BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu yang dilakukan oleh Rojas, S., dan Viera, berjudul "*Dynamic Forms: An Enhanced Interaction Abstraction Based on Forms*" telah mengusulkan pendekatan baru dalam memahami dan mengimplementasikan formulir interaktif. Penelitian ini mendalami konsep "*Dynamic Forms*" yang berfungsi sebagai abstraksi interaksi yang ditingkatkan yang berbasis formulir. Penelitian ini mengidentifikasi keterbatasan formulir konvensional, yang cenderung statis dan terbatas dalam interaksi pengguna.

Studi ini kemudian meluaskan konsep tersebut ke dalam konteks aplikasi khusus, dengan fokus pada implementasi *Dynamic Form* pada aplikasi *Sales Activity Tracking* menggunakan *Laravel*. Konteks ini menambah dimensi praktis pada konsep tersebut, dengan tujuan untuk mengatasi tantangan unik yang dihadapi PT. iClean dalam melacak aktivitas penjualan.

Sepanjang penelitian mereka, Rojas dan Viera menunjukkan bagaimana konsep *Dynamic Forms* meningkatkan interaksi pengguna dan fleksibilitas dalam pengisian formulir. Mereka menggambarkan bagaimana formulir dapat menyesuaikan diri dengan konteks dan perubahan data, sehingga menciptakan pengalaman pengguna yang lebih dinamis dan intuitif.

Penelitian terdahulu ini juga melibatkan eksperimen dan evaluasi untuk membuktikan efektivitas *Dynamic Forms* dalam berbagai konteks aplikasi. Hasilnya mengungkap bahwa pendekatan ini meningkatkan efisiensi dan kepuasan pengguna dalam mengisi formulir, serta memberikan manfaat dalam pengumpulan data yang lebih akurat.

Namun, ketika melihat penelitian ini, tampak adanya beberapa kekurangan dalam pendekatannya yang dapat saya hubungkan dengan penelitian ini. Penelitian terdahulu tidak menjelajahi implementasi konsep *Dynamic Form* dalam konteks aplikasi bisnis khusus, seperti yang saya kerjakan dalam penelitian ini. Fokus penelitian saya adalah mengatasi tantangan pelacakan aktivitas penjualan yang beragam dalam lingkungan bisnis nyata, khususnya di PT. iClean.

Dalam penelitian ini, saya terinspirasi oleh konsep *Dynamic Form* yang diusulkan oleh Rojas dan Viera. Saya berupaya mengambil pandangan mereka dan mengadaptasikannya lebih terfokus dalam konteks PT. iClean. Dengan mengidentifikasi titik lemah penelitian terdahulu dan mengaitkannya dengan fokus penelitian ini, saya berupaya memberikan kontribusi yang lebih konkret dan relevan dalam menawarkan solusi bagi tantangan dalam konteks aplikasi bisnis sehari-hari.

Persamaan penelitian sebelumnya dengan penelitian ini adalah sebagai berikut:

a) Objek yang diteliti adalah sama-sama *dynamic form*.

Perbedaan penelitian sebelumnya dengan penelitian ini adalah sebagai berikut:

a) Subjek dalam penelitian ini adalah iClean, pada penelitian sebelumnya membahas secara umum terkait *dynamic form.*
b) Fokus dalam penelitian ini adalah bagaimana membuat halaman yang bisa mengakomodasi transaksi dari PT. iClean yang dalam setiap transaksi bisa berbeda untuk setiap *client*-nya.

2.2 Rekayasa Aplikasi Web

Aplikasi Web merujuk pada sistem perangkat lunak yang berbasis pada teknologi World Wide Web. Aplikasi Web memiliki kemampuan untuk memproses masukan dari pengguna dan memberikan hasil dari masukan tersebut melalui antarmuka aplikasi Web (Jakob, 2011:12). Aplikasi Web sendiri dapat diartikan sebagai program aplikasi yang beroperasi di Internet atau intranet dan ekstranet. Pengguna menggunakan web browser pada komputer klien untuk menjalankan program yang berada di server. Meskipun proses berlangsung di server, seolah-olah semua itu terjadi pada mesin lokal pengguna (klien) (Suh, 2005:77-78).

