BAB II
LANDASAN TEORI

II.1. Sentimen Publik Terhadap Acara Televisi

Pada tahun 2008 stasiun televisi di Amerika Serikat melakukan suatu terobosan dengan mulai mengkombinasikan media sosial khususnya *micro-blogging* seperti twitter untuk menciptakan pengalaman sosial video di sepanjang peristiwa. Saat itu acara pertama yang menggunakan terobosan ini untuk acara debat perdana pemilihan presiden Amerika Serikat. Dibuatlah sebuah metodologi analisis yang menghitung jumlah nilai dari pesan sentimen yang ada di twitter dan direpresentasikan ke dalam bentuk visual. Hal tersebut dilakukan untuk membantu para wartawan atau humas untuk memahami dinamika sementara dari sentimen sebagai reaksi perdebatan acara.

Nama dari acara televisi yang melakukan hal tersebut adalah *Hack the Debate.* Pada acara tersebut mereka meminta partisipasi dari publik untuk mengomentari selama acara debat berlangsung. Menggunakan layanan Twitter, *tweet* yang dikirimkan oleh publik ditampilkan di TV di bawah berlangsungnya debat calon presiden antara Barack Obama dan John McCain pada saat itu. Acara tersebut menjadi pionir bagi acara lainnya untuk melakukan hal yang sama. Dengan banyaknya *tweet* membangun sebuah kesempatan untuk memahami sentimen publik yang diwakili oleh para pengguna twitter [6].

Salah satu media massa di Indonesia yaitu Tempo melakukan hal yang mirip dengan yang di Amerika Serikat. Namun yang dilakukan tempo tidak secara langsung ketika acara tersebut berlangsung, akan tetapi dilakukan dengan cara menghitung secara kualitatif dan tanpa adanya permintaan secara langsung. Media Tempo menilai acara televisi terburuk tahun 2013 memanfaatkan “kicauan” publik di twitter. Hasilnya, acara yang mendapatkan *mention* negatif paling banyak adalah Yuk Keep Smile. Acara yang ditayangkan di TransTV itu dianggap membosankan [7].

II.2. Text Mining

Text mining yang juga dikenal dengan *text data mining* atau pencarian pengetahuan di basis data textual adalah sebuah proses yang semi otomatis melakukan ekstraksi dari pola yang ada di *database*. Dari hasil ekstraksi tersebut munculah pengetahuan baru yang bisa dimanfaatkan untuk kepentingan pengambilan keputusan. *Text mining* mempunyai kesamaan dengan *data mining*. Keduanya memliki tujuan yang sama yaitu untuk memperoleh informasi dan pengetahuan dari sekumpulan data yang sangat besar. Data tersebut bisa berbentuk sebuah *database*. Namun keduanya memiliki perbedaan jenis data. *Data mining* memiliki input data dari data yang sudah terstruktur sedangkan *text mining* dimulai dengan data yang tidak terstruktur.

Pemanfaatan dari *text mining* secara nyata sangatlah luas. Areanya seluas data tekstual yang terbentuk seperti di area hukum dengan data putusan pengadilan, penelitian dengan data artikel penelitian, keuangan dengan data laporan triwulan, teknologi dengan data arsip paten, pemasaran dengan data komentar konsumen, dan di area lainnya. Sebagi contoh sebuah perusahaan membuat formulir yang biasa diisi apabila konsumennya ingin memuji, komplain, ataupun klaim garansi. Dari kartu formulir tersebut terbentuklah data yang sangant besar dan bisa digunakan untuk mengidentifikasi secara objektif produk dan layanan dari suatu perusahaan menggunakan text mining. Selain itu proses *text mining* yang dilakukan secara otomatis adalah dibidang komunikasi elektronik dan email. *Text mining* tidak hanya mengklasifikasikan dan menyaring email sampah, tetapi bisa juga memprioritaskan *email* secara otomatis berdasarkan tingkat kepentingannya [3].

