BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1. Tinjauan Studi

Terdapat beberapa penelitian yang berkaitan dengan penelitian ini. Namun dari beberapa penelitian terkait, ada dua penelitian terkait yang menjadi acuan dalam penelitian ini yaitu :

1. Penelitian yang dilakukan oleh Kishor S. Wagh dan R.C. Thool, Ph.D. Department of Information Technology SGGS College of Engineering and Technology, Nanded, dengan judul "Web Service Provisioning on Android Mobile Host". Pada penelitian tersebut dilakukan integrasi serta pertukaran data antara webservice sebagai server dan android sebagai client. Dari hasil pengujian dan implementasi menunjukkan bahwa proses integrasi data antara client dan server telah *compatible* (sesuai) dan tidak memiliki tampilan interface yang berbeda sehingga user mudah dalam memahami. Pembuatan webservice pada penelitian tersebut menggunakan arsitektur RESTFULL, dimana jenis arsitektur ini menggunakan method berupa GET, PUT, DELETE, HEAD, dan POST. Berikut adalah tabel hasil *performance* dari webservice :

Tabel 2.1 Hasil Performance dari webservice

Load Level	Total Request	Total Respon	Jumlah Sukses Respon	%error respon	Rata-rata waktu eksekusi (ms)
1	57	57	57	0.0%	28
5	107	107	107	0.0%	32
10	122	122	122	0.0%	24
15	136	136	136	0.0%	34
25	143	143	143	0.0%	67

Tabel tersebut menunjukkan bahwa webservice berjalan lancar dengan presentase error sebesar 0.0%. [1]

2. Penelitian yang dilakukan oleh Santosh M.Herur dan Prof. Vinayak P. Miskin, Department of Electronics and Communication Engineering, SDM College of Engineering and Technology Dharwad, dengan judul Efficient Wireless Thermal Printing From Android Application.

   Penelitian tersebut menghasilkan sebuah aplikasi android yang terhubung pada device printer melalui sambungan tanpa kabel (*wireless*) yaitu Bluetooth. Hasil dari pengujian menunjukkan bahwa aplikasi android sukses terhubung dengan printer dan dapat mencetak data berupa tulisan maupun gambar.[4]