Rekayasa Web adalah pendekatan ilmiah, teknis, dan manajemen yang sistematis dalam pembuatan, penyebaran, dan perawatan aplikasi berbasis Web berkualitas tinggi (Jakob, 2011:42). Tujuan inti dari rekayasa Web adalah mengelola keanekaragaman dan kompleksitas dalam pembuatan aplikasi Web untuk menghindari potensi kegagalan yang serius (Suh, 2005:3). Walaupun menggunakan prinsip-prinsip dasar dari rekayasa perangkat lunak, rekayasa Web berfokus pada teknik dan aktivitas pengelolaan yang khusus untuk aplikasi berbasis Web (Pressman, 2001:769).

Rekayasa Web tidak identik dengan rekayasa perangkat lunak secara keseluruhan. Ia memanfaatkan prinsip-prinsip dasar dari rekayasa perangkat lunak

namun menekankan kesamaan teknik dan pendekatan manajemen (Pressman, 2001:769).

Ahmad, et al (2012:281) menyatakan bahwa berbagai metode pengembangan konvensional seperti waterfall dan spiral telah digunakan untuk mengembangkan aplikasi Web. Namun, metode-metode tersebut tidak sepenuhnya cocok untuk mengatasi perubahan persyaratan aplikasi Web yang berkelanjutan. Oleh karena itu, metode pengembangan Agile diajukan sebagai alternatif untuk menangani masalah-masalah yang tidak dapat ditangani oleh metode konvensional. Agile telah terbukti berhasil dalam berbagai proyek pengembangan Web (Kappel, et al, 2006:216).

2.3 Laravel

Laravel merupakan suatu kerangka kerja yang sangat berguna bagi pengembang web dalam memaksimalkan penggunaan bahasa pemrograman PHP dalam proses pembuatan situs web. Seperti yang telah diketahui, PHP adalah bahasa pemrograman yang sangat dinamis. Namun, kehadiran Laravel telah membuat PHP menjadi lebih efektif, cepat, aman, dan sederhana. Selain itu, framework ini secara konsisten memperkenalkan teknologi terbaru di antara kerangka kerja PHP lainnya.

Kerangka kerja ini mengadopsi struktur MVC, yang merupakan Model View Controller. Struktur MVC memisahkan data dari tampilan berdasarkan komponen-komponen aplikasi seperti manipulasi data, pengendali, dan antarmuka pengguna. Dengan pendekatan ini, Laravel menjadi mudah untuk dipelajari dan dapat mempercepat proses pembuatan prototipe aplikasi web. Selain itu, Laravel menyediakan banyak fitur bawaan seperti otentikasi, pengiriman email, pesan, sesi, dan antrian tugas. Fokus utama dari Laravel adalah pada pengalaman pengguna

akhir, sehingga mengutamakan kejelasan dan kesederhanaan dalam segala hal, mulai dari penulisan kode hingga antarmuka pengguna. Dengan demikian, Laravel mampu menghasilkan aplikasi web yang fungsional dan berjalan dengan lancar. Terdapat beberapa kelebihan dari penggunaan Laravel yaitu antara lain (Komputer, 2010):

1) Menghemat waktu, karena adanya struktur dan *library* yang disediakan sehingga tim pengembang hanya perlu fokus terhadap logika pemrograman.
2) Laravel memiliki standar yang baku sehingga dapat digunakan kembali pada proyek yang lain *(code reuse).*
3) Adanya dukungan komunitas.
4) Laravel memiliki kumpulan kode *best practice* yang teruji sehingga tim pengembang dapat meningkatkan kualitas dari pemrograman dari proyeknya.

