DSM TP 2015
Theory and Practice

6th International Summer School
on Domain-Specific Modeling

Antwerp, Belgium
24-28 August

Modelling Languages: (mostly) Concrete (Visual) Syntax

Hans Vangheluwe

Antwerp
25 August 2015

Syntax, **Semantics**, and all that Stuff

David Harel, Bernhard Rumpe.
Meaningful Modeling: What's the Semantics of "Semantics"?
IEEE Computer, vol. 37, no. 10, pp. 64-72, October, 2004.

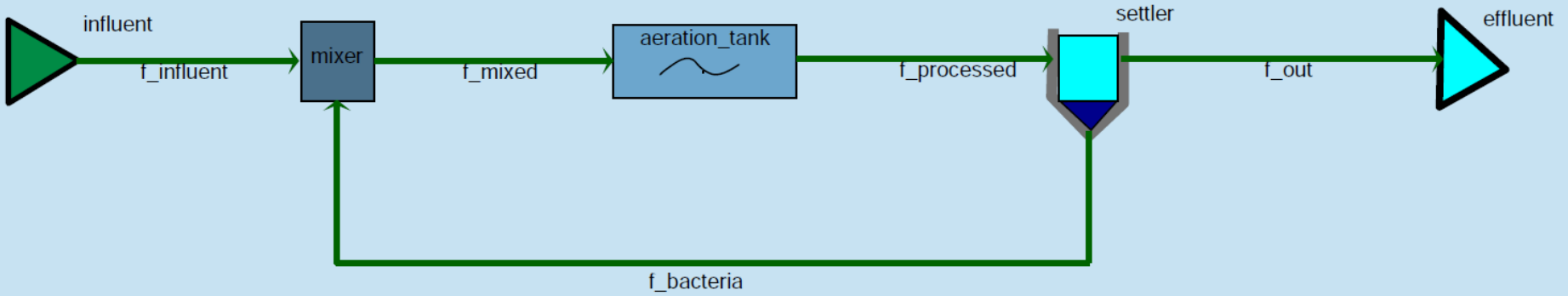
- "operational" semantics
- "denotational" (transformational) semantics

Operational vs. Denotational (Translational) semantics

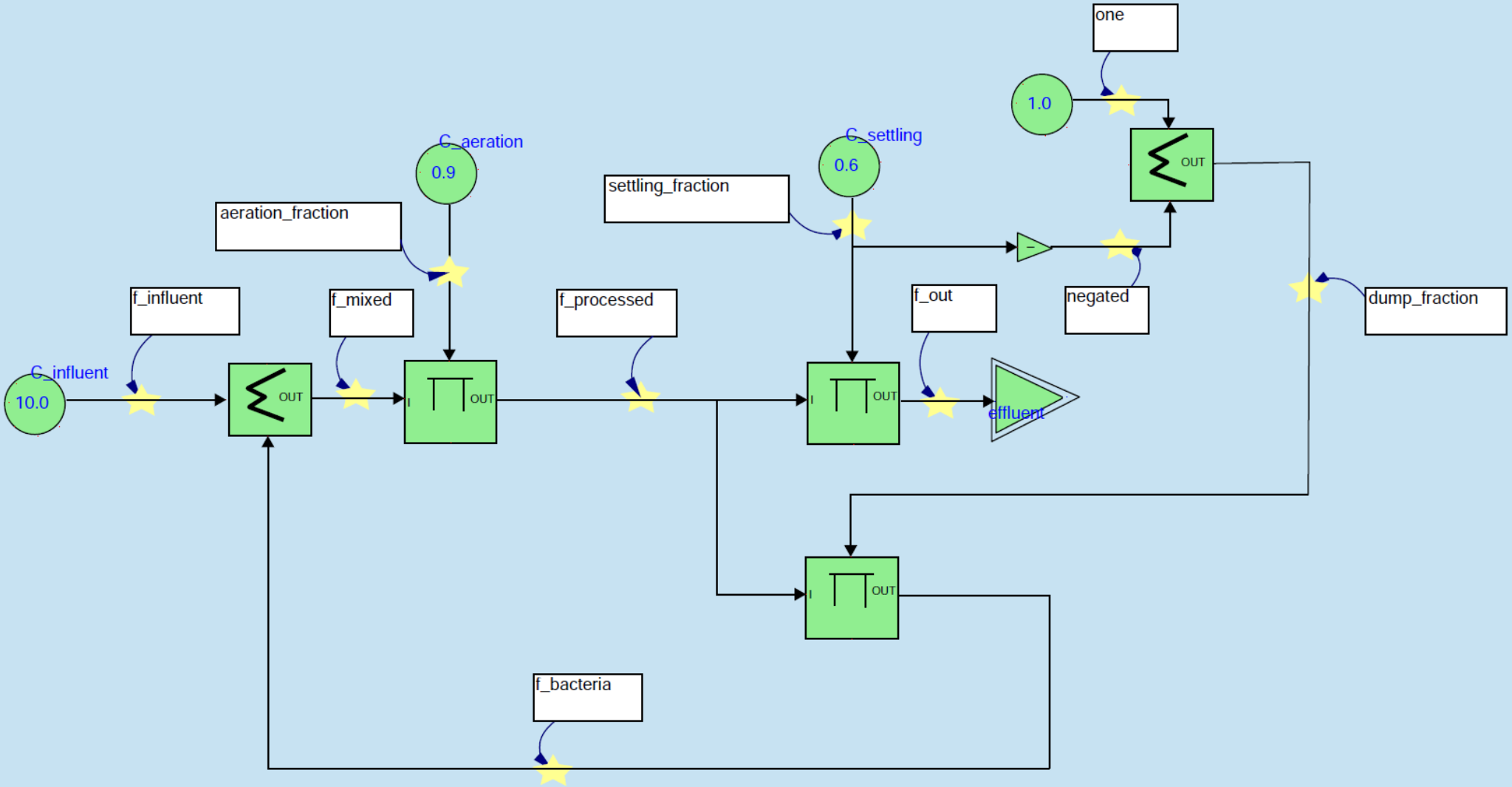
NATO's Sarajevo Waste Water Treatment Plant

www.nato.int/sfor/cimic/env-pro/waterpla.htm

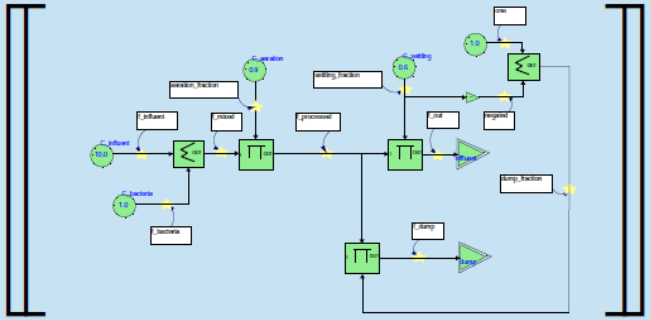
What does this WWTP model mean?



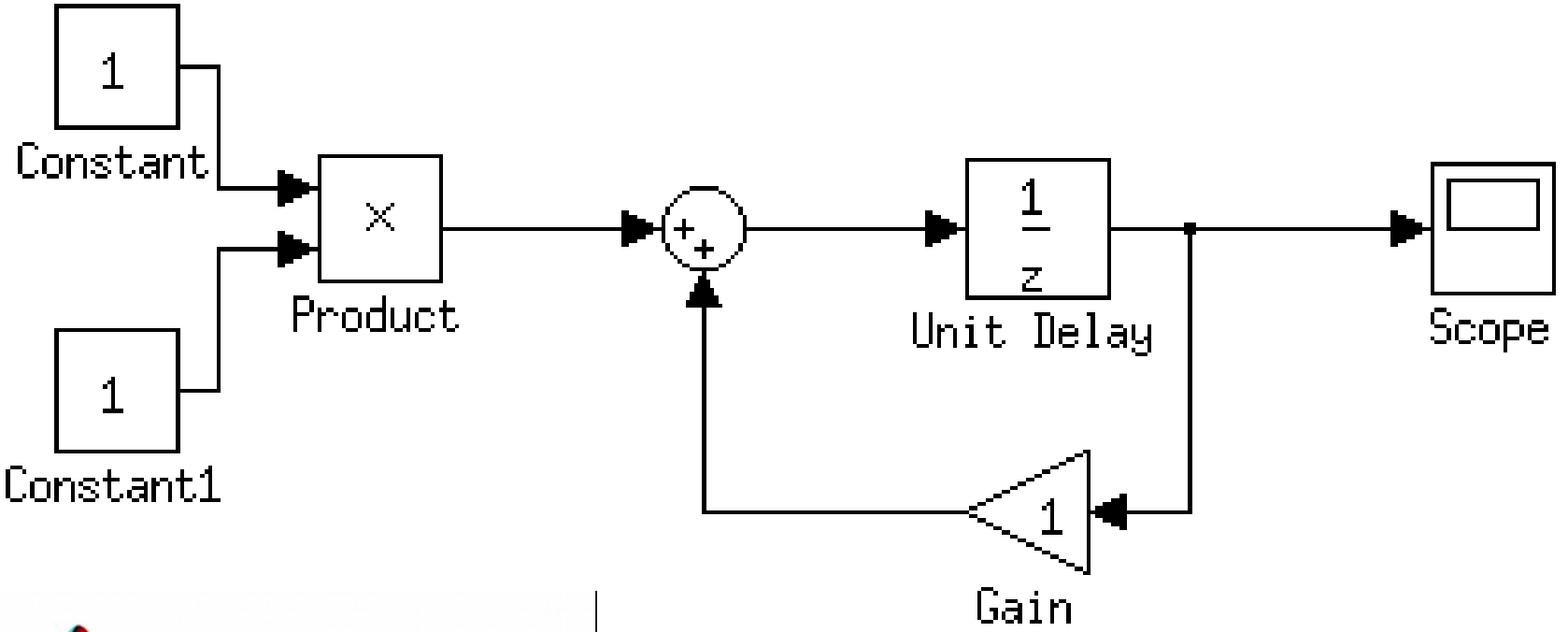
. . . its meaning (steady-state abstraction): Causal Block Diagram (CBD)



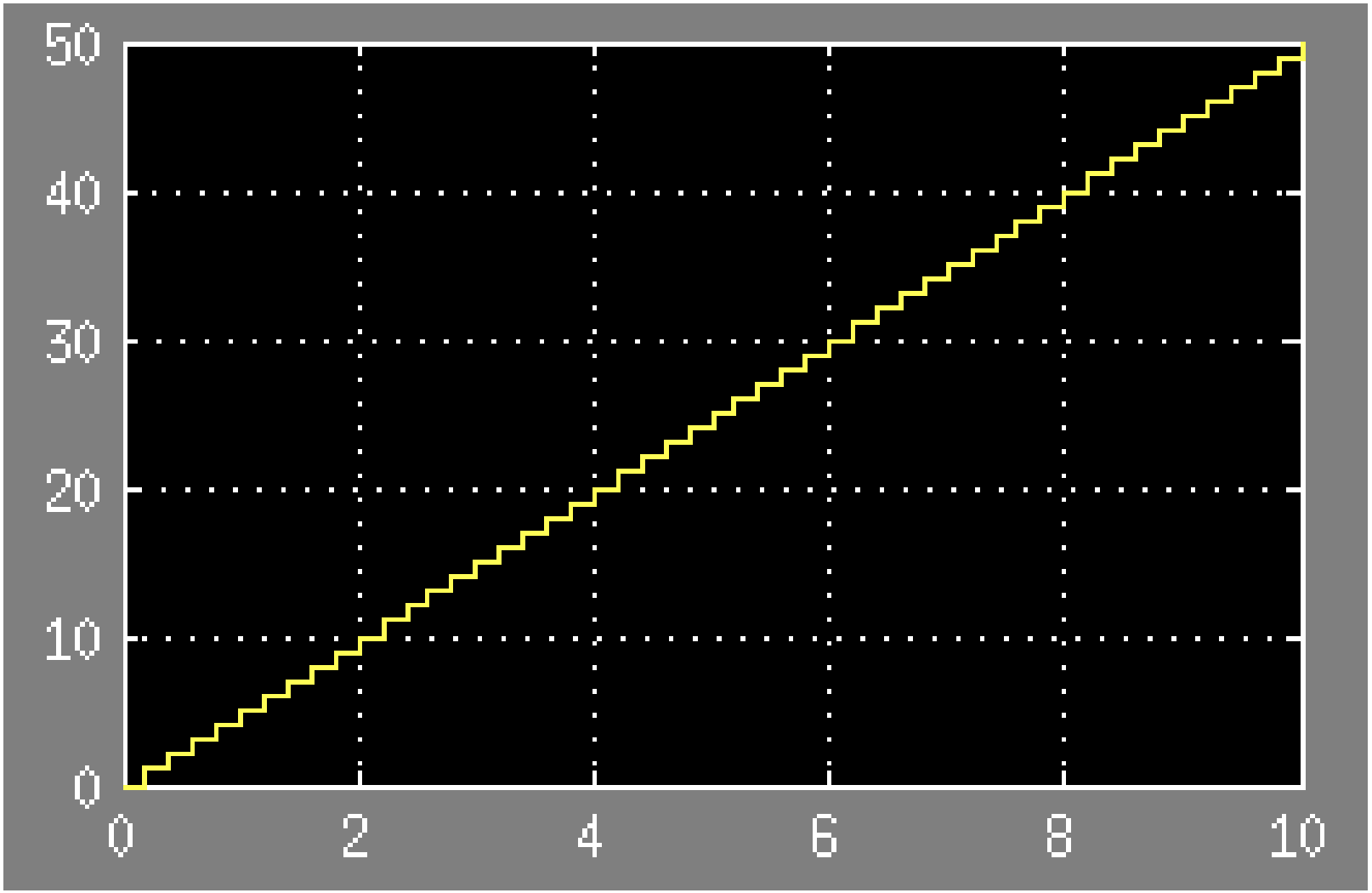
Meaning of the CBD . . . semantic mapping onto algEqns



