

ISBN : 978 - 602 - 14432 - 2- 4



PROCEEDING

SEMINAR NASIONAL PENDIDIKAN

STKIP SURYA

Tangerang, 15 Februari 2014



“Integrasi Keterampilan Abad 21 dalam Kurikulum 2013 untuk Mewujudkan Indonesia Jaya”

Penyelenggara:

Sekolah Tinggi Keguruan dan Ilmu Pendidikan SURYA

Jalan Scientia Boulevard Blok U Nomor 7 Gading Serpong

Tangerang Banten 15810

PROCEEDING

SEMINAR NASIONAL PENDIDIKAN



STKIP SURYA

Tangerang, 15 Februari 2014

“INTEGRASI KETERAMPILAN ABAD 21 DALAM KURIKULUM 2013 UNTUK MEWUJUDKAN INDONESIA JAYA”

Penanggung Jawab Proceeding

Jutri Taruna, Ph.D

TIM REVIEWER

- 1) Johannes Hamonangan Siregar., Ph.D
- 2) Ali Godjali, Ph.D
- 3) Josephine Kusuma, Ph.D
- 4) Dr. Nancy Susianna, M.Pd
- 5) Dr. Doddy Kustaryono, S.Si., Apt., MS., DEA.
- 6) Agus Purwanto, Ph.D

Diterbitkan Oleh:

Sekolah Tinggi Keguruan dan Ilmu Pendidikan Surya
(STKIP SURYA)

Alamat Penerbit:

Jalan Scientia Boulevard Blok U Nomor 7

Gading Serpong Tangerang Banten 15810

Telepon: 021-5464-196, 021-5464-535

Email: info@stkipsurya.ac.id , website : www.stkipsurya.ac.id

KATA PENGANTAR

Puji syukur kehadiran Tuhan Yang Maha Esa atas tersusunnya buku *proceeding* Seminar Nasional Sekolah Tinggi Keguruan dan Ilmu Pendidikan Surya. Tema seminar adalah Intregasi Keterampilan Abad 21 dalam Kurikulum 2013 untuk Mewujudkan Indonesia Jaya.

Dalam kesempatan ini perkenankan kami mengucapkan terima kasih kepada:

1. Prof. Yohanes Surya, Ph.D., Dr. Nancy Susianna, M.Pd., Onno W. Purbo, Ph.D., dan Bryan Holzer, M.BA sebagai pemakalah utama dalam seminar ini.
2. Ketua STKIP Surya yang telah memfasilitasi semua kegiatan seminar nasional ini.
3. Bapak/Ibu segenap panitia seminar nasional yang telah meluangkan waktu, tenaga, dan pemikirannya demi suksesnya seminar ini.
4. Bapak/Ibu peserta dan pemakalah.

Semoga buku *proceeding* ini dapat memberi kemanfaatan bagi kita semua, untuk kepentingan pengembangan ilmu pengetahuan dan teknologi. Harapan kami seminar ini dapat berkontribusi dalam menyelesaikan masalah pendidikan di Indonesia.

Tangerang, 15 Februari 2014

Ketua Panitia,

Jutri Taruna, Ph.D.

Daftar Isi

Halaman Judul	i
Kata Pengantar	ii
Sambutan Ketua STKIP Surya	iii
Daftar Isi	iv

Makalah Sidang Utama

No	Pemakalah	Judul	Halaman
1	Dr. Nancy Susianna, M.Pd	IMPLEMENTASI KETERAMPILAN ABAD 21 DALAM KURIKULUM 2013	MU – 1
2	Onno W Purbo, Ph.D	BEBERAPA PRINSIP TIK UNTUK PENDIDIKAN	MU – 14
3	Bryan Holzer, M.BA	21 st CENTURY EDUCATION IN THE UNITED STATES	MU – 27

Makalah Sidang Paralel

No	Pemakalah	Judul	Halaman
1	Dede Trie Kurniawan, Ida Hamidah	MODEL PEMBELAJARAN BERBASIS MASALAH BERBANTUAN WEBSITE PADA KONSEP FLUIDA STATIS UNTUK MENINGKATKAN PENGUASAAN KONSEP SISWA KELAS XI.	1 – 6
2	Francisca Harunning T	PENGARUH BLOG SEBAGAI MEDIA PEMBELAJARAN TERHADAP HASIL DAN MOTIVASI BELAJAR SISWA SMP PADA TOPIK LENSAN.	7 – 15
3	Asep Sutiadi	PROFIL KEMAMPUAN CALON GURU FISIKA DALAM MENINGKONSTRUKSI TES KOGNITIF BENTUK PILIHAN GANDA.	16 – 23
4	Heni Rusnayati, Ade Yeti Nuryantini, Sayida Al Adawi	PENERAPAN MEDIA KOMIK BUNYI UNTUK MENINGKATKAN PEMAHAMAN SISWA PADA MATERI AJAR BUNYI.	24 – 35
5	Intan Irawati	PENGEMBANGAN KETERAMPILAN MOTORIK PESERTA DIDIK MELALUI PEMBUATAN MEDIA PEMBELAJARAN FISIKA.	36 – 43
6	Niki Dian Permana P, Dea Nurul Utami, Dadi Rusdiana	PROFIL KETERAMPILAN BERPIKIR KRITIS SISWA PADA PEMBELAJARAN IMPULS DAN MOMENTUM.	44 – 51