2.4 Xampp

Menurut Hidayatullah dan Kawistara (2014) XAMPP merupakan salah satu paket *software web server* yang terdiri dari X (4 sistem operasi), Apache, MySQL, PHP dan Perl. XAMPP mendukung banyak sistem operasi seperti Windows, Linux, Mac, dan Solaris sehingga tidak masalah ketika berpindah ada perpindahan sistem operasi . XAMPP terdiri dari dari:

1. X yang berarti cross platform karena XAMPP bisa dijalankan sistem operasi manapun.
2. Apache sebagai web *server*.

3. MySQL sebagai Database Management System (DBMS).
4. PHP dan Perl sebagai bahasa pendukung.

2.5 MySQL

MySQL merupakan *database* yang paling popular di antara *database* – *database* yang lain dan banyak digunakan dalam membangun aplikasi web yang menggunakan database sebagai sumber dan pengelola datanya. Kepopuleran MySQL dikarenakan kemudahan dalam penggunaan, kinerja *query* yang cepat dan mencukupi kebutuhan *database* dalam lingkup perusahaan skala kecil dan menengah (Suryatiningsih dan Muhammad, 2009). MySQL juga merupakan program *database* yang dapat mengirim dan menerima data dengan kecepatan tinggi serta *multi user*. MySQL *database server* dapat menangani data yang memiliki volume besar dengan tidak membutuhkan *resource* yang besar. Kelebihan dari penggunaan MySQL dibandingkan dengan *database* lainnya antara lain (Komputer, 2010:5-7):

1) MySQL merupakan server paling cepat diantara *database* lainnya.
2) MySQL merupakan sistem manajemen *database* yang bersifat *Open Source* sehingga *free* atau bebas digunakan oleh individu atau instansi tanpa harus membeli atau membayar.
3) MySQL memiliki performa yang tinggi dan sederhana.
4) Database MySQL memahami bahasa SQL *(Structured Query Language)*.
5) MySQL dapat diakses melalui protocol ODBC *(Open Database Connectivity)* dari Microsoft sehingga MySQL dapat diakses oleh banyak *software*.
6) Akses *server* oleh semua klien dalam satu waktu.

7) Database MySQL dapat diakses dari semua tempat di internet dengan menggunakan hak akses tertentu.
8) MySQL mampu menyimpan data volume besar hingga ukuran Gigabyte.
9) MySQL dapat dioperasikan pada berbagai operating system seperti Linux, Windows, Solaris, dan lain – lain.

2.6 HTML

HTML atau *Hypertext Markup Language* dirancang dengan tujuan untuk menghubungkan satu halaman web dengan halaman web lainnya. HTML pertama kali dikembangkan pada 1989 oleh Tim Berners-Lee bersamaan dengan protocol HTTP *(Hypertext Transfer Protocol)*. HTML merupakan bahasa pemrograman web yang memberitahukan web *browser* bagaimana menyusun dan menyajikan konten di halaman web di mana dapat diartikan bahwa HTML merupakan sebuah pondasi dari web. HTML disusun menggunakan bahasa yang sederhana, sehingga mudah dalam pengaplikasiannya. Seiring dengan perkembangannya, saat ini HTML dapat menampilkan objek – objek seperti teks, tabel, tautan, gambar, audio dan video pada halaman web (Solichin, 2016:10).

Standar perintah HTML dikembangkan oleh sebuah konsorsium W3C atau *World Wide Web Consortium*. Terdapat beberapa standar versi HTML yang pernah dirilis oleh W3C antara lain HTML 2.0, HTML 3.2, HTML 4.0 yang direvisi menjadi HTML 4.01, XHTML yang merupakan standar HTML 4.01 dengan terdapat beberapa penyesuaian, dan hingga saat ini standar HTML terakhir yang dirilis adalah HTML5 dengan fitur baru seperti penanganan audio, video dan penyimpanan lokal (Solichin, 2016:10).