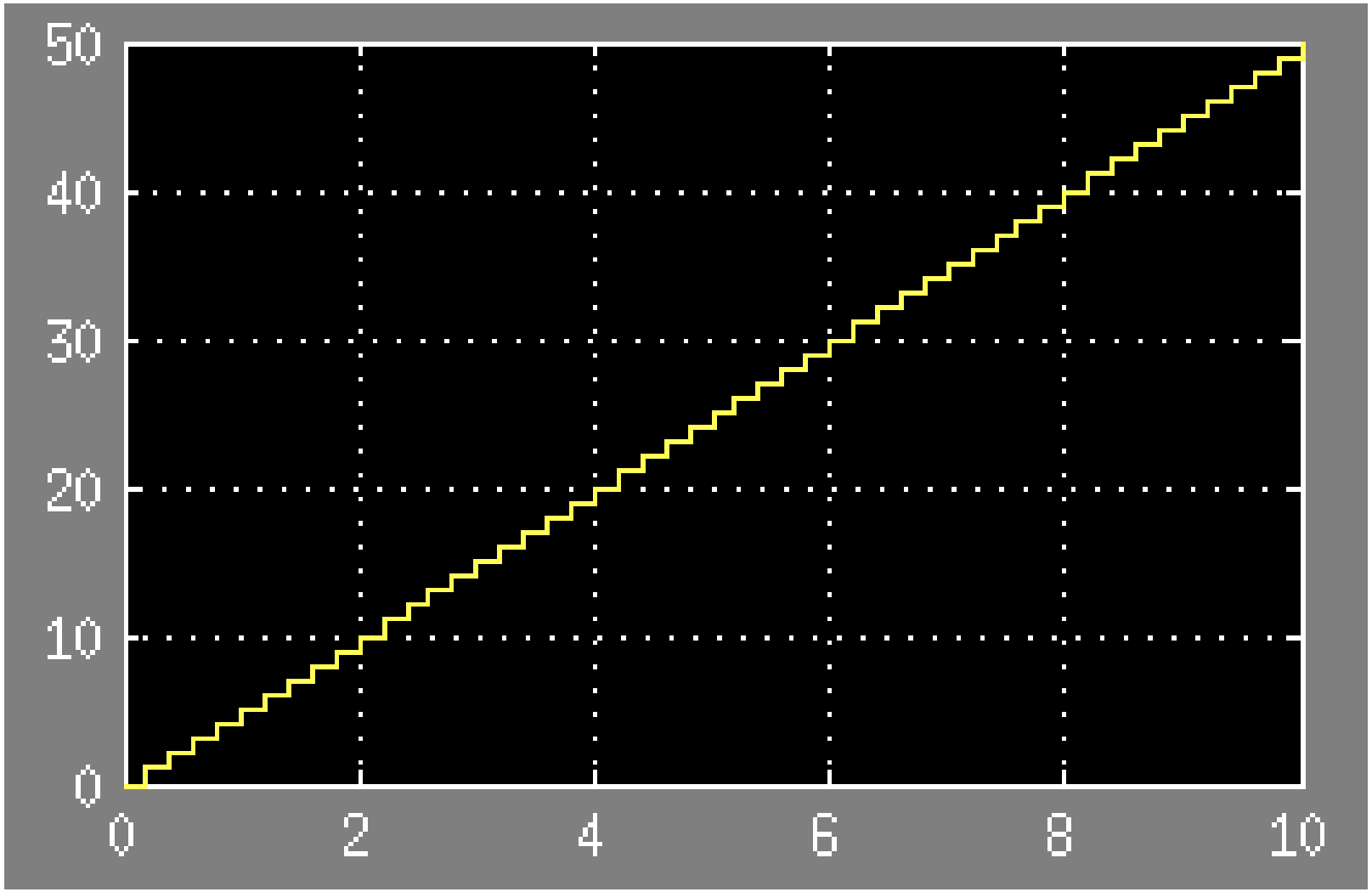
$$
\left\{
\begin{array}{rcl}
f\_influent & = & C\_influent \\
f\_bacteria & = & C\_bacteria \\
f\_mixed & = & f\_influent + f\_bacteria \\
aeration\_fraction & = & C\_aeration \\
f\_processed & = & aeration\_fraction * f\_mixed \\
settling\_fraction & = & C\_settling \\
negated & = & -settling\_fraction \\
one & = & 1 \\
dump\_fraction & = & one + negated \\
f\_dump & = & f\_processed * dump\_fraction \\
f\_out & = & settling\_fraction * f\_processed
\end{array}
\right.
$$

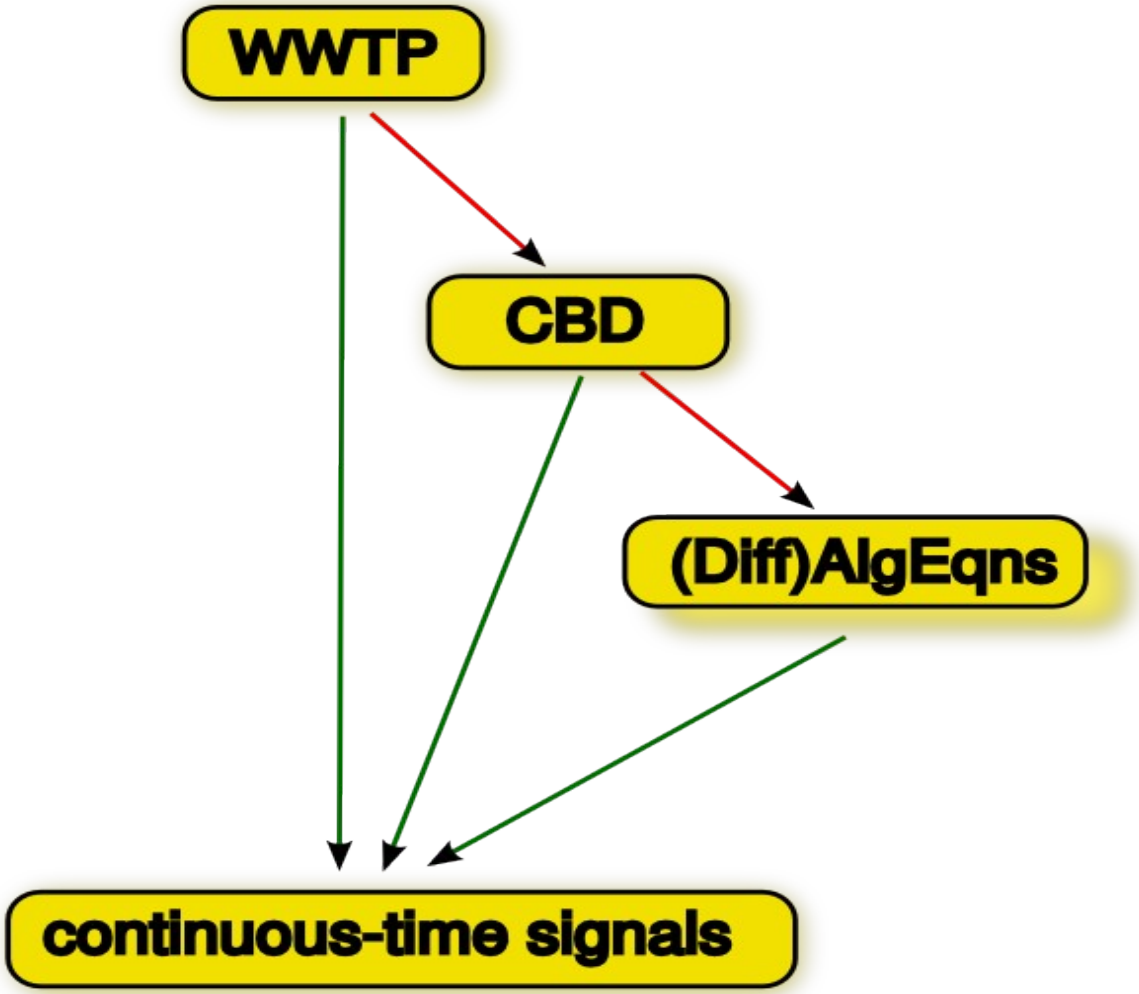


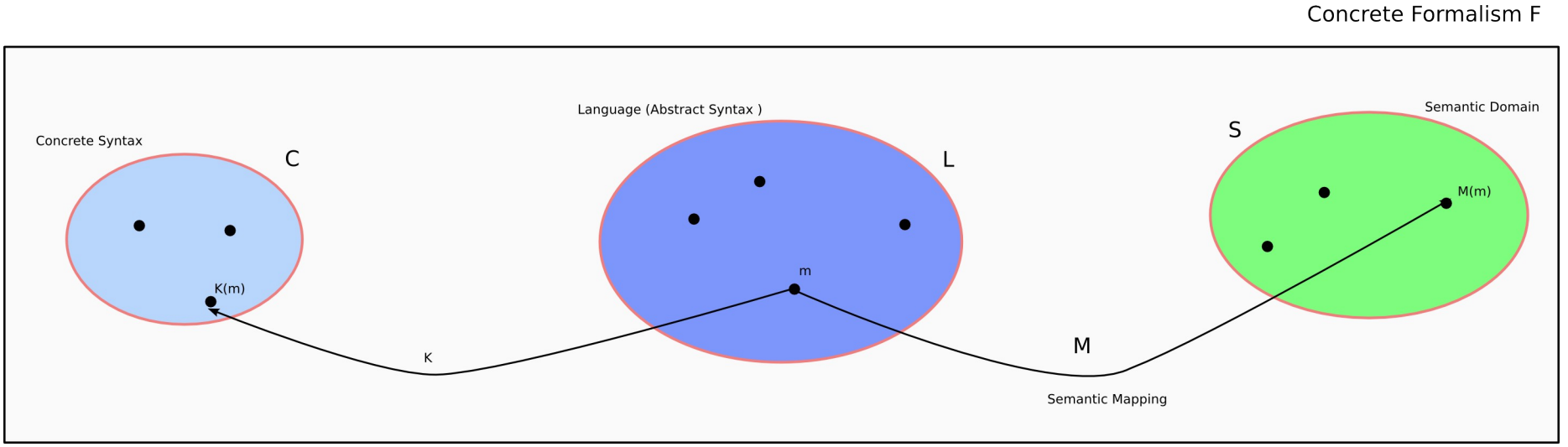
MATLAB®
SIMULINK®


50
40
30
20
10
0
0
2
4
6
8
10

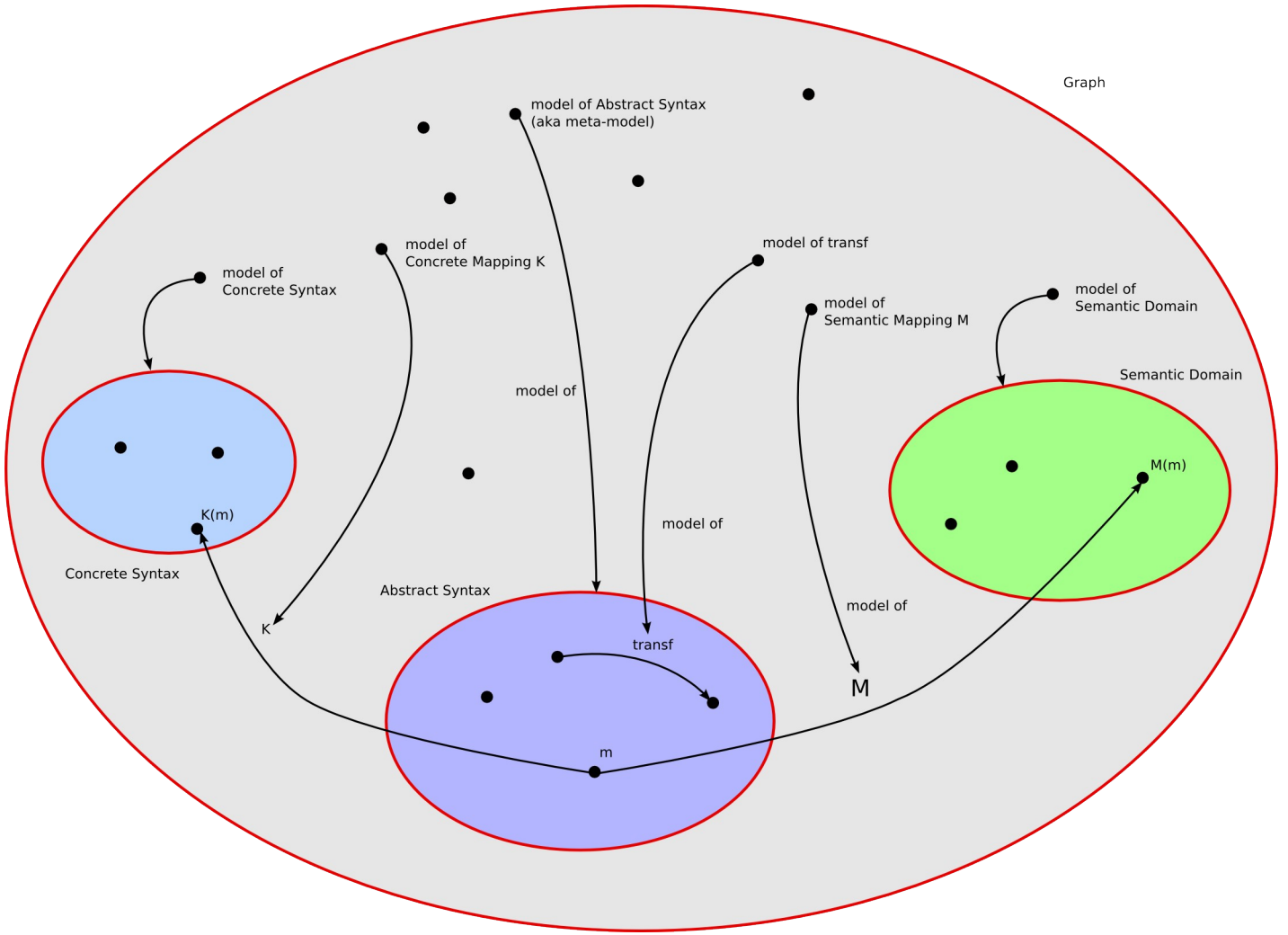
```
1: time_step ← 0
2: while not end_condition do
3:     schedule ← LOOPDETECT(DEPGRAPH(cbd))
4:     for gblock in schedule do
5:         COMPUTE(gblock)
6:     end for
7:     time_step ← time_step + 1
8: end while
```