No	Pemakalah	Judul	Halaman
16	Wijoyono	PEMBELAJARAN SAINS YANG MENYENANGKAN (<i>JOYFUL LEARNING</i>) DENGAN MEMANFAATKAN LINGKUNGAN MANGROVE SEBAGAI SUMBER BELAJAR KEANEKARAGAMAN ORGANISME DEKOMPOSISI.	127 - 135
17	Adi Kusjani, Badiyanto	ANALISIS PERBANDINGAN PENERAPAN PARADIGMA <i>OBJECT ORIENTED PROGRAMMING</i> DAN <i>ASPECT ORIENTED PROGRAMMING</i> PADA APLIKASI BERBASIS WEB.	137 - 146
18	Sherly Melinda	DESAIN PEMBELAJARAN UNTUK MENGEMBANGKAN KEMAMPUAN LOGIKA ANAK MENGGUNAKAN <i>SCRATCH</i> DI <i>RASPBERRY PI</i> .	147 – 156
19	Fadillah Hisyam, Fadjriah Hapsari	EFEKTIFITAS METODE BELAJAR <i>E-LEARNING</i> UNTUK PELAJARAN NON EKSAK (Studi Kasus Di Universitas X Jakarta Timur).	157 – 161
20	Siti Shinta Andayani, Raffy Hidayat	PENGEMBANGAN SISTEM PEMBELAJARAN <i>E-LEARNING</i> DENGAN METODE <i>LMS (LEARNING MANAGEMENT SISTEM)</i> MENGGUNAKAN <i>MOODLE</i> .	162 - 170
21	Marlindawati	PEMANFAATAN PERANGKAT LUNAK AJAR SEBAGAI MULTIMEDIA INTERAKTIF DALAM PENINGKATAN PEMAHAMAN MAHASISWA	171 – 178
22	Alfa Satya Putra	PENGEMBANGAN ALAT “ <i>TYPING TUTOR</i> ” UNTUK BELAJAR MENGETIK DENGAN <i>KEYBOARD</i> .	179 – 189
23	Dewanti Liem	ISU DAN TANTANGAN DALAM MENGAJAR <i>OBJECT ORIENTED PROGRAMMING</i>	190 – 196
24	Patmah Fatoni	PENGUKURAN PARAMETER KERMA DAN KUAT KERMA SUMBER BRAKITERAPI <i>LOW DOSE RATE IR-192</i> DI UDARA DAN AIR MENGGUNAKAN SIMULASI <i>MONTE CARLO EGSNRCMP</i>	197 - 207

**PROCEEDING SEMINAR NASIONAL PENDIDIKAN MATEMATIKA,
SAINS, DAN TIK STKIP SURYA 2014**

“Integrasi Keterampilan Abad 21 Dalam Kurikulum 2013
Untuk Mewujudkan Indonesia Jaya”

**ANALISIS PERBANDINGAN PENERAPAN PARADIGMA
OBJECT ORIENTED PROGRAMMING DAN ASPECT ORIENTED
PROGRAMMING PADA APLIKASI BERBASIS WEB**

Adi Kusjani¹⁾, Badiyanto²⁾

^{1,2)}STMIK Akakom Yogyakarta

adikusja@akakom.ac.id

badiyanto@akakom.ac.id

ABSTRAK

Paradigma Pemrograman Berorientasi Objek (*Object-Oriented Programming* atau *OOP*) telah menjadi paradigma utama pemrograman dalam beberapa tahun terakhir, yang menggeser paradigma pemrograman terstruktur. Kemudian muncul paradigma baru yaitu Pemrograman Berorientasi Aspek (*Aspect Oriented Programming* atau *AOP*) yang dikembangkan untuk memperbaiki permasalahan paradigma *OOP*. Paradigma *AOP* ini diperkirakan akan berkembang pesat dan menjadi acuan sebagai paradigma pemrograman bagi programmer. Pada penelitian ini, kami akan menganalisis perbandingan penerapan paradigma *OOP* dan *AOP* dalam sebuah program aplikasi berbasis web, dimana untuk penerapan paradigma *AOP* menggunakan *framework* *PostSharp 3.0.36*, dengan batasan pada fungsionalitas program yang meliputi: fungsi *logging*, otorisasi (*authorization*), dan penanganan kesalahan (*error handling*), yang akan diimplementasikan pada kedua paradigma pemrograman tersebut. Kaidah-kaidah yang digunakan untuk membandingkan kedua paradigma pemrograman ini meliputi: *reusability*, *understandability*, *maintainability* dan *testability*. Untuk membandingkannya digunakan beberapa metode *CK metric* berikut ini: *Weighted Methods per Class (WMC)*, *Response for Class (RFC)*, *Lack of Cohesion (LCOM)*, *Coupling Between Object Classes (CBO)*, *Depth of Inheritance Tree (DIT)*, dan *Number of Children (NOC)*. Dari Penelitian ini dapat diambil kesimpulan bahwa aplikasi *AOP* yang dibangun menggunakan *framework* *PostSharp 3.0.36*, lebih baik pada kaidah *reusability*, *maintainability* dan *testability*, sedangkan kaidah *understandability* aplikasi *OOP* lebih baik..

