The University of Sydney

# Modelling with Situations


## Eugene Doma

B.E., M.A.C.M., M.I.E.E.E

# Topics

1. Introduction to Modelling
2. What is a 'Situation' ?
3. A bit of theory
4. Application
5. Summing up

# Business Analyst as Architect

# *Building solutions that are:*

- Functional
- Feasible
- Reliable
- Adaptable
- Aesthetic

# Model Based Software Engineering

# A Model is :

- Small fragment of reality
- Properties
- Theory

= Abstraction

# George E. P. Box

"Essentially, all models are wrong,
but some are useful"

1987

Business Analysts

create

Models of the required solution

# Model Based Software Engineering Maturity Model

Bran Selic

IBM Distinguished Engineer, Rational Software
Co-chair OMG Task Force UML 2.0
President Malina Software Corp.



# MBSE Maturity Model

1 Code only

*What's a model ?*

Code

# MBSE Maturity Model

## 2 Code Visualisation

*The code is the model !*

# MBSE Maturity Model

## 3 Round Trip Engineering

*Manage code and model*

# MBSE Maturity Model

4 Model-centric

*The model is the code !*

# MBSE Maturity Model

5 Model only

*Who cares about the code ?*

Model

# Some successful applications

Automated doors, Base Station, Billing (In Telephone Switches), Broadband Access, Gateway, Camera, Car Audio, Convertible roof controller, Control Systems, DSL, Elevators, Embedded Control, GPS, Engine Monitoring, Entertainment, Fault Management, Military Data/Voice Communications, Missile Systems, Executable Architecture (Simulation), DNA Sequencing, Industrial Laser Control, Karaoke, Media Gateway, Modeling Of Software Architectures, Medical Devices, Military And Aerospace, Mobile Phone (GSM/3G), Modem, Automated Concrete Mixing Factory, Private Branch Exchange (PBX), Operations And Maintenance, Optical Switching, Industrial Robot, Phone, Radio Network Controller, Routing, Operational Logic, Security and fire monitoring systems, Surgical Robot, Surveillance Systems, Testing And Instrumentation Equipment, Train Control, Train to Signal box Communications, Voice Over IP, Wafer Processing, Wireless Phone

# What is a Situation?

## *"State of affairs; combination of circumstances"*

- Time
- Place
- Participants
- Relationships
- Processes

# 5 x W + H

- Who?
- When?
- Where?
- What?
- Why?
- How?

# Situation Theory

# CSLI

Stanford University

## Center For The Study Of Language And Information

# *Inter-disciplinary Research*

- Computer science
- Linguistics
- Mathematics
- Psychology
- Philosophy
- Political science
- Engineering

# Situation Theory

- "Unified mathematical theory of meaning and information content"

- "The world is viewed as a collection of objects, sets of objects, properties and relations"

# Cognitive Agents

- Make, measure and organise observations
- Prior knowledge
- Attunement
- Producers of information
  - Being or mechanism

# *Infon*

- Fundamental unit of information

  **<<r, a, b, c; 1>>**

- Relation : r
- Arguments: a, b, c, …
- Polarity: 1=true, 0=false
- Infon Algebra

# Example

```
<<insert_into, Card_9874, ATM_212; 1>>
<<keypad_entry, PIN 1234; 1>>
<<keypad_entry, Amount, $100; 1>>
<<button_press, Withdrawal; 1>>
<<tamper, Alarm; 0>>
```

- Infons 'hold in' situation

  'cash withdrawal request'