50
40
30
20
10
0
0
2
4
6
8
10


WWTP
CBD
(Diff)AlgEqns
continuous-time signals

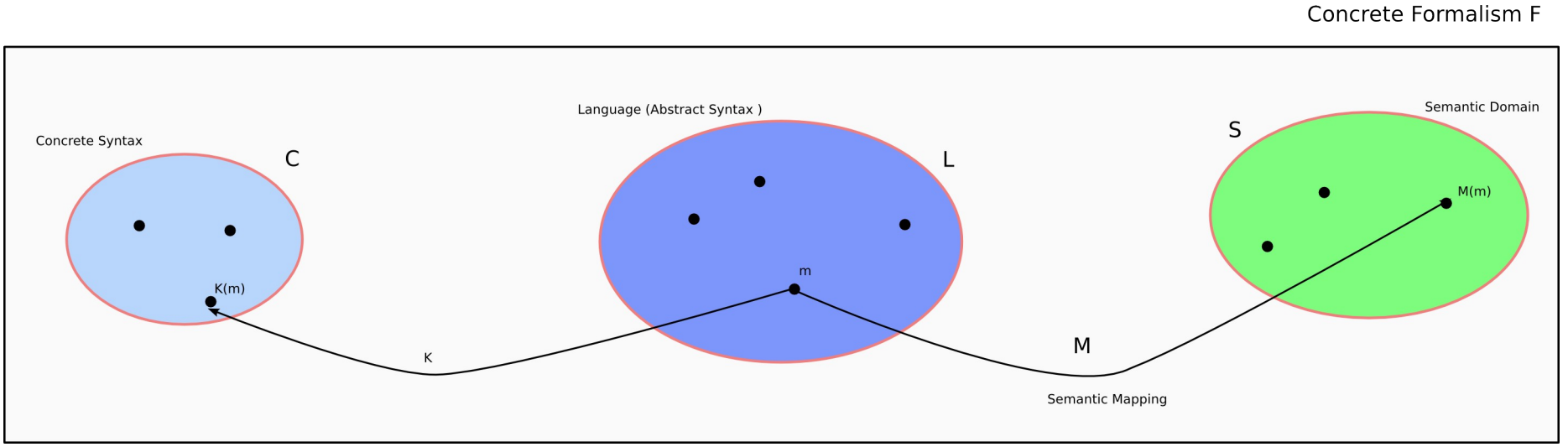

Concrete Formalism F
Concrete Syntax
C
K(m)
Language (Abstract Syntax)
L
m
Semantic Domain
S
M(m)
K
M
Semantic Mapping

Explicit Modelling of Modelling Languages/ Formalisms



What is the semantic domain of the Class Diagram formalism (when used as a meta-modelling language)?

Modelling Languages/ Formalisms



Textual Languages

"this sentence is very short"

- Individual letters in an alphabet
- Combined into words
- Combined in to sentences in a language

- Letters in words *specified* by regular expressions
- Words in a language *specified* by a grammar

- Symbols are combined by "is to the right of"

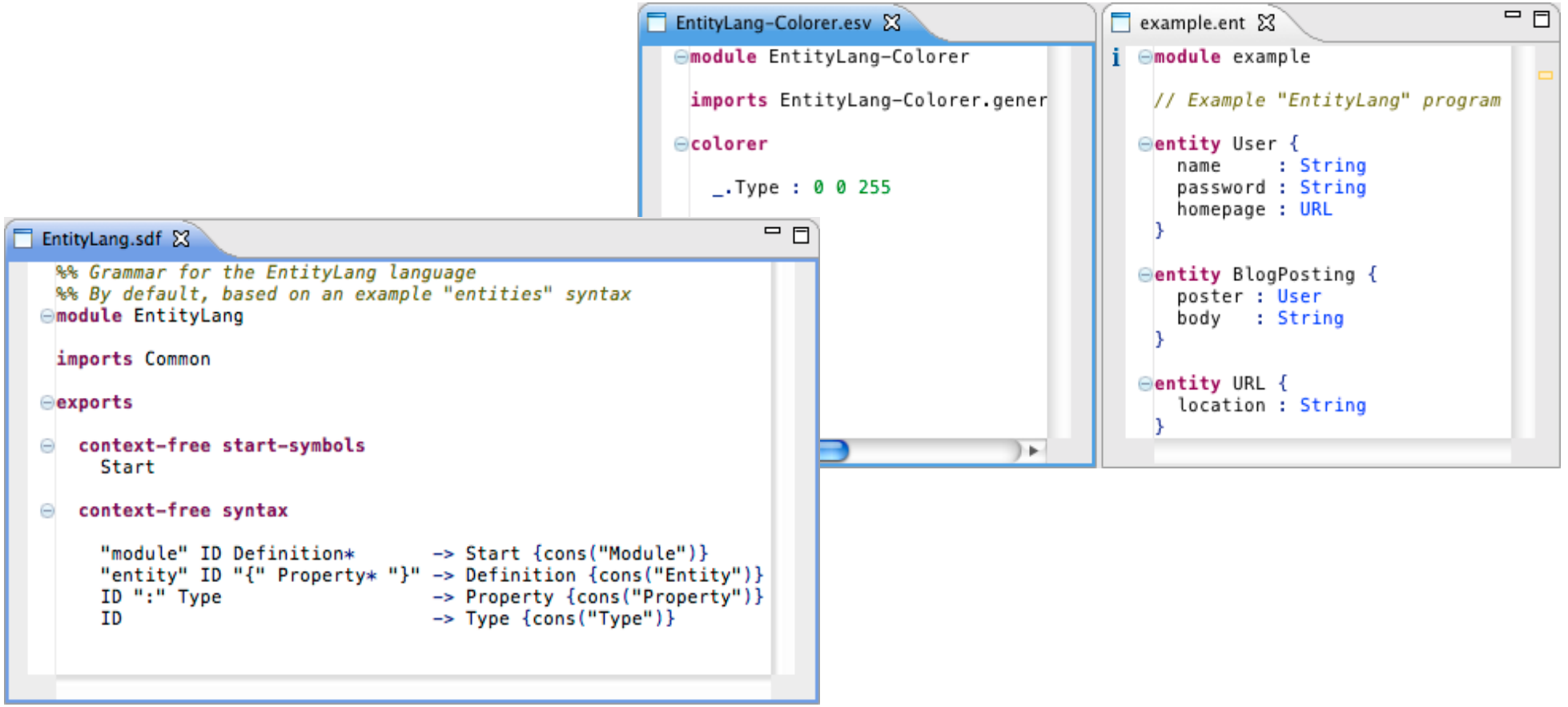
The Spoofax Language Workbench

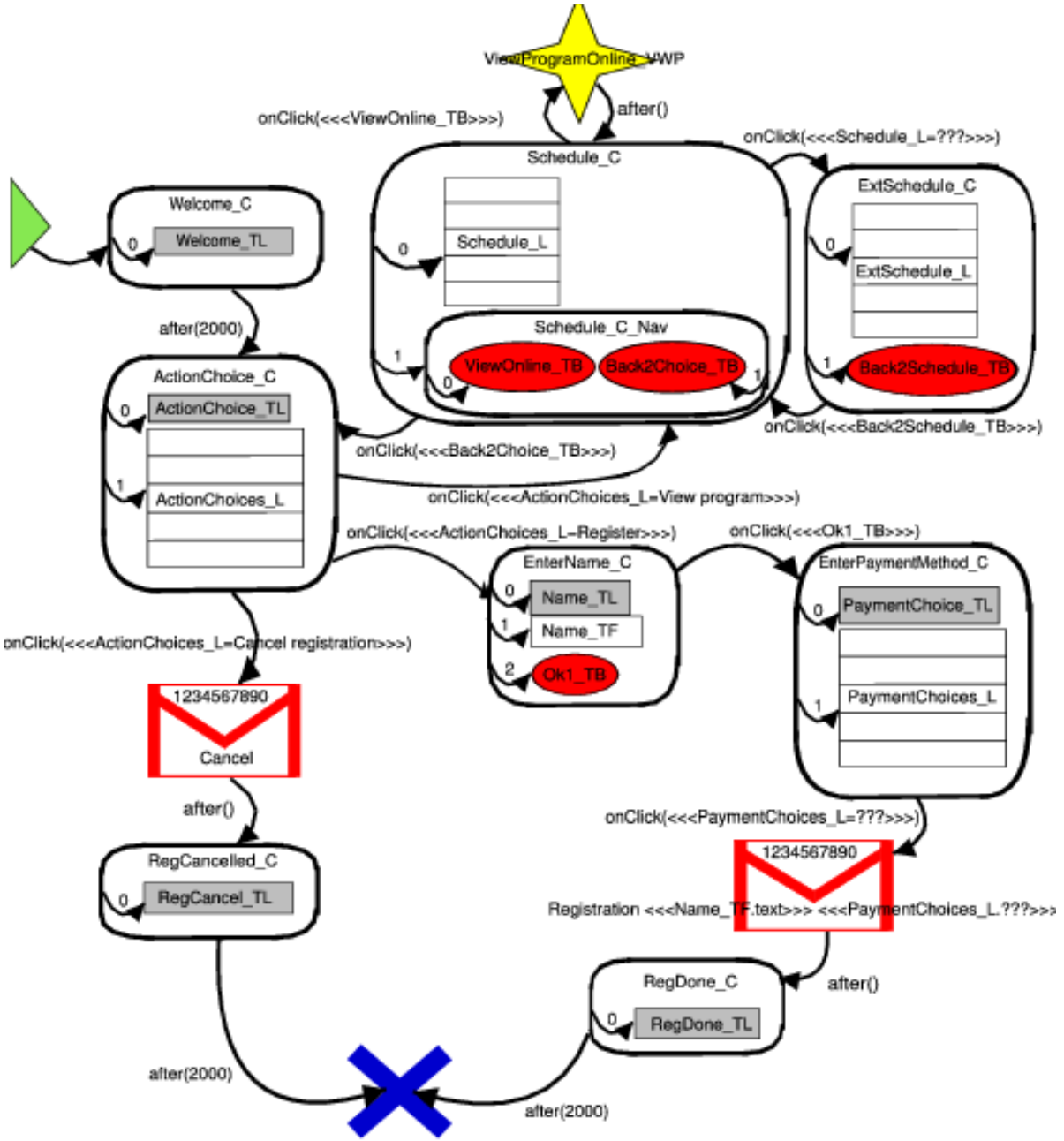
Report TUD-SERG-2010-014a

Rules for Declarative Specification of Languages and IDEs

Lennart C. L. Kats
Delft University of Technology
l.c.l.kats@tudelft.nl

Eelco Visser
Delft University of Technology
visser@acm.org



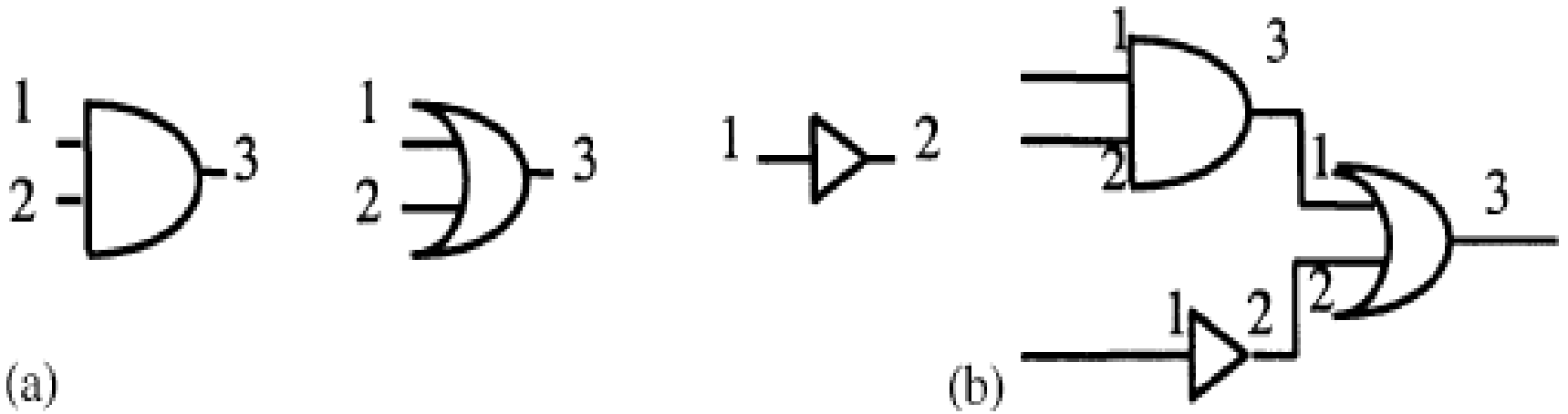

ViewProgramOnline_VWP
onClick(<<<ViewOnline_TB>>>)
after()
onClick(<<<Schedule_L=???>>>)
Schedule_C
Schedule_L
Schedule_C_Nav
ViewOnline_TB
Back2Choice_TB
ExtSchedule_C
ExtSchedule_L
Back2Schedule_TB
onClick(<<<Back2Schedule_TB>>>)
Welcome_C
Welcome_TL
after(2000)
ActionChoice_C
ActionChoice_TL
ActionChoices_L
onClick(<<<Back2Choice_TB>>>)
onClick(<<<ActionChoices_L=View program>>>)
onClick(<<<ActionChoices_L=Register>>>)
onClick(<<<Ok1_TB>>>)
EnterName_C
Name_TL
Name_TF
Ok1_TB
EnterPaymentMethod_C
PaymentChoice_TL
PaymentChoices_L
onClick(<<<ActionChoices_L=Cancel registration>>>)
1234567890
Cancel
after()
RegCancelled_C
RegCancel_TL
onClick(<<<PaymentChoices_L=???>>>)
1234567890
Registration <<<Name_TF.text>>> <<<PaymentChoices_L.???>>>
RegDone_C
RegDone_TL
after()
after(2000)
after(2000)

Journal of Visual Languages and Computing (2002) **13,** 573–600
doi:10.1006/S1045-926X(02)00025-3 available online at http://www.idealibrary.com on IDEAL®