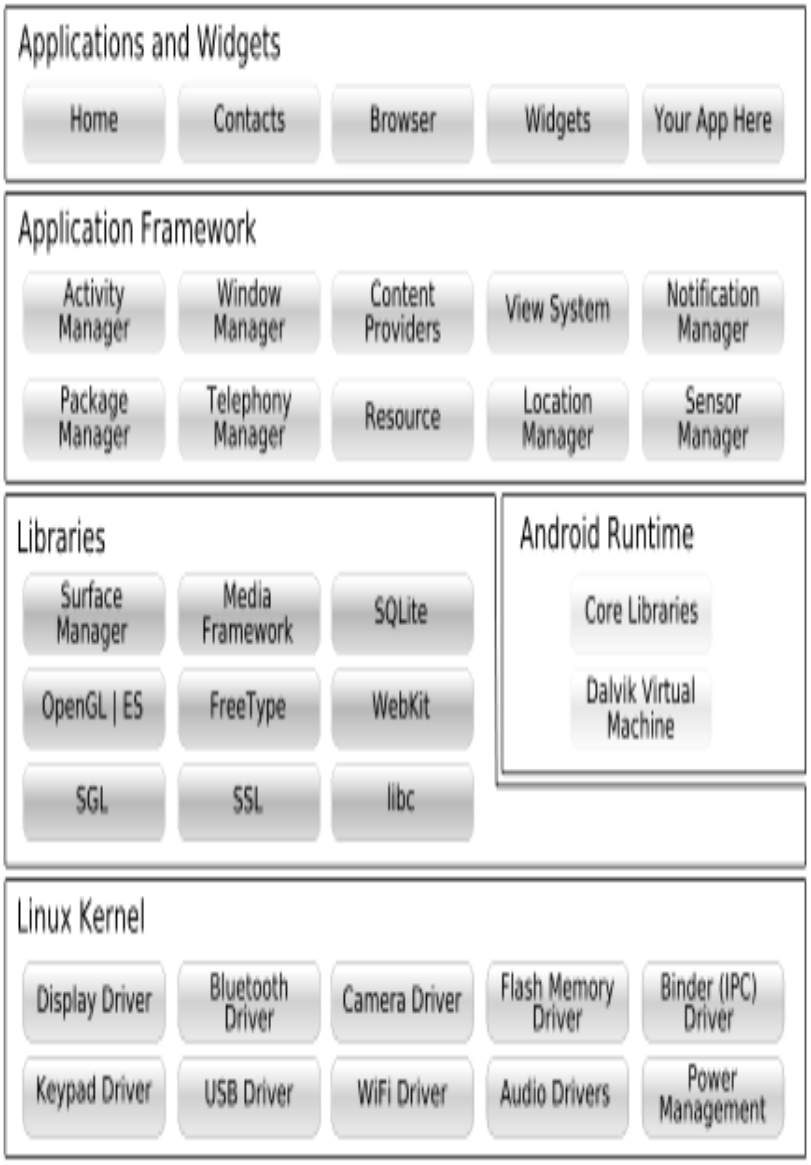
2.2. Tinjauan Pustaka

2.2.1. Android

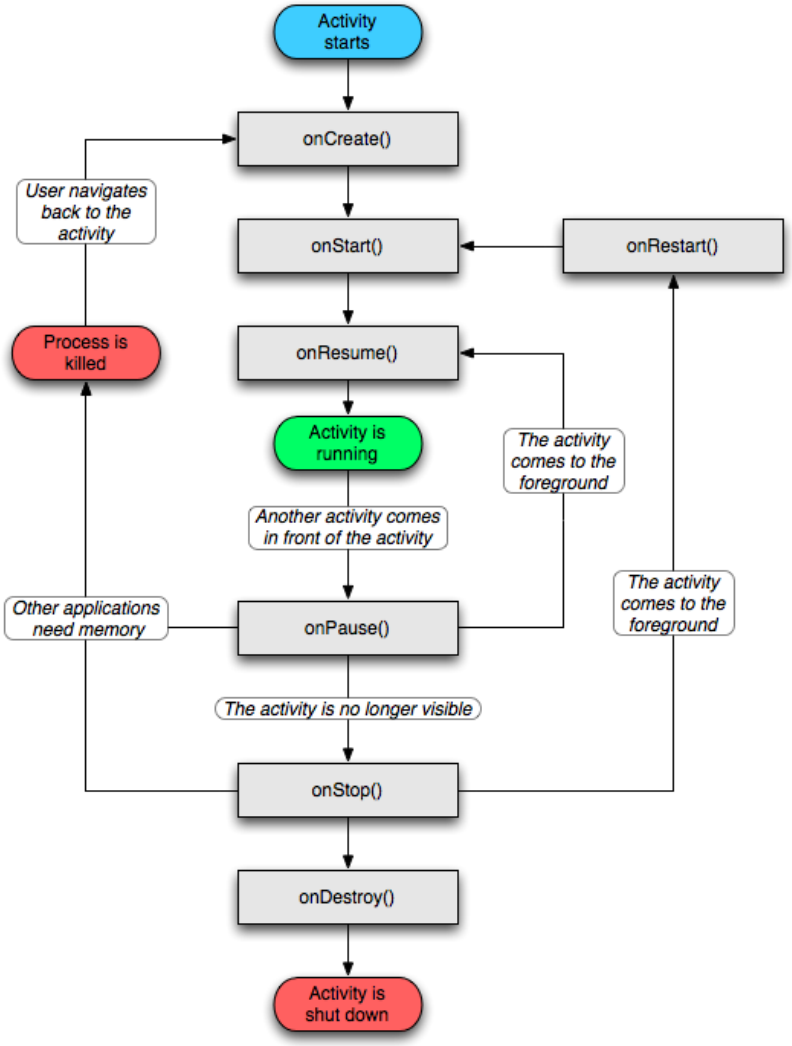
2.2.1.1 Definisi Android

Android merupakan sistem operasi berbasis java yang berjalan pada kernel 2.6 Linux. Aplikasi Android yang dikembangkan menggunakan bahasa pemrograman Java dan mudah menyesuaikan ke platform baru [5]. Android merupakan satu kumpulan lengkap perangkat lunak yang dapat berupa sistem operasi, *middleware,* dan aplikasi kunci perangkat *mobile.* Android terdiri dari satu tumpukan yang lengkap, mulai dari *boot loader, device driver,* dan fungsi-fungsi pustaka, hingga perangkat lunak API (*Application Programming Interface*), termasuk aplikasi SDK (*Software Development Kit*). Jadi sebenarnya Android bukanlah satu perangkat tertentu, melainkan sebuah platform yang dapat digunakan dan diadaptasikan untuk mendukung berbagai konfigurasi perangkat keras. Walaupun kelas utama perangkat yang didukung Android adalah telepon *mobile,* tetapi sekarang ini juga digunakan pada *electronic book readers, netbooks,tablet* dan *set-top boxes* (STB)[6]