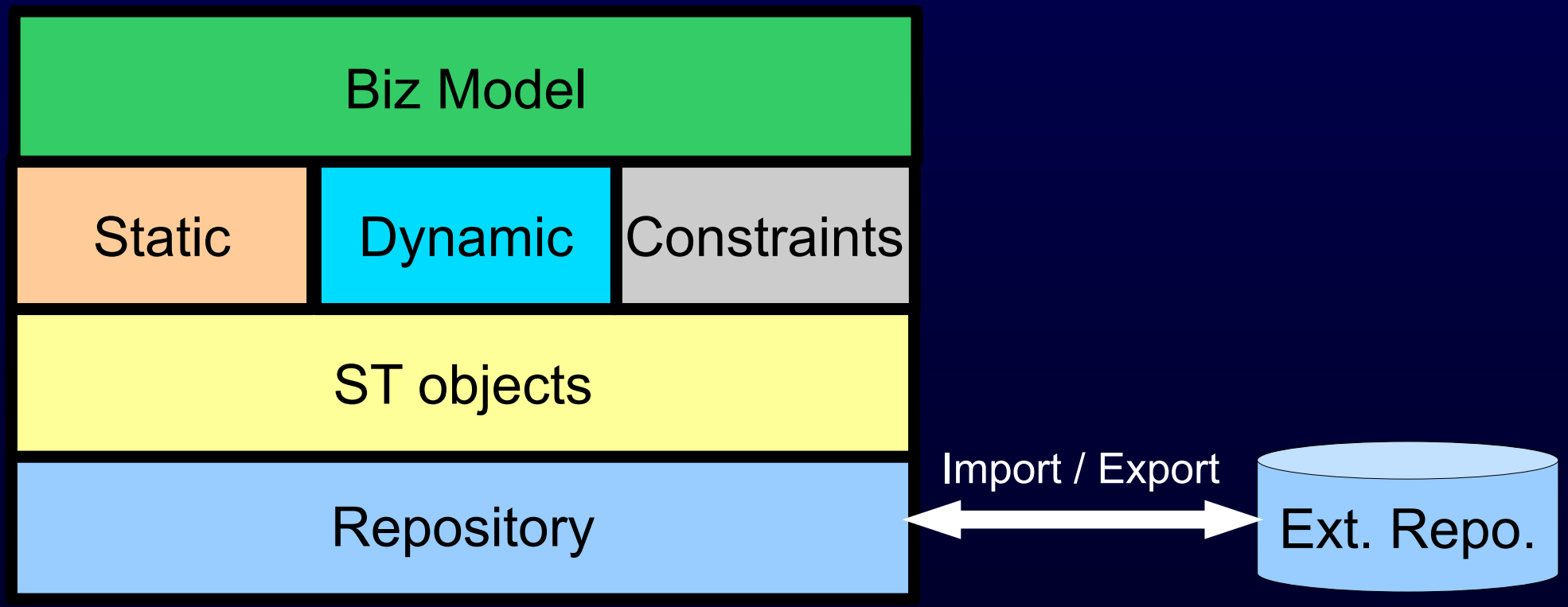
# ST Objects

- Types
- Relations
- Properties
- Constraints
- Situations
- Situation types

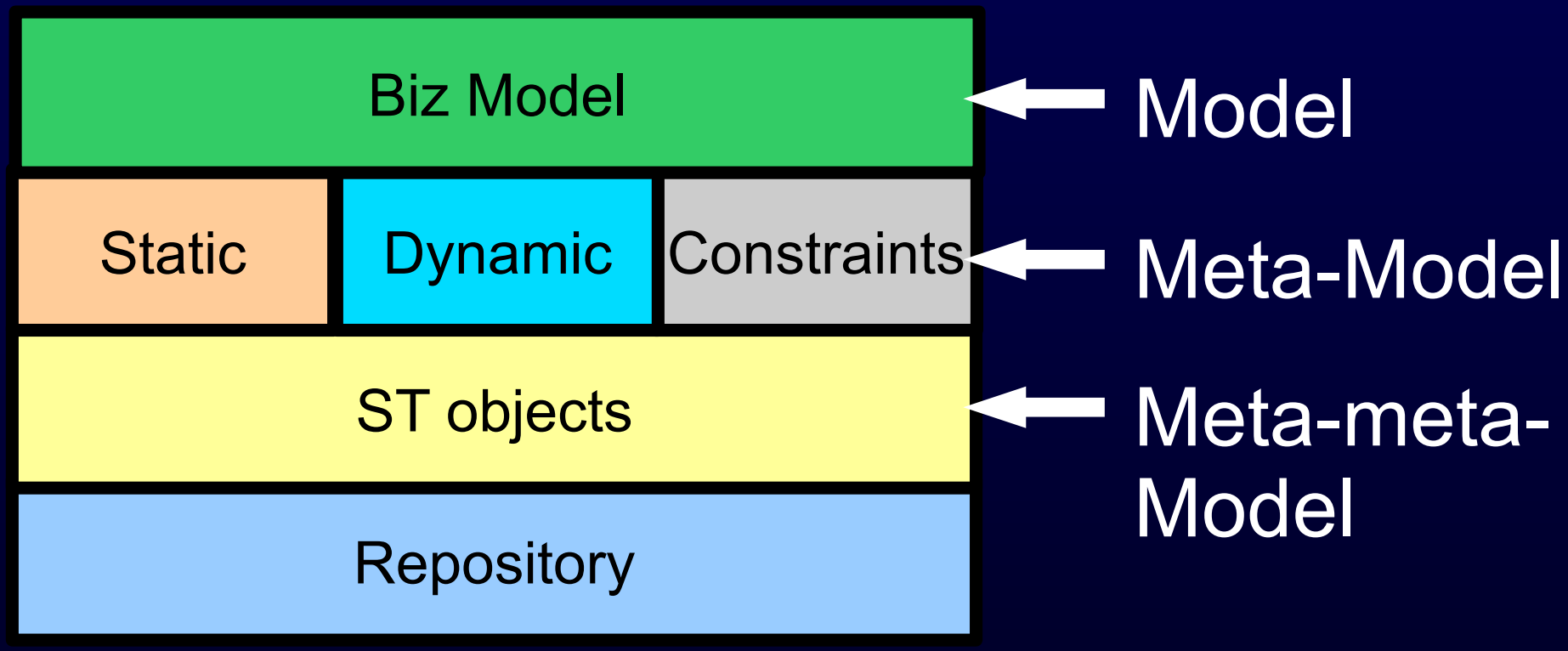
# Tool Support

- Infon fundamental unit of:
  - Storage
  - Manipulation
- Infon algebra
  - like Boolean algebra

# Toolset Architecture



# Toolset Architecture



# Some tools

- ASTL

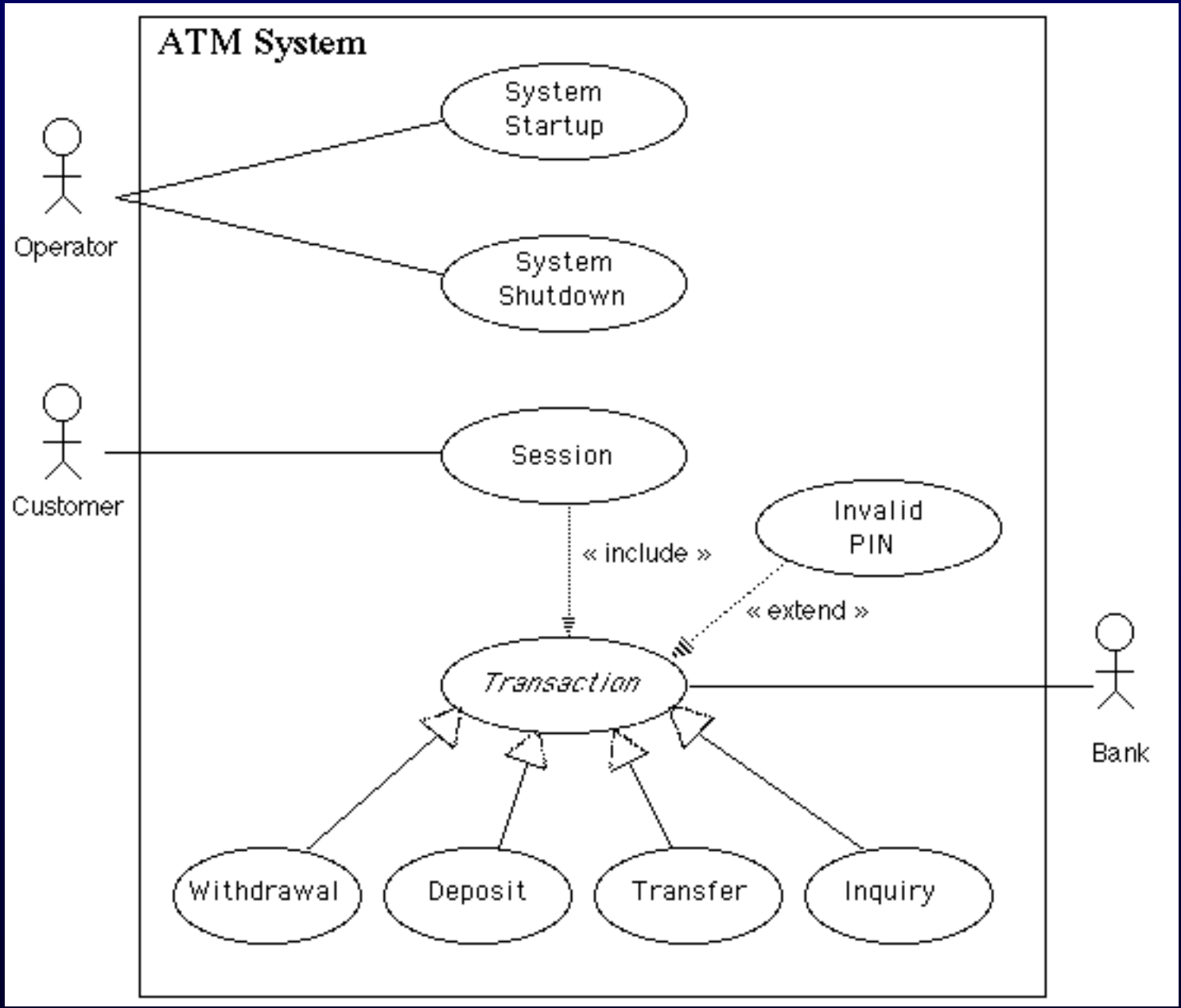
  A Situation Theoretic Language
- PROSIT

