# An Analysis Of Modularity In Aspect Oriented Design

(paper submitted for AOSD'04, under review)

Cristina Videria Lopes and Sushil Krishna Bajracharya (Authors)

ICS 229 Presentation, Fall 2004

#### Sushil K Bajracharya

sbajrach@ics.uci.edu

Donald Bren School of Information and Computer Sciences
Department of Informatics
University of California, Irvine

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

1

## Background

- 2 A fundamental theory of value based design
  - BC00] C. Y. Baldwin and K. B. Clark. Design Rules vol I, The Power of Modularity. MIT Press, 2000.
- 7 Thesis: Design is a value seeking process
  - "Modularization of a system can generate tremendous amounts of value in an industry, given that this strategy creates valuable options for module improvement." [Mac+]
- Pasic components of the theory
  - Six modular Operators for expressing design evolution
  - Modeling designs using DSMs (Dependency Structure Matrices) and hierarchy diagrams
  - Net Options Value as a mathematical model for valuing designs

[Mac+] Alan MacCormack, John Rusnak, and Carliss Baldwin. Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code.

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

#### Background (2)

- Applications of Baldwin and Clark's theory
  - System Design (IBM 360 family, UNIX) [BC00]
  - KWIC [Sul+01]
  - Aspects [our paper]

#### Terminology

- Design: abstract description of functionality and structure of an artifact.
- Medium for expressing design: A designer expresses the basic structure and configuration of design elements with a medium (s)he chooses to work with.
- Design parameters: attributes of the artifact that govern the variation in design. Choosing new values for parameters give new design options.
- Hierarchies: formed by partial ordering of parameter dependencies (for e.g. 'uses' relationships).
  - [Par02] D. L. Parnas. On a "Buzzword": Hierarchical structure. In Software pioneers: contributions to software engineering, Springer-Verlag New York, Inc., 2002.
- Abstraction: hides the complexity of the element.
- Information hiding: design for change.
  - [Par72] D. L. Parnas. On the criteria to be used in decomposing systems into modules. Commun. ACM, 15(12):1053-1058, 1972.

[Sul+01] K. J. Sullivan, W. G. Griswold, Y. Cai, and B. Hallen. The structure and value of modularity in software design. ACM SIGSOFT 2001

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UC

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

#### Terminology (2)

- Modules: strongly connected structural elements grouped together as a module
  - increases the range of manageable complexity
  - allows concurrent work
  - accommodates uncertainty
- Design rules and Interface:
  - decisions common to modules, that are unlikely to change are factored out as design rules
  - design rules constitute the interfaces to connect modules with each other
- Architecture: provides a framework that allows for both independence of structure and integration of function

Sushil Bajracharya

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

5

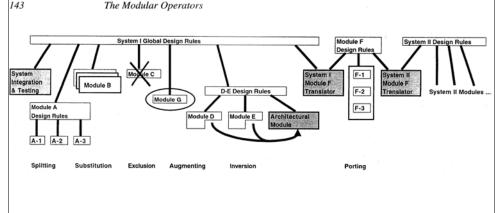
#### Outline

- Outline of the paper
  - Tracing design evolution of a web services application
    - ? Introducing aspects (aspect oriented modularization)
  - Heuristics and assumptions for analysis
  - Evaluating design changes
  - Conclusions

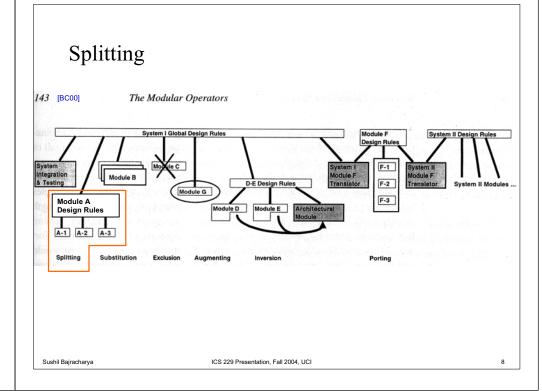
Sushil Bairacharya ICS 229 Presentation, Fall 2004, UC