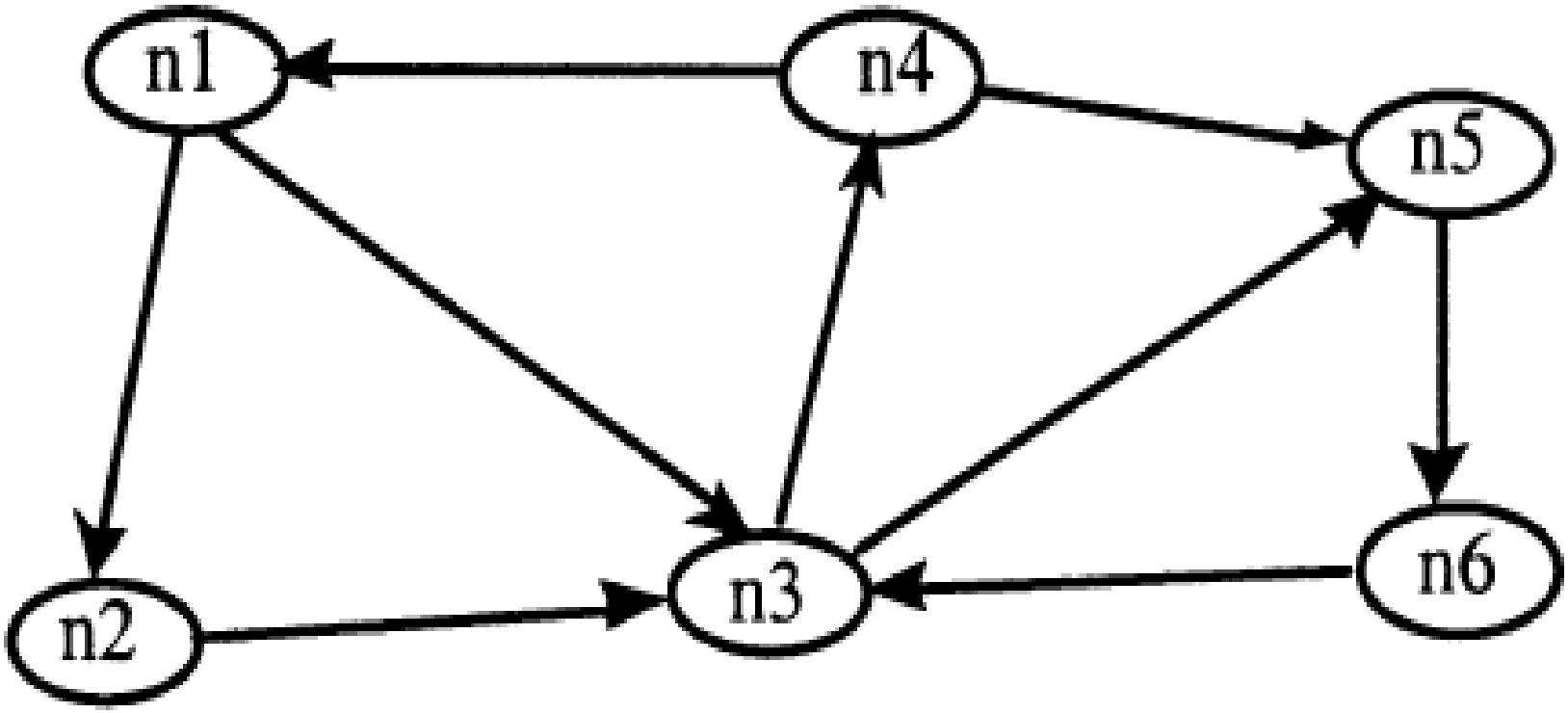
A Classification Framework to Support the Design of Visual Languages