  PROgramming in Situation Theory
- BABY - SIT
- IDEF 3

# Computer Assisted Business Services Engineering

- Graphical toolset
- Proof-of-concept for ST in biz.
- Everything is an infon
- Code generation - C#, Java, SQL

# Use Cases




http://www.cs.gordon.edu/courses/cps211/ATMExample/UseCases.html

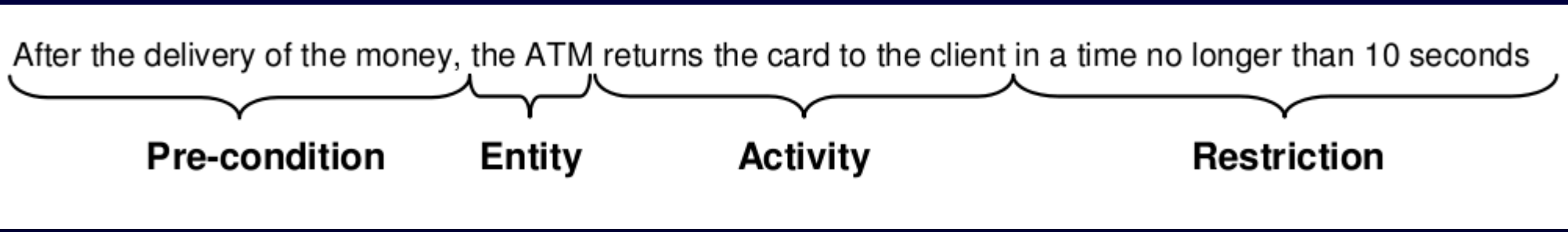
## Withdrawal Transaction Use Case

A withdrawal transaction asks the customer to choose a type of account to withdraw from (e.g. checking) from a menu of possible accounts, and to choose a dollar amount from a menu of possible amounts. The system verifies that it has sufficient money on hand to satisfy the request before sending the transaction to the bank. (If not, the customer is informed and asked to enter a different amount.) If the transaction is approved by the bank, the appropriate amount of cash is dispensed by the machine before it issues a receipt. (The dispensing of cash is also recorded in the ATM's log.)

A withdrawal transaction can be cancelled by the customer pressing the Cancel key any time prior to choosing the dollar amount.

# Sentence → Structure

[pre-condition] <actor> | <entity> <activity> [restriction]



| USE CASE | Withdraw | |
|---|---|---|
| ABSTRACT | If Client is a member of the network, the ATM allows him to withdraw money from any of his accounts. | |
| ACTOR | Client | |
| PRE-CONDITION | ATM ready for operation | |
| DESCRIPTION | *Normal flow the events* | |
| | 1) The Client inserts his card into the ATM. | (! (insert client card atm)) |
| | 2) The ATM checks card validity. | (! (validate atm card)) |
| | 3) The ATM displays a prompt for password. | (! (prompt atm password)) |
| | 4) The Client types his password. | (! (typein client password)) |
| | 5) The ATM checks password validity. | (! (validate atm password)) |
| | 6) The ATM displays a list of options. | (! (display atm options)) |
| | 7) The ATM displays a prompt for option. | (! (prompt atm option)) |
| | 8) The Client chooses the withdraw option. | (! (choose client withdrawOption)) |
| | 9) The ATM displays a list of accounts. | (! (display atm listAccount)) |
| | 10) The ATM displays a prompt for account. | (! (prompt atm account)) |
| | 11) The Client chooses an account from the list. | (! (choose client account)) |
| | 12) The ATM displays a prompt for amount. | (! (prompt atm amount)) |
| | 13) The Client enters the amount. | (! (typein client amount)) |
| | 14) The ATM checks if the account balance is sufficient for debiting the amount. | (! (check atm balance amount)) |
| | 15) The ATM delivers the money to the client. | (! (deliver atm money client)) |
| | 16) The ATM registers the transaction. | (! (register atm transaction)) |
| | 17) The ATM updates the account balance. | (! (update atm balance)) |
| | 18) The ATM prints a receipt. | (! (print atm receipt)) |
| | 19) The ATM ejects the card. | (! (eject atm card) |
| POST-CONDITION | The Client gets his money and his card | |