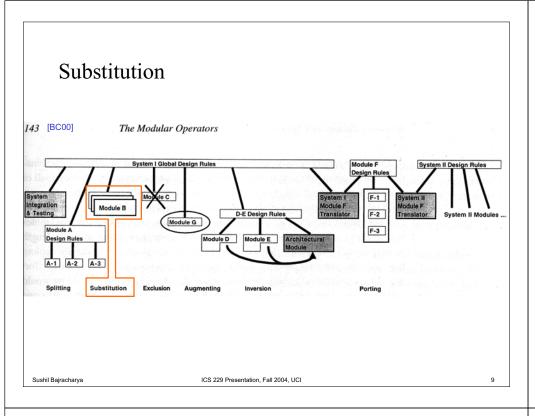
6

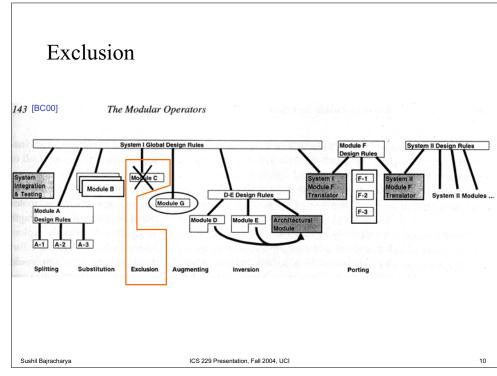
# Modular Operators • Splitting, Substitution, Augmentation, Exclusion, Inversion, and Porting [BC00] The Modular Operators

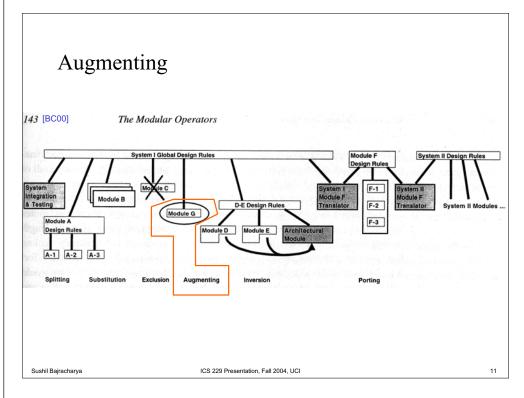


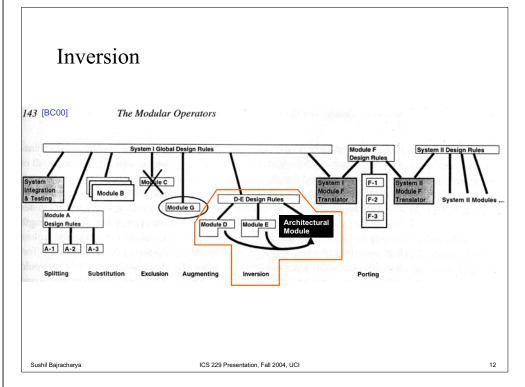
ICS 229 Presentation, Fall 2004, UCI

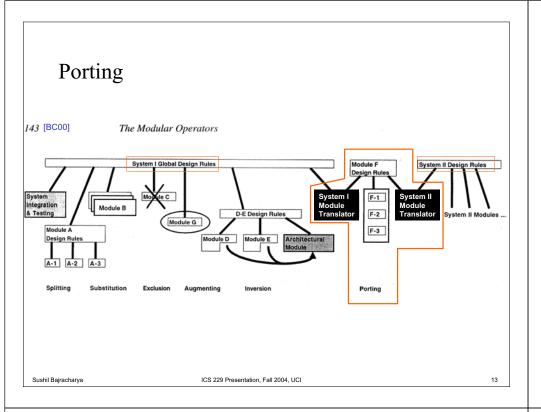






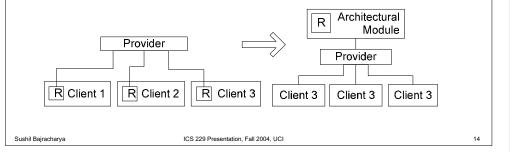






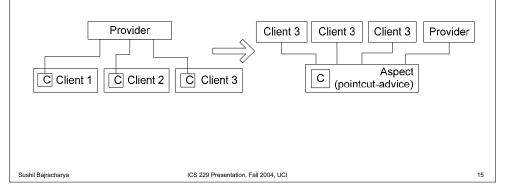
#### Aspect Oriented Modularization

- Aspect Oriented Modularization as a variant of Inversion
- · The effect of Inversion
  - Factors out redundant parameters
  - Adds architectural modules
  - Changes the hierarchy



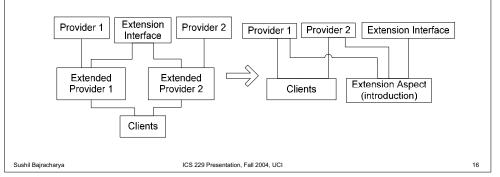
#### Aspect Oriented Modularization (2)

- · Effect of aspects with pointcut-advice mechanism
  - Removes scattered code
  - Adds 'Aspects'
  - Changes the hierarchy



#### Aspect Oriented Modularization (3)

- Effect of aspects with 'introductions'
  - Modifies the hierarchy
  - Factors out common parameters
  - Is this inversion?