G. COSTAGLIOLA*, A. DELUCIA†, S. OREFICE‡ AND G. POLESE*

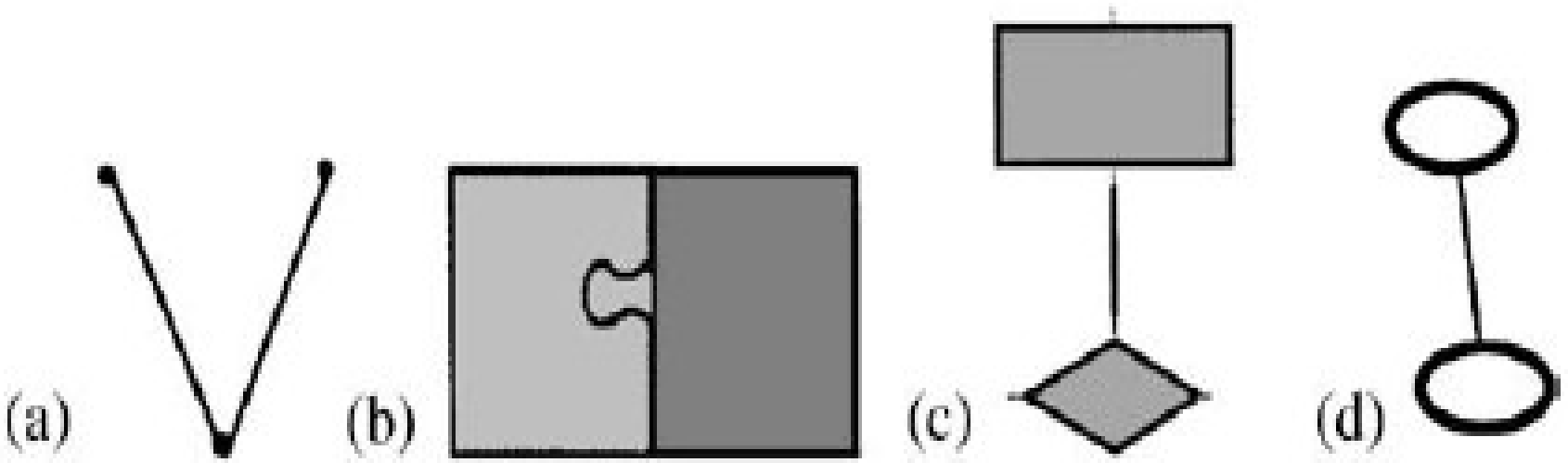
Plex



Graph



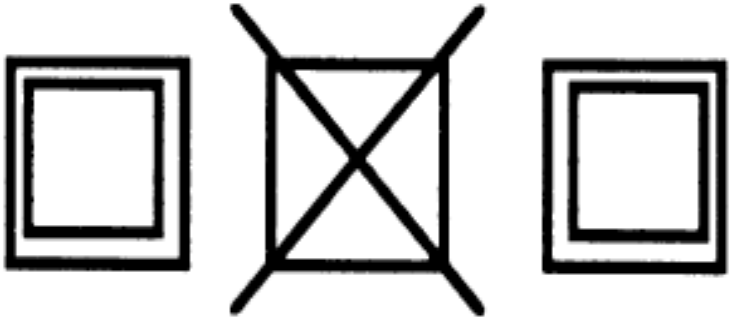
Connection Types

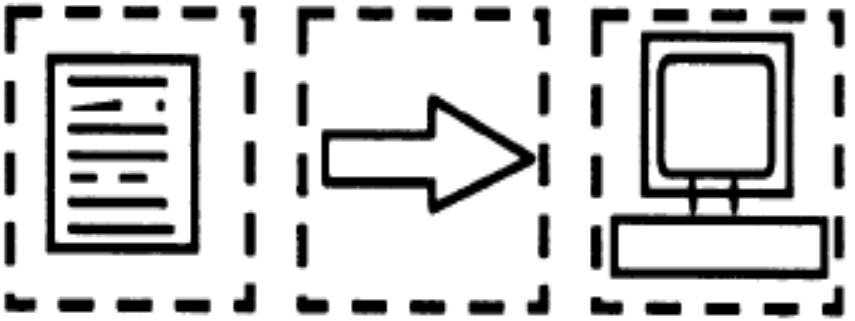


String

Sequence

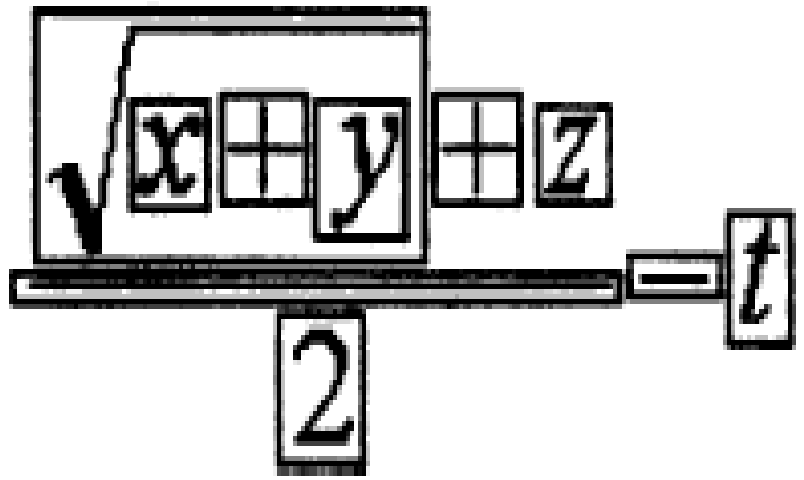
Iconic



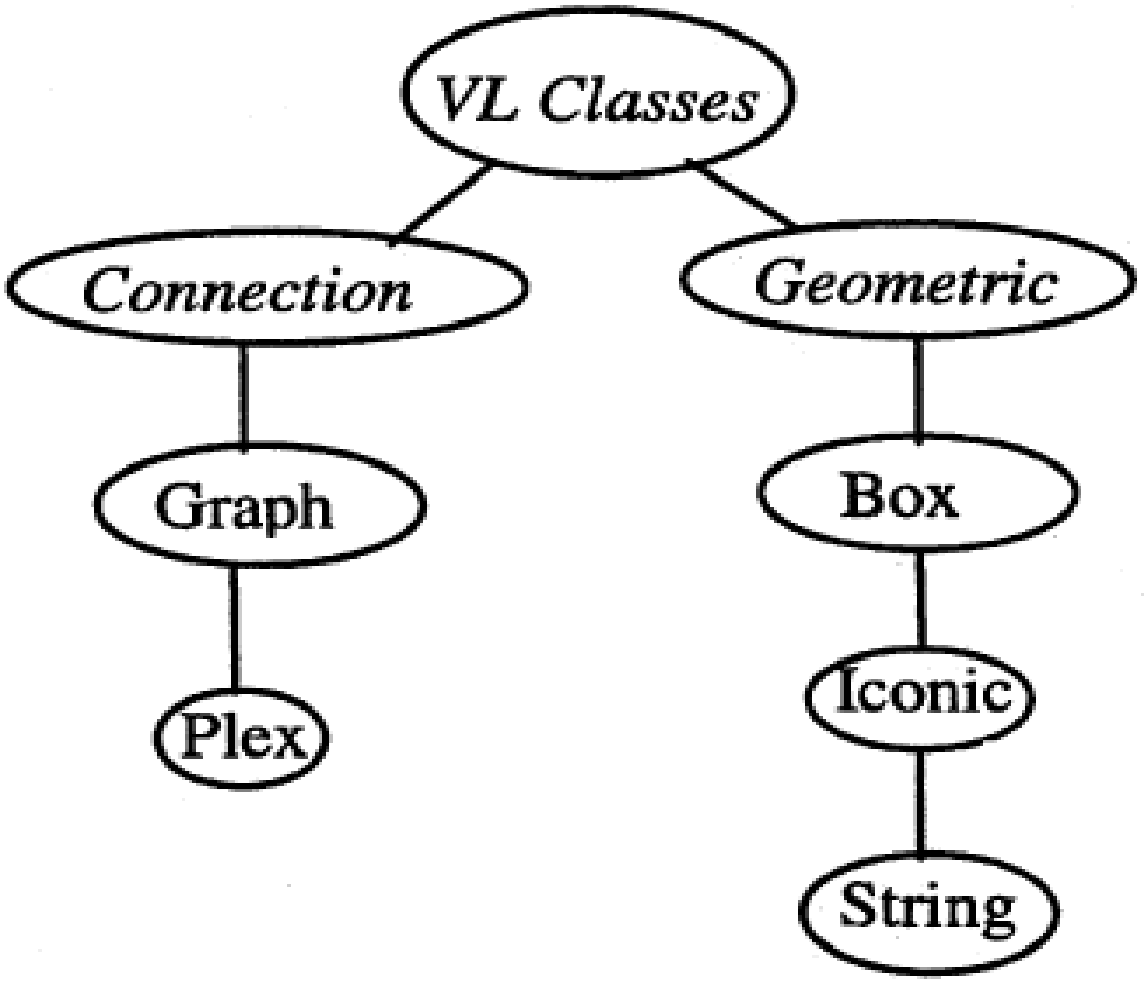


Box

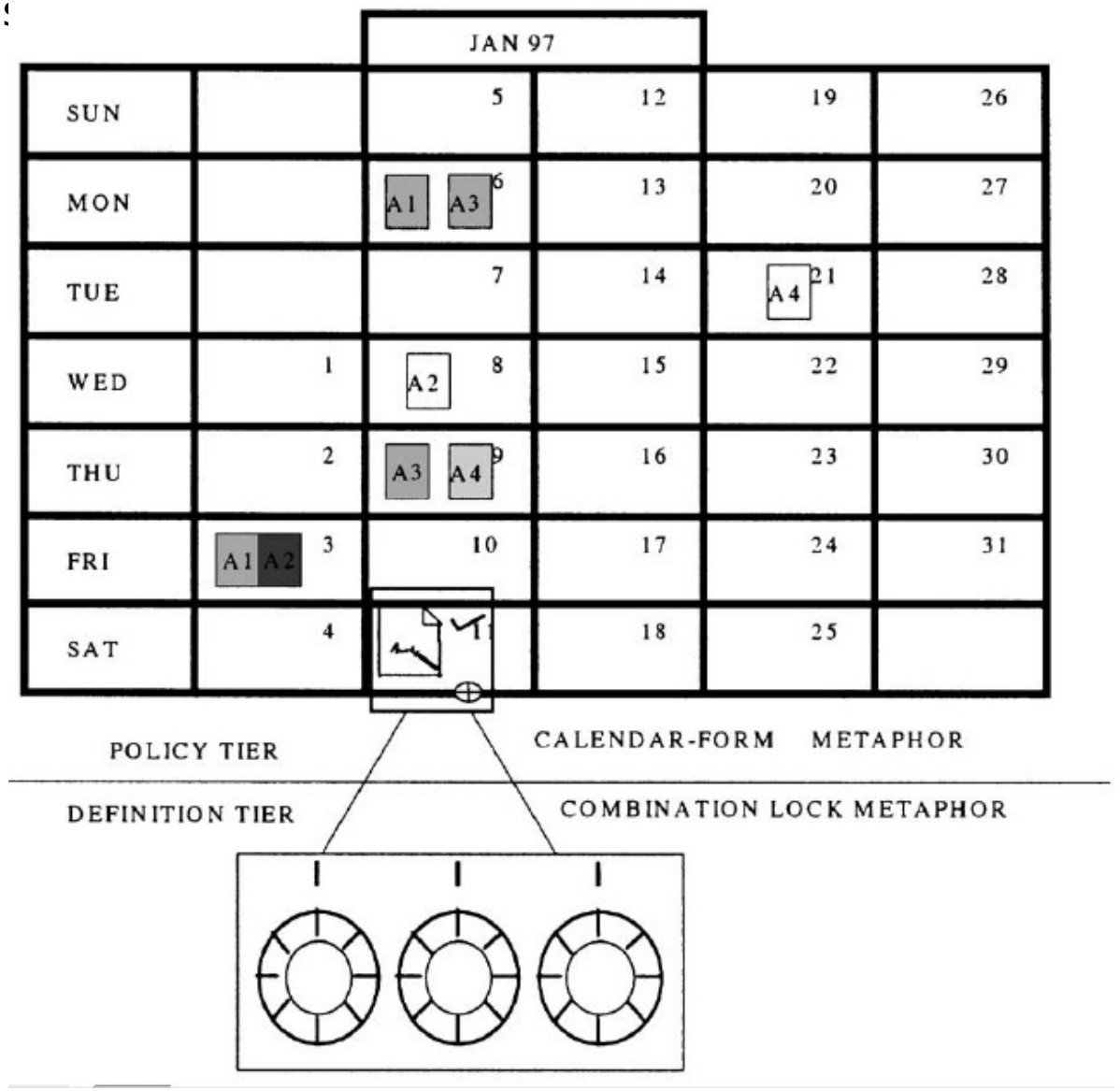
$$\frac{\sqrt{x+y+z}}{2} - t$$



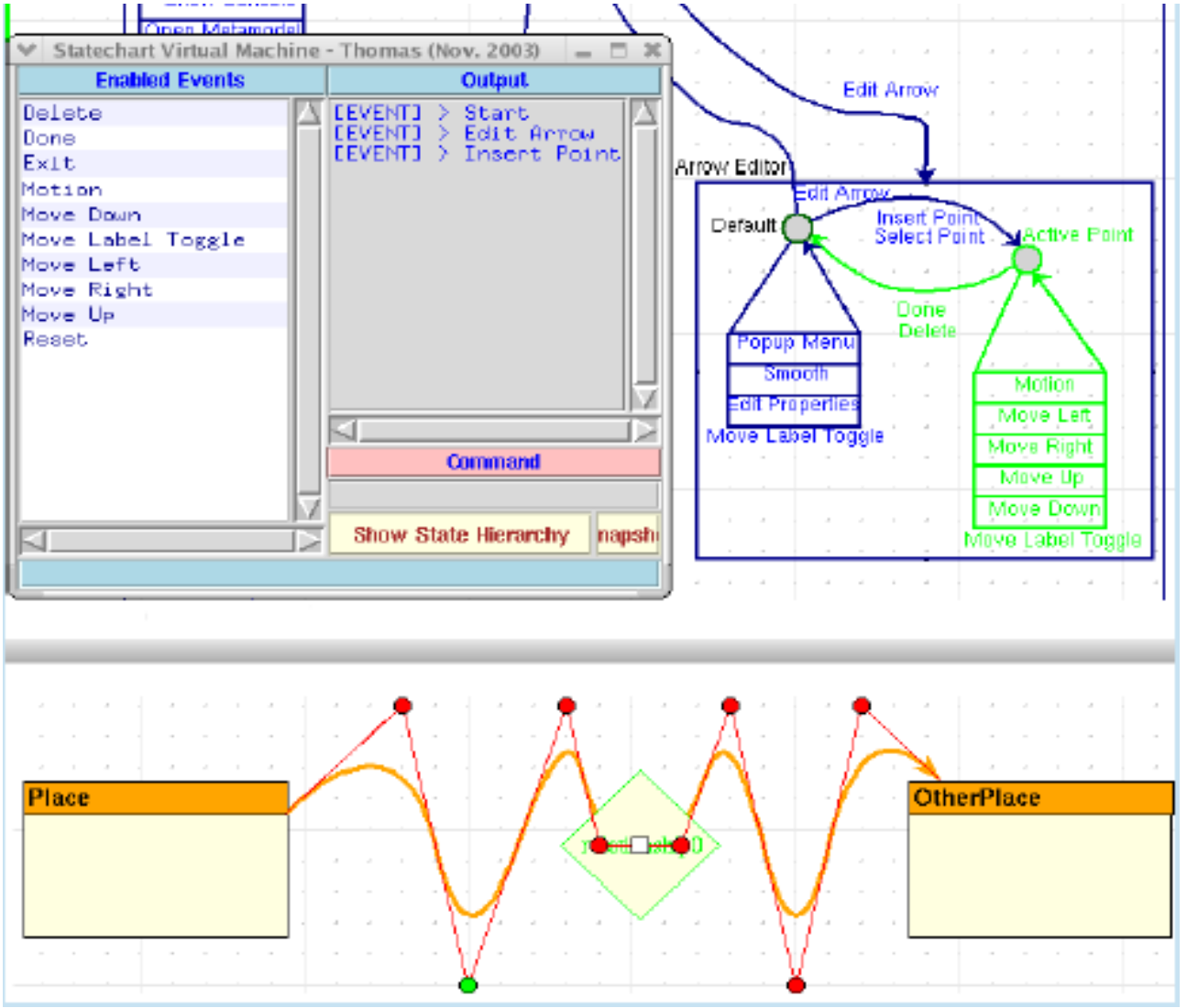
Visual Language Classes



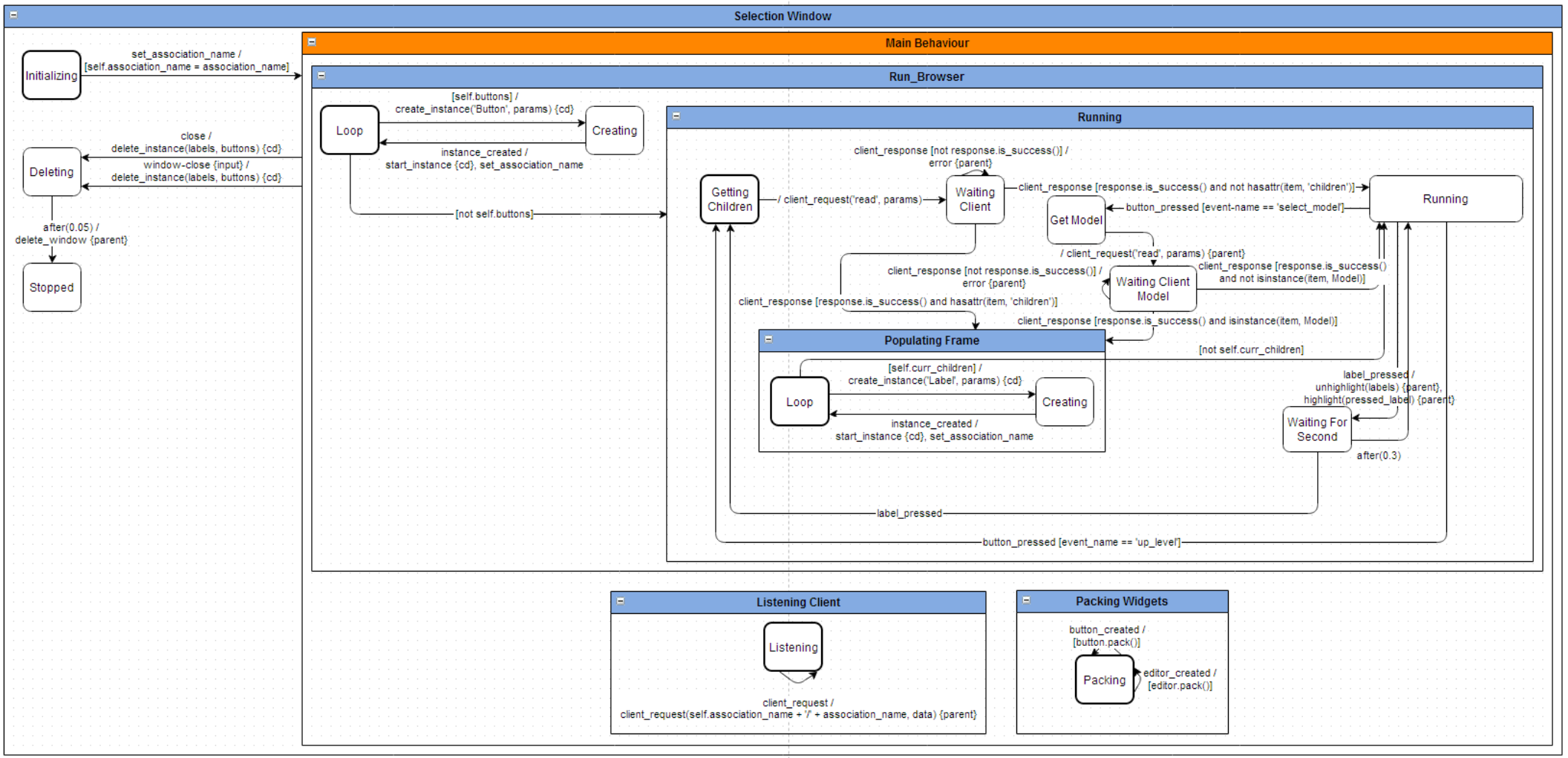
Hybrid Languages



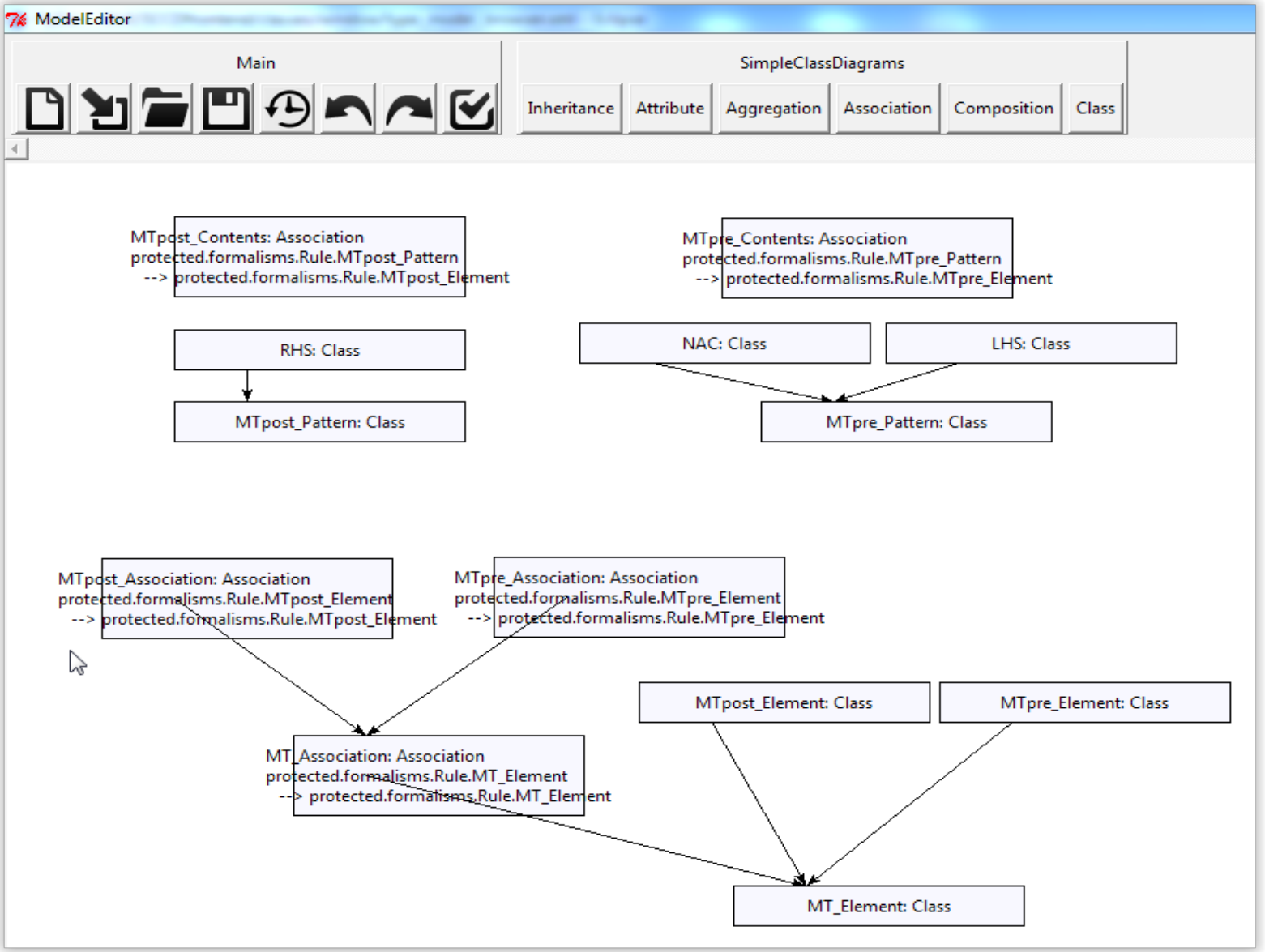
Syntax-directed Visual Editors: model behaviour



