. Mesin *Virtual Dalvik* dioptimalkan untuk perangkat *mobile*
3. *Integrated browser* berdasarkan *engine open source WebKit*
4. Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi OpenGL ES 1.0 (Opsional akselerasi *hardware*)
5. SQLite untuk penyimpanan data
6. *Media Support* yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM *Telephony* (tergantung *hardware*)
7. *Bluetooth*, EDGE, 3G, dan WiFi (tergantung *hardware*)
8. Kamera, GPS, kompas, dan *accelerometer* (tergantung *hardware*)
9. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging*, profil, dan kinerja memori, dan *plugin* untuk IDE *Eclipse*.

2.9.5 *Java Development Kit* (JDK)

JDK (*Java Development Kit*) adalah sebuah perangkat peralatan yang digunakan untuk membangun perangkat lunak dengan menggunakan bahasa pemrograman Java. JDK berjalan diatas sebuah *virtual machine* yang dinamakan JVM (*Java Virtual Machine*) [9].

Didalamnya terdapat tiga komponen utama, yaitu Java API, Java *Virtual Machine*, dan Java *Compiler*. Dokumentasi JDK berisi spesifikasi API, deskripsi fitur, panduan pengembang, referensi halaman untuk perkakas JDK dan utilitas, demo, dan *link* ke informasi terkait. Dokumentasi ini juga tersedia dalam bundel yang dapat diunduh dan diinstal pada komputer. JDK merupakan perangkat

minimal yang harus dimiliki oleh pemograman Java. Berikut adalah nama-nama folder yang biasanya terdapat setelah menginstal JDK [19].

Tabel 2. 4 Folder dalam JDK

Nama Folder	***Keterangan***
Bin	Berisi kompiler dan *file-file* program lainnya
Demo	Berisi program demo yang dapat digunakan untuk mempelajari Java
Include	Berisi *file header* C untuk *native code*
Jre	Berisi *file* dan *folder* untuk Java *Runtime Environment*
Lib	*File-file library* yang digunakan sebagai API
Sample	Berisi program demo yang dapat digunakan untuk mempelajari Java
Src	Berisi *source code* dari *library*, berada dalam file .zip

2.9.6 Eclips *Integrated Development Environment* (Eclips IDE)

IDE adalah program komputer yang memiliki beberapa fasilitas yang diperlukan dalam pembangunan perangkat lunak. Tujuan dari IDE adalah untuk menyediakan semua utilitas yang diperlukan dalam membangun perangkat lunak. Eclipse adalah sebuah IDE untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform-independent*) [3].

2.9.7 *Android Development Tools* (ADT)

ADT adalah plugin yang didesain untuk IDE *Eclipse* yang memberikan kemudahan dalam mengembangkan aplikasi android dengan menggunakan IDE Eclipse. Dengan menggunakan ADT untuk *Eclipse* akan memudahkan dalam membuat aplikasi *project* android, membuat GUI aplikasi, dan menambahkan komponen-komponen lainnya, selain itu dapat digunakan untuk menjalankan aplikasi menggunakan Android SDK melalui *Eclipse*. Dengan ADT dapat melakukan pembuatan package android (.apk) yang digunakan dalam distribusi apliaksi android.

2.10 Java

Inovasi bahasa komputer dimotivasi oleh dua faktor: perbaikan dalam seni pemrograman dan perubahan dalam lingkungan komputasi, tidak terkecuali Java [8]. Dibangun di atas warisan yang kaya dari C dan C + +, Java menambahkan perbaikan dan fitur yang mencerminkan keadaan seni dalam pemrograman saat ini. Menanggapi munculnya lingkungan online, Java menawarkan fitur yang merampingkan pemrograman untuk arsitektur yang sangat terdistribusi.

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM).

Versi awal Java ditahun 1996 sudah merupakan versi rilis sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

1. java.lang: Peruntukan kelas elemen-elemen dasar.
2. java.io: Peruntukan kelas input dan output, termasuk penggunaan berkas.
3. java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
4. java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI)
6. java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

Seperti telah dibahas sebelumnya, banyak jenis komputer dan sistem operasi yang terhubung ke Internet. Untuk program-program untuk secara dinamis didownload ke semua berbagai jenis platform, beberapa sarana untuk menghasilkan kode dieksekusi portabel diperlukan. Mekanisme yang sama yang

membantu menjamin keamanan juga membantu menciptakan portabilitas karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang populer digunakan dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Java memliki beberapa kelebihan dibandingkan dengan bahasa pemograman lain, diantaranya:

1. Multiplatform.

   Kelebihan utama dari Java ialah dapat dijalankan di beberapa platform / sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan.

2. OOP (*Object Oriented Programming*)

   Object Oriented Programming merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Berbeda dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

   Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

3. *Library Class* yang Lengkap.

   Java terkenal dengan kelengkapan *library* (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan *library* ini ditambah dengan keberadaan komunitas Java

yang besar yang terus menerus membuat *library* baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Mewarisi kekayaan C/C++.

   Memiliki sintaks seperti bahasa pemrograman C/C++ dan mewarisi kekayaan yang dimiliki C/C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java.

5. Pengumpulan sampah otomatis

   Memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

Selain dari beberapa kelebihan yang dimiliki oleh java, terdapat pula beberapa kekurangan yang dimiliki java, diantaranya:

1. Mudah didekompilasi.

   Dekompilasi adalah proses membalikkan hasil kompilasi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan bytecode yang menyimpan banyak atribut bahasa tingkat tinggi. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/direverse-engineer.

2. Penggunaan memori yang banyak.

   Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal. Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru, tetapi bagi komputer yang telah berumur lebih dari 4 tahun akan menyebabkan masalah, karena ketersesiaan memori yang terbatas.