

PENERAPAN METODE GOMS UNTUK EVALUASI PERANGKAT LUNAK PEMODELAN VISUAL BERBASIS OPEN SOURCE

Harijanto Pangestu

Information Systems Department, School of Information Systems, Binus University
Jln. K.H. Syahdan No. 9, Palmerah, Jakarta Barat 11480
harijantopangestu@binus.ac.id

ABSTRACT

Software evaluation is needed to determine the usability level of the software. Good usability is having the following characteristics: effective to use (effectiveness), efficient to use (efficiency), safe to use (safety), having good utility (utility), easy to learn (learnability) and easy to remember how to use (memorability). Analytical evaluation is one evaluation approach that does not involve users. The evaluation is conducted by experts associated with the product. One method used in the analytical evaluation is GOMS, a method to analyze a task related to human computer interaction. GOMS does not provide accurate calculation of how users interact with the system, yet it estimates the time need to do a task dealing with the system. With the evaluation of UML users are expected to be able to use UML tools that appropriate to particular needs based on open source.

Keywords: *software evaluation, usability, GOMS, tools UML, open source*

ABSTRAK

Evaluasi perangkat lunak sangat dibutuhkan untuk mengetahui tingkat usability dari perangkat lunak tersebut. Usability yang baik memiliki karakteristik effective to use (effectiveness), efficient to use (efficiency), safe to use (safety), having good utilize (utility), easy to learn (learnability) dan easy to remember how to use (memorability). Analytical evaluation adalah salah satu pendekatan evaluasi yang tidak melibatkan pengguna. Evaluasi dilakukan oleh ahli yang berhubungan dengan produk tersebut. Salah satu metode yang dipakai dalam analytical evaluation adalah GOMS yaitu metode untuk menganalisis suatu tugas (task) dalam hubungannya dengan human computer interaction. Metode GOMS tidak memberikan suatu hasil perhitungan yang akurat tentang bagaimana pengguna berinteraksi dengan sistem tetapi memberikan estimasi waktu yang dibutuhkan untuk mengerjakan suatu tugas dalam berhubungan dengan sistem. Dengan adanya evaluasi ini diharapkan pengguna UML dapat menggunakan tools UML yang tepat sesuai dengan kebutuhan khusus berbasis open source.

Kata kunci: *evaluasi perangkat lunak, usability, GOMS, UML, tools UML, open source*

PENDAHULUAN

Unified Modeling Language (UML) yaitu satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Whitten et al., 2004). Untuk menggambar UML dibutuhkan perangkat lunak pemodelan visual. Sekarang ini banyak sekali perangkat lunak pemodelan visual baik berbasis *open source* maupun berbasis *proprietary* (berbayar). Untuk itu diperlukan evaluasi perangkat lunak tersebut sehingga pengguna dapat memilih sesuai kebutuhannya.

Menurut Helen Sharp et al. (2006) ada tiga pendekatan evaluasi yaitu: (1) *usability testing* – melibatkan tipe-tipe pengguna tertentu pada tipe tugas tertentu. Jumlah dan macam-macam kesalahan yang dibuat pengguna dicatat demikian pula dengan waktu yang dibutuhkan pengguna untuk menyelesaikan tugas tersebut. Kegiatan pengguna diawasi dan direkam dengan menggunakan *video recorder*; (2) *field studies* – pengguna melakukan kegiatan dalam lingkungan yang sewajarnya. Pendekatan ini bertujuan untuk memahami bagaimana pengguna melakukan kegiatan dan bagaimana produk dapat membantu kegiatan pengguna; (3) *analytical evaluation* – tidak melibatkan pengguna. Evaluasi dilakukan oleh ahli yang berhubungan dengan produk tersebut.

Dalam penelitian ini perangkat lunak pemodelan visual yang diteliti dibatasi hanya yang berbasis *open source* karena mudah diperoleh dan dana yang dibutuhkan relatif tidak besar. *Tools UML open source* yang akan dievaluasi adalah: Umlpad versi 3.2, Violet UML editor versi 0.21.1, ArgoUML versi 0.26.2, DIA versi 0.97, StarUML versi 5.0.2.1570, UMLet versi 9.1, Winbrello (Umbrello UML Modeller versi 1.5.71)

Perangkat lunak pemodelan visual ini dievaluasi untuk mengetahui tingkat *usability*-nya. *Usability* yang baik memiliki karakteristik *Effective to use (effectiveness)*, *Efficient to use (efficiency)*, *Safe to use (safety)*, *Having good utility (utility)*, *Easy to learn (learnability)* dan *Easy to remember how to use (memorability)* (Helen Sharp et al., 2006). *Usability* pada umumnya adalah mengenai tingkat kepastian dimana sebuah produk mudah dipelajari, efektif, dan menyenangkan menggunakannya dari sudut pandang pengguna. *Usability* merupakan hal yang sangat penting karena jika sebuah produk memiliki fungsi yang baik tetapi tidak memiliki *usability* yang baik, produk tersebut tidak dapat digunakan dengan maksimal dan kemungkinan produk tersebut tidak dapat dipakai dan ditinggalkan oleh penggunanya. *Usability* yang tinggi akan menjamin sebuah produk dapat digunakan dengan efektif, efisien, aman digunakan, memiliki fungsi yang baik, mudah dipelajari dan mudah digunakan.