#### Aspect Oriented Modularization (4)

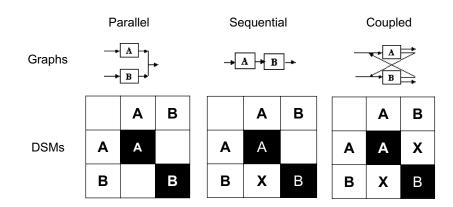
- · What is similar to Inversion?
  - Captures common parameters and moves them to a single module
  - Changes the levels of the parameters in the hierarchy
- What is different from Inversion?
  - Introduces aspects that depend on existing module
  - Whereas, inversion introduces modules on which existing modules are dependent

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

17

#### DSMs – Dependency Matrix of Design Parameters



Three configurations of design parameters that characterize a system [DSMWEB] www.dsmweb.org

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

#### Partitioning a DSM

- -Transforming the DSM into a nearly lower triangular form.
- -Goal: define a task order, reduce feedback information flows

	Α	В	С	D	Ε	F	G
Α	Α	<b>#</b>	-X	X			
В		В					Х
С	X	X	С			Х	X
D		X		D			
Ε			X		Е	X	
F						F	
G				Х		Х	G

	F	В	D	G	С	Α	E
F	F						
В		В		Х			
D		Х	D				
G	Х		Х	G			
С	Х	Х		Х	С	Х	
Α			Х		Х	Α	
Е	Х				X		Е



[DSMWEB]

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

#### Clustering in DSM

		Α	В	С	D	Ε	F	G	Н	Т	J	K	L	N	М	0	Р
Radiator	Α	Α	Х														
Engine fan	В	Х	В			Г	П		П						П		Г
Heater Core	С			С		Г											Х
Heater Hoses	D				D												
Condenser	Ε		Х			Е	Х		Х								Г
Compressor	F					Х	F		Х	Х							
Evaporator Case	G							G									Х
Evaporator Core	н		Г	П		Х	х		н	х		П			Г	П	х
Accumulator	Т	П	г	Г	П	г	Х	П	Х		г	П		г	Г	П	г
Refrigeration Controls	J										J						
Air Controls	ĸ											ĸ					Г
Sensors	L												L				
Command Distribution	М	П	Г	П	П	Г	П	П	П		П			М	П		Г
Actuators	N														N		
Blower Controls	0	П	Г	П	П	Г	Г	П	П	П	П	П			Г	0	х
Blower Motor	Р	П	г	х	П	г	Г	Х	Х		П	П		П	П	Х	P

Goal: finding subsets of DSM elements (i.e. clusters or modules) that are mutually exclusive or minimally interacting subsets

[DSMWEB]



	П	D	J	K	L	М	N	Α	В	E	F	1	Н	C	P	0	G
tadiator	D	D													36		
ingine fan	J		J											_	9		
leater Core	K			K											Q.	l.	
leater Hoses	L	Г			Ц		Г										Г
Condenser	M	Г	П	Г	П	М	Г										Г
Compressor	N						N								7815		
vaporator Case	A			Г			Г	A	X							4715	
vaporator Core	В							X	В	X					96		
ccumulator	E		1						X	E	X		X	(i)	(8)		
tefrigeration Controls	F									X	F	Х	X	9	9	(3)	
ir Controls	1		П								Х	1	Х			10	
ensors	Н									X	X	X	H		X		
Command Distribution	С		П	Г			Г							C	X		
ctuators	P			Г			Г						X	X	P	X	X
lower Controls	0														X	0	
lower Motor	G											×-			X		G

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

#### Categories of Design Parameters

- Simple clustering of design parameters into following categories
  - External Parameters
    - 2 Libraries, Frameworks, External Services.
      - External Parameter is a particular category of Environment Parameter introduced in [Sul+01]
  - Design Rules
    - Parameters that are less likely to be changed.
    - Serve as interface between modules
  - Application Modules
    - Perform application specific tasks
  - Subsidiary Modules
    - <sup>7</sup> Contribute to subsidiary functionalities (non-functional requirements)
  - Application Controller
    - modules that use the design rules as interface to access the functionalities provided by application modules

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

21

#### **Example Application**

- Tracing design evolution of an example application WineryLocator starting from an existing application StoreLocator.
- StoreLocator locates coffee stores in USA
- WineryLocator locates wineries in California
- Design Changes:
  - StoreLocator
  - WineryLocator, First version (Splitting and Substitution on StoreLocator)
  - WineryLocator with logging feature (Augmentation)
  - WineryLocator, Hiding design rules from external parameters (Refining Inversion)
  - WineryLocator with Aspect-oriented modularization