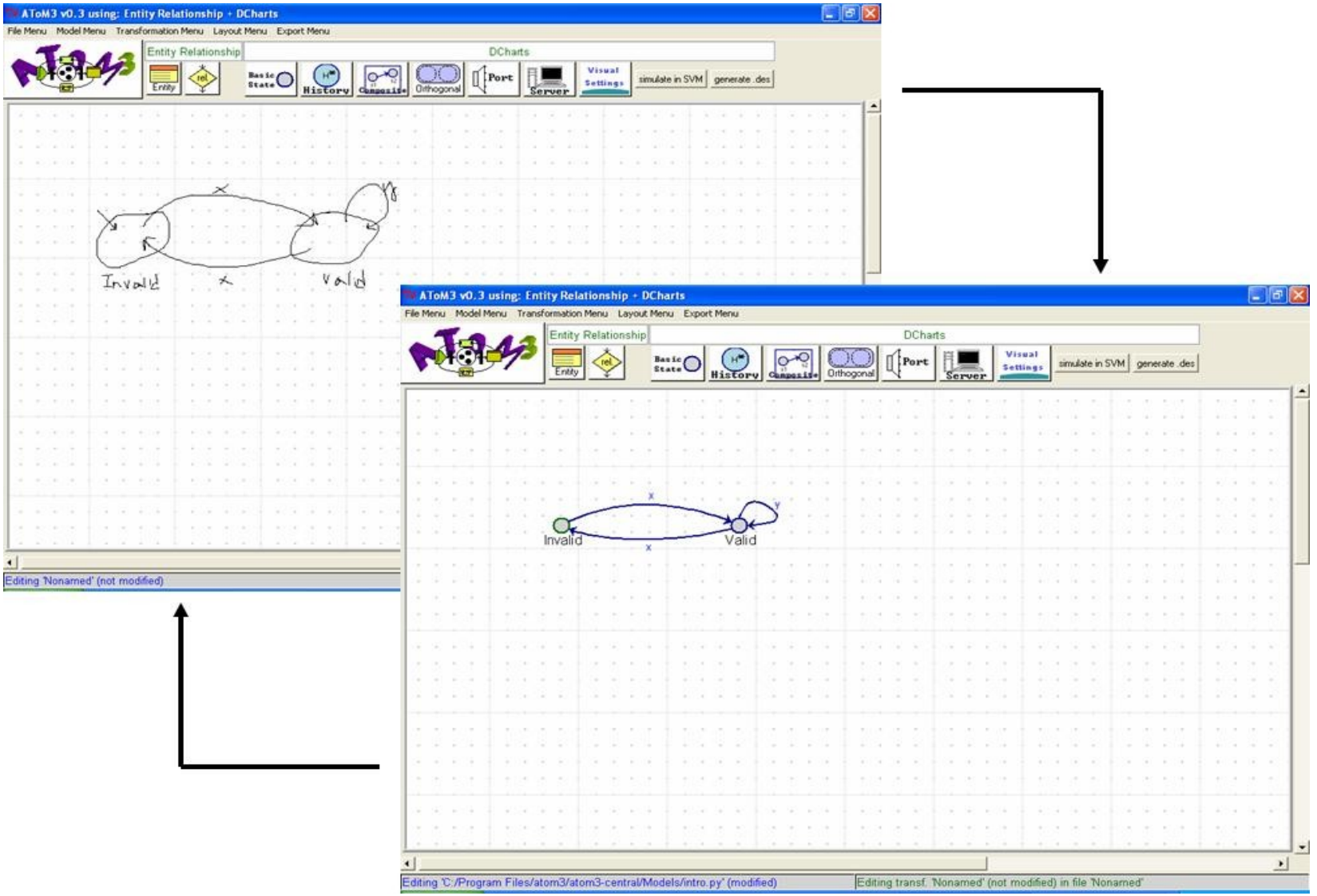
Syntax-directed Visual Editors: model behaviour



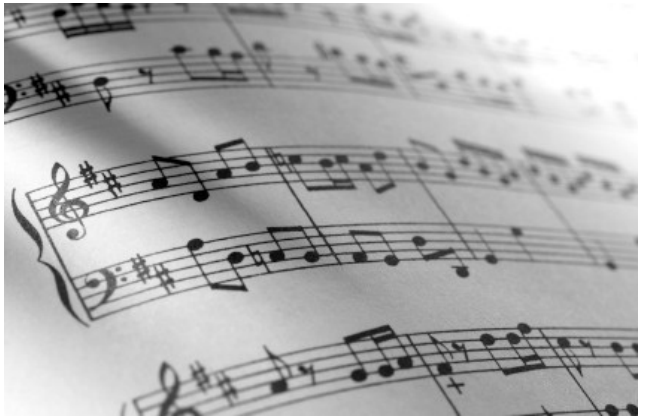
Syntax-directed Visual Editors: model behaviour

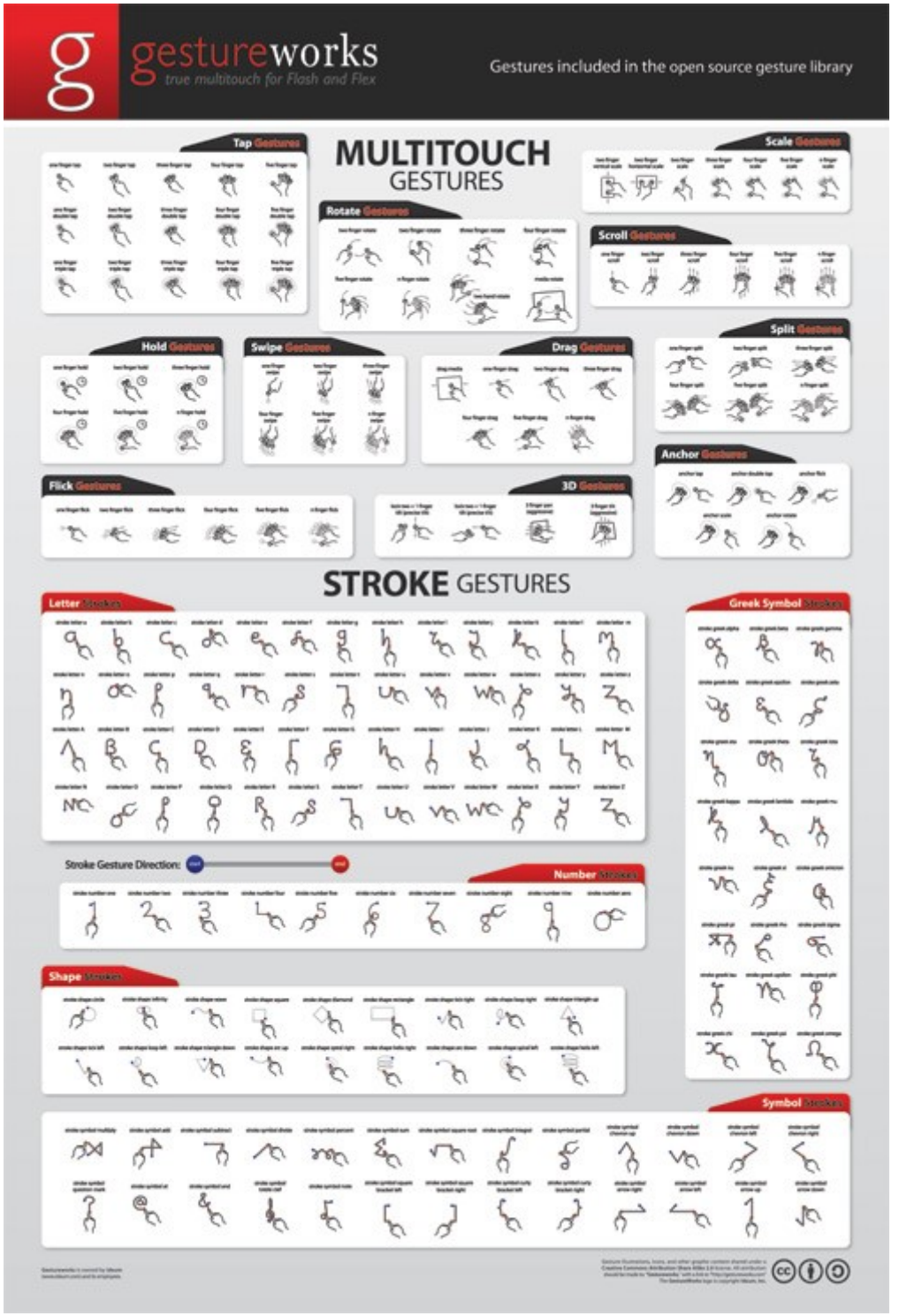


Syntax-directed Visual Editors: freehand (early stages of multi-domain project)



Different Media:
Gestural Interaction, Sound, ...





IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 35, NO. 5, NOVEMBER-DECEMBER 2009

The "Physics" of Notations: Towards a Scientific Basis for Constructing Visual Notations in Software Engineering

Daniel L. Moody, *Member, IEEE*

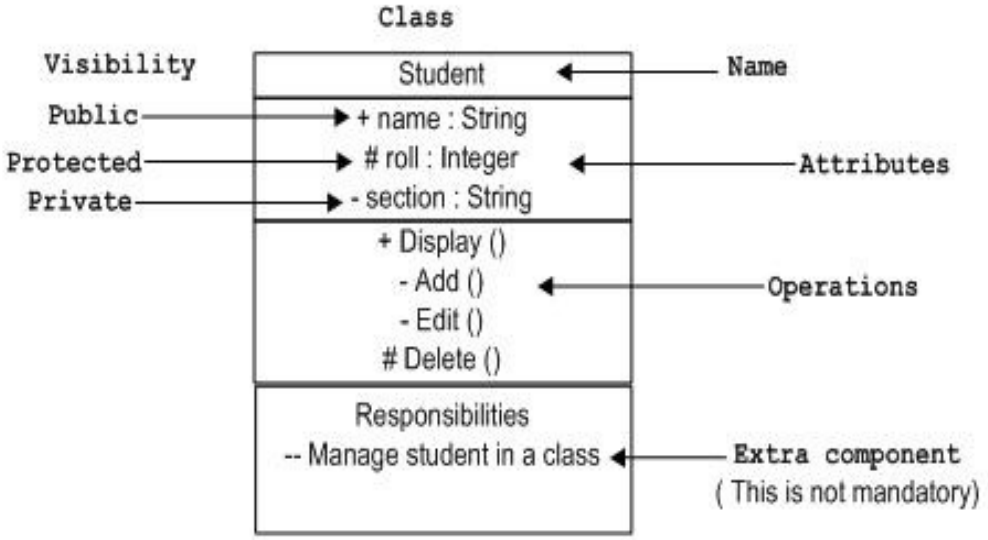
Introduction

- Visual notations pre-date textual ones
- Visual notations are important for Modelling and Software Engineering
- Humans are excellent pattern recognizers
- Need cognitively efficient and effective notations.
  Cognitive effectiveness =
  speed, ease and accuracy with which a representation can be processed by the human mind

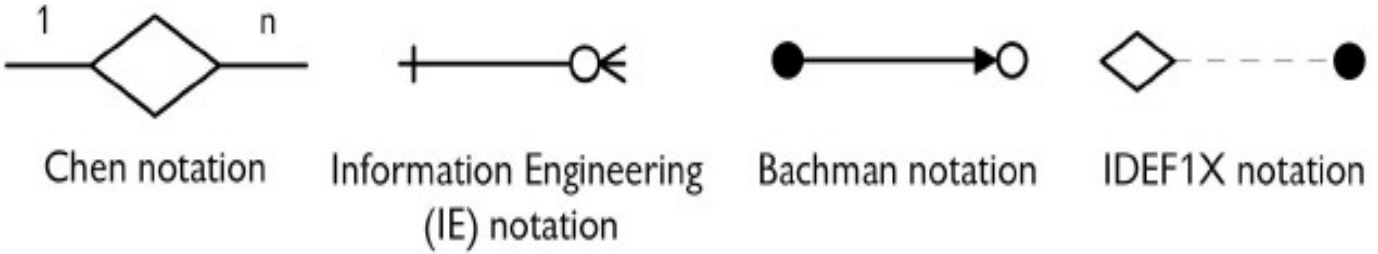
a DSVL @ Lascaux

Introduction/Rationale

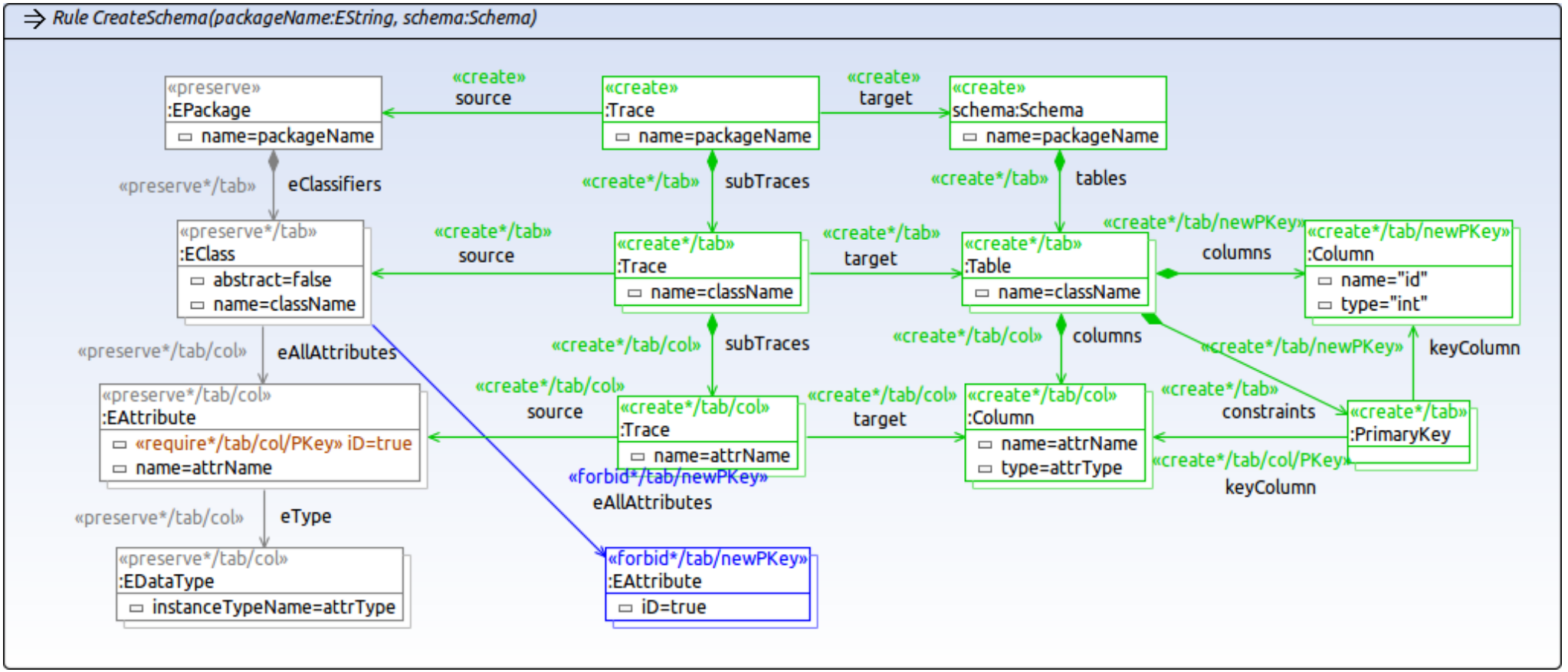
Visual notations are often introduced without underlying theory or rationale

