# CONCERN-DRIVEN DEVELOPMENT

#### Gunter Mussbacher / Jörg Kienzle

Software Engineering Laboratory

Department of Electrical and Computer Engineering / School of Computer Science

McGill University

Montreal, Canada

Email: {Gunter.Mussbacher, Joerg.Kienzle}@mcgill.ca

#### THE VISION

Some years from now...
a software engineer is tasked to build a new application...

She broadly identifies domain-specific and generic concerns

She can access a virtual software engineering bookshelf filled with generic reusable concern units

- Complete set of models from requirements to implementation
- Interactive guidelines
  - Variations of the concern + their implications on system goals
  - Composition instructions

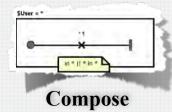
A concern specialist is on call, if help is needed

#### EXAMPLE

A P P Select Desired Features taking tradeoffs into consideration

#### Compose

Feature level: F1 reuses F2, G1 impacts G2



#### Workflow level:

• When an input reaches the system

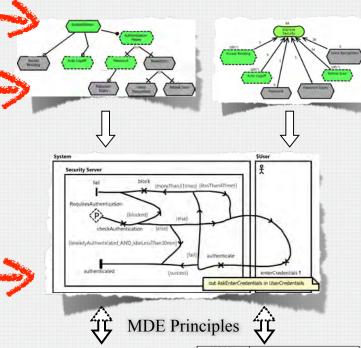
aspect bCMSDomainWithAuthentication extends bCMSDomain depends on UserCredentials					
structural view					
Instantiations: UserCredentials:					

#### Compose

#### Design level:

 Affected objects (e.g., Coordinator, First Aid Personnel, Drivers in a Crisis Management System





			message view Authenticate.ex	new Authenticate execute		
			caller: Caller   target: Auti			
			execute(c)	UserManager WorkflowExecutor		
pect UserCredi	entials extends Input, Output, Workflow depends on Map	[Authenticatable]		rput := getCurrentinput()		
ructural view			1 1 1	input:		
OCIUTAL FIOR	UserCredentialsInput - Storo userrame	lAuthenticatable	1 : :	userCredentialsInput UserCredentialsInput		
CheckAuthentic	cation String password	- String password - boolean blocked				
execute/Loca/Con	- String getUsername()	- boolean authenticated		um: UserManager		
AskEnter Crede		- int triats - Time idle		- years)		
AskEnterCrede	- setPassword(String pass)	- String getPassword()	alt (a fu mult)	perPass := gePassword) a: Authenticatable		
execute(LocalCor	AskEnterCredentialsOutput	- boolean isBlocked()		nerPass (oetPassword)		
Authentical		- setSlocked(boolean b) - boolean isAuthenticated()	1 1 5	esetide()		
	- boolean timeout	- setAuthenticated(boolean aut)	1 1 5	etAuthenticated(hoedPass = userPass)		
- execute/Loca/Con	- setNotAuthenticated() - setTimeout()	- incrementTrials() - int getTrials()		etCurrent(Isseria) c: LocalContext		
Block		- int getTrials() - resetTrials()	1 1 8	et.unectusera)		
execute/Loca/Con		- Time getide()	alt [typedPass = use			
execuse/Local.or	max.	0.1 A current/Jean		setChoosenPath("CheckAuthenticated", "success"		
_				wse(Trials)		
	UserManager	LocalContext	lenel	erChoosesPath/*CheckAutherscated*, *tail*)		
- 1	- UserManager petUserManager()	Locationian		ncrement Triale()		
13	- Muthenticatable getUser(String n) +	Authenticatable getCurrentUsen)   setCurrentUsen(Authenticatable a)	1 11 5	::: getTrials()		
	addUser(Authenticatable a, String n)	securenosen(Authenticatable a)	MR (1 > 20 or	etChoosenPath/*CheckTrials*, *moreThan0Times*		
stantiations:			Fast	etChoosenPuth*CheckTrials*, TessThan4Tmes*		
	stomizableNode CheckAuthentication, AskEnterCredent stomizableInoutData UserCredentialsInput	ists, Block, Authenticate	I 11177 #	BICHOOSEPER(CHICKINGE, NEETINGH INNEY)		
	stomizableOutputData - AskEnterCredentialsOutput					
ip: IDal	ta - UserManager; IKey - String; Nalue - (Authenticatab	le; getValue → getUser;	(else)	elChooserPath("CheckAutherticated", "tail")		
	→ addUser	Implementation:	1 1 5	etChoosenPath("CheckTrials", "lessThan4Times")		
	gleton - UserManager; getInstance - getUserManager	Time: java.util.Date				
essage view Cho	sckAuthentication.execute		message view Block execute			
ratter Catter	target: CheckAuthentication   c: LocalContext	cometaclesson		lock c: LocalContext		
execute)	0	WorkflowExecutor	execute(c)	a perCurrent/berrit		
	a := getCurrentUser() a: Authen	licatable		a: Authenticatable		
: E	t [a la malf] b is inflocted)	Cormeteriasson	: :	-		
1 1		Time	message view AskEnterCreder	tials execute		
11	alt [b = true] setChoosenPath("CheckBlock", "blocked")		Caller: Caller   Target: AskE	nterCredentials   c: LocalContext   cometaclassico		
11	letel setChoosenPath("CheckBlock", "else")		esecuteici	WorkflowExecutor		
	aut := isAuthenticated()		***************************************	a :: getCurrentUser() a: Authenticatable		
	ct := getSystemTime()		I :	output := create() output: AskEnterCredentialsOutput		
	ide :: getide()		·	output: AskEnterCreosentialsCutput		
	alt Flaud AA co - sole < 10001	<b>⊣</b> ⊓ ' ∶	alt (a la realt)	ssut := isAuthenticated)		
	setAutherticated(true)		alt (suth)	se(Timeout)		
	resetide()		I			
	ssetChoosenPath("CheckAuthenticatedAndIdle", "at	eadyAuthorEcated_And_idjeLhasThan30min*)	[stat]	seffictAuthenticated()		
	- leter		L			
	setChoosenPath("CheckAutherScaSonAndidle", "	ise')	[also]	(seffictAuthenticated)		
Li						
- [	[else] setChoosenPath("CheckBlock", "else")		1	i :: getCurrentinput) i: Input		
	setChoosenPath("CheckAuthenScationAndidle", "	ise')	1	s is getSender)		