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

22

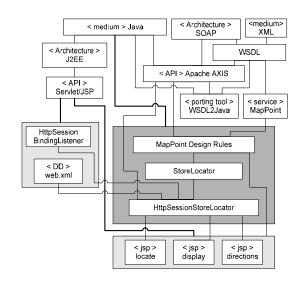
24

#### Requirements

- StoreLocator
  - Functional
    - Finding accurate locations (starting point)
    - · Getting List of coffee store
    - Generating navigable maps and driving directions
  - Non-Functional
    - Providing authentication mechanism for accessing MapPoint webservices

- WineryLocator
  - Functional
    - Finding accurate locations (starting point)
    - Getting list of wineries according to users preference
    - Generating wineries tour of all wineries
    - Generating directions and maps for the tour
  - Non-Functional
    - Authentication
    - Logging access to WebServices

#### StoreLocator – Hierarchy diagram



Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

#### StoreLocator - DSM

			1	2	3	4	5	6	7	8	9	10	11
$\Box$	< service > MapPoint	1	*										
EP >	< API > Apache AXIS	2		*									
П	< API > Servlet	3			*								
Ľ	HttpSessionBindingListener	4			Χ	*							
<dr></dr>	MapPoint Design Rules	5	Х	Х			*						
<am></am>	StoreLocator	6	Γ				X	*					
٧	HttpSessionStoreLocator	7	L.,	Х		Χ	<u>.</u>	Х	*				Х
^	< jsp > locate	8	Γ.		X		X		Χ	*	Х		
ΑC	< jsp > display	9			Χ		X		Χ	Х	*	Х	
٧	< jsp > directions	10			Χ		ίX		Χ	Х		*	
	< DD > web.xml	11	Х	Χ	Χ		i						*

Sushil Bairacharva

ICS 229 Presentation, Fall 2004, UCI

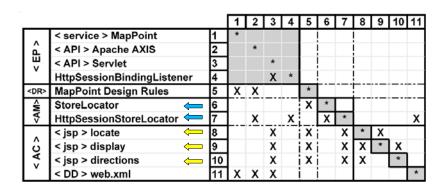
#### StoreLocator - DSM

			1	2	3	4	5	6	7	8	9	10	11
Γ.	< service > MapPoint	1	*										
EP >	< API > Apache AXIS	2		*									
	< API > Servlet	3			*					!			
•	HttpSessionBindingListener	4			Χ	*				<u>.                                    </u>			
<dr></dr>	MapPoint Design Rules	5	Х	Х			*			Ε			
<am></am>	StoreLocator	6	Γ				X	*		Γ			
Υ>	HttpSessionStoreLocator <	7	L.,	Х		Χ	<u>.</u>	Х	*				Χ
	< jsp > locate	8	Γ		X		X		Χ	*	Х		
AC >	< jsp > display	9			Χ		ΙX		Χ	Х	*	Х	
۷ ×	< jsp > directions	10			Χ		iΧ	i	X	Х		*	
	< DD > web.xml	11	Х	Χ	Χ								*

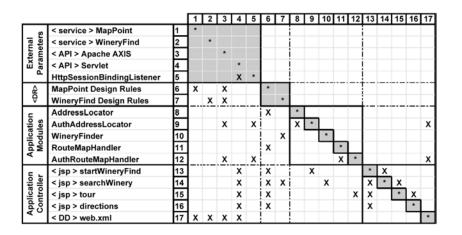
Sushil Bairacharva

ICS 229 Presentation, Fall 2004, UCI

#### StoreLocator - DSM



#### WineryLocator, first version



Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

#### WineryLocator, first version

			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
υ	< service > MapPoint	1	*																
nal	< service > WineryFind <	2		*						i									
External	< API > Apache AXIS	3			*														
External Parameters	< API > Servlet	4				*				!									
	HttpSessionBindingListener	5				Х	*			<u>.</u>									
<dr></dr>	MapPoint Design Rules	6	Х		Х			*		F									
ô	WineryFind Design Rules	7		Х	Х				*										
_	AddressLocator	8	l					X		*									
es li	AuthAddressLocator	9			Х		Х			Х	*								х
Application Modules	WineryFinder -	10							Х			*							
do₩	RouteMapHandler	11						Х					*						
٩	AuthRouteMapHandler	12	l		Х		Х	L					Х	*					Х
= _	< jsp > startWineryFind	13				Х		Х			Х				*	Х			
iệ ệi	< jsp > searchWinery	14				Χ		X	Х	i		Х			Х	*	Х		
lica	< jsp > tour	15				Х		Х						Х	Х		*	Х	
Application Controller	< jsp > directions	16				Х		Х		!					Х			*	
~ ~	< DD > web.xml	17	Х	Х	Х	Х													*