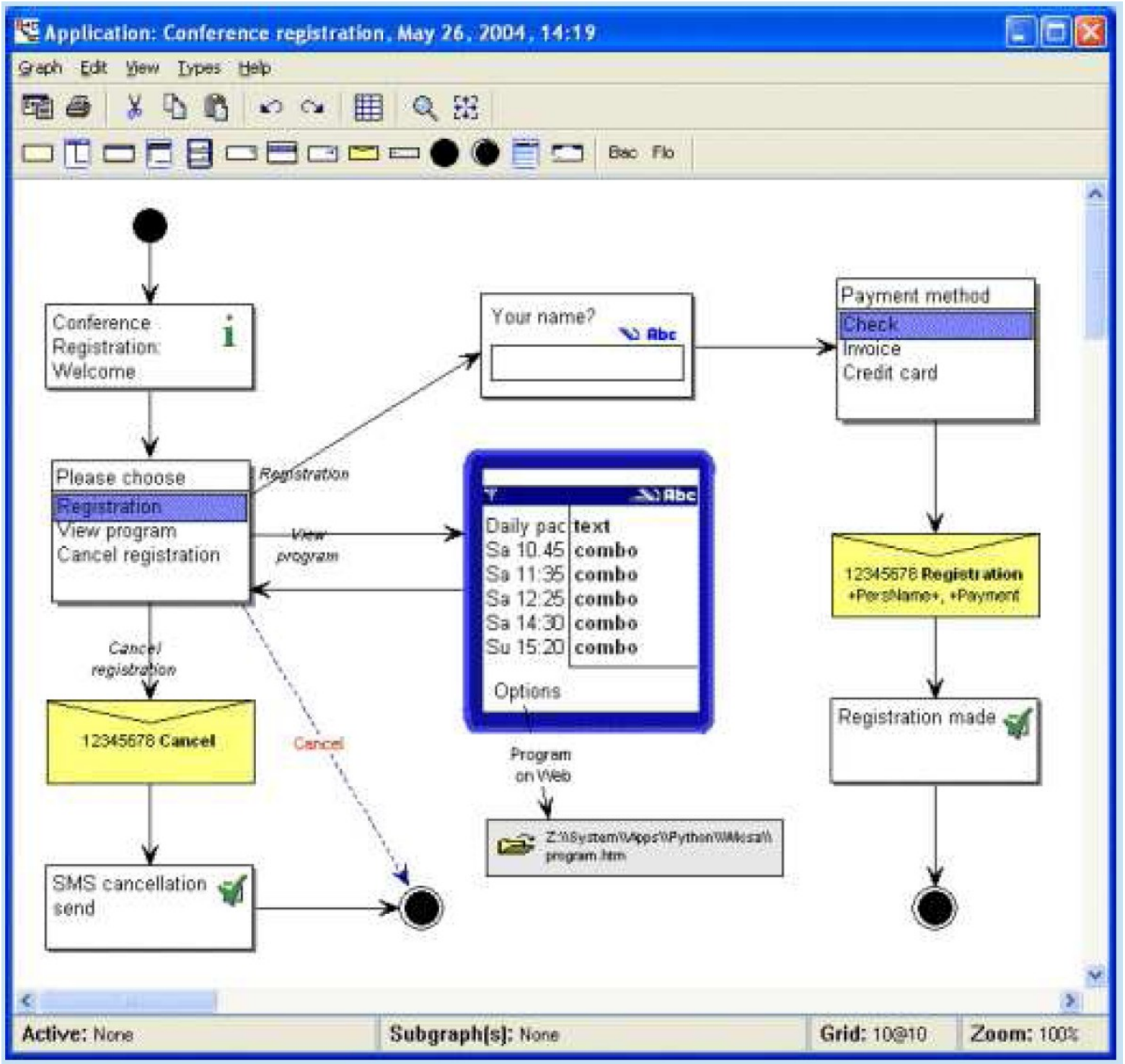


Many visual notations for same concepts.

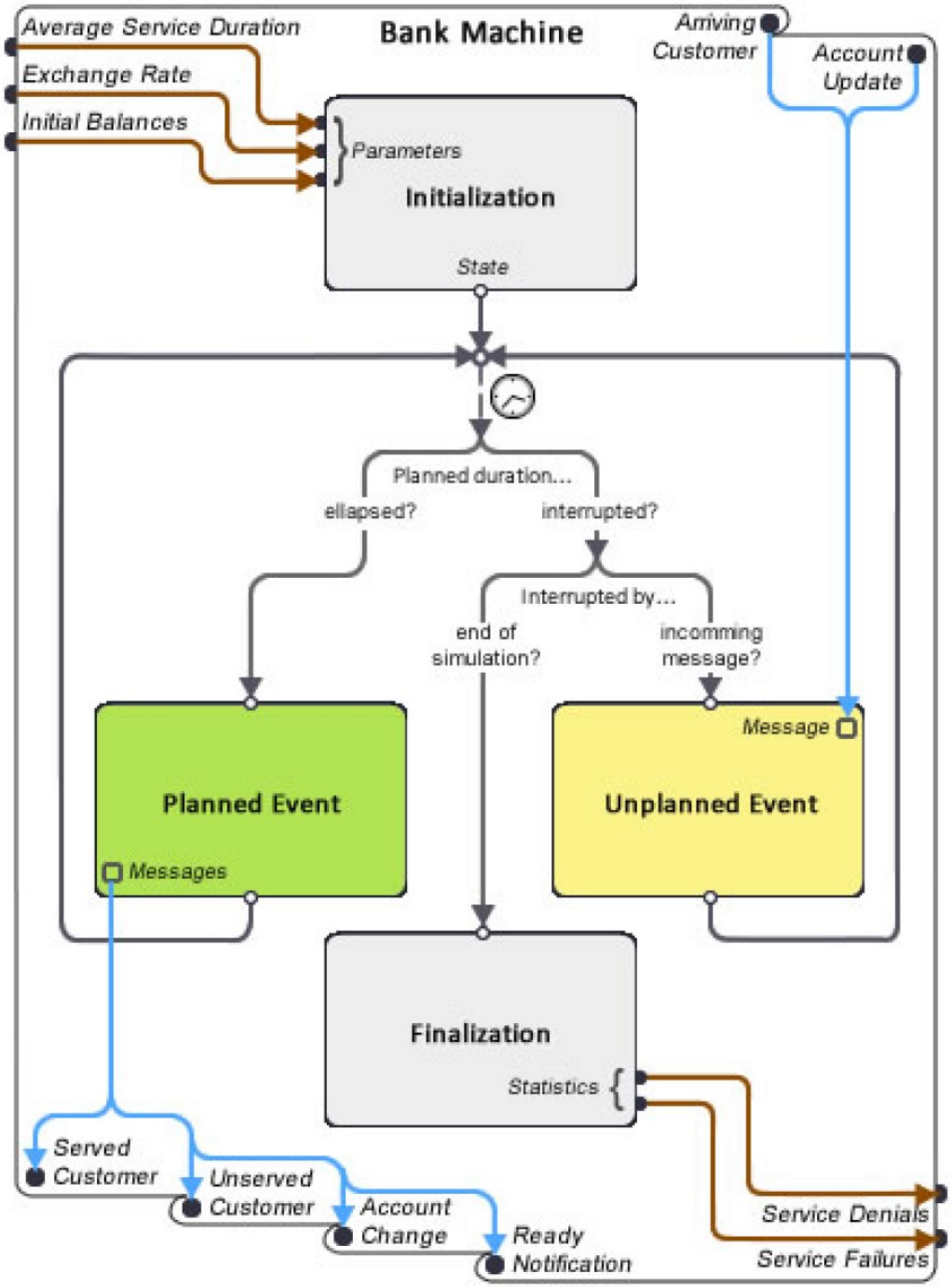


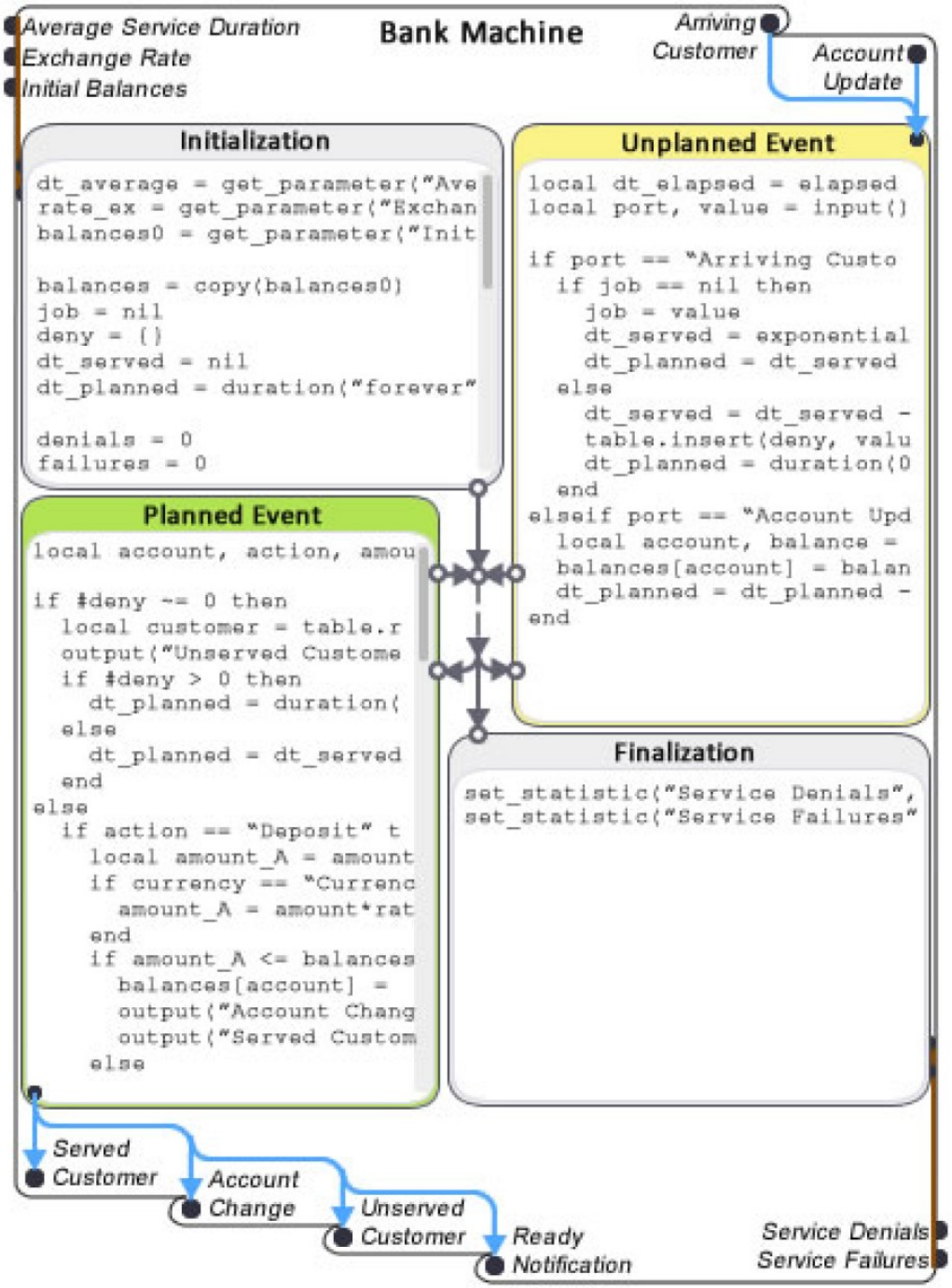
No rigorous way to compare effectiveness and hence no clear design goal.