II.3. Analisis Sentimen

Sejarah analisis sentimen pertama kali muncul pada sebuah jurnal karya Das, Chen, dan Tong pada tahun 2001, bahasan yang mereka angkat susuai dengan minat mereka yaitu menganalisis sentimen pasar. Analisis sentimen adalah mengekstraksi pendapat, sentimen, evaluasi, dan emosi orang tentang suatu topik tertentu yang tertulis menggunakan teknik pemrosesan bahasa alami. Sejumlah karya-karya besar lainnya menyebutkan analisis sentimen fokus pada aplikasi spesifik yang

mengklasifikasikan mengenai sifat yang berlawanan (antara positif dan negatif). Dari pengertian tersebut menjadi sebuah fakta yang menyebabkan beberapa penulis bahwa istilah analisis sentimen mengacu pada tugas yang sempit atau spesifik. Namun saat ini banyak yang menafsirkan istilah analisis sentiment lebih luas lagi yang berarti cara pengkomputasian pendapat, sentimen, dan subjektifitas pada teks [8].

Sistem analisis sentimen saat ini sudah banyak diterapkan di hampir setiap bisnis dan domain sosial karena opini menjadi pusat semua aktivitas manusia dan menjadi kunci yang mempengaruhi sikap seseorang. Dari keyakinan dan persepsi realita, dan pilihan yang dibuat sesorang, sebagian besar dipengaruhi oleh opini orang lain. Karena alasan itulah, keputusan sesorang yang diambil sering berdasarkan opini atau pendapat orang lain. Hal tersebut tidak hanya berlaku bagi individu tetapi juga bagi organisasi.

II.4. Regular Expression

Regular expression atau yang biasa disingkat dengan regex adalah sebuah untaian teks untuk menggambarkan pencarian sebuah pola. Regex biasa digunakan untuk pencarian atau memanipulasi teks. Pola yang dibentuk oleh regex mungkin akan cocok sekali, beberapa kali, atau tidak sama sekali untuk teks yang diberikan. Regex didukung oleh banyak bahasa pemrograman, seperti PHP, C#, Java dan banyak bahasa pemrograman lainnya. Berikut adalah aturan-aturan penulisan *regular expression* dalam bahasa pemrograman Java [9].

1. Pencocokan simbol umum

Regular expression menyediakan pola yang bisa digunakan untuk mencocokan simbol-simbol yang umum pada suatu teks. Pada Tabel II-1 akan dijelaskan *regex* yang digunakan beserta deskripsinya.

Tabel II-1 Daftar Simbol Umum Regex

Regular Expression	Deskripsi
.	Mencocokan dengan karakter apapun
^regex	Menemukan kata regex yang ada di awal baris.
regex$	Menemukan kata regex yang ada di akhir baris.
[abc]	Tanda kurung siku digunakan untuk mencocokan salah satu huruf yang ada di dalamnya. Contoh digunakan untuk mencocokan dengan huruf a atau b atau c.
[abc][de]	Mencocokan dengan huruf a atau b atau c kemudian diikuti dengan huruf d atau e.
[^abc]	Tanda sisipan yang muncul dalam tanda kurung siku sebagai tanda negasi. Contoh digunakan untuk mencocokan dengan huruf apapun kecuali a atau b atau c.
[a-d1-7]	Mencocokan dengan deretan huruf yang yang ada dari a hingga d dan 1 sampai 7.
a|b	Menemukan a atau b.
Ab	Menemukan a yang kemudian diikuti dengan b.
a!b	Mrnrmukan a yang kemudian diikuti bukan dengan b

2. Metacharacters

Metacharacter berikut memiliki arti yang ditentukan dan membuat pola umum yang lebih mudah digunakan. Berikut contohnya pada Tabel II-2 .

Tabel II-2 Daftar *Metacharacter Regex*

Regular Expression	Deskripsi
\d	Mencocokan dengan angka, lebih sederhana dari [0-9]
\D	Mencocokan dengan bukan angka, lebih sederhana dari [^0-9]
\s	Mencocokan dengan spasi, lebih sederhana dari [\t\n\x0b\r\f]
\S	Mencocokan dengan bukan spasi, lebih sederhana dari [^\s]
\w	Mencocokan dengan alphanumerik, lebih sederhana dari [a-zA-Z_0-9]
\W	Mencocokan dengan bukan alphanumerik, lebih sederhana dari [^\w]

3. Quantifier

Quantifier mendefinisikan seberapa sering sebuah elemen dapat terjadi. Berikut contoh dan deskripsi pada Tabel II-3.