Perangkat lunak dalam hal ini *Tools UML open source* yang dievaluasi diharapkan dapat membantu pengembangan sistem mulai dari tahap analisis, desain, sampai dengan tahap implementasi. Karena itu diagram UML yang diuji coba meliputi *Use-Case Model Diagrams*, *Class diagrams*, *Sequence diagrams*, *Activity diagrams*, *Component diagrams* serta *Deployment diagrams*.

Ben Shneiderman dan Catherine Plaisant (2005) menemukan tingkat keahlian pemakai sebagai berikut: (1) *novice (first-time users)* – profesional yang mengerti konsep tentang tugasnya tetapi memiliki konsep pengetahuan antarmuka yang dangkal. Perancangan yang cocok adalah dengan membatasi jumlah pilihan, umpan balik yang informatif, *user manual* dan tutorial *online* yang efektif; (2) *knowledgeable intermittent users* – profesional yang telah memiliki konsep tugas yang stabil dan konsep antarmuka yang luas tapi sulit mengingat atau menguasai struktur menu atau lokasi-lokasi fitur yang tersedia. Perancangan yang cocok adalah dengan menyediakan struktur menu yang rapi, konsisten, kejelasan antarmuka, perlindungan dari bahaya karena eksplorasi fitur; (3) *expert frequent users* – profesional yang terbiasa dengan konsep tugas dan antarmuka. Mereka ingin menyelesaikan

pekerjaannya dengan cepat. Perancangan yang cocok adalah dengan menyediakan makro, *shortcuts*, singkatan dan lain sebagainya.

METODE

GOMS

GOMS (*goals, operators, methods, dan selection rules*) adalah metode untuk menganalisis suatu tugas (*task*) yang dikembangkan oleh Stuart Card, Thomas P. Moran and Allen Newell pada tahun 1983 di dalam bukunya *The Psychology of Human Computer Interaction*. *Goals* berarti tujuan yang ingin dicapai oleh pengguna, *operators* berarti tindakan dasar yang harus dilakukan pengguna di dalam menggunakan sistem, *methods* berarti cara-cara yang dapat dilakukan untuk mencapai tujuan dengan menggunakan *operators* yang ada, dan *selection rules* berarti pilihan metode yang digunakan.

Metode GOMS tidak memberikan suatu hasil perhitungan yang akurat tentang bagaimana pengguna berinteraksi dengan sistem tetapi memberikan estimasi sebagai prediksi waktu yang dibutuhkan untuk mengerjakan suatu tugas dalam berhubungan dengan sistem.

Dalam melakukan analisis suatu tujuan dapat didekomposisi menjadi tujuan-tujuan yang lebih kecil yang selanjutnya sampai dengan level operator dasar. Dengan demikian GOMS merupakan suatu metode yang baik untuk mendeskripsikan bagaimana seorang ahli melakukan suatu tugas dalam hubungannya dengan suatu sistem yang sedang diuji. Karena itu GOMS bukan untuk menganalisis tugas-tugas dimana langkah-langkahnya kurang dipahami dan tidak dilakukan pada pengguna awam atau yang tidak berpengalaman.

Keystroke Level Model (KLM)

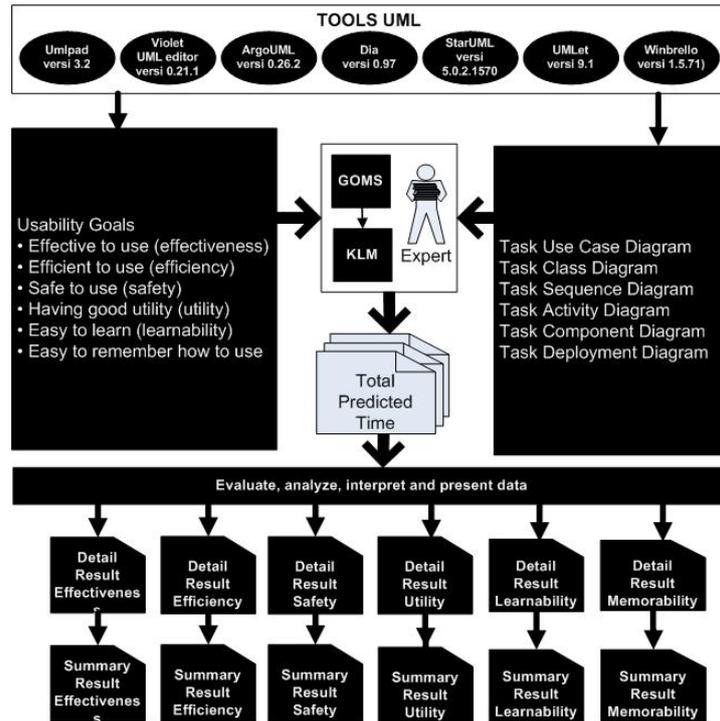
KLM adalah suatu metode untuk menganalisis suatu tugas (*task*) yang dikembangkan oleh Stuart Card, Thomas P. Moran and Allen Newell pada tahun 1983 di dalam bukunya *The Psychology of Human Computer Interaction*. KLM merupakan bentuk model GOMS tingkat terendah. Tabel 1 berikut memuat ketentuan *operators* dan waktu pengerjaan menurut Helen Sharp et al. (2006):