Kata Kunci: *aspect oriented programming, authorization, error handling, logging, CK metric.*

PENDAHULUAN

Pemrograman berorientasi objek (*object-oriented programming* disingkat *OOP*) merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam *kelas-kelas* atau *objek-objek*. Lebih jauh lagi, pendukung *OOP* mengklaim bahwa *OOP* lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya (pemrograman terstruktur), dan pendekatan *OOP* lebih mudah dikembangkan dan dirawat. Namun permasalahan dapat muncul jika didalam objek-objek tersebut terdapat suatu aspek berbeda yang unik. Hal ini akan menimbulkan permasalahan *code tangling* yaitu kondisi dimana struktur dari program tersebut tidak dapat ditelusuri dengan baik karena adanya *code-code* yang sama di beberapa bagian yang berbeda dalam program yang mengakibatkan aplikasi

menjadi sulit dikelola dan diubah. Permasalahan lainnya *scattering* yaitu kejadian dimana beberapa aspek yang sama muncul pada setiap kelas yang membutuhkannya. Kemudian muncul jenis paradigma baru yaitu Penrograman Berorientasi Aspek (*Aspect Oriented Programming* atau AOP) yang dikembangkan untuk memperbaiki paradigma OOP. AOP merupakan sebuah teknik yang melengkapi OOP dan berfungsi untuk mengatasi masalah diatas. Jika AOP dianalogikan dengan proses normalisasi perancangan basis data, maka AOP merupakan teknik normalisasi *cross cutting concern* yang tersebar pada sebuah sistem, dan mengelompokkan *cross cutting concern* tersebut menjadi sebuah *aspect*.

Berdasarkan latar belakang maka perlu adanya analisa perbandingan penerapan teknik perangkat lunak OOP dengan AOP dan sejauh mana kelebihan menerapkan paradigma AOP pada aplikasi berbasis web dibandingkan dengan paradigma OOP. Pada penelitian ini aplikasi OOP dibangun menggunakan ASP.NET MVC 3 Web Application dengan bahasa C# sedangkan AOP dibangun menggunakan ASP.NET MVC 3 Web Application dengan bahasa C# dan *framework* PostSharp 3.0.36.

Penelitian ini bertujuan untuk membuktikan kelebihan penerapan teknik AOP dibandingkan dengan teknik OOP, dengan mengevaluasi berdasarkan kaidah-kaidah perangkat lunak yang berkualitas menggunakan beberapa metode *CK metric*. Penelitian ini diharapkan dapat mengetahui kelebihan maupun kekurangan penerapan paradigma AOP pada suatu aplikasi, sehingga dapat dijadikan pertimbangan untuk menerapkan paradigma AOP pada aplikasi yang akan dibangun, khususnya yang menggunakan bahasa C# dengan *framework* ASP.NET MVC Web application dan PostSharp 3.0.36.

METODE PENELITIAN

Metode penelitian yang dilakukan meliputi tahap-tahap sebagai berikut:

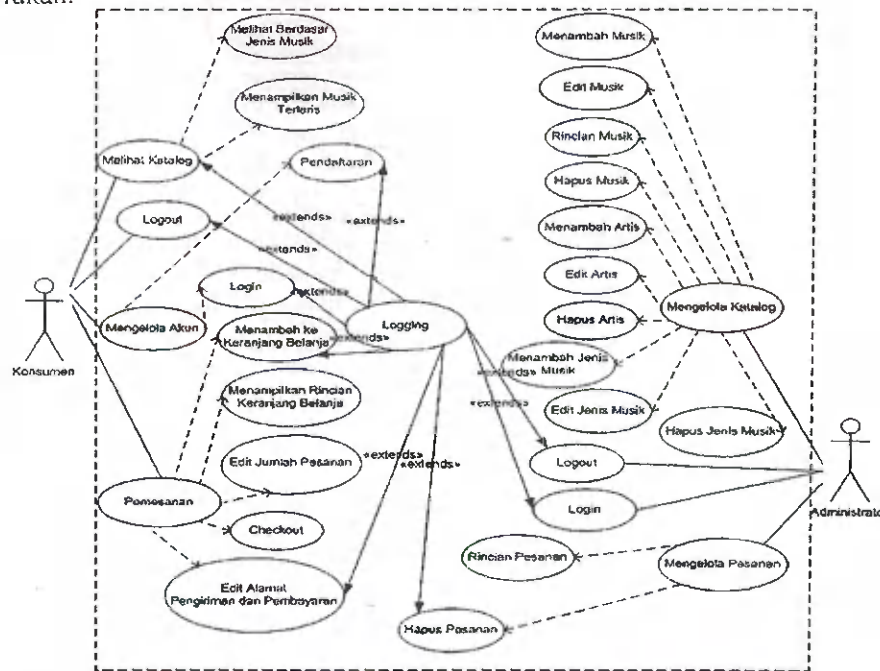
1. Membangun aplikasi berbasis web dengan teknik OOP menggunakan ASP.NET MVC 3 Web Application dengan bahasa C# dan AOP dibangun menggunakan ASP.NET MVC 3 Web Application dengan bahasa C# dan *framework* PostSharp 3.0.36., dimana kedua aplikasi memiliki sistem yang sama, sedangkan yang berbeda pada teknik mengimplementasikan, yang satu dengan paradigma OOP sedangkan yang lain dengan paradigma AOP.
2. Melakukan pengujian pada aplikasi yang dibangun dengan OOP dan AOP dan memastikan sistem kedua aplikasi berjalan dengan baik, dimana masing-masing aplikasi memiliki fungsionalitas program sebagai berikut: fungsi *logging*, otorisasi (*authorization*), dan penanganan kesalahan (*exception handling*)
3. Membandingkan kedua aplikasi web OOP dan AOP, dengan kaidah-kaidah yang menunjukkan kualitas dari perangkat lunak meliputi: *reusability*, *understandability*, *maintainability* dan *testability*
4. Membandingkannya menggunakan metode *CK metric* yang meliputi:
 - *Weighted Methods per Class* (WMC) digunakan untuk mengukur banyaknya method yang diimplementasikan dalam kelas atau jumlah keseluruhan kompleksitas method (CC).
 - *Response for Class*(RFC), digunakan untuk menghitung banyaknya method yang kemungkinan di eksekusi sebagai *response* atas message objek dari kelas tersebut.
 - *Lack of Cohesion* (LCOM), digunakan untuk mengukur derajat kemiripan method oleh variabel input data atau atribut dalam class, dengan cara menghitung jumlah method