Tabel II-3 Daftar *Quantifier Regex*

Regular Expression	Deskripsi	Contoh
*	Terjadi kemunculan tidak sama sekali atau berkali-kali. Lebih sederhana dari {0,}.	a* menemukan tidak sama sekali atau berkali-kali kemunculan huruf a
+	Terjadi kemunculan sekali atau berkali-kali. Lebih sederhana dari {1,}	a+ menemukan sekali atau berkali-kali kemunculan huruf a
?	Terjadi kemunculan tidak sama sekali atau sekali. Lebih sederhana dari {0,1}	A? menemukan tidak sama sekali atau tepat satu kali kemunculan huruf a
{x}	Terjadi kemunculan sebanyak x	\d{5} mencari untuk angka yang memiliki tiga digit.
{x,y}	Terjadi kemunculan sebanyak x hingga ke y.	\d{1-5} berarti \d harus muncul meninmal satu dan paling banyak lima kali.

4. Backslash

Backslash digunakan di dalam *regular expression* memiliki arti yang ditentukan dalam Java. Sebelumnya telah dibahas penggunaan secara implisit penggunaan backslah. Dalam implementasinya ketika ingin mendefiniskan \w , maka harus menggunakan \\w di regex yang dibuat. Jika ingin mendefiniskan backslahes dan tanda baca lainnya maka menggunakan *double backslashes* diikuti dengan tanda baca.

II.5. Preprocessing

Tahan *preprocessing* diperlukan untuk membersihkan data dari yang tidak diperlukan, dengan tujuan pada tahap masuk ke dalam metode *Naïve Bayes Classiffier* lebih optimal dalam perhitungannya. Pada tahap ini melibatkan rekognisi dari isi dan struktur teksnya. Adapun tahapan-tahapan dari *preprocessing*.

1. *Convert Emoticon*

Emoticon merupakan salah satu cara pengungkapan ekspresi perasaan secara tekstual. Hal tersebut akan membantu dalam menentukan sentiment dalam suatu kalimat atau tweet sekalipun. Setiap *emoticon* akan dikonversikan ke dalam *string*

yang bersesuaian. Jenis *emoticon* yang akan diproses oleh adalah *emoticon western style*, dikarenakan jenis tersebut banyak digunakan atau menjadi standar di semua platform seperti *web* ataupun *mobile*. Pada Tabel II-4 dijelaskan hasil pengklasifikasian arti dari *emoticon western style* secara umum.

Tabel II-4 Daftar *Emoticon* Yang Akan Dikonversi

Emoticon	Deskripsi
:-) :) :o) :] :3 :c) :> =] 8) =) :} :^) :つ)	Senang
:-D :D 8-D 8D x-D xD X-D XD =-D =D =-3 =3 B^D	Ketawa
:-|| :@ >:(	Benci
:$ >:[:-(:(:-c :c :-< :つC :< :-[:[:{ ;(:'-(:'(D:< D: D8 D; D= DX v.v D-':	Kecewa
<3 ;-) ;) *-) *) ;-] ;] ;D ;^) :-,	Suka

Pada Tabel II-5 dijelaskan contoh penerapan dari *convert emoticon* menjadi kata.

Tabel II-5 Contoh Penerapan Dari *Convert Emoticon*

Data Latih	
Input	**Output**
Gokil,kocak,gila,seru,ngakak, ea Cuma di Indonesia Lawak Klub ;)	Gokil,kocak,gila,seru,ngakak, ea Cuma di Indonesia Lawak Klub senang
Jelek yks sekarang mah :-(	Jelek yks sekarang mah kecewa

2. *Cleansing*

Clenasing merupakan proses membersihkan kata-kata yang tidak diperlukan untuk mengurangi noise. Kata yang dihilangkan adalah URL, *hashtag (#)*, *username* (@username), dan email. Selain itu juga tanda baca seperti titik(.), koma(,), dan tanda baca yang lainnya akan dihilangkan. Pada Tabel II-6 dijelaskan contoh penerapan dari *cleansing*:

Tabel II-6 Contoh Penerapan *Cleansing*

Data Latih	
Input	**Output**
Saya suka video @YouTube http://youtu.be/wjLPx8J1dDs?a MATA NAJWA - Memilih Wakil Rakyat - Iwan Fals, Ahok, JK & DS (Full)	Saya suka video MATA NAJWA Memilih Wakil Rakyat Iwan Fals Ahok JK DS Full
Stand Up Comedy Anang! stand up comedy Anang Lucu Banget Terbaru 2014 indonesia metro tv http://www.youtube.com/watch?v=JoSg0a4Q8vY …	Stand Up Comedy Anang stand up comedy Anang Lucu Banget Terbaru indonesia metro tv

3. *Case Folding*

Case folding merupakan tahapan merubah bentuk kata-kata menjadi sama bentuknya, baik semuanya menjadi *lower case* ataupun menjadi *upper case*. Pada Tabel II-7 dijelaskan contoh dari penerapan proses *case folding*.