Tabel 1
Ketentuan Operator dan Waktu Pengerjaan

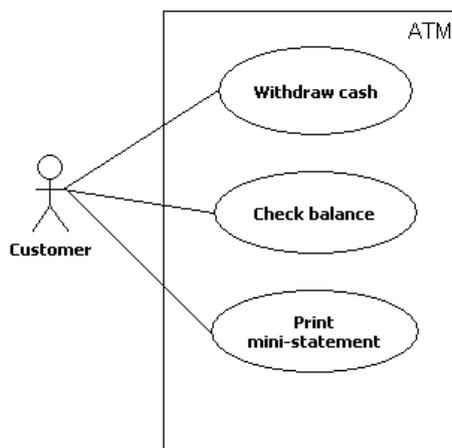
Operator Name	Description	Time
K	Pressing a single key or button	0.35 (average)
	Skilled typist (55 wpm)	0.22
	Average typist (40 wpm)	0.28
	User unfamiliar with the keyboard	1.20
	Pressing shift or control key	0.08
P	Pointing with a mouse or other device to a target on a display	1.10
P _i	Clicking the mouse or similar device	0.20
H	Homing hands on the keyboard or other device	0.40
D	Draw a line using a mouse	Variable depending on the length of line
M	Mentally prepare to do something (e.g. make a decision)	1.35
R (t)	System response time – counted only if it causes the user to wait when carrying out their task	t

Operator “K - Keystroking” berarti menekan suatu tuts pada papan tuts, operator ”P - Pointing” berarti memindahkan *mouse* ke target atau menekan tombol pada *mouse*, operator ”H - Homing” berarti memindahkan tangan dari *mouse* ke papan tuts, operator “D – Drawing” berarti menggambar dengan menggunakan *mouse*, operator “M – Mental Preparation” berarti mempersiapkan sikap mental untuk suatu aksi fisik, operator “R – Response” berarti respons sistem dalam waktu ketika pengguna harus menunggu dalam usaha menyelesaikan suatu tugas.

HASIL DAN PEMBAHASAN



Gambar 1. Evaluasi menggunakan GOMS-KLM



Gambar 2. Use case diagram (sumber: Simon Bennet et al., 2005).

Contoh GOMS-UseCase

Goal: Menggambar simbol UseCase

Operators: Move-mouse
Click mouse button
Type Characters

Methods:

Untuk menggambar simbol UseCase:

Click→move-mouse→click (*click methods*)

Untuk memberi nama UseCase:

Type Characters (*type characters methods*)

Selection rules:

Untuk menggambar simbol UseCase *click methods*.

Untuk memberi nama UseCase gunakan *type characters methods*.

Keystroke Level Model (KLM)

Keystroke Level Model merupakan suatu metode untuk menganalisis suatu tugas (*task*). KLM merupakan bentuk model GOMS tingkat terendah. Hasil KLM dari tugas *Use Case Diagram* yang dikerjakan oleh penulis akan digunakan untuk memilih tools UML dalam hal mengerjakan tugas-tugas diagram UML berikutnya. Tiga Tools UML yang terpilih akan digunakan dalam kuesioner. Hasil akhir dari KLM untuk seluruh tugas menggambar diagram akan dibandingkan dengan hasil kuesioner yang dikerjakan oleh responden. Tabel 2 berikut memuat Contoh KLM untuk Menggambar Use Case “Withdraw Cash” Menggunakan Click Methods:

Tabel 2

Contoh Keystroke Level Model (KLM) untuk Menggambar Use Case “Withdraw Cash” Menggunakan Click Methods:

Description	Operator	Duration (sec)	Description	Operator	Duration (sec)
Mentally prepare	M	1.35	Type “t”	K	0.22
Move cursor to icon “UseCase” in toolbox	P	1.10	Type “h”	K	0.22
Click icon “UseCase”	P ₁	0.20	Type “d”	K	0.22
Mentally prepare	M	1.35	Type “r”	K	0.22
Move cursor to stage	P	1.10	Type “a”	K	0.22
Click cursor to draw	P ₁	0.20	Type “w”	K	0.22
Homing hands on the keyboard	H	0.40	Type “space”	K	0.22
Mentally prepare	M	1.35	Type “C”	K	0.22
Pressing shift key	K	0.35	Type “a”	K	0.22
Type “W” (good typist)	K	0.22	Type “s”	K	0.22
Type “i”	K	0.22	Type “h”	K	0.22
			Total predicted time		10.26

Tabel 3 menjelaskan kemampuan tools UML open source untuk menggambar diagram-diagram UML. Berdasarkan penelitian penulis perangkat lunak Dia, UMLet dan Winbrello dapat digunakan untuk menggambar *Use-Case diagrams*, *Class diagrams*, *Sequence diagrams*, *Activity diagrams*, *Component diagrams* serta *Deployment diagrams*. Dengan demikian perangkat lunak Dia, UMLet dan Winbrello akan dipakai oleh penulis untuk tahap penelitian selanjutnya.