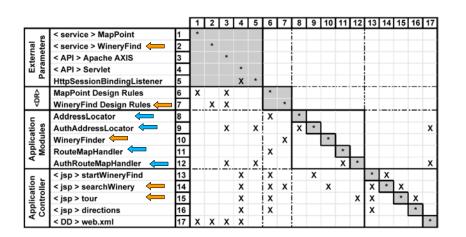
Sushil Bajracharya

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

29

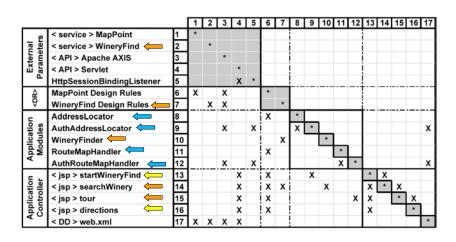
#### WineryLocator, first version



Sushil Bairacharva

ICS 229 Presentation, Fall 2004, UCI

# WineryLocator, first version



ICS 229 Presentation, Fall 2004, UCI

Sushil Bajracharya

31

# WineryLocator with Logging

			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
o o	< service > MapPoint	1	*																	$\Box$
External Parameters	< service > WineryFind	2		*						i	i									
External	< API > Apache AXIS	3			*															
Exa	< API > Serviet	4				*														
	HttpSessionBindingListener	5				Х	*				L									
<dr></dr>	MapPoint Design Rules	6	Х		Х			*												
ô	WineryFind Design Rules	7	L	Х	Х				*		<u>.                                    </u>					L				
<sm></sm>	WebServicesLogger	8								*										
_	AddressLocator	9						Х		х	*									
tio	AuthAddressLocator	10			Х		Х				Х	*								х
Application Modules	WineryFinder	11							Х	X			*							
gŏ	RouteMapHandler	12						Х		x				*						
`	AuthRouteMapHandler	13	L		Х		Х			!				Х	*					х
٦.	< jsp > startWineryFind	14				Х		Х		-		Х				*	Х			
를	< jsp > searchWinery	15				Х		Х	Х				Х			Х	*	Х		
ica tr	< jsp > tour	16				Х		X		i					Х	х		*	Х	
Apllication Controller	< jsp > directions	17				Х		Х								Х			*	
	< DD > web.xml	18	Х	Х	Х	Х														*

ICS 229 Presentation, Fall 2004, UCI

#### WineryLocator with Logging

			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
	< service > MapPoint	1	*																	
External Parameters	< service > WineryFind	2	1	*							i									
External	< API > Apache AXIS	3			*						į									
Exa	< API > Servlet	4				*					!									
	HttpSessionBindingListener	5				Х	*				<u>L</u>					L				
<dr></dr>	MapPoint Design Rules	6	х		Х			*			i									
0	WineryFind Design Rules	7	l	X	Х				*		<u>i </u>					Ĺ				
<sm></sm>	WebServicesLogger —	8	$I_{-}$							٠										
_	AddressLocator	9						Х		х	*									
Application Modules	AuthAddressLocator	10			Х		Х				х	*								Х
lica du	WineryFinder	11							Х	X			*							
dd⊗	RouteMapHandler	12						х		X				*						
٩	AuthRouteMapHandler	13	$\mathbb{L}_{-}$		Х		Х			!				Х	*					Х
۲ .	< jsp > startWineryFind	14				Х		Х				Х				*	Х			
ië ë	< jsp > searchWinery	15				Х		Х	Х		i		Х			Х	*	х		
Apllication Controller	< jsp > tour	16				Х		X		i	i				х	х		*	Х	
P S	< jsp > directions	17				Х		Х			į					Х			*	
	< DD > web.xml	18	х	Х	Х	Х					!									*

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

35

Sushil Bairacharva

ICS 229 Presentation, Fall 2004, UCI

# WineryLocator with Logging

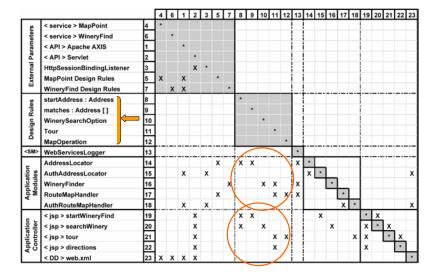
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
S	< service > MapPoint	1	*																	$\Box$
External Parameters	< service > WineryFind	2		*						i	i					i				
External	< API > Apache AXIS	3			*					į						į				
Ex	< API > Servlet	4				*				!	!									
	HttpSessionBindingListener	5				Х	*			<u>.                                    </u>	L.,					L.,				
<dr></dr>	MapPoint Design Rules	6	х		Х			*												
0	WineryFind Design Rules	7	L	Х	Х				*		<u>.                                    </u>					Ĺ.,				l
<sm></sm>	WebServicesLogger —	8								*										
_	AddressLocator	9						Х		x	*									
tio	AuthAddressLocator	10			Х		Х				Х	*								x
pplicatio Modules	WineryFinder	11							Х	X			*							
Application Modules	RouteMapHandler	12						х		į 🗴				*						
4	AuthRouteMapHandler	13	L		Х		Х			!				Х	*					х
c .	< jsp > startWineryFind	14				Х		Х				Х				*	Х			
ti e	< jsp > searchWinery	15				Х		Х	Х				Х			Х	*	Х		
ica tro	< jsp > tour	16				Х		X		i	i				Х	х		*	Х	
Apllication Controller	< jsp > directions	17				Х		Х		į						х			*	
	< DD > web.xml	18	х	Х	Х	Х				!										*