Tabel II-7 Penerapan Tahapan *Case Folding*

Data Latih	
Input	**Output**
Saya suka video MATA NAJWA Memilih Wakil Rakyat Iwan Fals Ahok JK DS Full	saya suka video mata najwa memilih wakil rakyat iwan fals ahok jk ds full
Stand Up Comedy Anang stand up comedy Anang Lucu Banget Terbaru indonesia metro tv	stand up comedy anang stand up comedy anang lucu banget terbaru di indonesia metro tv

4. *Convert Negation*

Seperti halnya ilmu matematika, dalam bahasa terdapat kata yang dapat membalikan arti dari kata tersebut atau bersifat negasi. Kata-kata yang bersifat negasi adalah “kurang”, “tidak”, “enggak”, “ga”, “nggak”, “tak”, dan “gak”. Pada Tabel II-8 dijelaskan contoh penerapannya.

Tabel II-8 Contoh Penerapan *Convert Negation*

Sebelum Convert Negation	Setelah Convert Negation
gak seru final indonesia idol tanpa virzha sabar virzha sukses pulang	ga_seru final indonesia idol tanpa virzha sabar virzha sukses pulang

5. Tokenizing

Tokenizing bekerja untuk mengidentifikasi kata-kata dalam teks menjadi beberapa urutan yang terpotong oleh spasi atau karakter spesial. Berikut contoh penerapan dari *tokenizing*.

Tabel II-9 Penerapan Tahapan *Tokenizing*

Data Latih	
Input	**Output**
stand up comedy anang stand up comedy anang lucu banget terbaru di indonesia metro tv	String 1 : stand String 2 : up String 3 : comedy String 4 : anang String 5 : stand String 6 : up

Data Latih	
Input	**Output**
	String 7 : comedy String 8 : anang String 9 : lucu String 10 : banget String 11 : terbaru String 12 : di String 13 : indonesia String 14 : metro String 15 : tv

6. *Stopping*

Stopping berperan untuk membuang kata-kata yang sering muncul dan bersifat umum, kurang menunjukan relevansinya dengan teks. Kata-kata yang akan dibuang tersebut didefinisikan dalam *stopword list.* Contoh beberapa kata yang sering masuk ke dalam *stopword list* adalah "sebuah", "yang", dan "itu". Berikut adalah beberapa daftar *stoplist* yang disimpan dalam *database*.

Tabel II-10 Stoplist Proses *Stopping*

Masih	Dong	ke	Ada	Yoi
Malam	Ya	loe	Pada	Yang
Ini	Dan	juga	Kita	Saya
untuk	Dari	bagi	Iya	di mana
kapan	Bisa	mana	Itu	Sih
sudah	Bikin	dengan	Anda	Begitu
entah	Lalu	yuk	Aku	Adalah
gue	Nanti	tunggu	Tau	Kemarin

Berikut adalah contoh penerapan dari tahapan *stopping*:

Tabel II-11 Contoh Penerapan *Stopping*

Data Latih	
Input	**Output**
acara apaan tu d anteve pesbuker sih prtunjukan tp jgn kasih liat yg ngeri dong	acara apaan anteve pesbuker prtunjukan jgn kasih liat ngeri

7. *Stemming*

Stemming adalah tahapan untuk membuat kata yang berimbuhan kembali ke bentuk asalnya. Contohnya kata "memberikan" setelah melewati tahap ini maka akan menjadi "beri" [10].

Tabel II-12 Penerapan tahapan *stemming*

Data Latih	
Input	**Output**
materi	Materi
kick	kick
andy	andy
barusan	baru
bagus	bagus
bnget	bnget
menginspirasi	insprirasi
sekali	sekali
ampe	ampe
netes	netes
aer	aer
mata	mata

II.6. Naïve Bayes Classifier

Naïve bayes classifier adalah penggolong menggunakan statistik sederhana berdasarkan teorema bayes yang mengasumsikan bahwa keberadaan atau ketiadaan dari suatu fitur tertentu dari suatu kelas tidak berhubungan dengan keberadaan atau ketiadaan fitur lainnya. Sifatnya yang sebagai model probabilitas, *naïve bayes classifier* bisa dilatih dengan efisien sebagai *supervised learning*. Pada pengaplikasiannya, parameter estimasi untuk model naïve bayes menggunakan metode kemungkinan maksimum. Dengan kata lain, masih bisa bekerja dengan model naïve bayes tanpa harus mempedulikan bayesian probabilitas atau metode bayesian lainnya.