Android memiliki arsitektur sistem [7] sebagai berikut:



Gambar 2.1: Arsitektur Platform Android



Gambar 2.2 : Daur Hidup Android

Penjelasan dari daur hidup android [8] adalah sebagai berikut :

a. onCreate()

Method ini dipanggil saat activity pertama kali dibuat. Di sinilah inisialisasi dilakukan, seperti membuat UI (User Interface), mendapatkan data yang diperlukan dan lainnya.

b. onStart()

Method ini dipanggil sesaat sebelum activity ditampilkan ke pengguna.

c. onResume()

Method ini dipanggil ketika activity berinteraksi dengan pengguna.

d. onPause()

Method ini berjalan ketika activity berada di balik layar (background), tidak terlihat oleh pengguna tapi masih berjalan. Biasanya hal ini terjadi saat ada activity lainnya yang dijalankan. Di state inilah seharusnya data program kita disimpan ke persistent state.

e. onStop()

Method ini berjalan ketika activity sudah tidak terlihat lagi oleh pengguna dalam waktu yang cukup lama dan activity tidak diperlukan untuk sementara waktu

f. onRestart()

Jika method ini dipanggil, berarti activity sedang ditampilkan ulang ke pengguna dari state berhenti (stop).

g. onDestroy()

Method ini dipanggil sebelum activity dimusnahkan (hilang dari memori)

Dalam pengembangan aplikasi Android perlu dipahami beberapa komponen dasarnya [9], diantaranya :

a. Views

Views adalah element user Interface (UI) yang membentuk dasar dari sebuah *user interface*. View dapat berupa sebuah tombol, label, kolom teks, atau banyak elemen UI lainnya.

b. Activity

Activity adalah sebuah konsep dari UI. Sebuah *activity* biasanya memrepresentasikan sebuah layar dari sebuah

aplikasi. Biasanya *activity* memiliki satu atau lebih *view*, tetapi bisa juga tidak memiliki *view*.

c. Intent

Sebuah Intent didefinisikan sebagai "*intention*" atau "niat" untuk melakukan beberapa pekerjaan. Intent dapat digunakan untuk :

1. Mengirim pesan.
2. Memulai *service.*
3. Menjalankan activity
4. Menampilkan halaman web

d. Content Provider

Dengan menggunakan content provider, kita dapat mengekspos data dan menggunakan data dari aplikasi lain.

e. Service

Service pada android menyerupai *service* pada Windows ataupun plat*form* lain. Aplikasi yang berpotensi berjalan lama ditempatkan pada *background*. Contoh : aplikasi e-mail.

f. AndroidManifest.XML

AndroidManifest.XML mendefinisikan konten dan *behavior* dari aplikasi yang dibuat. Contoh : daftar *activity* yang dibuat, *permission* dan fitur ponsel yang digunakan oleh aplikasi

2.2.1.2 Bahasa Pemrograman Java

Java adalah bahasa pemrograman serbaguna. Java dapat digunakan untuk membuat program sederhana sebagaimana membuatnya dengan bahasa seperti pascal atau C++. Java juga mendukung sumber daya internet dan juga Java mendukung aplikasi client/server, baik dalam jaringan local maupun jaringan berskala luas [10]

Java terbagi dalam 3 kategori, yaitu:

1. Java 2 Standart Edition (J2SE) merupakan edisi standar dari Java 2. J2SE lebih difokuskan pada pemrograman Desktop dan Applet (aplikasi yang dapat dijalankan di browser web).
2. Java 2 *Entetprise Edition* (J2EE) merupakan edisi perluasan dari J2SE, edisi ini ditujukan untuk aplikasi berskala besar (Enterprise), seperti pemrograman menggunakan database dan diatur di server.
3. Java 2 *Mobile Edition* (J2ME) merupakan edisi khusus dari java dan subset dari edisi J2SE. Edisi ini untuk pemrograman dengan peralatan-peralatan kecil seperti PDA, handphone, pager, dan lain-lain.