Selanjutnya Tabel 4 menjelaskan *total predicted time* perangkat lunak yang diuji dalam menggambar *Use Case Diagram*. Sesuai dengan Tabel 3, berdasarkan penelitian penulis, perangkat lunak Dia, UMLet dan Winbrello mampu untuk menggambar *Use-Case Diagrams* dengan baik dan lengkap. Dengan demikian perangkat lunak Dia, UMLet dan Winbrello akan dipakai oleh penulis untuk tahap penelitian selanjutnya.

Tabel 3
Diagram UML vs Tools UML Open Source

DIAGRAM UML	Umleditor Ver 3.0.1	UML Pad Ver 3.2	Violet Ver 0.21.1	ArgoUML Ver 0.26.2	Dia Ver 0.97	StarUML Ver 5.0.2.1570	Umllet Ver 9.1	Winbrello Ver 1.5.71
1 Use-Case diagrams		√	√	√	√	√	√	√
2 Class diagrams	√	√	√	√	√	√	√	√
3 Object diagrams			√		√			
4 Sequence diagrams		√	√	√	√	√	√	√
5 Collaboration diagrams				√	√	√		√
6 Statechart diagrams		√	√	√	√	√	√	√
7 Activity diagrams		√	√	√	√	√	√	√
8 Component diagrams					√	√		√
9 Deployment diagrams				√	√	√	√	√
Installation	copy	copy	copy	setup	setup	setup	copy	copy
 Tipe File	Executable Jar File	Application	Executable Jar File	Application	Application	Application	Executable Jar File	Application

Tabel 4
Total Predicted Time - Task Use Case Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)								
	Umleditor Ver 3.0.1	UML Pad Ver 3.2	Violet Ver 0.21.1	ArgoUML Ver 0.26.2	Dia Ver 0.97	StarUML Ver 5.0.2.1570	Umllet Ver 9.1	Winbrello Ver 1.5.71	
1 Menggambar actor "Customer"	-	9.78	11.06	10.26	10.26	9.16	12.86	9.38	
2 Memberi nama actor "Customer"	-	10.88	10.66	11.36	10.26	10.26	15.76	10.48	
3 Menggambar UseCase "Withdraw cash"	-	10.88	10.66	11.36	10.26	10.26	15.76	10.48	
4 Memberi nama UseCase "Withdraw cash"	-	12.42	12.20	12.90	11.80	11.80	17.30	12.02	
5 Menggambar UseCase "Check balance"	-	18.60	10.60	18.60	18.60	18.60	32.40	9.20	
6 Memberi nama UseCase "Check balance"	-	-	-	15.45	-	-	-	-	
7 Menggambar UseCase "Print mini-statement"	-	-	-	5.65	7.30	7.30	15.31	7.30	
8 Memberi nama UseCase "Print mini-statement"	-	-	-	-	10.66	-	-	8.28	
9 Menggambar simbol association	-	-	-	-	-	-	-	-	
10 Menghapus multiplicity	-	-	-	-	-	-	-	-	
11 Menggambar simbol System Boundary	-	-	-	-	-	-	-	-	
12 Memberi nama System Boundary	-	-	-	-	-	-	-	-	
TOTAL PREDICTED TIME	-	62.56	55.18	85.58	79.14	67.38	109.39	67.14	

Berdasarkan kemampuan perangkat lunak mendukung diagram-diagram UML dan *Total Predicted Time* (Tabel 5 – 10) untuk *Use Case Diagram*, perangkat lunak Dia, UMLet dan Winbrello merupakan perangkat lunak yang akan di evaluasi lebih lanjut.