2.7 PHP

PHP atau *Personal Home Page* (Situs Personal) adalah bahasa pemrograman *script* yang paling banyak digunakan pada saat ini di mana sering digunakan pada pemrograman situs web dinamis, walaupun tidak menutup kemungkinan digunakan pada pemakaian yang lain. Menurut Solichin (2016:11-13) PHP merupakan salah satu bahasa pemrograman berbasis web yang memiliki beberapa keunggulan dibandingkan dengan bahasa pemrograman lainnya, yaitu antara lain:

1) PHP dapat diunduh dan digunakan dengan gratis, di mana PHP dikembangkan oleh komunitas *open source* dan selalu didistribusikan secara gratis dari, oleh dan untuk web *developer*.
2) PHP berlisensi GNU *General Publish License* (GPL) hal tersebut merupakan sebuah jaminan bahwa semua versi PHP akan selalu dibagikan dengan gratis. Walaupun untuk setiap program dan aplikasi yang dihasilkan dengan menggunakan PHP akan memiliki lisensinya sendiri dari pihak web *developer*.
3) PHP memiliki performa yang handal dan efisien dimana dapat melayani jutaan akses di setiap harinya.
4) PHP mendukung hamper di semua perangkat database, seperti MySQL, Oracle, PostgreSQL, Informix, Interbase, Sybase, MariaDB hingga SQLite bahkan beberapa di antara database tersebut sudah dapat terhubung dengan PHP secara default.
5) PHP dirancang secara khusus untuk aplikasi berbasis web, oleh karena itu PHP menyediakan banyak pustaka (library) bawaan yang terkait dengan web agar dapat langsung digunakan.

6) PHP dapat dijalankan pada hampir semua sistem operasi *(cross platform)*.
7) Perintah – perintah pada bahasa pemrograman PHP mudah dipelajari karena sebagian besar diadopsi dari bahasa pemrograman populer lainnya seperti C/C++, Java dan Perl.

2.8 CSS

CSS, atau Cascading Style Sheets, adalah sebuah dokumen yang berisi perintah untuk memisahkan isi (content) dengan tata letak (layout) dalam desain halaman web. Tujuan utama CSS adalah menyediakan template gaya (style) yang memungkinkan penulisan kode yang lebih mudah untuk halaman-halaman web yang akan dibuat. Dengan menggunakan CSS, pengaturan tampilan keseluruhan situs web dapat diatur dengan mudah hanya dengan menggantikan atribut atau perintah dalam style CSS dengan atribut CSS yang diinginkan.

Saat ini, CSS telah menjadi pilihan style yang populer karena menyediakan berbagai kemudahan dan memiliki atribut yang lengkap. Dengan mengimplementasikan CSS, pengembang web dapat lebih mudah mengontrol tampilan dan format dari halaman-halaman web secara konsisten dan efisien (Kurniawan dan Kom, 2013:1-2).

Secara umum, CSS digunakan untuk mengatur tampilan dan format halaman web yang dibuat dengan HTML dan XHTML. Spesifikasi CSS diatur oleh W3C (World Wide Web Consortium), sebuah lembaga yang mengembangkan standar dan pedoman untuk teknologi web.

Dengan menggunakan CSS, halaman web dapat memiliki tampilan yang konsisten dan dapat menyesuaikan diri dengan berbagai resolusi layar yang

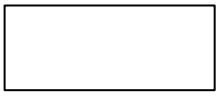
digunakan oleh pengunjung. Hal ini memungkinkan halaman web yang sama akan tampak berbeda secara visual pada perangkat dengan resolusi layar yang berbeda, tanpa perlu bergantung pada penggunaan tabel seperti yang biasa terjadi pada HTML klasik.

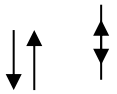
Selain itu, dengan penerapan CSS, halaman HTML atau XHTML pada situs web yang sama dapat ditampilkan secara berbeda dalam hal tampilan gaya atau skema warna yang digunakan. Hal ini memungkinkan pengembang untuk dengan mudah mengubah tampilan keseluruhan situs web dengan hanya mengubah kode CSS tanpa harus menyentuh struktur HTML-nya (Suryatiningsih dan Muhammad, 2009).