2.2.2. Client Server

Arsitektur jaringan *client server* merupakan pengembangan dari arsitektur *file server*. Arsitektur ini adalah model konektivitas pada jaringan yang mengenal adanya *server* dan *client*, dimana masing-masing memiliki fungsi yang berbeda satu sama lain. *Server* dapat berbagi pakai data, aplikasi dan *peripheral* seperti *harddisk, printer, modem* dan lain-lain. Oleh karena itu, tidak jarang juga tercipta sebutan *print server, communication server* dan lain sebagainnya. Prinsip kerjanya sangat sederhana, dimana *server* akan menunggu permintaan dari *client*, memproses dan memberikan hasilnya kepada *client*. Sedangkan *client* akan mengirimkan permintaan ke *server*, menunggu proses dan melihat visualisasi hasil prosesnya.

Sistem *Client* dan *Server* terdiri atas dua komponen (mesin) utama, yaitu Client dan Server. Client berisi aplikasi basis data dan server berisi DBMS dan basis data. Setiap aktifitas yang dikehendaki para pemakai akan lebih dahulu ditangani oleh client. Client menangani proses yang menjadi tanggung jawabnya. Jika ada proses yang harus melibatkan data yang tersimpan pada basis data yang terletak di server, barulah client mengadakan hubungan denga server. Pada bentuk sistem client server untuk memenuhi kebutuhan client akan megirimkan pesan atau perintah *Query* pengambilan

data. Selanjutnya server yang menerima pesan tersebut akan menjalankan *Query t*ersebut dan hasilnya akan dikirimkan kembali ke client. Dengan begitu, transfer datanya jauh lebih efisien.

Komponen dasar Client Server adalah:

1. Client

   Client merupakan terminal yang digunakan oleh pengguna untuk meminta layanan tertentu yang dibutuhkan. Terminal client dapat berupa PC, ponsel, komunikator, robot, televisi dan peralatan lain yang membutuhkan informasi.

2. Middleware

   Midleware merupakan komponen perantara yang memungkinkan client dan server untuk saling terhubung dan berkomunikasi satu sama lain. *Midleware* ini dapat berupa *Transaction Monitor* /TP. *Remote Procedure Call* atau *Object Request Broker*/ORB.

3. Server

   Server merupakan komputer khusus yang bertugas melayani aplikasi-palikasi jaringan / pihak yang menyediakan layanan. Server ini akan dapat berupa basis data SQL, Monitor TP, server *groupware*, server objek dan web. Secara umum, server berperan menerima pesan permintaan layanan dari client, memproses permintaan tersebut dan mengirimkan hasil permintaan kepada client.

2.2.3. UML

UML adalah bahasa untuk menspesifikasi, menvisualisasi, membangun dan mendokumentasikan *artifacts* (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, *artifact* tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya. Selain itu UML adalah bahasa pemodelan yang menggunakan konsep orientasi *object*. UML dibuat oleh *Grady Booch*, *James Rumbaugh*, dan *Ivar Jacobson* di bawah bendera *Rational Software Corp*. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai perspektif. UML tidak

hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan. [11]

2.2.3.1 *Use Case Diagram*

Komponen yang terlibat dalam *Use Case Diagram* diantaranya yaitu :

1. *Actor*

   Pada dasarnya *actor* bukanlah bagian dari *use case diagram*, namun untuk dapat terciptanya suatu *use case diagram* diperlukan beberapa *actor* dimana *actor* tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem. Sebuah *actor* mungkin hanya memberikan informasi inputan pada sistem, hanya menerima informasi dari sistem atau keduanya menerima dan memberi informasi pada sistem, *actor* hanya berinteraksi dengan *use case* tetapi tidak memiliki kontrol atas *use case*. *Actor* digambarkan dengan *stick man*. *Actor* dapat digambarkan secara secara umum atau spesifik, dimana untuk membedakannya kita dapat menggunakan *relationship.*

2. *Use Case*

   Use case adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun. Cara menentukan Use Case dalam suatu sistem:

   - Pola perilaku perangkat lunak aplikasi.
   - Gambaran tugas dari sebuah *actor*.
   - Sistem atau “benda” yang memberikan sesuatu yang bernilai kepada *actor*.
   - Apa yang dikerjakan oleh suatu perangkat lunak (bukan bagaimana cara mengerjakannya).