#### CONCERN CHARACTERISTICS (1)

 Encapsulates composable reusable models of a generic concern (i.e., not productspecific) with well-defined interfaces for all models







- Reaches across all software development phases
  - Each model uses the most appropriate formalism to express the model properties and composition rules relevant at current level of abstraction

### CONCERN CHARACTERISTICS (2)

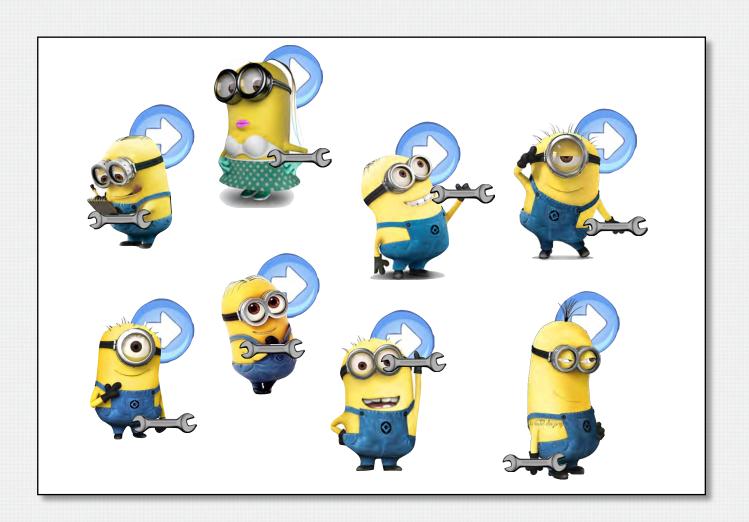
- Impact evaluation of variations
  - Modelled as generally as possible including all relevant variations
  - Guidance on how to choose among variations
  - Known impact on high-level system & stakeholder goals