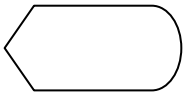
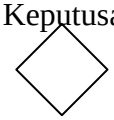
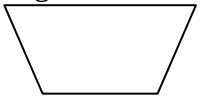
2.9 Flow Chart

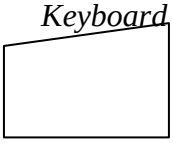
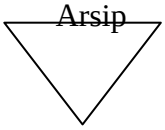
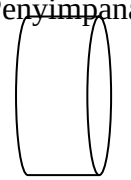
Flowchart merupakan penggambaran langkah – langkah dan urutan prosedur dari sebuah program dalam bentuk grafik. *Flow chart* berguna untuk membantu analis dan programmer dalam menyelesaikan masalah dengan menjabarkannya ke dalam segmen yang lebih kecil dan membantu dalam menganalisis alternatif pengoperasian (Pressman, 2015).

Tabel 2.1 Simbol dalam *Flowchart*

Simbol	**Keterangan**
Proses 	Menunjukkan pengolahan yang dilakukan oleh komputer.

Dokumen 	Menyatakan input berasal dari dokumen dalam bentuk kertas atau output yang dicetak di kertas.
Garis alir 	Menghubungkan antara simbol satu dengan yang lain.
Terminator 	Awal atau akhir dari suatu kegiatan.

Simbol	**Keterangan**
Display 	Menyatakan peralatan output yang digunakan yaitu layar, plotter, printer dan sebagainya.
Keputusan 	Pemilihan proses berdasarkan kondisi yang ada.
Kegiatan Manual 	Menunjukkan pengolahan/proses yang tidak dilakukan oleh komputer.
Store Data 	Menyatakan input yang berasal dari disk atau disimpan dalam disk.

Input – Output 	Menyatakan proses input dan output tanpa tergantung dengan jenis peralatannya.
Keyboard 	Pemasukan data secara manual on – line keyboard.
Arsip 	Disimpan secara offline.
Penyimpanan 	Menyatakan input yang berasal dari proses komputer disimpan dalam database.

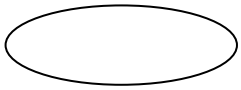
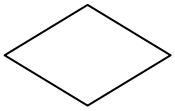
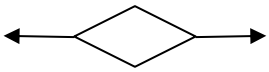
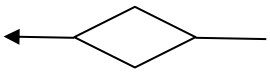
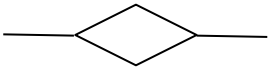
Sumber: (Pressman, 2015)

2.10 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) merupakan suatu penyajian data menggunakan entitas *(entity)* dan relasi *(relationship)*. ERD adalah peralatan pembuatan model data yang paling fleksibel dan dapat digunakan untuk berbagai pendekatan yang mungkin diikuti sebuah organisasi dalam pengembangan sistem. Definisi lainnya mengenai *Entity Relationship Diagram* (ERD) yaitu sekumpulan cara atau peralatan untuk mendeskripsikan data atau objek nyata yang disebut entitas serta hubungan antar entitas – entitas tersebut dengan menggunakan beberapa notasi. Komponen – komponen dari ERD dijelaskan pada tabel di bawah ini.

Tabel 2.2 Komponen – komponen ERD

Notasi	Komponen	Keterangan

	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lainnya
	Atribut	Properti yang dimiliki oleh suatu entitas, di mana dapat mendeskripsikan karakteristik dari entitas tersebut.
	Relasi	Menunjukkan hubungan di antara sejumlah entitas yang berbeda.
	Relasi 1 : 1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua.
	Relasi 1 : N	Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak himpunan entitas lain.
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas pada himpunan entitas yang pertama dapat berhubungan

		dengan banyak entitas pada himpunan entitas kedua, dan sebaliknya.