- yang tidak memiliki irisan atribut dengan method lainya dikurangi dengan method yang memiliki irisan atribut dengan method lainnya.
- *Coupling Between Object Classes (CBO)*, digunakan untuk menghitung jumlah class lainya yang *non-inheritance* dimana class tersebut di *couple*(didalam satu class memanggil method dari class lainya).
- *Depth of Inheritance Tree (DIT)*, digunakan untuk mengukur kedalaman dari suatu class pada *inheritance hierarchy tree*, dengan cara menghitung jumlah tingkatan dari kelas node ke root dari *inheritance hierarchy tree*.
- *Number of Children (NOC)* merupakan jumlah *subclass* yang diturunkan langsung dari suatu *class*.

HASIL DAN PEMBAHASAN

1. Use Case Diagram

Dalam penelitian ini *use case* diagram dipisahkan menjadi 3 berdasarkan fungsionalitas tambahan yang kemudian akan dijadikan aspek, yaitu: *Logging*, *Authorization* dan *Exception*. Berikut ini salah satu contoh *use case* diagram, dengan aspek *logging*, dapat dilihat pada gambar 1. Dalam *use case* diagram ini, fungsionalitas tambahan *logging* yang merupakan fungsi perekaman suatu kejadian dan menyimpan hasil rekaman tersebut yang biasanya dapat dianalisa jika diperlukan.



Gambar 1. Use case diagram dengan aspek logging

2. Class Diagram

Class diagram adalah diagram yang menunjukkan *class-class* yang ada dari sebuah sistem dan hubungannya secara logika. *Class* diagram dari sistem aplikasi yang dibuat dapat dilihat pada gambar 2.

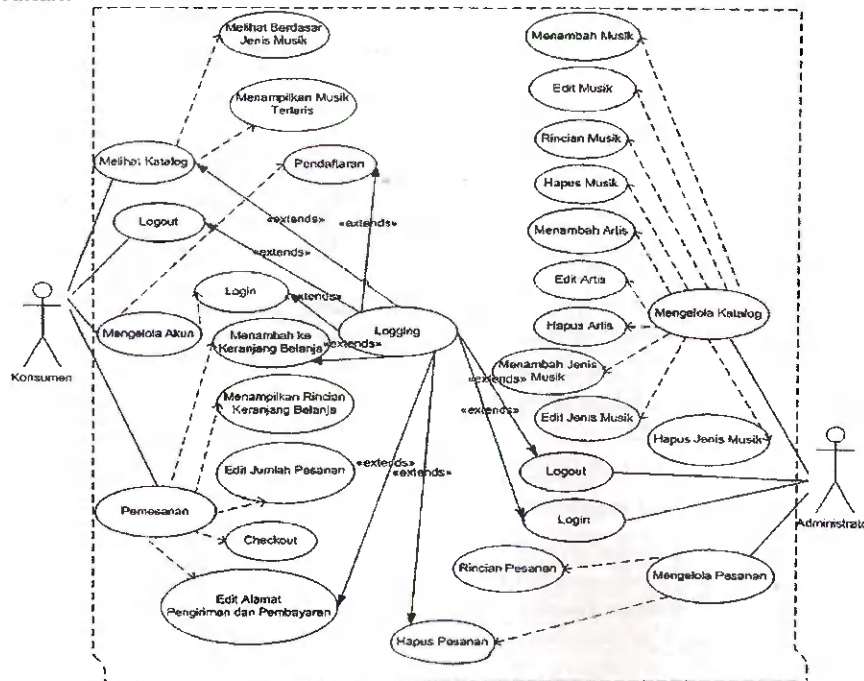
yang tidak memiliki irisan atribut dengan method lainya dikurangi dengan method yang memiliki irisan atribut dengan method lainya.

- *Coupling Between Object Classes* (CBO), digunakan untuk menghitung jumlah class lainya yang *non-inheritance* dimana class tersebut di *couple*(didalam satu class memanggil method dari class lainya).
- *Depth of Inheritance Tree* (DIT), digunakan untuk mengukur kedalaman dari suatu class pada *inheritance hierarchy tree*, dengan cara menghitung jumlah tingkatan dari kelas node ke root dari *inheritance hierarchy tree*.
- *Number of Children* (NOC) merupakan jumlah *subclass* yang diturunkan langsung dari suatu *class*..

HASIL DAN PEMBAHASAN

1. Use Case Diagram

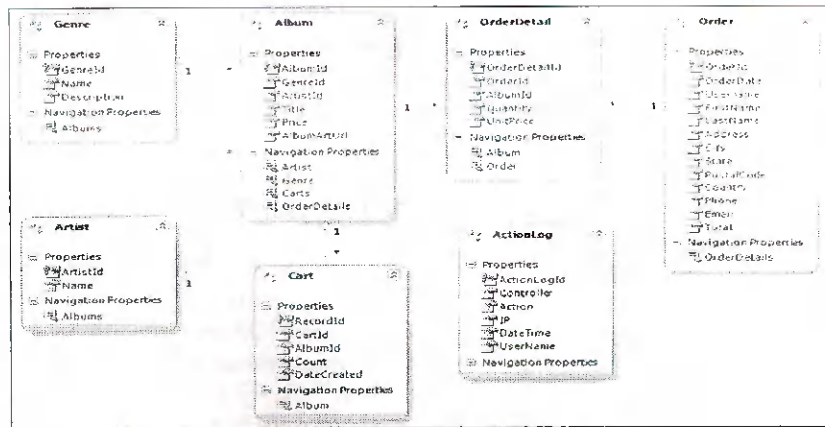
Dalam penelitian ini *use case* diagram dipisahkan menjadi 3 berdasarkan fungsionalitas tambahan yang kemudian akan dijadikan aspek, yaitu: *Logging*, *Authorization* dan *Exception*. Berikut ini salah satu contoh *use case* diagram, dengan aspek *logging*, dapat dilihat pada gambar 1. Dalam *use case* diagram ini, fungsionalitas tambahan *logging* yang merupakan fungsi perekaman suatu kejadian dan menyimpan hasil rekaman tersebut yang biasanya dapat dianalisa jika diperlukan.