3. Relasi

Ada beberapa relasi yang terdapat pada *use case diagram:*

- *Association,* menghubungkan link antar element.
- *Generalization,* disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
- *Dependency,* sebuah element bergantung dalam beberapa cara ke element lainnya.
- *Aggregation,* bentuk association dimana sebuah elemen berisi elemen lainnya.

Tipe relasi/ *stereotype* yang mungkin terjadi pada *use case diagram*:

- **<<include>>**, yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya.
- **<<extends>>**, kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm.
- <<**communicates**>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association* . Ini merupakan pilihan selama asosiasi hanya tipe *ralationship* yang dibolehkan antara *actor* dan *use case*.

4. *Use Case Diagram*

Adalah gambaran graphical dari beberapa atau semua *actor*, *use case*, dan interaksi diantaranya yang memperkenalkan suatu sistem.

2.2.3.2 *Activity Diagram*

Activity Diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Menguntungkan untuk membuat *activity diagram* pada awal pemodelan proses untuk membantu memahami keseluruhan proses. *Activity diagram* juga bermanfaat untuk menggambarkan *parallel behavior* atau menggambarkan interaksi antara beberapa *use case.*

Elemen-elemen activity diagram :

a. Status *start* (mulai) dan *end* (akhir)
b. Aktifitas yang merepresentasikan sebuah langkah dalam *workflow*
c. *Transition* menunjukkan terjadinya perubahan status aktivitas (*Transitions show what state follows another*).
d. Keputusan yang menunjukkan alternatif dalam *workflow*.
e. *Synchronization bars* yang menunjukkan *subflow parallel*. *Synchronization bars* dapat digunakan untuk menunjukkan *concurent threads* pada *workflow* proses bisnis.
f. *Swimlanes* yang merepresentasikan *role* bisnis yang bertanggung jawab pada aktivitas yang berjalan.

2.2.4. MySQL

MySQL (*My Structure Query Language*) adalah sebuah program pembuatan database yang bersifat *open source*, artinya siapa saja boleh menggunakannya. Dapat dijalankan pada semua plat form baik windows maupun linux. MYSQL merupakan program bersifat jaringan sehingga dapat digunakan untuk aplikasi multi user (banyak pengguna). Saat ini database mysql telah digunakan hampir semua progamer database. [11]

Kelebihan MYSQL adalah menggunakan bahasa query standar yang dimiliki SQL. SQL adalah suatu bahasa permintaan yang terstruktur yang telah distandarkan untuk semua program database. Sebagai program penghasil database, MYSQL tidak dapat berjalan endiri tanpa adanya sebuah palikasi lain (interface). MYSQL dapat diukung hampir semua program aplikasi baik yang open source maupun tidak. Mysql memiliki prompt utama yang disebut mysql.

Adapun perintah-perintah dalam MYSQL diantaranya adalah sebagai berikut:

a. MYSQL, yaitu perintah untuk masuk kedalam sql
b. *Show tables*, yaitu perintah utuk melihat tabel
c. *Show databases*, yaitu perintah untuk melihat database
d. *Desc*, perintah untuk menampilkan tabel atau database yang baru dibuat
e. *Insert into*, perintah untuk menyisipkan isi kedalam tabel
f. *Create*, yaitu perintah untuk membuat tabel atau database
g. *Select,* yaitu perintah untuk menyeleksi isi yang akan ditampilkan sesuai dengan permintaan
h. *Update,* yaitu perintah untuk memperbaharui atau mengganti isi file
i. *Delete,* yaitu perintah untuk menghapus isi file