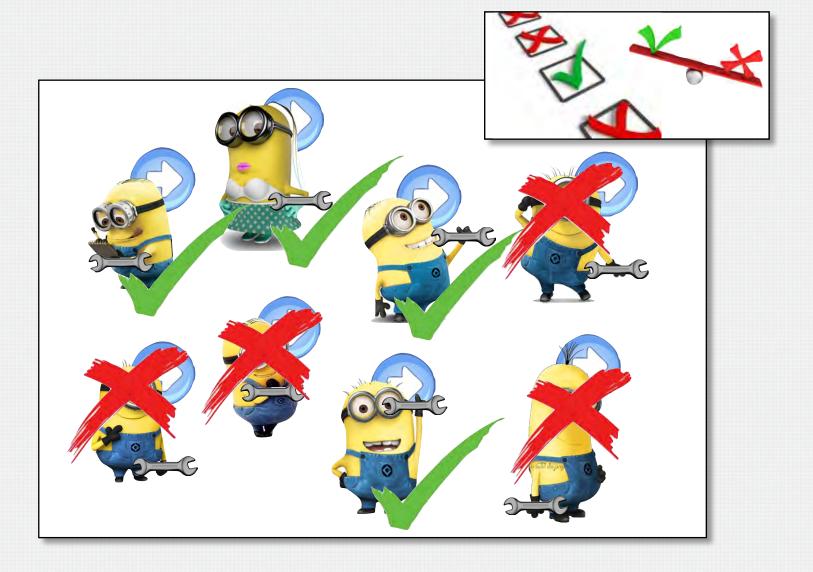


- \* Defines model transformations
  - Link models/compositions from one phase to next
  - Avoid duplication of effort
  - Preserve properties

