BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Telah ada beberapa penelitian yang dilakukan terkait dengan penerapan Metode SAW, diantaranya adalah :

1. Penelitian yang dilakukan oleh Ria Eka Sari dengan judul Metode Fuzzy Simple Additive Weighting (SAW) Untuk Seleksi Penerimaan Karyawan (Studi Kasus : CV. Asia Exotica), (2015), dimana pada pembahasan ini pihak CV. Asia Exotica mengalami hambatan dalam penyeleksian berkas dikarenakan banyaknya berkas yang masuk harus dicocokkan dengan begitu banyak kriteria yang diinginkan perusahaan. Untuk membantu perusahaan dalam mempermudah dalam pengambilan keputusan maka penyelesaiannya dengan metode *Simple Additive Wieighting(SAW)* yang diharapkan informasi yang dihasilkan bisa menjadi rekomendasi keputusan oleh Pengambil Keputusan. Hasil yang didapat dari penelitian ini adalah Didapatkan hasil analisa perankingan masing-masing alternatif adalah A1 = 0.775, A2 = 0.875, A3 = 0.575, A4 = 0.600 dan A5 = 0.825. Dan nilai tertinggi adalah pada A2 = 0.875 sangat direkomendasikan untuk diterima sebagai karyawan berdasarkan kriteria yang diinginkan oleh pihak perusahaan. Metode *Simple Additve Weighting (SAW)* ternyata mampu menghasilkan analisis dan informasi serta menyelesaikan masalah yang multiple kriteria menjadi suatu informasi berupa keputusan yang dapat digunakan oleh pihak perusahaan dalam mengambil

keputusan untuk seleksi penerimaan karyawan, dan Metode *Simple Additve Weighting (SAW)* dapat diterapkan pada perusahaan CV. Asia Exotica dalam seleksi karyawan yang sesuai dengan kriteria yang ditetapkan oleh perusahaan.

2. Penelitian yang dilakukan oleh Lili Tanti, (2015) yang berjudul Pemilihan Pegawai Berprestasi Berdasar Evaluasi Kinerja Pegawai Dengan Metode SAW. Tujuan dari penelitian ini adalah untuk Untuk memudahkan pihak manajemen khususnya bagian kepegawaian untuk mendapatkan hasil yang lebih akurat dalam pemilihan pegawai berprestasi. Menerapkan metode Fuzzy Multiple Attribute Decision Making (FMADM) dengan metode Simple Additive Weighting (SAW) dalam pemilihan pegawai berprestasi. Dengan metode perangkingan diharapkan lebih tepat dan akurat karena sudah didasarkan pada kriteria dan bobot yang sudah ditetapkan sehingga dapat menentukan siapa yang lebih berhak mendapat penghargaan tersebut. Aplikasi ini diharapkan dapat membantu perusahaan dalam menilai kinerja pegawai sekaligus pengukur kinerja dari Univeristas Potensi Utama, sehingga institusi dapat berkembang dengan pesat sesuai visi dan misi serta tujuan dari institusi tersebut.
3. Penelitian yang dilakukan oleh Aris Rakhmadi, Bambang Efirianto, (2016) yang berjudul Sistem Penilaian Karyawan Terbaik Menggunakan Metode Simple Additive Weighting Pada Dealer Motor. Tujuan dari penelitian ini adalah untuk merancang sistem informasi penilaian karyawan pada Dealer Kondang Motor menggunakan metode SAW. Langkah metode SAW adalah

penentuan kriteria yang dijadikan acuan dalam pengambilan keputusan, dalam hal ini Ci, penentuan rating kecocokan alternatif untuk setiap kriteria, pembuatan matriks keputusan yang didasarkan pada kriteria (Ci), kemudian melakukan normalisasi matriks yang didasarkan pada persamaan yang disesuaikan dengan jenis atribut (atribut benefit/keuntungan ataupun atribut cost/biaya) sehingga diperoleh matriks ternormalisasi R. Hasil akhir adalah proses perankingan, penjumlahan dari perkalian matriks ternormalisasi R dengan vektor bobot sehingga didapat nilai terbesar yang dipilih sebagai alternatif terbaik (Ai) sebagai solusinya. Berdasarkan hasil analisis penilaian karyawan dengan SAW, Informasi menunjukkan bahwa Wulan sari dengan nilai 0.79 merupakan karyawan terbaik, sedangkan Hafid dengan nilai 0.58 merupakan karyawan dengan performa yang paling buruk.