2.2.5. Agile Scrum

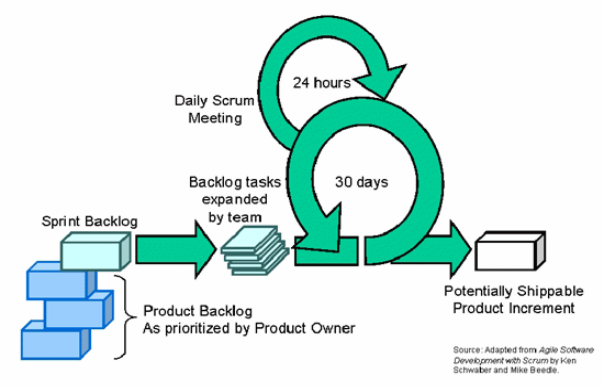
2.2.5.1 *Agile Software Development*

Agile Software Development adalah salah satu metodologi dalam pengembangan sebuah perangkat lunak. Kata *Agile* berarti bersifat cepat, ringan, bebas bergerak, waspada. Kata ini digunakan sebagai kata yang menggambarkan konsep model proses yang berbeda dari konsep model – model proses yang sudah ada. Konsep *agile software developoment* dicetuskan oleh *Kent Beck* dan 16 rekannya dengan mengatakan bahwa *Agile Software Development* adalah cara membangun software dengan melakukannya dan membantu orang lain membangunnya

sekaligus. Model Agile proses yang akan penulis gunakan dalam pengembangan sistem yaitu model *Scrum.*[12]

2.2.5.2 *Scrum*

Scrum adalah pendekatan tangkas untuk pengembangan perangkat lunak. Diperkenalkan oleh Jeff Sutherland tahun awal tahun 1990an, pengembangan berikutnya dilakukan oleh Schwaber dan Beedle. [12] [13]



Gambar 2.3 Proses Agile Scrum

Aktifitas Scrum yaitu :

1. Aktivitas Backlog adalah daftar kebutuhan yang jadi prioritas klien. Daftar dapat bertambah
2. Aktivitas Sprints: unit pekerjaan yang diperlukan untuk memenuhi kebutuhan yang ditetapkan dalam backlog sesuai dengan waktu yang ditetapkan dalam time-box (biasanya 30hari). Selama proses ini berlangsung backlog tidak ada penambahan
3. Aktivitas Scrum Meeting: pertemuan 15 menit perhari untuk evaluasi apa yang dikerjakan, hambatan yang ada, dan target penyelesaian untuk bahan meeting selanjutnya
4. Aktivitas Demo:penyerahan software increment ke klien didemonstrasikan dan dievaluasi oleh klien.

Pada implementasi metodologi Scrum pembagian team dibagi menjadi tiga role yaitu:

1. Product Owner

   Product owner mewakili suara customer dan bertanggungjawab terhadap team yang akan mengimplementasi dari requirement ke implementasi. Product owner biasanya menulis daftar fitur produk berdasarkan diskusi dalam model user story dan memperioritaskan daftar fitur yang dimasukkan kedalam product backlog. Satu team Scrum akan mempunyai satu product owner dan juga anggota team development. Direkomendasikan bahwa role product owner tidak digabungkan dengan role ScrumMaster.

2. Scrum Master

   ScrumMaster bertugas untuk membawa team dari hambatan-hambatan dalam pengembangan produk. Scrum Master bertanggungjawab atas kemajuan pengembangan produk.

3. *Team*

   Team yang bertanggungjawab dalam realisasi produk jadinya. Biasanya satu team terdiri sampai 5-9 orang dengan ketrampilan yang dimiliki bervariasi yaitu *analisa, desain, develop, test, technical communication* hingga dokumentasi. Setiap anggota team dituntut untuk bekerja sendiri dan mengatur manajemen sendiri dalam koridor dalam satu team.

2.2.6. Webservice

2.2.6.1 Pengertian *Webservice*

Webservice adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan [14]. *Webservice* diartikan sebagai sebuah antarmuka yang menggambarkan sekumpulan operasi-operasi yang dapat diakses

melalui jaringan, misalnya *internet* dalam bentuk pesan *XML* (*eXtensible Markup Language*) [15].

Webservice menyediakan standar komunikasi di antara berbagai aplikasi *software* yang berbeda-beda dan dapat berjalan di berbagai *platform* maupun *framework* [16]. Teknologi pada *webservice* dapat mengubah kemampuan transactional web, yaitu kemampuan web untuk saling berkomunikasi dengan pola *Program-to-Program* (P2P). Fokus web selama ini didominasi oleh komunikasi program-to-user dengan interaksi *bussiness-to-consumer* (B2C), sedangkan transactional web akan didominasi oleh program-to-program dengan interaksi business-to-bussiness. [16]

2.2.6.2 Arsitektur Webservice

Komponen Webservice adalah sebagai berikut [17] :

1. *Extensible Markup Language (XML)*

   XML merupakan dasar yang penting atas terbentuknya *Web Services. Web Service* dapat berkomunikasi dengan aplikasi-aplikasi yang memanggilnya dengan menggunakan *XML*, karena *XML* berbentuk teks sehingga mudah untuk ditransportasikan menggunakan protocol HTTP. Selain itu, *XML* juga bersifat *platform* independen sehingga informasi di dalamnya bisa dibaca oleh aplikasi apapun pada *platform* apapun selama aplikasi tersebut menerjemahkan tag-tag *XML*.