Tabel 5
Total Predicted Time – Task Class Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)		
	Dia	Umlet	Winbrello
	Ver 0.97	Ver 9.1	Ver 1.5.71
1 Menggambar Class "Company"		9.34	10.24
2 Visible attributes dan operations Class "Company"	14.74	-	6.95
3 Menggambar Class "Department"	13.02	13.54	11.12
4 Menambah atribut "name:Name" pada Class "Departement"	13.86	7.10	12.56
5 Menggambar Class "Office"	13.02	12.44	10.02
Menambah atribut pada Class Office			
6 - attribute "address: String"	26.43	11.76	23.83
7 - attribute "voice: Number"			
8 Menggambar Class "Headquarters"		13.76	11.34
9 Visible attributes dan operations Class "Headquarters"	15.84	-	6.95
10 Menggambar Class "Person"	11.92	12.44	10.02
Menambah attribute pada Class "Person"			
11 - attribute "name: Name"	38.69	16.11	34.79
12 - attribute "employeeID: Integer"			
13 - attribute "title: String"			
Menambah operation pada Class "Person"			
14 - operation "getPhoto(p:Photo)"	43.65	27.31	44.95
15 - operation "getSoundBite()"			
16 - operation "getContactInformation()"			
17 - operation "getPersonalRecords()"			
18 Menggambar Class "ContactInformation"	14.91	15.43	13.01
19 Menambah atribut "address: String" pada Class "ContactInformation"	14.96	8.20	34.79
20 Menggambar Class "PersonelRecords"	14.25	14.77	12.35
Menambah attribute pada Class "PersonalRecords"			
21 - attribute "taxID"	26.86	11.72	26.86
22 - attribute "employmentHistory"			
23 - attribute "salary"			
Dari Class "Company" ke Class "Department"			
24 - Menggambar composition	11.60	8.60	8.40
25 - Menambah multiplicity	10.60	7.41	11.90
Dari Class "Company" ke Class "Office"			
26 - Menggambar composition	11.60	8.60	8.40
27 - Menambah multiplicity	10.60	7.41	11.90
Pada Class "Departement"			
28 - Menggambar composition itself"	19.50	13.00	5.50
29 - Menambah multiplicity	10.60	7.41	14.30
Dari Class "Department" ke Class "Office"			
30 - Menggambar association	7.30	8.60	7.30
31 - Menambah nama dan multiplicity	15.45	19.75	19.95
Dari Class "Department" ke Class "Person"			
32 - Menggambar association	15.20	8.60	8.60
33 - Tidak menampilkan direction			
34 - Menambah role "member"	16.62	20.19	18.82
35 - Menambah multiplicity			
Dari Class "Department" ke Class "Person"			
36 - Menggambar association	15.20	8.60	8.60
37 - Tidak menampilkan direction			
38 - Menambah role "manajer"	15.83	19.40	19.04
39 - Menambah multiplicity			
40 - Menggambar dan memberi nama Constraint	13.21	15.26	-
Dari Class "Office" ke Class "Headquarters"			
41 - Menggambar generalization	7.30	8.60	8.60
Dari Class "Person" ke Class "ContactInformation"			
42 - Menggambar dependency	7.30	8.60	8.60
Dari Class "Person" ke Class "PersonnelRecord"			
43 - Menggambar dependency	7.30	8.60	8.60
Pada Class "PersonnelRecord"			
44 - Menggambar dan memberi nama interface	15.76	22.86	-
TOTAL PREDICTED TIME	473.12	375.41	438.29

Tabel 6
Total Predicted Time – Task Sequence Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)		
	Dia	Umlet	Winbrello
	Ver 0.97	Ver 9.1	Ver 1.5.71
1 Menggambar simbol object dan memberi nama "c:Client"	9.51	14.36	10.81
2 Menggambar simbol object dan memberi nama "":Transaction"	10.04	14.89	11.34
3 Menggambar simbol object dan memberi nama "p:ODBCProxy"	10.52	13.62	11.34
4 Memberi nama stereotype "transient" pada object "":Transaction"	10.53	11.33	-
5 Memberi <i>lifeline</i> pada Object "c:Client"	8.60	7.50	-
Memberi <i>activation</i> pada Object "c:Client"		7.50	-
6 Memberi <i>lifeline</i> pada Object "":Transaction"	8.60	7.50	-
Memberi <i>activation</i> pada Object "":Transaction"		7.50	-
7 Memberi <i>lifeline</i> pada Object "p:ODBCProxy"	8.60	7.50	-
Memberi <i>activation</i> pada Object "p:ODBCProxy"		7.50	-
8 Menggambar <i>message</i> dan memberi <i>message-type</i> "<< create >>" dari <i>activation-Object</i> "c:Client" ke <i>object</i> "":Transition"	13.80	15.25	13.65
9 Menggambar <i>message</i> dan memberi <i>message-label</i> "setAction(a,d,o)" dari <i>activation-Object</i> "c:Client" ke <i>object</i> "":Transition"	21.27	16.22	14.62
10 Menggambar <i>message</i> dan memberi <i>message-label</i> "setValues(d,3,4)" dari <i>activation-object</i> "":Transition" ke <i>activation-object</i> "p:ODBCProxy"	21.27	16.22	13.32
11 Menggambar <i>message</i> dan memberi <i>message-label</i> "setValues(a,"CO")" dari <i>activation-object</i> "":Transition" ke <i>activation-object</i> "p:ODBCProxy"	22.89	17.84	16.24
12 Menggambar <i>message</i> , memberi <i>message-type</i> "return" dan memberi <i>message-label</i> "committed" dari <i>activation-Object</i> "":Transition" ke <i>object</i> "c:Client"	20.18	19.04	12.03
13 Menggambar <i>message</i> dan memberi <i>message-type</i> "<< destroyed >>" dari <i>activation-Object</i> "c:Client" ke <i>object</i> "":Transition"	13.80	15.47	13.87
14 Menggambar tanda <i>destruction</i> pada <i>lifetime</i> pada <i>object</i> "":Transition"	5.25	5.25	-
TOTAL PREDICTED TIME	184.86	204.49	117.22