II.2. Landasan Teori

II.2.1. Sistem Pendukung Keputusan / *Decision Support System* (DSS)

SPK sebagai sebuah sistem berbasis komputer yang membantu dalam proses pengambilan keputusan. SPK sebagai sistem informasi berbasis komputer yang adaptif, interaktif, fleksibel, yang secara khusus dikembangkan untuk mendukung solusi dari pemasalahan manajemen yang tidak terstruktur untuk meningkatkan kualitas pengambilan keputusan. Dengan demikian dapat ditarik satu definisi tentang SPK yaitu sebuah sistem berbasis komputer yang adaptif, fleksibel, dan interaktif yang digunakan untuk memecahkan masalah-masalah tidak terstruktur

sehingga meningkatkan nilai keputusan yang diambil. (Jurnal Seminar Nasional Informatika 2015, Ria Eka Sari)

II.2.2. Komponen Sistem Pendukung Keputusan

Aplikasi komponen-komponen sistem pendukung keputusan dapat terdiri dari subsistem, diantaranya :

1. Subsistem manajemen data**.** Subsistem manajemen data mencakup satu *database* yang berisi data yang relevan untuk situasi dan dikelola oleh sistem manajemen basisdata *(Data Base Management Systems (DBMS))*. Subsistem manajemen data dapat diinterkoneksikan dengan data *warehouse* perusahaan, suatu repositori untuk data perusahaan yang relevan untuk pengambil keputusan. Biasanya data disimpan atau diakses *via server web databasel*. Subsistem manajemen data dapat diinterkoneksikan dengan data *warehouse* perusahaan.
2. Subsistem manajemen model**.** Merupakan paket perangkat lunak yang memasukkan model keuangan, statistik, ilmu manajemen, atau model kuantitatif lainnya yang memberikan kapabilitas analitik dan manajemen perangkat lunak yang tepat.
3. Subsistem antarmuka pengguna. Pengguna berkomunikasi dengan dan memerintahkan DSS melalui subsistem ini. Pengguna adalah bagian yang dipertimbangkan dari sistem. Para peneliti menegaskan bahwa beberapa kontribusi unik dari DSS berasal dari interaksi yang intensif antara komputer dan pembuat keputusan.

4. Subsistem manajemen berbasis-pengetahuan. Subsistem ini dapat mendukung semua subsistem lain atau bertindak sebagai komponen independen. Ia memberikan inteligensi untuk memperbesar pengetahuan si pengambil keputusan (Jurnal Seminar Nasional Informatika 2015, Ria Eka Sari).

II.2.3. Ciri-Ciri Sistem Pendukung Keputusan

Menurut Jurnal Ria Eka Sari (2014), adapun ciri-ciri sebuah SPK seperti yang dirumuskan oleh Alters Keen adalah sebagai berikut [3]:

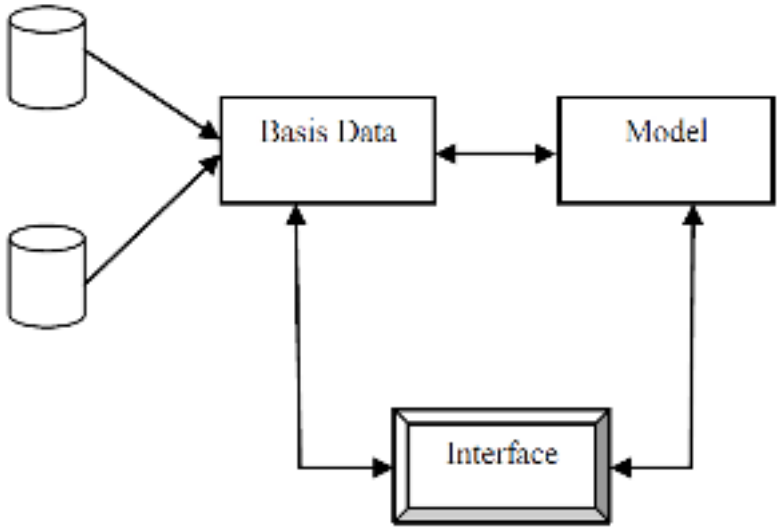
1. SPK ditujukan untuk membantu pengambilan keputusan-keputusan yang kurang terstruktur dan umumnya dihadapi oleh para manajer yang berada di tingkat puncak.
2. SPK merupakan gabungan antara kumpulan model kualitatif dan kumpulan data.
3. SPK memiliki fasilitas interaktif yang dapat mempermudah hubungan antara manusia dengan komputer.
4. SPK bersifat luwes dan dapat menyesuaikan dengan perubahan-perubahan yang terjadi.