Gambar 1. Use case diagram dengan aspek logging

2. Class Diagram

Class diagram adalah diagram yang menunjukkan *class-class* yang ada dari sebuah sistem dan hubungannya secara logika. *Class* diagram dari sistem aplikasi yang dibuat dapat dilihat pada gambar 2.



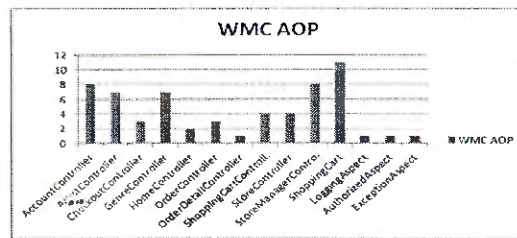
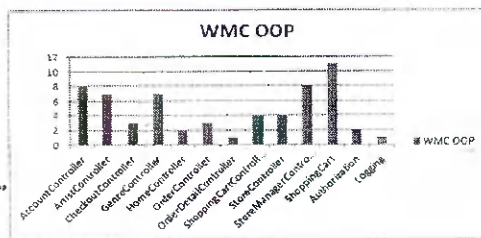
Gambar 2. Class diagram sistem aplikasi

3. Analisa Aplikasi

- Berikut tabel dan diagram hasil perhitungan WMC pada aplikasi OOP dan AOP.

Tabel 1. Hasil Perhitungan WMC dari Aplikasi OOP dan AOP

Nama Kelas	WMC OOP	Nama Kelas	WMC AOP
AccountController	8	AccountController	8
ArtistController	7	ArtistController	7
CheckoutController	3	CheckoutController	3
GenreController	7	GenreController	7
HomeController	2	HomeController	2
OrderController	3	OrderController	3
OrderDetailController	1	OrderDetailController	1
ShoppingCartController	4	ShoppingCartController	4
StoreController	4	StoreController	4
StoreManagerController	8	StoreManagerController	8
ShoppingCart	11	ShoppingCart	11
Authorization	2	LoggingAspect	1
Logging	1	AuthorizedAspect	1
Total	61	ExceptionAspect	1
		Total	61



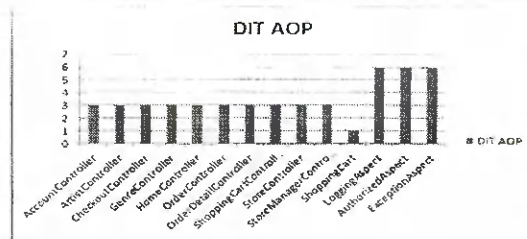
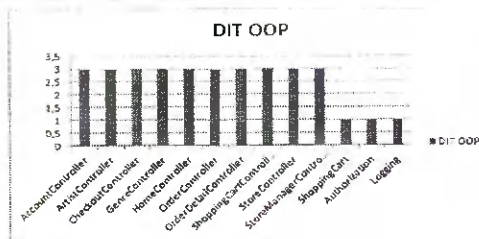
Gambar 3. Diagram Hasil Perhitungan WMC dari Aplikasi OOP dan AOP

Hasil perhitungan WMC dari masing-masing aplikasi OOP dan AOP sama dengan 61. Hasil perhitungan WMC berpengaruh terhadap *understandability*, *maintenance*, *reusability* dan *testability*, maka dapat disimpulkan pengukuran dengan metrik *Weighted Methods per Class*, menunjukkan tingkat *understandability*, *maintenance*, *reusability* dan *testability*, dari kedua aplikasi OOP dan AOP sama.

- Berikut tabel dan diagram hasil perhitungan DIT pada aplikasi OOP dan AOP.

Tabel 2. Hasil Perhitungan DIT dari Aplikasi OOP dan AOP

Nama Kelas	DIT OOP	Nama Kelas	DIT AOP
AccountController	3	AccountController	3
ArtistController	3	ArtistController	3
CheckoutController	3	CheckoutController	3
GenreController	3	GenreController	3
HomeController	3	HomeController	3
OrderController	3	OrderController	3
OrderDetailController	3	OrderDetailController	3
ShoppingCartController	3	ShoppingCartController	3
StoreController	3	StoreController	3
StoreManagerController	3	StoreManagerController	3
ShoppingCart	1	ShoppingCart	1
Authorization	1	LoggingAspect	6
Logging	1	AuthorizedAspect	6
Total	33	ExceptionAspect	6
		Total	49



Gambar 4. Diagram Hasil Perhitungan DIT dari Aplikasi OOP dan AOP

Hasil perhitungan DIT dari aplikasi OOP sama dengan 33 dan dari aplikasi AOP sama dengan 49. Hasil perhitungan DIT berpengaruh terhadap *understandability* dan *reusability*, maka dapat disimpulkan pengukuran dengan metrik *Depth of Inheritance Tree*, menunjukkan tingkat *understandability* dari aplikasi OOP lebih baik dibandingkan dengan aplikasi AOP, sedangkan tingkat *reusability* aplikasi AOP lebih baik dibandingkan dengan OOP.