## Winery Locator with new Design Rules

			4	6	1	2	3	5	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22 23
	< service > MapPoint	4	-	ů	÷	-	_	_	-	ů	9	10		12	113	17	10	10	.,,	10	15	20	211	22 23
External Parameters	< service > Maprolit < service > WineryFind	6	1						Н															
E E	< API > Apache AXIS	1	1						Н															
ara	< API > Serviet	2	1						Н															
<u> =</u>	HttpSessionBindingListener	3	1			х																		
E	MapPoint Design Rules	5	x		х	^			Н	Н														
ă		7	<b>∤</b> ^	х																				
$\vdash$	WineryFind Design Rules startAddress : Address	8	Н		_	_	_	_			_	_	_		i	<del> </del>								
Design Rules		-	-							Ĥ					Н	-								
2	matches : Address []	9	-							Н					_	-								
E E	WinerySearchOption	10	1									•			_									
es	Tour	11	┖										*		_	_								
Ľ	MapOperation	12	L	L	L		L.,	L						*		<u>i</u>								
<sm></sm>	WebServicesLogger	13	Ι												*									
_	AddressLocator	14						х		х	Х				х	*								
tio es	AuthAddressLocator	15	1		х		х			į					į	х	٠							X
pplicatio	WineryFinder	16							х			х	х		x			٠						
Application Modules	RouteMapHandler	17	1					х					х	х	x				*					
<u> </u>	AuthRouteMapHandler	18	1		х		х												х	٠				х
_	< jsp > startWineryFind	19	T			х				x	х				[		х				٠	Х	П	
tion	< jsp > searchWinery	20	1			х				х		х						х			х	٠	х	
Application Controller	< jsp > tour	21	1			х							х	х		i				х	x		٠	х
Application Controller	< jsp > directions	22	1			х							х								х		$\neg$	•
_	< DD > web.xml	23	x	х	х	х																		

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

#### Winery Locator with new Design Rules

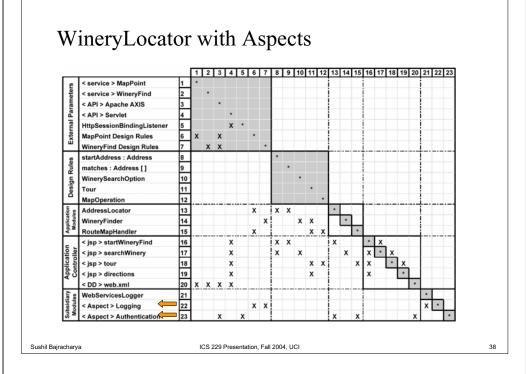


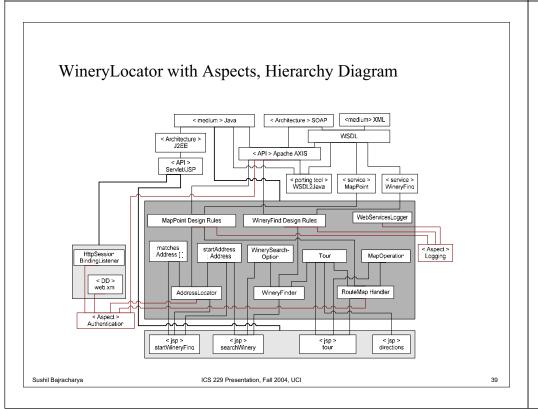
Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

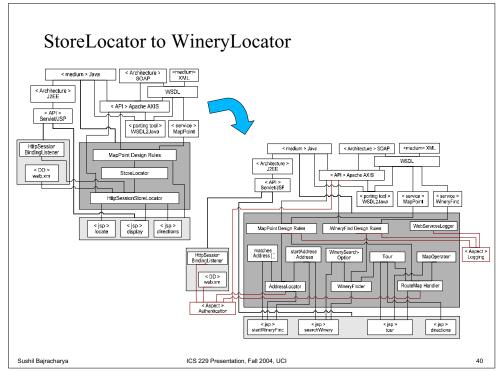
#### Winery Locator with new Design Rules 4 6 1 2 3 5 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 service > MapPoint service > WineryFind API > Apache AXIS < API > Servlet MapPoint Design Rules WineryFind Design Rules matches : Address [] WinerySearchOption Tour MapOperation <SM> WebServicesLogger AddressLocator AuthAddressLocato WinervFinder RouteMapHandler < jsp > startWineryFind < jsp > searchWinery < jsp > directions < DD > web.xml