Tabel 7

Total Predicted Time – Task Activity Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)		
	Dia	Umlet	Winbrello
	Ver 0.97	Ver 9.1	Ver 1.5.71
1 Menggambar simbol Swimlane dan memberi nama "Customer"	17.31	12.71	17.31
2 Menggambar simbol Swimlane dan memberi nama "Sales"	16.65	12.05	16.65
3 Menggambar simbol Swimlane dan memberi nama "Warehouse"	17.53	12.93	17.53
4 Menggambar simbol Start State	3.95	5.05	3.95
5 Menggambar simbol Activity dan memberi nama "Request Product"	12.05	14.25	17.80
6 Menggambar simbol Activity dan memberi nama "Process Order"	11.61	13.81	17.36
7 Menggambar simbol Activity dan memberi nama "Pull Materials"	11.83	14.03	17.58
8 Menggambar simbol Activity dan memberi nama "Ship order"	10.95	13.15	16.70
9 Menggambar simbol Activity dan memberi nama "Receive order"	10.95	13.81	17.36
10 Menggambar simbol Activity dan memberi nama "Bill customer"	11.61	13.81	17.36
11 Menggambar simbol Activity dan memberi nama "Pay bill"	10.51	12.71	16.26
12 Menggambar simbol Activity dan memberi nama "Close order"	11.61	13.37	16.92
13 Menggambar simbol Transition dari "start-state" ke state "Request product"	8.60	10.80	6.60
14 Menggambar simbol Fork antara "Request product" ke "Process order"	5.30	7.50	5.30
15 Menggambar simbol Transition dari "Request product" ke "fork"	8.60	10.80	-
16 Menggambar simbol Transition dari "fork" ke "join"	8.60	10.80	-
17 Menggambar simbol Transition dari "fork" ke "Process order"	8.60	10.80	-
18 Menggambar simbol Transition dari "Process order" ke "Pull materials"	8.60	10.80	6.60
19 Menggambar simbol Transition dari "Pull material" ke "Ship order"	8.60	10.80	6.60
20 Menggambar simbol Transition dari "Ship order" ke "join"	8.60	10.80	-
21 Menggambar simbol Join setelah "Ship order"	8.60	7.50	-
22 Menggambar simbol fork	5.30	7.50	5.30
23 Menggambar simbol Transition dari "join" ke "fork"	8.60	10.80	-
24 Menggambar simbol Transition dari "fork" ke "Receive order"	8.60	10.80	-
25 Menggambar simbol Transition dari "fork" ke "Bill customer"	8.60	10.80	-
26 Menggambar simbol Transition dari "Receive order" ke "join"	8.60	10.80	-
27 Menggambar simbol Transition dari "Bill customer" ke "join"	8.60	10.80	-
28 Menggambar simbol Transition dari "join" ke "Pay bill"	8.60	10.80	-
29 Menggambar simbol Transition dari "Pay bill" ke "Close order"	8.60	10.80	6.60
30 Menggambar simbol End State	5.65	5.05	3.95
31 Menggambar simbol Transition dari "Close order" ke "End State"	8.60	10.80	6.60
TOTAL PREDICTED TIME	300.41	341.23	240.33

Tabel 8

Total Predicted Time – Task Component Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)		
	Dia	Umllet	Winbrello
	Ver 0.97	Ver 9.1	Ver 1.5.71
1 Menggambar simbol dan memberi nama Component "image.java"	9.82	15.00	9.82
2 Menggambar simbol dan memberi nama Component "component.java"	10.70	15.88	10.70
3 Menggambar simbol <i>Dependency</i>	7.30	10.80	6.60
4 Menggambar simbol <i>Interface</i>	7.30	10.80	-
5 Memberi nama "ImageObserver" pada simbol <i>interface</i> (khusus Dia dan UMLet)	9.31	8.16	-
6 Memberi nama "ImageObserver" pada simbol <i>dependency</i> (khusus Winbrello)	-	-	8.23
TOTAL PREDICTED TIME	44.43	60.64	35.35

Tabel 9

Total Predicted Time – Task Deployment Diagram

ACTIVITY	TOTAL PREDICTED TIME (in seconds)		
	Dia	Umllet	Winbrello
	Ver 0.97	Ver 9.1	Ver 1.5.71
1 Menggambar simbol <i>node</i> dan memberi node "kiosk"	11.70	13.90	11.70
2 Menggambar simbol <i>node</i> dan memberi node "console"	12.14	14.34	12.14
3 Menggambar simbol <i>node</i> dan memberi node "server"	11.92	14.12	11.92
4 Menggambar simbol <i>node</i> dan memberi node "RAID farm"	12.80	15.00	12.80
5 Menggambar simbol <i>Link dari node "kiosk" ke node "server"</i>	7.50	10.80	6.60
Memberi nama "<< 10-T Ethernet >>" pada simbol <i>Link dari node "kiosk" ke node "server"</i>	13.13	14.88	13.13
6 Menggambar simbol <i>Link dari node "console" ke node "server"</i>	7.50	10.80	6.60
Memberi nama "<< RS-232 >>" pada simbol <i>Link dari node "console" ke node "server"</i>	11.59	13.34	11.59
7 Menggambar simbol <i>Link dari node "server" ke node "RAID farm"</i>	7.50	10.80	6.60
TOTAL PREDICTED TIME	95.78	117.98	93.08