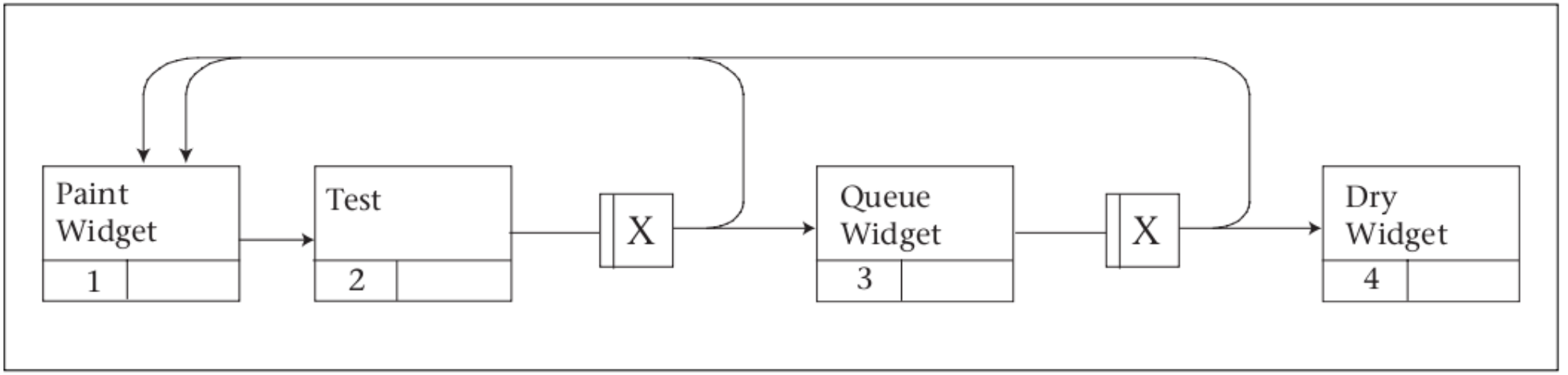
http://www.inf.puc-rio.br/wer02/zip/Writing_use_cases(3).pdf

# Elements

- Title
- Goal
- Context
- Pre-conditions
- Resources
- Actors
- Episodes
- Post-conditions

# Process Modelling

# IDEF 3 - Process diagram




http://www.aaai.org/ojs/index.php/aimagazine/article/viewPDFInterstitial/1719/1617

```
(define-object
  :name widget
  :constraints (Widget widget))

(define-object
  :name painter
  :constraints (Paint_Sprayer painter))

(define-object
  :name oven
  :constraints (Oven oven))

(define-activity-role
  :id Act-1
  :name Paint_Widget
  :successors 2
  :preconditions
    (or (not (Painted widget (beginof ?occ)))
        (not (Adequate (Paint_Coverage widget (beginof ?occ)))))
  :postconditions
    (Painted widget (endof ?occ)))

(define-activity-role
  :id Act-2
  :name Test_Coverage
  :successors 1 3
  :preconditions (Painted widget (beginof ?occ))
  :postconditions (Adequate (Paint_Coverage widget) (endof ?occ)))

(define-activity-role
  :id Act-3
  :name Dry_Widget
  :successors
  :preconditions (Adequate (Paint_Coverage widget) (beginof ?occ))
  :postconditions (Dry widget (endof ?occ))
```

# Elements

- Objects
- Activity roles
- Constraints
- Pre-conditions
- Post-conditions

# Summing Up

# MBSE

- Secret of success in many projects
- Supported by UML 2.0
- BAs – drive SDLC
- Improved communication
- Simulation
- Validation
- Code generation

# Infons

- Fundamental unit for model construction
- Manipulated by toolset
- Object oriented

# Situations

- Scenario description
- Static semantics
- Dynamic semantics
- Explicit constraints

# Questions & Discussion

www.eelab.usyd.edu.au/PEOPLE/Eugene

# Thank you !