2. *Simple Object Access Protocol* (SOAP)

   XML saja tidak cukup agar *Web Service* dapat berkomunikasi dengan aplikasi yang lainnya. *XML* yang digunakan untuk saling bertukar informasi antara *web services* dengan aplikasi yang lainnya harus menggunakan sebuah format standard yang dapat dimengerti oleh keduanya. Format itulah yang dikenal dengan nama SOAP. SOAP (*Simple Object Access Protocol*) merupakan suatu format standar *XML* yang

digunakan untuk melakukan proses request dan responses antara *web services* dengan aplikasi yang memanggilnya. Dokumen SOAP yang digunakan untuk melakukan request disebut dengan SOAP *request,* sedangkan dokumen SOAP yang diperoleh dari *web services* disebut dengan SOAP *responses*.

3. *Web Service Definition Language* (WSDL)

   Untuk mengetahui *method-method* apa saja yang disediakan oleh *Web services,* memerlukan sebuah dokumen bernama WSDL. WSDL (*Web Services Descriptiom Language*) adalah sebuah dokumen dalam format *XML* yang isinya menjelaskan informasi detail sebuah *Web services.* Di dalam WSDL dijelaskan method-method apa saja yang tersedia dalam *Web services,* parameter apa saja yang diperlukan untuk memanggil sebuah method, dan apa hasil atau tipe data yang dikembalikan oleh method yang dipanggil tersebut.

2.2.6.3 Teknologi REST

REST merupakan singkatan dari Representative State Transfer. Pertama kali ditemukan dalam disertasi seorang program doctor bernama Roy Thomas Fiellding pada tahun 2000 [18]. REST adalah sebuah metode dalam menyampaikan resource melalui media web. Sedangkan *resource* sendiri didefinisikan sebagai segala sesuatu yang dapat disimpan di dalam sebuah computer dan ditampilkan sebagai urutan bit, misalnya sebuah dokumen, tabel dalam sistem basis data, atau hasil dari sebuah perhitungan. [19]

2.2.6.4 Arsitektur REST

Arsitektur REST dibangun dengan sifat sebagai berikut [20] :

a. *Addressability*

Dalam prinsip ini seluruh sumberdaya atau resource harus tersedia melalui sebuah alamat unik, pengalamatan ini dilakukan dengan menggunakan URI (*Unique Resource Identifiers*)

b. *Uniform Interface*

Semua interaksi sebaiknya dibangun dengan interface yang seragam Restful service menampilkan semua resource dan interaksinya dengan *interface* yang seragam, dalam metode REST antarmuka yang digunakan adalah menggunakan HTTP. HTTP menawarkan semua operasi yang diperlukan, dikenal dan tersebar luas. Semua interaksi antara klien dan sumber daya (*resource*) didasarkan pada metode dasar HTTP.

c. *Representation-oriented*

Representasi menjelaskan dalam bentuk apa data sedang dipertukarkan antara *client* dan *server*. Pada umumnya data dipertukarkan dalam bentuk XML, JSON dan HTML.

d. *Statelessnes*

Setiap interaksi antara *client* dan *server* harus memiliki state sendiri (atau dengan kata lain tidak dipengaruhi *session client*). *Jadi* server *hanya akan memantau* resource *state* bukan *client session.*

e. *Hypermedia As The Engine of Application State* (HATEOAS)

Hypermedia sebagai state dari sebuah aplikasi (HATEOAS), menyatakan REST dapat menggunakan *link* untuk menghubungkan sumber daya atau resource ke sumber daya lain yang berkaitan.