ICS 229 Presentation, Fall 2004, UCI

Sushil Bairacharva







#### NOV (Net Options Value)

$$V = S_0 + NOV_1 + NOV_2 + ... + NOV_n$$
 (1)

$$NOV_{i} = \max_{ki} \{ \sigma_{i} n_{i}^{1/2} Q(k_{i}) - C_{i}(n_{i}) k_{i} - Z_{i} \}$$
 (2)

$$Z_i = \sum_{j-sees-i} cn_j \tag{3}$$

General expression for NOV of a modular design

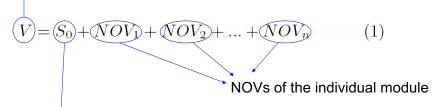
Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UCI

41

#### NOV (2)

Total value of the system

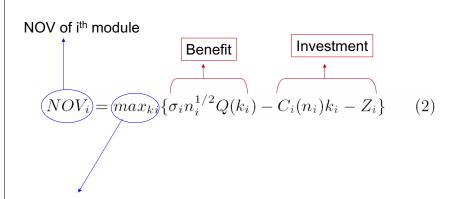


Value of the unmodularized system (usually normalized to 0)

Sushil Bairacharva

ICS 229 Presentation Fall 2004 LICE

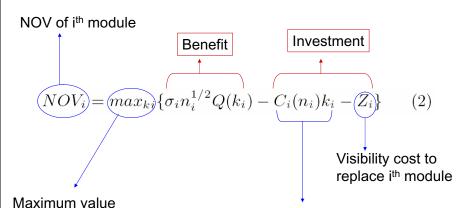
# NOV(3)



Maximum value out of kexperiments for the i<sup>th</sup> module

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

NOV(3)



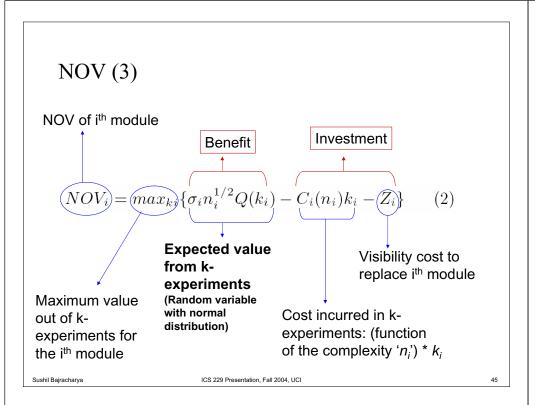
experiments for experiments: (function the i<sup>th</sup> module of the complexity ' $n_i$ ') \*  $k_i$ 