- Berikut tabel dan diagram hasil perhitungan NOC pada aplikasi OOP dan AOP.

Tabel 3. Hasil Perhitungan NOC dari Aplikasi OOP dan AOP

Nama Kelas	NOC OOP	Nama Kelas	NOC AOP
AccountController	0	AccountController	0
ArtistController	0	ArtistController	0
CheckoutController	0	CheckoutController	0
GenreController	0	GenreController	0
HomeController	0	HomeController	0
OrderController	0	OrderController	0
OrderDetailController	0	OrderDetailController	0
ShoppingCartController	0	ShoppingCartController	0
StoreController	0	StoreController	0
StoreManagerController	0	StoreManagerController	0
ShoppingCart	0	ShoppingCart	0
Authorization	0	LoggingAspect	0
Logging	0	AuthorizedAspect	0
Total	0	ExceptionAspect	0
		Total	0



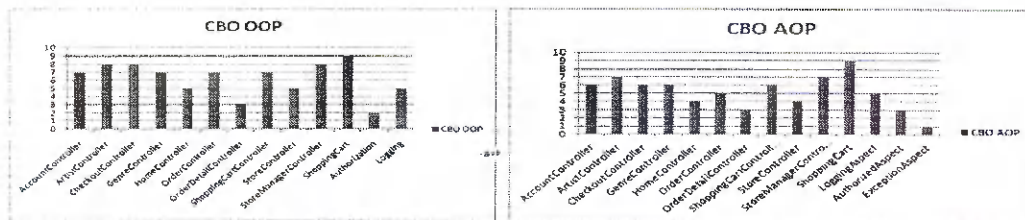
Gambar 5. Diagram Hasil Perhitungan NOC dari Aplikasi OOP dan AOP

Hasil perhitungan NOC dari masing-masing aplikasi OOP dan AOP sama dengan 0. Hasil perhitungan NOC berpengaruh terhadap *reusability* dan *testability*, maka dapat disimpulkan pengukuran dengan metrik *Number of Children*, menunjukkan tingkat *reusability* dan *testability* dari kedua aplikasi OOP dan AOP sama.

- Berikut tabel dan diagram hasil perhitungan CBO pada aplikasi OOP dan AOP.

Tabel 4. Hasil Perhitungan CBO dari Aplikasi OOP dan AOP

Nama Kelas	CBO OOP	Nama Kelas	CBO AOP
AccountController	7	AccountController	6
ArtistController	8	ArtistController	7
CheckoutController	8	CheckoutController	6
GenreController	7	GenreController	6
HomeController	5	HomeController	4
OrderController	7	OrderController	5
OrderDetailController	3	OrderDetailController	3
ShoppingCartController	7	ShoppingCartController	6
StoreController	5	StoreController	4
StoreManagerController	8	StoreManagerController	7
ShoppingCart	9	ShoppingCart	9
Authorization	2	LoggingAspect	5
Logging	5	AuthorizedAspect	3
Total	81	ExceptionAspect	1
		Total	72



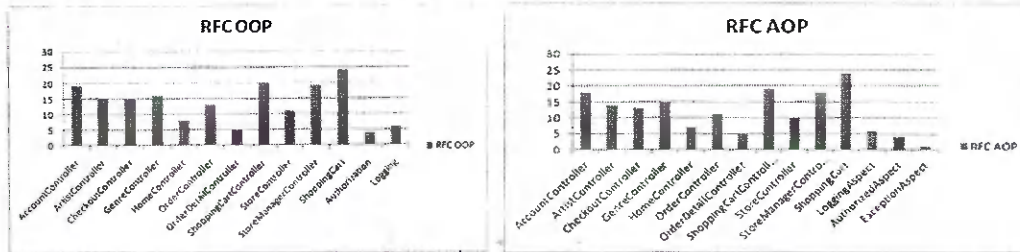
Gambar 6. Diagram Hasil Perhitungan CBO dari Aplikasi OOP dan AOP

Hasil perhitungan CBO dari aplikasi OOP sama dengan 81 dan dari aplikasi AOP sama dengan 72 dimana hasil perhitungan CBO berpengaruh terhadap *reusability* dan *maintainability*, maka dapat disimpulkan pengukuran dengan metrik *Coupling Between Object*, menunjukkan tingkat *reusability* dan *maintainability* dari aplikasi AOP lebih baik dibandingkan dengan OOP.

- Berikut tabel dan diagram hasil perhitungan RCF pada aplikasi OOP dan AOP.

Tabel 5. Hasil Perhitungan RFC dari Aplikasi OOP dan AOP

Nama Kelas	RFC OOP	Nama Kelas	RFC AOP
AccountController	19	AccountController	18
ArtistController	15	ArtistController	14
CheckoutController	15	CheckoutController	13
GenreController	16	GenreController	15
HomeController	8	HomeController	7
OrderController	13	OrderController	11
OrderDetailController	5	OrderDetailController	5
ShoppingCartController	20	ShoppingCartController	19
StoreController	11	StoreController	10
StoreManagerController	19	StoreManagerController	18
ShoppingCart	24	ShoppingCart	24
Authorization	4	LoggingAspect	6
Logging	6	AuthorizedAspect	4
Total	175	ExceptionAspect	1
		Total	165



Gambar 7. Diagram Hasil Perhitungan RFC dari Aplikasi OOP dan AOP

Dengan melihat hasil perhitungan RFC diatas, dimana hasil perhitungan RFC dari aplikasi OOP sama dengan 175 dan dari aplikasi AOP sama dengan 165, dimana hasil perhitungan RFC berpengaruh terhadap *reusability* dan *testability*, maka dapat disimpul-kan pengukuran dengan metrik *Response for Class*, menunjukkan tingkat *reusability*, *maintainability* dan *testability* dari aplikasi AOP lebih baik dibandingkan dengan OOP.