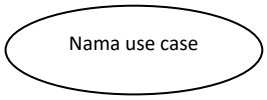
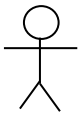
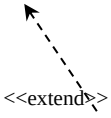
2.11 Unified Modeling Language (UML)

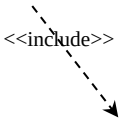
Unified Modeling Language (UML) merupakan bahasa spesifikasi standar yang digunakan sebagai dokumentasi, spesifikasi dan membangun sistem perangkat lunak. *Unified Modeling Language* (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program aplikasi berorientasi objek (OOP). UML juga diartikan sebagai metodologi untuk mengembangkan sistem OOP dan sekelompok alat untuk mendukung pengembangan sistem tersebut. Terdapat empat jenis diagram yang paling sering digunakan untuk membangun aplikasi berorientasi objek yaitu *use case diagram, sequence diagram, collaboration diagram,* dan *class diagram* (Imbar dan Hartanto, 2011).

2.12 Use Case

Use Case merupakan cara formal yang digunakan untuk menggambarkan bagaimana sebuah sistem bisnis berinteraksi dengan lingkungannya. Sebuah *Use Case* Diagram digambarkan untuk mendeskripsikan visualisasi dari adanya interaksi antar pengguna atau aktor dengan sebuah sistem. *Use Case* juga digunakan untuk mengetahui fungsi – fungsi yang ada dalam sebuah sistem informasi dan pengguna atau aktor yang memiliki hak untuk menggunakan fungsi tersebut. *Use Case* Diagram memiliki beberapa simbol dalam penggunaannya:

Tabel 2.3 Simbol dalam *Use Case Diagram*

Simbol	Keterangan
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor, dinyatakan dengan menggunakan kata kerja di awal frasa nama *use case*.
Aktor 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
Asosiasi `	Komunikasi antar aktor dengan *use case* yang berpartisipasi pada *use case*.
Extensi 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek.

Generalisasi 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* di mana fungsi salah satunya adalah fungsi umum.
Include 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2.13 Class Diagram

Class Diagram merupakan gambaran struktur sistem dari pendefinisian kelas – kelas yang akan dibuat untuk membangun sistem. Di mana kelas – kelas tersebut memiliki atribut dan metode atau operasi. Terdapat beberapa simbol pada *Class Diagram* yang digambarkan pada tabel di bawah ini:

Tabel 2.4 Simbol pada *Class Diagram*

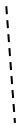
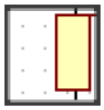
Simbol	Keterangan
Kelas 	Kelas pada struktur sistem.
Antarmuka/interface 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.

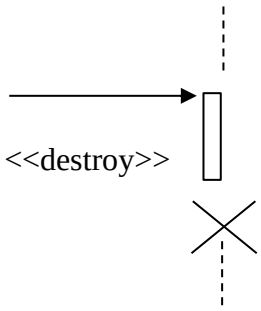
Asosiasi	Relasi antar kelas dengan makna umum, asosiasi juga disertai dengan *multiplicity*.
Asosiasi Berarah 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
Generalisasi	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum – khusus).
Kebergantungan/dependency 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi 	Relasi antar kelas dengan makna semua bagian *(whole part)*.

2.14 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku objek pada *Use Case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Sequence Diagram juga memiliki beberapa simbol yang digambarkan pada tabel berikut:

Tabel 2.5 Simbol dalam *Sequence Diagram*

Simbol	Keterangan
Aktor  Atau nama_aktor	Orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar sistem tersebut.
Garis Hidup/Lifeline 	Menyatakan kehidupan dari sebuah objek.
Objek Nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan.
Waktu Aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
Pesan Tipe Create <<create>>	Menyatakan sebuah objek membuat objek lain, arah panah mengarah pada objek yang dibuat.
Pesan Tipe Call 1:nama_method()	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi/metode.

Pesan Tipe Send 1 : masukan	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
Pesan Tipe Return 1 : keluaran	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
Pesan Tipe Destroy  	Menyatakan suatu objek mengakhiri hidup objek yang lain. Arah panah mengarah pada objek yang diakhiri.