Tabel 10

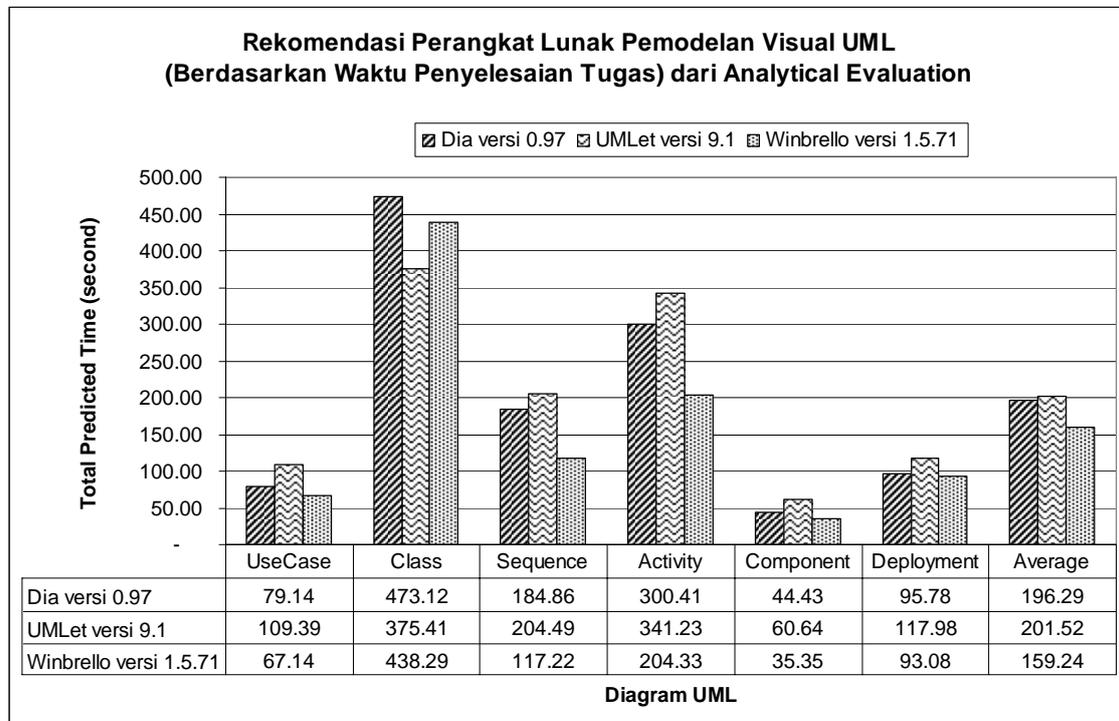
Summary Total Predicted Time

	Dia versi 0.97	UMLet versi 9.1	Winbrello versi
Task Use Case Diagram	79.14	109.39	67.14
Task Class Diagram	473.12	375.41	438.29
Task Sequence Diagram	184.86	204.49	117.22
Task Activity Diagram	300.41	341.23	204.33
Task Component Diagram	44.43	60.64	35.35
Task Deployment Diagram	95.78	117.98	93.08
Total Predicted Time (in seconds)	1,177.74	1,209.14	955.41

Rekomendasi Perangkat Lunak Pemodelan Visual

Berdasarkan Tabel 10 Perangkat Lunak DIA merupakan perangkat lunak dengan *total predicted time* yang terkecil. Winbrello memiliki nilai yang lebih kecil dikarenakan ada beberapa aktivitas yang tidak dapat dilakukan. UMLet merupakan perangkat lunak dengan tingkat *total predicted time* pada posisi kedua.

Perbedaan waktu rata-rata untuk menyelesaikan tugas menggambar diagram UML antara perangkat lunak UMLet dan perangkat lunak DIA hanya berbeda sedikit nilai (Gambar 3). Untuk seluruh tugas menggambar diagram UML, dalam *analytical evaluation* perangkat lunak UMLet versi 9.1 memiliki nilai yang lebih besar dibandingkan dengan perangkat lunak DIA versi 0.97. Untuk menggambar *Class Diagram* UMLet versi 9.1 memiliki *total predicted time* yang lebih kecil dibandingkan dengan perangkat lunak DIA versi 0.97 dan Winbrello versi 1.571



Gambar 3. Waktu penyelesaian tugas berdasarkan *analytical evaluation*.