- Berikut tabel dan diagram hasil perhitungan LCOM pada aplikasi OOP dan AOP.

Tabel 6. Hasil Perhitungan LCOM dari Aplikasi OOP dan AOP

Nama Kelas	LCOM OOP	Nama Kelas	LCOM AOP
AccountController	16	AccountController	16
ArtistController	1	ArtistController	1
CheckoutController...	1	CheckoutController	1
GenreController	1	GenreController	1
HomeController	0	HomeController	0
OrderController	1	OrderController	1
OrderDetailController	0	OrderDetailController	0
ShoppingCartController	0	ShoppingCartController	0
StoreController	4	StoreController	4
StoreManagerController	0	StoreManagerController	0
ShoppingCart	9	ShoppingCart	9
Authorization	0	LoggingAspect	0
Logging	0	AuthorizedAspect	0
Total	33	ExceptionAspect	0
		Total	33

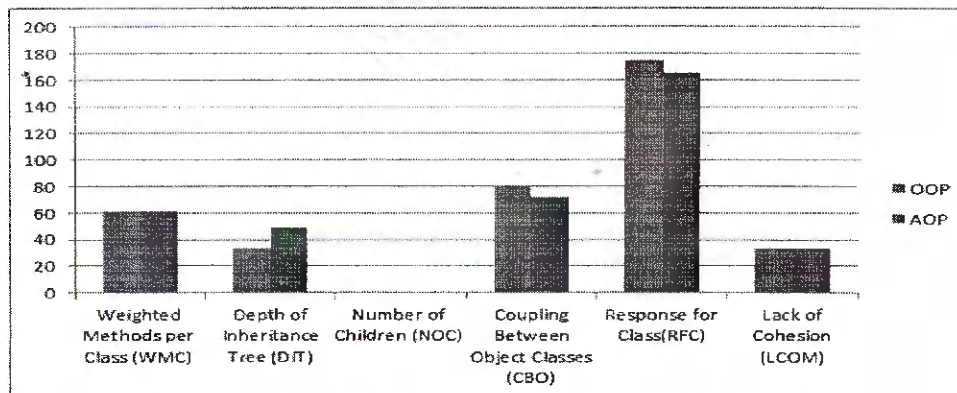
Dengan melihat hasil perhitungan LCOM diatas, dimana hasil perhitungan LCOM dari aplikasi OOP dan AOP sama dengan, dimana hasil perhitungan LCOM berpengaruh terhadap *reuseability*, *maintainability* dan *understanability*. Maka dapat kita simpulkan pengukuran dengan metode metrik *Lack of Cohesion in Methods*, menunjukkan tingkat *reuseability*, *maintainability* dan *understanability* dari aplikasi AOP dan OOP sama.

5.2.2.7 Analisa Perbandingan OOP dan AOP dengan Metode CK Metric.

Berikut ini tabel dan diagram dari metode CK metric, untuk mengukur kaidah-kaidah yang menunjukkan kualitas dari perangkat lunak meliputi: *reusability*, *understandability*, *maintainability* dan *testability*.

Tabel 7. Hasil Perbandingan OOP dan AOP dengan Metode CK Metric

Metode CK Metric	OOP	AOP
Weighted Methods per Class (WMC)	61	61
Depth of Inheritance Tree (DIT)	33	49
Number of Children (NOC)	0	0
Coupling Between Object Classes (CBO)	81	72
Response for Class(RFC)	175	165
Lack of Cohesion (LCOM)	33	33



Gambar 9. Diagram Hasil Perbandingan OOP dan AOP dengan Metode CK Metric

Dengan melihat diagram hasil perbandingan aplikasi OOP dan AOP, terlihat untuk metode *Weighted Methods per Class (WMC)*, *Number of Children (NOC)* dan *Lack of Cohesion in Methods (LCOM)*, masing-masing memiliki nilai yang sama, yang berarti dengan pengukuran ketiga metode metrik ini kedua aplikasi memiliki tingkat *reusability*, *understandability*, *maintainability* dan *testability* yang sama. Untuk metode *Depth of Inheritance Tree (DIT)* nilai aplikasi OOP lebih rendah dibandingkan dengan AOP, yang berarti tingkat *understandability* dari aplikasi OOP lebih baik dibandingkan dengan aplikasi AOP, sedangkan tingkat *reusability* aplikasi AOP lebih baik dibandingkan dengan OOP. Untuk metode *Coupling Between Object Classes (CBO)* nilai aplikasi OOP lebih tinggi dibandingkan dengan AOP, yang berarti dengan pengukuran ini tingkat *reusability* dan *maintainability* dari aplikasi AOP lebih baik, Sedangkan untuk metode *Response for Class* nilai aplikasi AOP lebih rendah dibandingkan dengan OOP, yang berarti dengan pengukuran ini tingkat *reusability*, *maintainability* dan *testability* aplikasi AOP lebih baik.