Meskipun banyak sekali asumsi-asumsi yang disederhanakan, *naïve bayes classifier* telah terbukti mampu menyelesaikan masalah dunia nyata yang kompleks. Keunggulan yang dimiliki *naïve bayes classifier* adalah hanya membutuhkan sejumlah data latih yang kecil untuk memperkirakan parameter yang diperlukan untuk klasifikasi . Berikut adalah model matematis untuk *naïve bayes classifier*:

$$p(C|F_1, \dots, F_n) = \frac{p(C)p(F_1, \dots, F_n|C)}{p(F_1, \dots, F_n)} \quad \text{(II.1)}$$

II.7. Percentage Split

Algoritma pengklasifikasi yang diterapkan dalam penelitian ini adalah algoritma *naïve bayes classifier*, akan dievaluasi tingkat keberhasilannya dalam mengklasifikasikan dokumen. Dari banyak metode yang ada, dalam penelitian ini menggunakan metode yang digunakan untuk mengevaluasi algoritma *naïve bayes classifier* tersebut menggunakan *percentage split*. Algoritma *naïve bayes classifier* akan diuji dengan cara dari data latih yang digunakan akan dibagi dua untuk dijadikan sebagai data uji sebanyak n-% dan sisanya tetap sebagai data latih. Tentunya jumlah data latih harus lebih banyak dari data uji. Hasil *training* dari data latih yang sudah tadi akan digunakan untuk mengklasifikasikan data uji dan hasilnya menunjukan tingkat keakurasian algoritma *naïve bayes classifier* yang diimplementasikan [11].

II.8. Pemrograman Berorientasi Objek

Pemrograman berorientasi objek adalah sebuah pendekatan untuk pembangunan perangkat lunak di mana struktur dari perangkat lunak didasari dari objek yang saling berinteraksi satu sama lain untuk menyelesaikan suatu tugas. Sudut pandang dari pemrograman berorientasi objek menyelesaikan masalah dengan mengunakan objek, persis dengan apa yang ada di dunia nyata. Pemrograman berorientasi objek (PBO) dirancang untuk menutup kelemahan dari pemrograman berorientasi prosedural yang sulit untuk dikelola dan *debug* ketika kode yang dibuat telah mencapai beribu-ribu baris. Maka diperkenalkanlah struktur pemrograman untuk memecah kode ke dalam beberapa segmen atau biasa disebut prosedur atau fungsi. Dengan struktur tersebut kode mengalami perbaikan, tetapi secara program dapat melakukan fungsionalitas yang lebih kompleks. Berikut adalah konsep mendasar mengenai bahasa permrograman berorientasi objek.

1. Kelas

Kelas adalah struktur yang akan digunakan oleh objek yang berisikan atribut-atribut yang mendefiniskan data yang akan diolah dan juga berisikan *method* yang dapat berbentuk prosedur ataupun fungsi untuk dapat mengolah data. Dalam

konteks PBO biasanya didasari dari pemodelan dari objek yang ada di dunia nyata dalam pemrograman untuk menyelesaikan masalah. Sebagai contoh, dilakukan pemodelan untuk sepeda untuk dijadikan sebuah kelas. Dalam sebuah sepeda misalkan terdiri dari roda, rangka, dan setir. Ketiga unsur tersebut bisa disebut sebagai atribut. Kemampuan dari sepeda itu bisa berjalan dan melompat. Kemampuan tersebut bisa disebut *method* yang ada di dalam kelas.

2. Objek

Objek adalah instansiasi dari sebuah class. Kemampuan yang bisa dilakukan oleh suatu objek bergantung apa dari isi yang didefinisikan dari class tersebut.