II.2.4. Komponen-komponen *Decision Support System* (DSS)

Komponen *Decision Support System* dapat berupa :

1. Subsistem manajemen data, subsistem manajemen data memasukkan satu database yang berisi data yang relevan untuk situasi dan dikelola oleh perangkat lunak yang disebut sistem manajemen database (DBMS)

2. Subsistem manajemen model, merupakan paket perangkat lunak yang memasukkan model keuangan, statistik, ilmu manajemen atau model kuantitatif lainnya yang memberikan kapabilitas analitik dan manajemen perangkat lunak yang tepat.
3. Antarmuka Pengguna, pengguna berkomunikasi dengan memerintahkan DSS melalui subsistem ini. *(Sumber : Ria Eka Sari, Alfa Saleh, 2014)* Adapun gambar yang memperlihatkan komponen sistem pendukung keputusan dapat dilihat pada gambar II.1 berikut.



Gambar II.1. Model Konseptual *Decision Support System* (DSS)

(Sumber : Ria Eka Sari, Alfa Saleh, 2014)

II.3. Metode SAW

Metode *SAW* merupakan metode yang paling dikenal dan paling banyak digunakan dalam menghadapi situasi *MADM*. Metode ini mengharuskan pembuat keputusan menentukan bobot bagi setiap atribut. Skor total untuk sebuah alternatif diperoleh dengan menjumlahkan seluruh hasil perkalian antara rating (yang dapat

dibandingkan lintas atribut) dan bobot tiap atribut. Rating tiap atribut haruslah bebas dimensi dalam arti telah melewati proses normalisasi sebelumnya. Pada dasarnya metoda ini berdasarkan konsep pembobotan rata-rata. Pembuat keputusan secara langsung menentukan bobot "kepentingan relatif" pada masing-masing peta tematik. Total nilai masing-masing alternatif didapatkan dengan mengalikan bobot yang ditentukan untuk masingmasing atribut dan menjumlahkan hasil atribut-atribut tersebut. Menurut *Thill* saat skor keseluruhan semua alternatif dihitung, alternatif dengan nilai tertinggi akan dipilih. Evaluasi aturan keputusan masing-masing alternatif, Ai, seperti rumus sebagai berikut :

Ai = Wj.Xij …….. (1)

Dimana Xij : adalah alternatif ke i pada atribut j.

Wj : adalah normalisasi bobot (Wj=1).

Bobot-bobot tersebut menunjukkan pentingnya atribut secara relatif. Alternatif yang paling dipilih diseleksi dengan mengidentifikasi nilai Ai maksimum.

Ai (i=1,2,...,m) …….. (2)

Metode *SAW* sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode *SAW* adalah mencari penjumlahan terbobot dari rating kerja pada setiap alternatif pada semua atribut (fishburn,1967),(MacCrimmon,1968). Metode *SAW* membutuhkan proses normalisasi matriks keputusan (X) ke skala yang dapat diperbandingkan dengan semua rating alternatif yang ada

$$R_{ij} = \begin{cases} \dfrac{X_{ij}}{Max_i \; X_{ij}} & \text{Jika J adalah atribut keuntungan (Benefit)} \\ \dfrac{Min_i \; X_{ij}}{X_{ij}} & \text{Jika J adalah atribut biaya (Cost)} \end{cases} \quad \text{.....(3)}$$

dimana :

Ri j = nilai rating kinerja ternormalisasi

Xi = nilai atribut yang dimiliki dari setiap kriteria

Max xij = nilai terbesar dari setiap kriteria i

Min xij = nilai terkecil dari setiap kriteria i

Benefit = jika nilai terbesar adalah terbaik

Cost = jika nilai terkecil adalah terbaik

Dimana *rij* adalah rating kinerja ternormalisasi dari alternatif Ai pada atribut *Cj; i*=1,2,..., *m* dan *j*=1,2,...,*n*.

Nilai preferensi untuk setiap alternatif (Vi) diberikan sebagai:

$$V_i = \sum_{j=1}^{n} W_j R_{ij} \quad \text{.....(4)}$$

Di mana :

Vi = rangking untuk setiap alternatif

wj = nilai bobot dari setiap kriteria

rij = nilai rating kinerja ternormalisasi

Nilai *Vi* yang lebih besar mengindikasikan bahwa alternatif *Ai* lebih terpilih. (Ria Eka Sari, 2015)

II.4. *Unified Modelling Language (UML)*

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai negara dapat mengerti pemodelan perangkat lunak. Seperti yang kita ketahui bahwa menyatukan banyak kepala

untuk menceritakan sebuah ide dengan tujuan untuk memahami hal yang sama tidaklah mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang.

Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan oleh banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modelling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasi, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak.

"UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung"

UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek. Seperti yang kita ketahui bahwa banyak hal di dunia sistem informasi yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkungan dan konteksnya. Begitu juga dengan

perkembangan penggunaan UML bergantung pada *level* abstraksi penggunaannya. Jadi belum tentu pandangan yang berbeda dalam penggunaan UML adalah suatu yang salah, tapi perlu ditelaah dimanakah UML digunakan dan hal apa yang ingin divisualkan. (Shalahuddin, M. dan Rosa A.S, 2014:137).

II.5. *Use Case Diagram*

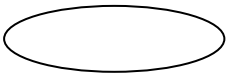
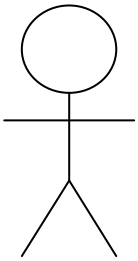
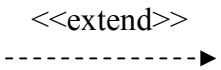
Use case atau diagram *use case* merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case.*

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

   Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel II.1 Simbol-Simbol *Use Case Diagram*

Simbol	Nama	Keterangan
	*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
	*Actor*	Orang, proses, atau sistem lain yang berorientasi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
	Association	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
	*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
	Generalization	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

<<include>> ----------▶	*Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:156, Rekayasa Perangkat Lunak)

II.6. *Activity Diagram*

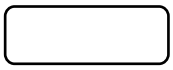
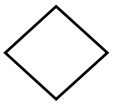
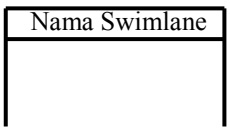
Diagram aktivitas atau *activity diagram* menggambarkan *work flow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor,* jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

Tabel II.2 Simbol *Activity Diagram*

Simbol	Nama	Keterangan
	Status Awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
	Percabangan/ *Decision*	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
	Penggabungan/ *Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:162, Rekayasa Perangkat Lunak)

II.7. *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur system dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

a. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas

b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar Antara dokumentasi perancangan dan perangkat lunak sinkron. Banyak berbagai kasu, perancangan kelas yang dibuat tidak sesuaidengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai.

Kelas-kelas yang ada pada struktur system harus dapat melakukan fungsi-fungsi-fungsi sesuai dengan kebutuhan system sehingga pembuat perangkat lunak dapat membuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

a. Kelas main : kelas yang memiliki fungsi awal dieksekusi ketika system dijalankan.
b. Kelas yang menangani tampilan system (*view*) : kelas yang mendefenisikan dan mengatur tampilan ke pemakai.
c. Kelas yang diambil dari pendefenisian *use case* (*controller*) : kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefenisian *use case*, ,kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
d. Kelas yang diambil dari pendefenisian data (*model)* : kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Semua table yang dibuat di basis data dapat dijadikan kelas, namun untuk table dari

hasil ralasi atau atribut mutivalue pada ERD dapat dijadikan kelas tersendiri dapat juga tidak asalkan pengaksesannya dapat dipertanggungjawabkan atau tetap ada didalam perancangan kelas.

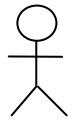
II.8. *Sequence Diagram*

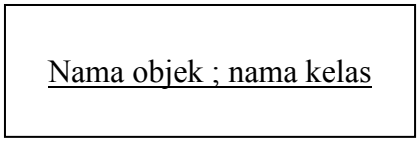
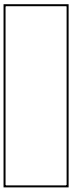
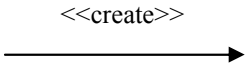
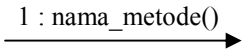
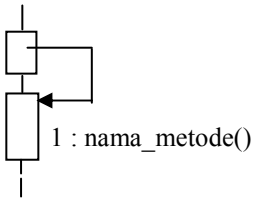
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirmkan dan diterima antar objek. Oleh karean itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimilik kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat scenario yang ada pada *use case*.

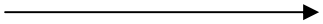
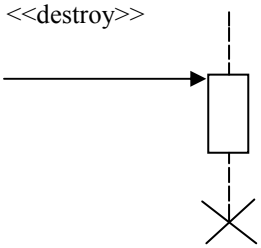
Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefenisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefenisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel II.3. Simbol *Sequence Diagram*

Simbol	Deskripsi
 Nama aktor Atau Nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yanng berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama actor

Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
Pesan tipe create 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe call 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yanng dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.

Pesan tipe send 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
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 diakhri, sebaiknya jika ada create maka ada destroy.

(Sumber : Shalahuddin, M. dan Rosa A.S, 2014:165-167, Rekayasa Perangkat Lunak)