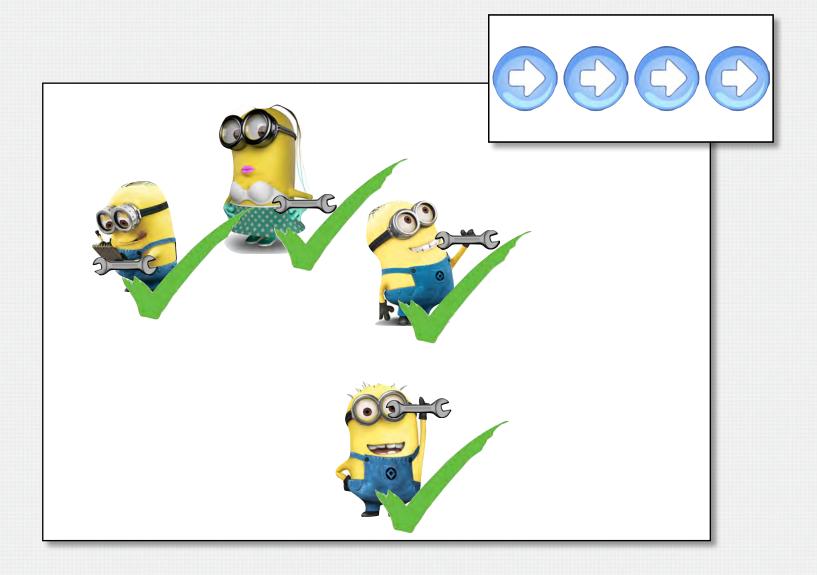
### EXAMPLE CONCERN



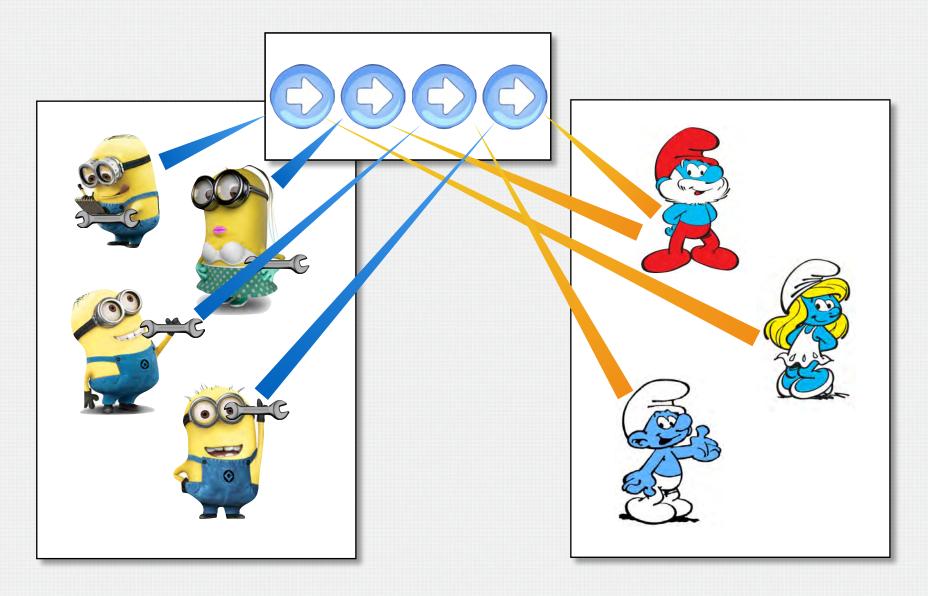
### VARIATION INTERFACE



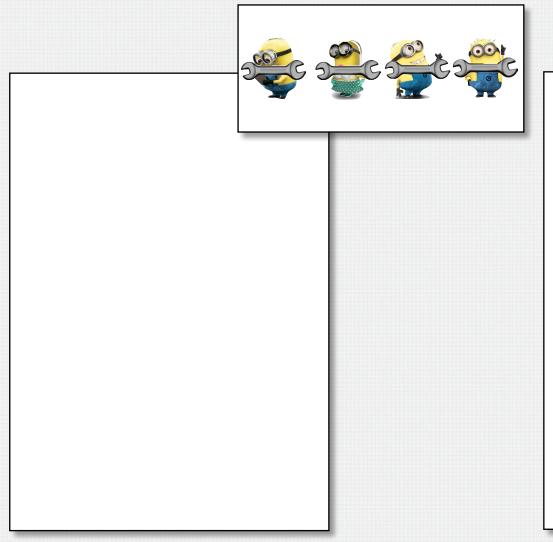
### CUSTOMIZATION INTERFACE

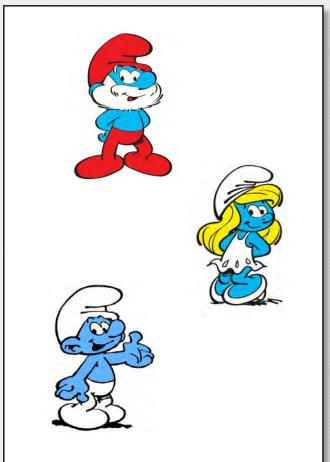


#### CUSTOMIZATION INTERFACE



### USAGE INTERFACE





#### COMPOSED APPLICATION



#### CONCERN REUSE PROCESS

- 1. Use the variation interface of the concern to select the most appropriate feature(s)
  - That provides the desired functionality
  - That maximizes positive impact on relevant non-functional application properties
  - → This generates the generic models for the selected feature(s) of the concern
- 2. Use the customization interface of the generated models to adapt the generic model elements to the application-specific model(s)
  - → This generates the application-specific models for the selected feature(s) of the concern
- 3. Use the selected concern features within the application model(s) according to the usage interface

## THE CORE APPROACH

Application-Specific Concern

Reuse a Concern ⇔
Compose at all Levels

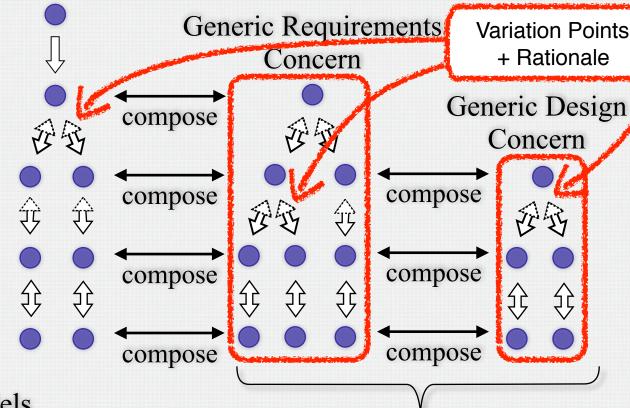
Requirements Elicitation

Specification & Analysis

Architecture Design

Detailed Design

Implementation



Model / Set of Models

Automated Model Transformation + Completion

Automated Model Transformation

Reusable Concern Library "Software Concern Line"

# THE CORE APPROACH

Requirements Elicitation

Specification & Analysis

Architecture Design

Detailed Design

Implementation

Notation (AoURN)

• Goal Models (GRL)

• Worldlow Models (Use Case Many

Aspect-oriented User Requirements

Workflow Models (Use Case Maps)

Reusable Aspect Models (RAM)

- Application-Specific Design Models
- RAM Workflow Concern
- Reusable Design Concern Models

Java Code

All of these Models of a Concern can be reused!

Tool Support is Essential!