3. Abstraksi

Ketika membuat objek dalam aplikasi PBO, penting untuk menggunakan konsep dari abstraksi. Objek hanya berisi informasi yang relevan dengan konteks dari aplikasi. Contoh untuk pembangunan aplikasi pengiriman. Akan dibuat sebuah objek dengan nama produk dengan atribut ukuran dan berat. Warna akan menjadi informasi yang tidak berhubungan dan bisa diabaikan sebagi atribut dari objek produk. Di kasus lain seperti pembangunan aplikasi pencatatan pemesanan, warna menjadi informasi yang berhubungan dan bisa dijadikan atribut dari suatu objek.

4. Enkapsulasi

Enkapsulasi adalah sebuat perses yang tidak mengizinkan akses langsung kepada data. Jika ingin mengakases dari data tersebut, diharuskan menggunakan objek yang memiliki tanggung jawab untuk terhadap data tersebut. Menggunakan contoh sebelumnya, ketika akan menggunakan *method* berjalan tidak bisa dilakukan secara langsung. Kelas sepeda tersebut harus diinstansiasi terlebih dahulu menjadi objek sepeda sehingga bisa mengakses *method* berjalan.

5. *Polymorphism*

Polymorphism adalah kemempuan untuk merespon pesan yang sama dengan cara yang berbeda. Misalkan ada dua objek yaitu sepeda dan motor. Dalam objek tersebut memiliki kemampuan method yang sama yaitu berjalan. Ketika *method* berjalan dari kedua objek tersebut dipanggil. Kedua objek tersebut menampilkan rekasi yang berbeda, sepeda berjalan dengan cara mengayuh pedal, sedangkan motor berjalan dengan cara menarik gas.

6. *Inheritance*

Inheritance dalam PBO untuk mengklasifikasikan objek di dalam program berdasarkan karakteristik umum. Dengan *inheritance* pemrograman lebih mudah karena bisa mengkombinasikan karakteristik umum ke dalam kelas induk kemudian bisa diturunkan ke kelas anaknya. Contoh dibuat kelas kendaraan yang berisikan *method* berjalan . Kemudian dibuat kelas kendaraan darat. Dalam kelas kendaraan darat tidak perlu membuat *method* berjalan di kelas kendaraan darat cukup dengan melakukan *inheritance* atau pewarisan *method* yang ada di kelas kendaraan ke kelas kendaraan darat [12].

II.9. Unified Modeling Language

Unified modeling language (UML) adalah bahasa pemodelan standar untuk pembangunan perangkat lunak ataupun sistem yang dibangun dengan pendekatan *object-oriented*. Dalam merancang sistem, model memiliki peranan penting yaitu untuk mengelola kompleksitas. Pemodelan membantu untuk fokus menghubungkan aspek penting dari perancangan sistem. Sebuah model adalah sebuah abstraksi dari benda nyata. Model yang dibangun adalah penyederhanaan dari sistem yang dibangun. Memberikan maksud dan kelangsungan hidup sistem untuk dipahami, dievaluasi, dikritisasi bahkan lebih cepat dibandingkan harus langsung ke sistem sebenarnya.

UML itu sendiri sudah mengalami beberapa generasi, dimulai dari UML 1.0, 1.5, dan yang terakhir 2.0. Versi pertama dari UML memberikan komuniasi rancangan secara jelas, menyampaikan esensi dari suat perancangan, bahkan menangkap peta kebutuhan fungsional dari solusi perangkat lunak. Bagaimanapun, dunia berubah secara fundamental dengan pengenalan pemodelan sistem, dibanding

hanya pemodelan perangkat lunak. Dari perubahan itulah yang membuat versi UML itu sendiri berkembang hingga kini 2.0. Secara keseluruhan ada 13 diagram yang dapat dimanfaatkan dari UML 2.0 ini. Namun dalam penelitian ini hanya akan diterapkan empat diagram. Tabel II-13 menjelasan mengenai diagram yang akan digunakan.

Tabel II-13 Diagram-Diagram Uml 2.0 yang Digunakan

Nama Diagram	Deskripsi
Use Case	Menggambarkan interaksi antara sistem yang akan dibangun dengan pengguna atau sistem eksternal. Juga membantu dalam pemetaan kebutuhan ke dalam sistem
Activity	Menggambarkan aktivitas secara sekuensial dan pararel dalam sistem.
Class	Menggambarkan hubungan *class*, *interface* dalam sistem.
Sequence	Menggambarkan interaksi diantara *object* dimana urutan dari interaksinya sangat penting