Berdasarkan hasil di atas, ada beberapa hal yang harus diperhatikan pada saat penerapan metode GOMS-KLM ini yaitu: (1) *effective to use* – dapat diketahui dengan melihat berapa banyak tugas (*task*) yang dapat diselesaikan oleh perangkat lunak tersebut; (2) *efficient to use* – dapat diketahui dengan melihat berapa *total predicted time* yang diperoleh untuk menyelesaikan suatu tugas (*task*) oleh perangkat lunak tersebut; (3) *safe to use* – mengandung pengertian “*free from error*” sulit untuk dievaluasi dengan menggunakan metode ini karena kesalahan-kesalahan pada saat melakukan tugas (*task*) tersebut tidak dicatat dalam *total predicted time* yang diperoleh; (4) *having good utility* – sulit untuk dievaluasi dengan menggunakan metode ini; (5) *easy to learn* – sulit untuk dievaluasi dengan menggunakan metode ini karena dilakukan oleh pengguna yang sudah mahir menggunakan perangkat lunak ini; (6) *easy to remember how to use* – sulit untuk dievaluasi dengan menggunakan metode ini karena metode ini dilakukan oleh pengguna yang sudah mahir menggunakan perangkat lunak ini.

PENUTUP

Berdasarkan hasil penelitian dan analisis yang telah disampaikan pada paragraf sebelumnya, dapat diambil beberapa simpulan sebagai berikut: (1) *functional requirement* mutlak harus dipenuhi tetapi *non-functional requirement* pun dalam hal ini sisi user interface tidak boleh diabaikan karena

akan meningkat *usability* perangkat lunak tersebut; (2) pengelompokan notasi dalam *toolbox tools UML* akan membantu pengguna menggambar diagram UML; (3) perangkat lunak pemodelan visual berbasis *open source* harus dilakukan evaluasi secara berkala karena perangkat lunak ini harus dapat menyesuaikan dengan perkembangan versi UML selanjutnya; (4) dalam melakukan *analytical evaluation* perlu dilakukan juga *usability testing* untuk melengkapi beberapa kelemahan dari *analytical evaluation*; (5) berdasarkan *analytical evaluation* dengan metode GOMS-KLM, perangkat lunak DIA versi 0.97 memiliki *effective to use* dan *efficient to use* yang lebih baik dibandingkan dengan perangkat lunak UMLet versi 9.1 dan perangkat lunak Winbrello versi 1.5.71, sedangkan untuk kriteria *usability* yang lain perlu dilakukan penelitian pendukung; (6) perangkat lunak DIA versi 0.97 belum dapat dikatakan memiliki *usability* yang baik karena hanya memiliki *effective to use* dan *efficient to use* yang baik dibandingkan dengan perangkat lunak UMLet versi 9.1 dan perangkat lunak Winbrello versi 1.5.71 sedangkan kriteria *usability* lainnya belum diketahui.

Untuk meningkatkan penelitian selanjutnya, diberikan saran-saran sebagai berikut: (1) *analytical evaluation* sebaiknya dilengkapi dengan *usability testing*; (2) penelitian ini dapat ditingkatkan dengan melakukan evaluasi *CASE tools UML* berbasis *open source* mulai dari tahap pemodelan visual sampai dengan *generate source code*; (3) beberapa saran untuk peningkatan perangkat lunak yang dievaluasi sehingga meningkatkan *usability* yaitu: (a) perlunya informasi (tool tips) untuk notasi di *toolbox* sehingga tidak membingungkan pengguna khususnya untuk perangkat lunak Winbrello; (b) perlu penambahan notasi dan penyempurnaan perangkat lunak sehingga sesuai dengan perkembangan versi UML selanjutnya (ada beberapa notasi tidak terdapat di perangkat lunak Winbrello dan ada beberapa hal tidak dapat digambarkan dengan menggunakan perangkat lunak ini). Contoh: notasi Interface dan Constraint tidak dapat digambarkan pada class diagram, notasi Activation tidak dapat digambarkan pada sequence diagram, kesulitan koneksi antara notasi *fork*, notasi *join*, dan notasi *activity* pada *activity diagram*, notasi *interface* tidak dapat digambarkan pada *component diagram*; (c) perlunya Kemudahan koneksi (keterhubungan antar notasi) dalam penggambaran suatu diagram UML untuk perangkat lunak DIA dan UMLet dilakukan secara manual sedangkan untuk perangkat lunak Winbrello sudah terhubung secara otomatis; (d) stabilitas perangkat lunak perlu disempurnakan (perangkat lunak Winbrello kadang-kadang menutup secara otomatis bila terjadi kesalahan sehingga semua yang telah dikerjakan menghilang dan menyulitkan *user*).

DAFTAR PUSTAKA

- Bennet, S., Skelton, J., Lunn, K. (2005). *Schaum's Outline of UML* (2nd edition). New York: McGraw-Hill.
- Card, S., Moran, Thomas P., & Newell, Allen. (1983). *The Psychology of Human Computer Interaction*. New Jersey: Lawrence Erlbaum Associates.
- Sharp, H., Rogers, Y., Preece, J. (2006), *Interaction Design: Beyond Human-Computer Interaction* (2nd edition). New Jersey: John Wiley & Sons.
- Shneiderman, Ben & Plaisant, Catherine. (2005). *Designing The User Interface: Strategies for Effective Human-Computer Interaction* (4th edition). Boston: Pearson Education.
- Whitten, Jeffrey L.; Bentley, Lonnie D., & Dittman, Kevin C. (2004). *Metode Desain dan Analisis Sistem* (edisi 6). Jakarta: Andi dan McGraw-Hill Education.