SIMPULAN DAN SARAN

1. Simpulan

Berdasarkan hasil penelitian dan pembahasan maka dapat di buat beberapa kesimpulan dari perbandingan aplikasi OOP yang dibangun dengan ASP.NET MVC dan AOP yang dibangun dengan ASP.NET MVC menggunakan *framework* PostSharp, sebagai berikut:

- Dengan metode *Weighted Methods per Class* (WMC), *Number of Children* (NOC) dan *Lack of Cohesion in Methods*, menunjukkan kedua aplikasi yang dibangun memiliki tingkat *reusability*, *understandability*, *maintainability* dan *testability* yang sama.
- Dengan metode *Depth of Inheritance Tree* (DIT), menunjukkan tingkat *understandability* aplikasi OOP lebih baik, sedangkan tingkat *reusability* aplikasi AOP lebih baik.
- Dengan metode *Coupling Between Object Classes*, menunjukkan tingkat *reusability* dan *maintainability* dari aplikasi AOP lebih baik.
- Dengan metode *Response for Class*, menunjukkan tingkat *reusability*, *maintainability* dan *testability* aplikasi AOP lebih baik.
- Dari empat hal diatas dapat disimpulkan, aplikasi AOP lebih baik pada kaidah *reusability*, *maintainability* dan *testability*, sedangkan kaidah *understandability* OOP lebih baik.

2. Saran

Untuk pengembangan lebih lanjut dari penelitian ini kami menyarankan untuk aplikasi yang di buat lebih kompleks dan memiliki jumlah kelas yang banyak mengandung *cross-cutting concerns*.

DAFTAR PUSTAKA

- Mahendra, Adhari C., 2004, *Pengantar Pemrograman Berbasis Aspek (AOP)*, <http://dataku.50webs.com/download/PengantarPemrogramanBerbasisAspek%28AOP%29.pdf>, diakses 10 Januari 2013.
- Prasojo, Nugroho Gito., 2005, *Penerapan teknik aspect oriented programming dengan studi kasus: sistem card fraud monitoring*, <http://lib.ui.ac.id/harvest/index.php/record/view/41869>, diakses 10 Februari 2013.
- Xerox Palo Alto Research Center, 2007, *A Study on Exception Detection and Handling Using Aspect-Oriented Programming*, <http://www2.parc.com/csl/groups/sda/publications/papers/PARC-AOP-RG97/for-web.pdf>, diakses 12 Desember 2012.
- El-Ahmadi, Abdellatif., "Software Quality Metrics for Object Oriented Systems", Technical University of Denmark, Kongens Lyngby, 2006.
- Jawadekar, Waman S., "Software Engineering: Principles and Practice", Tata McGraw-Hill, New Delhi, 2004.
- SATC., "Software Quality Metrics for Object Oriented System Enviroments", NASA Goddard Space Flight Center, Grenbelt Maryland, 1995.
- Barbacci, Mario R. "Software Quality Atributes: Modifiability and Usability", Carnegie Mellon University, Pittsburgh, 2004.
- Fenton, Norman E., "Software Metrics: A Rigourous and Practical Approach", PWS Publishing Company, Boston, 1997.
- Rhamdani, "Evaluasi Kualitas Perangkat Lunak Berorientasi Objek", [Online]. Available: <http://repository.ipb.ac.id/handle/123456789/17981>, Computer Science IPB, Bogor, 2008.
- Chidamber, S. dan Chris F. Kemerer., "Metrics Suite for Object Oriented Design", IEEE Transaction on Software Engineering, vol.20 no.6, 1994.
- Arthana, Resika dan Sunario, "Matric Software", [Online]. Available: <http://www.rey1024.com/index.php?s=Metric+Software>, 2011.
- Suhanto, Agus., "Mengenal ASP .NET MVC", Indonesia .NET Developer Community, 2008.
- Kurniawan, Erick., *Pemrograman Web Dinamis dengan ASP.NET 4.5*, Andi, Yogyakarta, 2012.
- Amri, M. Choirul., "Pengantar ASP.NET", [Online]. Available: <http://komputer34.files.wordpress.com/2008/12/pengantar-aspnet.pdf>, 2003.

ISBN : 978 – 602 – 14432 – 2 – 4

- Subagyo, Hendro., “*ASP.NET*”. <http://brainmatics.com/asp-net/>, diakses 10 Oktober 2013
- Hyrazz., “*Apa itu Microsoft SQL Server*”. <http://mugi.or.id/blogs/9lagank/archive/2011/06/25/apa-itu-microsoft-sql-server.aspx>, diakses 20 Agustus 2013
- Wikipedia, *Pemrograman berorientasi objek*, http://id.wikipedia.org/wiki/Pemrograman_berorientasi_objek, diakses 15 Agustus 2013.
- <http://sdn.vlsm.org/share/ServerLinux/node173.html>, diakses 15 Oktober 2013.
- Puspita, Nindya., “*Exception handling*”. <http://nindyapuspita178.wordpress.com/2013/05/07/bab-6-exception-handling/>, diakses 15 Oktober 2013.
- Sakti, Bima., “*SharpCrafters PostSharp 2.1.6.12*”, <http://airsalahdino.blogspot.com/2012/04/sharpcrafters-postsharp-21612-keygen.html>, diakses 16 Oktober 2013.



SERTIFIKAT SEMINAR NASIONAL PENDIDIKAN
"Integrasi Keterampilan Abad 21 dalam Kurikulum 2013
untuk Mewujudkan Indonesia Jaya"
Nomor : 003/LP/PM/401/2014


diberikan kepada:
Adi Kusjani, S.T

sebagai:
Pemakalah

diselenggarakan oleh:

Sekolah Tinggi Keguruan dan Ilmu Pendidikan SURYA
Pada Tanggal 15 Februari 2014

Ketua STKIP Surya


Eddy Yusuf, Ph. D