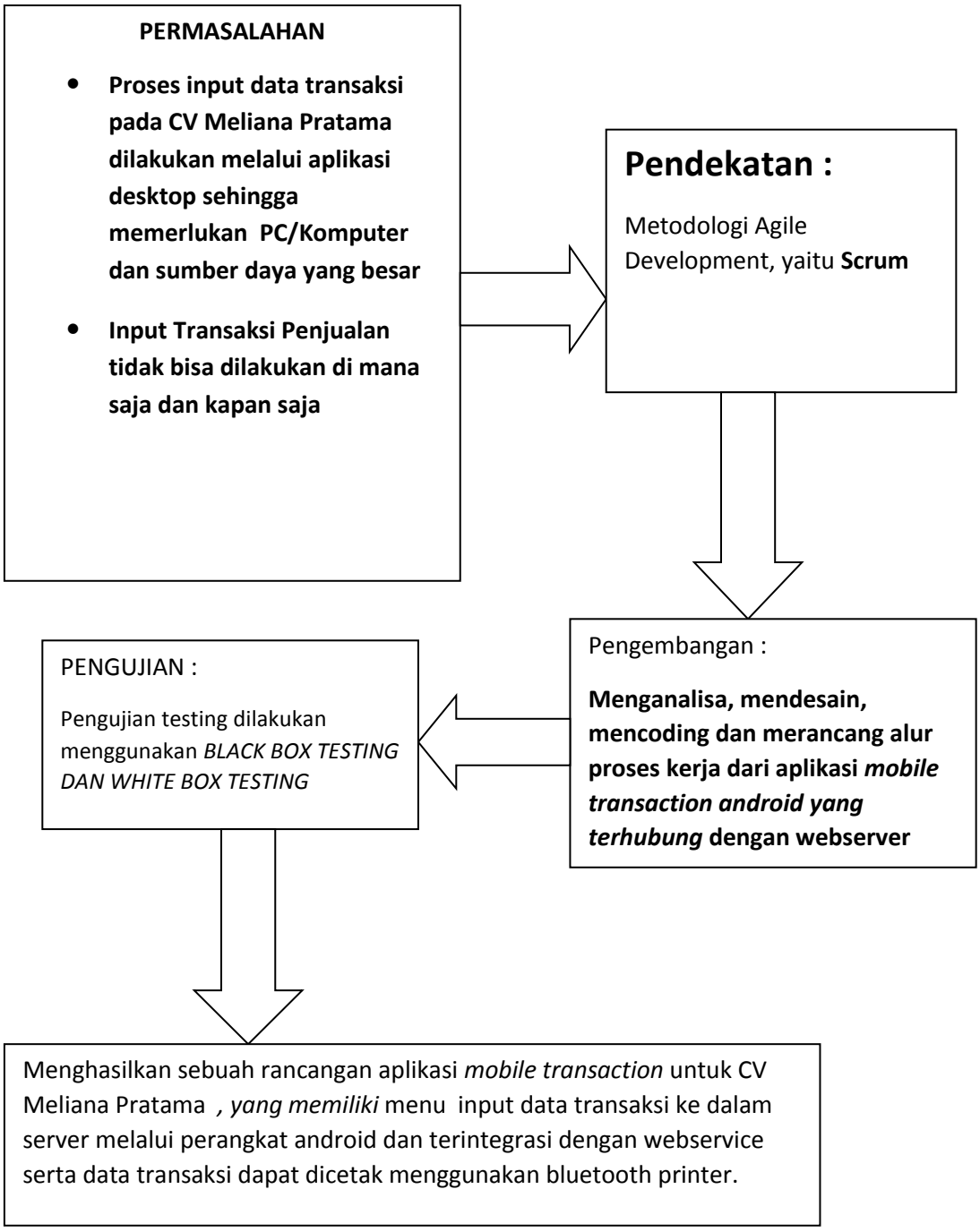
2.2.6.5 JSON (JavaScript Object Notation)

JSON adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh computer. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman JavaScript, Standar ECMA-262 Edisi ke-3 Desember 19999. Pertukaran data dengan format JSON sangat ideal karena format JSON berbasis teks dan terbaca oleh manusia, serta digunakan untuk merepresentasikan struktur data sederhana dan larik asosiatif (disebut objek) [21].

2.3. Obyek Penelitian

Dalam Proyek Tugas Akhir ini penulis mengadakan penelitian pada CV Meliana Pratama Semarang yang beralamat di jalan Pandanaran No 89 Semarang

2.4. Kerangka Pemikiran



Gambar 2.4 : Kerangka Pemikiran