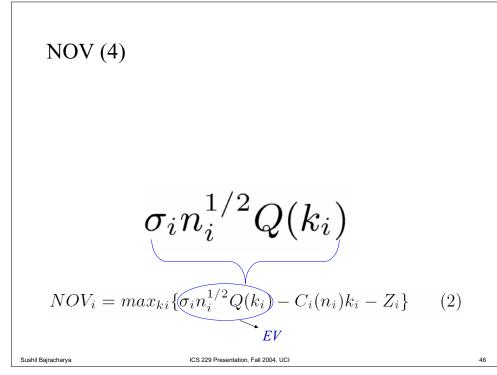
Sushil Bajracharya

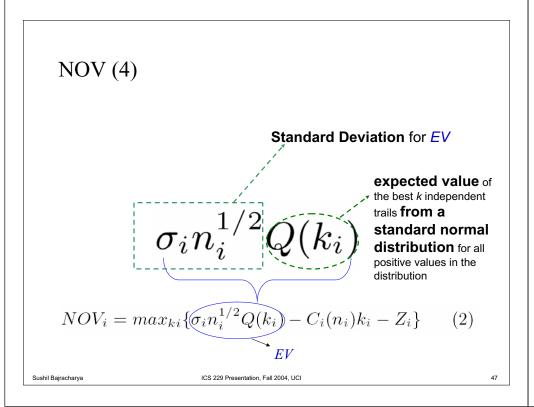
out of k-

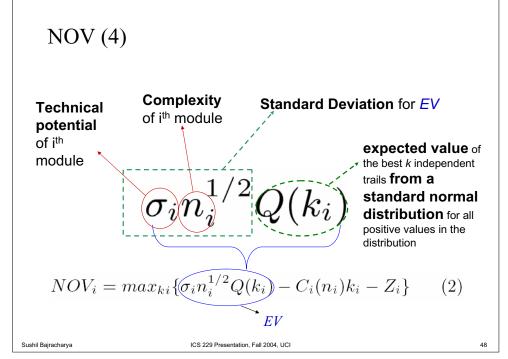
ICS 229 Presentation, Fall 2004, UCI

Cost incurred in k-

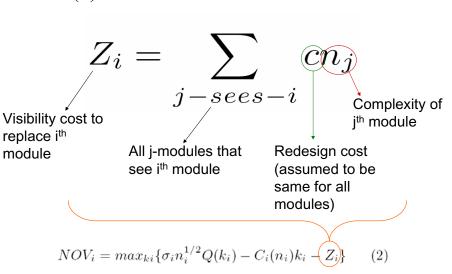








#### NOV (5)



#### Assumptions for NOV analysis

- External parameters are excluded as modules for NOV analysis because they are not subjected to further experimentation.
- We treat all parameters under design rules as a single module.
- All other design parameters are treated as individual modules.
- Redesign cost ' $c_i$ ' of a single module = 1
  - that leads to maximum value of 's' to be 2.5.
    - $rac{1}{2}$  (s  $N^{1/2}Q(1) cN = 0$  and Q(1) = 0.4)
      - Breakeven assumption [BC00]

Sushil Bajracharya

ICS 229 Presentation, Fall 2004, UC

#### Assumptions (2)

Sushil Bairacharya

- 7 Technical Potential of a module =  $s_i$ 
  - $s_i = f(e_i, p_i) = (e_i + 1) X p_i$
- $e_i$  = number of external parameters that the  $i^{th}$  module depends on

ICS 229 Presentation Fall 2004 LIC

- $p_i = i^{th}$  module's relevance to the *end users* 
  - $-p_i = 2$ , for Application Controller modules
  - $-p_i = 1$ , for functional modules, subsidiary modules, design rules
  - $-p_i = 0$ , for web.xml
- $_{?}$  Value of 's' for all modules scaled relatively, using  $s_{\it max}$  = 2.5
- These assumptions are elaborations of observations made for s in [Sul+01]

#### Assumptions (3)

- Module Complexity =  $n_i$
- $_{?}$  N = the complexity of the whole design
  - = the total tasks performed by all the modules in the system.
- ? Complexity of a module
  - = (total number of tasks it performs) / N

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI 51 Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI

#### Observations

· NOV results for different design variants

Design	ID	NOV	$I_c$ %	$I_s$ %	$I_{w1}$ %
StoreLocator	s	0.72	NA	NA	NA
WineryLocator	w1	1.38	91.41	91.42	NA
WineryLocator	w2	1.41	2.18	95.6	2.18
with Logging					
WineryLocator	w3	1.59	12.59	120.2	15.05
with design					
rules for appli-					
cation					
WineryLocator	w4	1.76	24.55	143.6	27.28
with Aspects					

 $I_c$  = Cumulative increase in value

 $\vec{l_s}$  = Net increase in value with respect to 's'

 $I_{w1}$  = Net increase in value with respect to 'w1'

Sushil Bajracharya

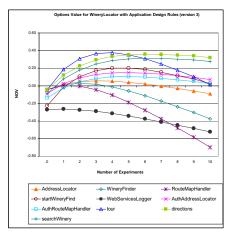
Sushil Bajracharya

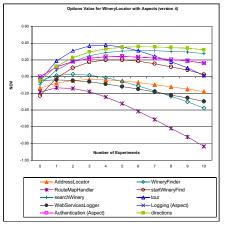
ICS 229 Presentation, Fall 2004, UCI

53

#### Observations (2)

· Effect of Aspect-oriented modularization on option values





Sushil Bajracharya ICS 229 Presentation, Fall 2004, UC

#### Conclusions

- ? Results
  - Aspects can be treated as modules (of design)
  - Aspects add value to an existing design
    - Aspect oriented modularization makes augmentation more profitable even if the added modules have comparatively low technical potential
- 2 Limitations
  - Only two models of aspect oriented mechanism considered
  - General Expression for NOV used instead of individual expressions for operators
  - Distinction between aspect oriented modularization and inversion still is unclear
  - Simple heuristics for determining technical potential of the modules

ICS 229 Presentation, Fall 2004, UCI

- feasibility of modeling and evaluating finer design changes (e.g. various forms of big and small refactorings)
- Applying standard DSM operations on DSMs for software
- Limitations of DSMs

# Finally

- 2 Questions?
- Comments
- ? Thank You!

Sushil Bajracharya ICS 229 Presentation, Fall 2004, UCI