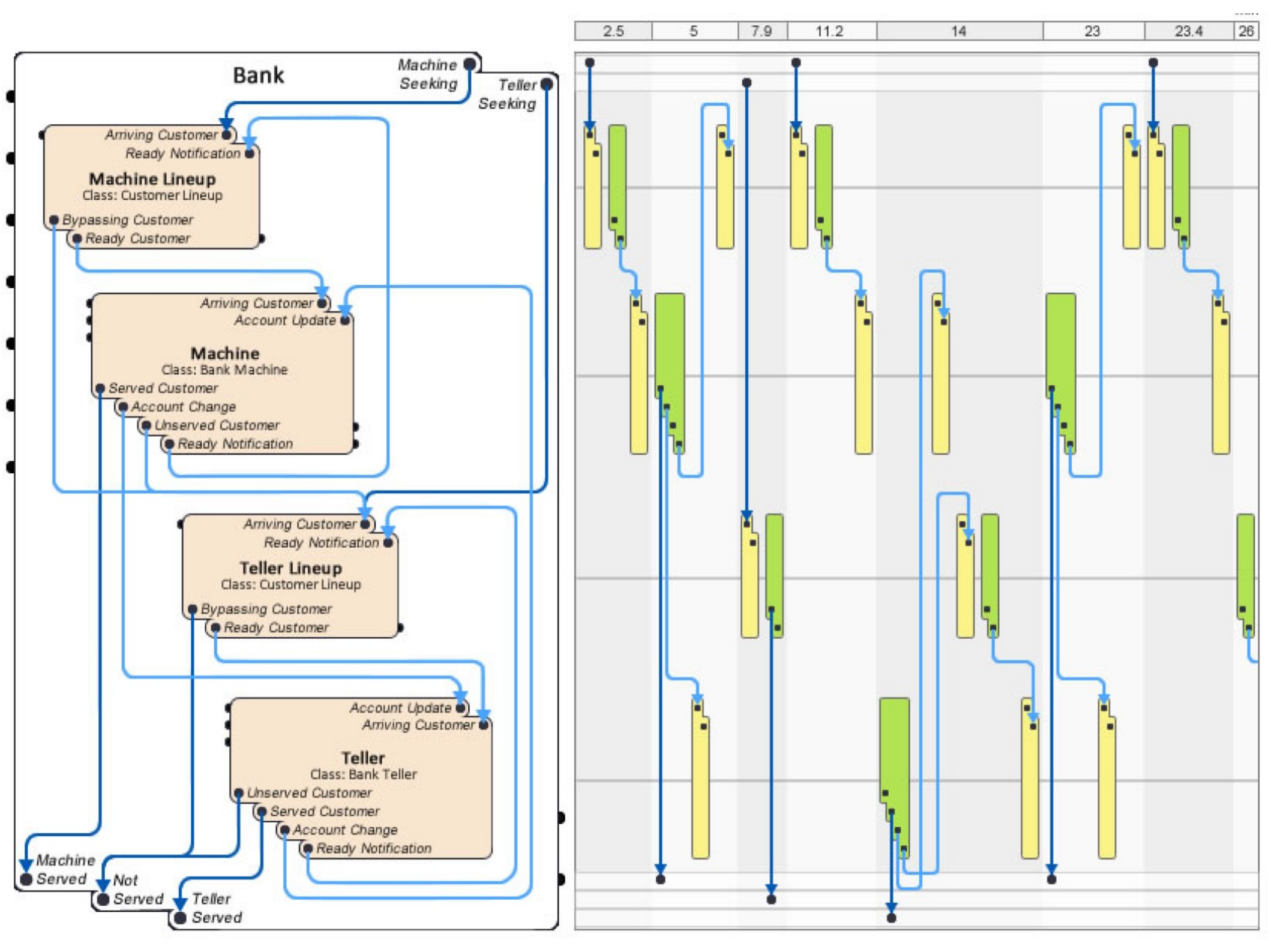

Rule CreateSchema(packageName:EString, schema:Schema)
«preserve»
:EPackage
name=packageName
«create»
source
«create»
:Trace
name=packageName
«create»
target
«create»
schema:Schema
name=packageName
«preserve*/tab»
eClassifiers
«create*/tab»
subTraces
«create*/tab»
tables
«preserve*/tab»
:EClass
abstract=false
name=className
«create*/tab»
source
«create*/tab»
:Trace
name=className
«create*/tab»
target
«create*/tab»
:Table
name=className
«create*/tab/newPKey»
columns
«create*/tab/newPKey»
:Column
name="id"
type="int"
«preserve*/tab/col»
eAllAttributes
«create*/tab/col»
subTraces
«create*/tab/col»
columns
«create*/tab/newPKey»
keyColumn
«preserve*/tab/col»
:EAttribute
«require*/tab/col/PKey» iD=true
name=attrName
«create*/tab/col»
source
«create*/tab/col»
:Trace
name=attrName
«create*/tab/col»
target
«create*/tab/col»
:Column
name=attrName
type=attrType
«create*/tab»
constraints
«create*/tab»
:PrimaryKey
«forbid*/tab/newPKey»
eAllAttributes
«create*/tab/col/PKey»
keyColumn
«preserve*/tab/col»
eType
«preserve*/tab/col»
:EDataType
instanceTypeName=attrType
«forbid*/tab/newPKey»
:EAttribute
iD=true


Application: Conference registration, May 26, 2004, 14:19
Graph Edit View Types Help
Conference Registration: Welcome
Your name?
Abc
Payment method
Check
Invoice
Credit card
Please choose
Registration
View program
Cancel registration
Registration
View program
Daily pac text
Sa 10.45 combo
Sa 11:35 combo
Sa 12:25 combo
Sa 14:30 combo
Su 15:20 combo
Options
12345678 Registration
+PersName+, +Payment
Cancel registration
12345678 Cancel
Cancel
Program on Web
Registration made
SMS cancellation send
Active: None
Subgraph(s): None
Grid: 10@10
Zoom: 100%

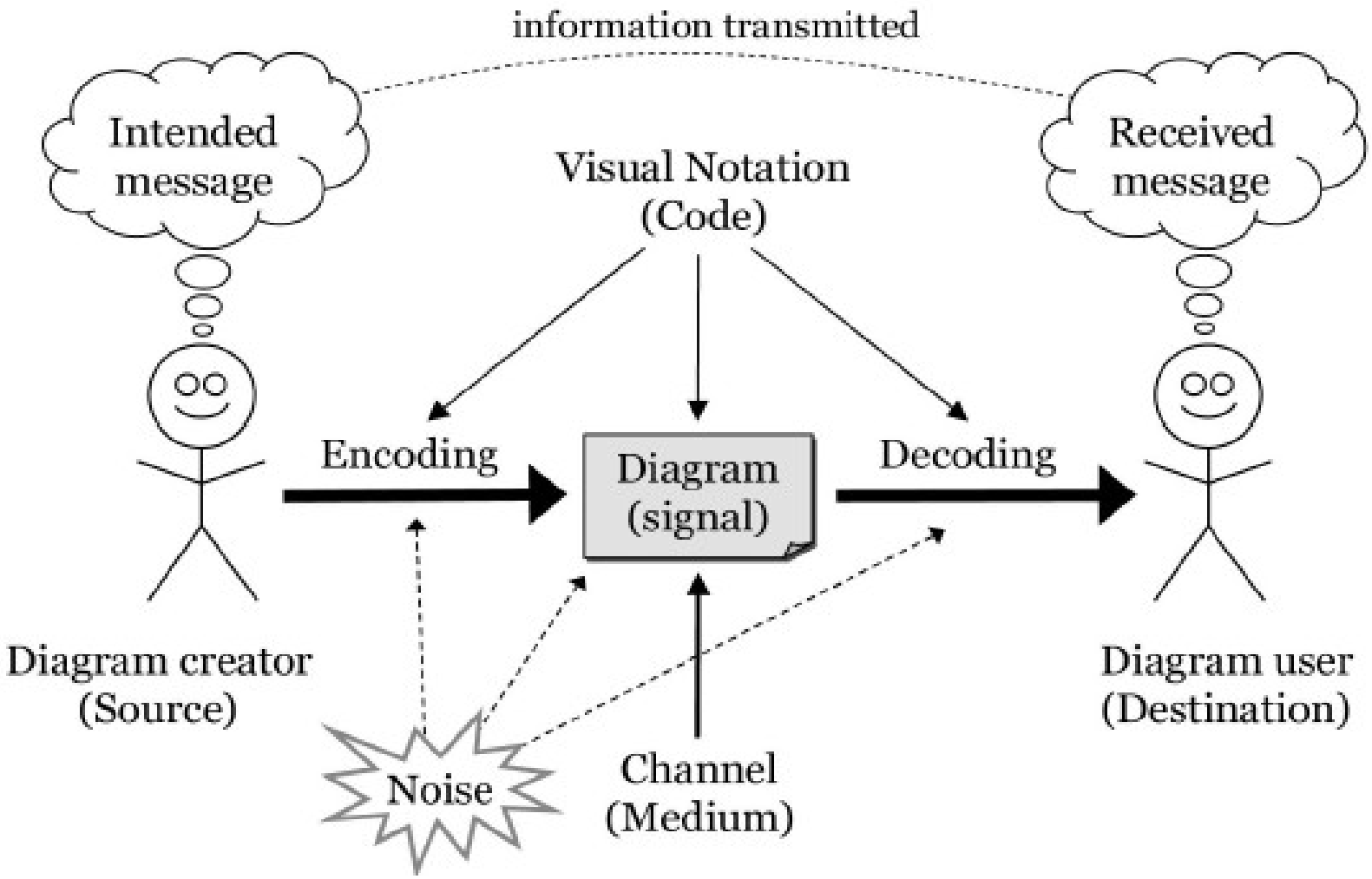

Conference registration
Daily pa...
995,- USD
Sa 10.4...
T1: Smith
T2: Jackson
T3: Holland
Sa 11:3...
T4: Cohen
Sa 12:2...
T7: Dawson
Ok
Cancel
ABC
C


Average Service Duration
Exchange Rate
Initial Balances
Bank Machine
Arriving Customer
Account Update
Parameters
Initialization
State
Planned duration...
elapsed?
interrupted?
Interrupted by...
end of simulation?
incomming message?
Message
Planned Event
Unplanned Event
Messages
Finalization
Statistics
Served Customer
Unserved Customer
Account Change
Ready Notification
Service Denials
Service Failures

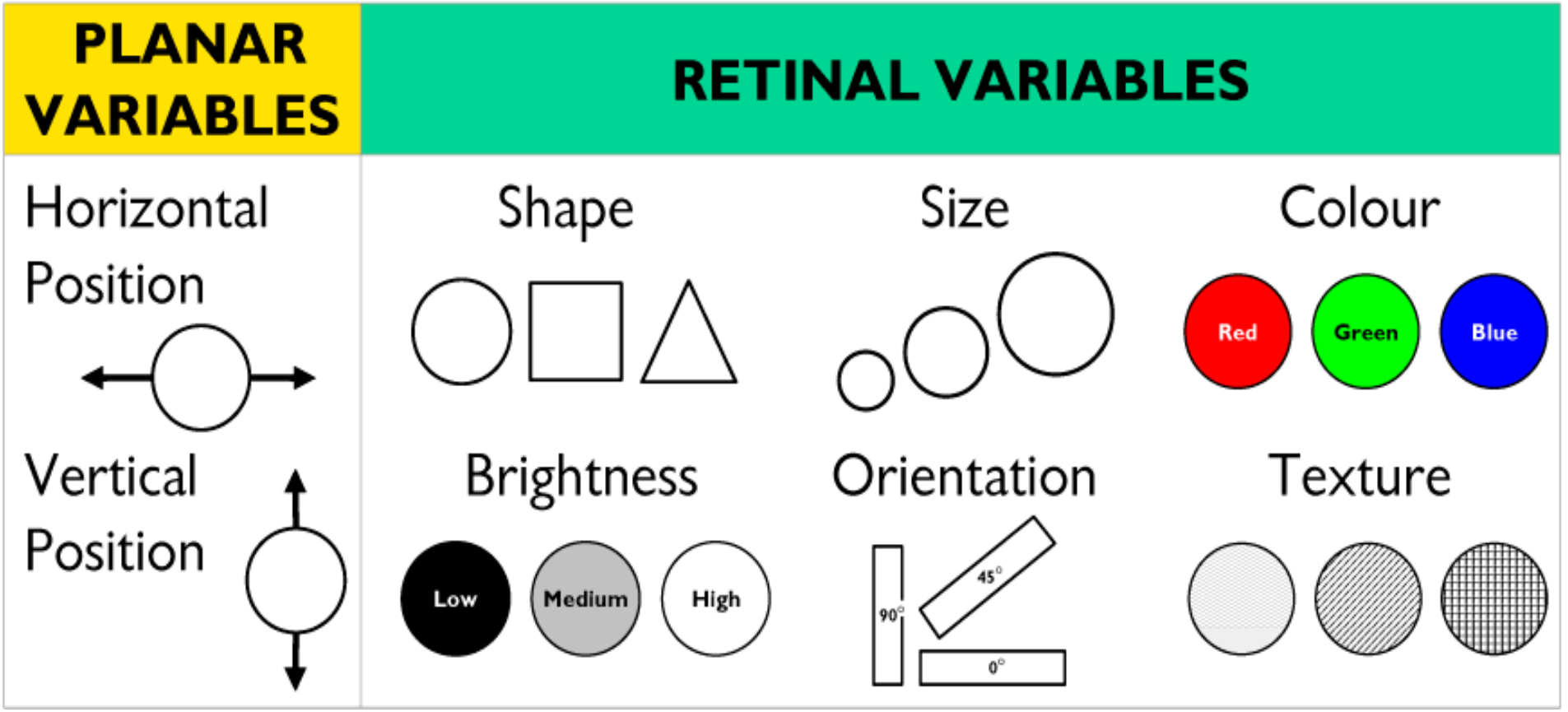

Average Service Duration
Exchange Rate
Initial Balances
Bank Machine
Arriving Customer
Account Update
Initialization
dt_average = get_parameter("Ave
rate_ex = get_parameter("Exchan
balances0 = get_parameter("Init
balances = copy(balances0)
job = nil
deny = {}
dt_served = nil
dt_planned = duration("forever"
denials = 0
failures = 0
Unplanned Event
local dt_elapsed = elapsed
local port, value = input()
if port == "Arriving Custo
if job == nil then
job = value
dt_served = exponential
dt_planned = dt_served
else
dt_served = dt_served -
table.insert(deny, valu
dt_planned = duration(0
end
elseif port == "Account Upd
local account, balance =
balances[account] = balan
dt_planned = dt_planned -
end
Planned Event
local account, action, amou
if #deny ~= 0 then
local customer = table.r
output("Unserved Custome
if #deny > 0 then
dt_planned = duration(
else
dt_planned = dt_served
end
else
if action == "Deposit" t
local amount_A = amount
if currency == "Currenc
amount_A = amount*rat
end
if amount_A <= balances
balances[account] =
output("Account Chang
output("Served Custom
else
Finalization
set_statistic("Service Denials",
set_statistic("Service Failures"
Served Customer
Account Change
Unserved Customer
Ready Notification
Service Denials
Service Failures


Bank
Machine Seeking
Teller Seeking
Arriving Customer
Ready Notification
Machine Lineup
Class: Customer Lineup
Bypassing Customer
Ready Customer
Arriving Customer
Account Update
Machine
Class: Bank Machine
Served Customer
Account Change
Unserved Customer
Ready Notification
Arriving Customer
Ready Notification
Teller Lineup
Class: Customer Lineup
Bypassing Customer
Ready Customer
Account Update
Arriving Customer
Teller
Class: Bank Teller
Unserved Customer
Served Customer
Account Change
Ready Notification
Machine Served
Not Served
Teller Served
2.5
5
7.9
11.2
14
23
23.4
26

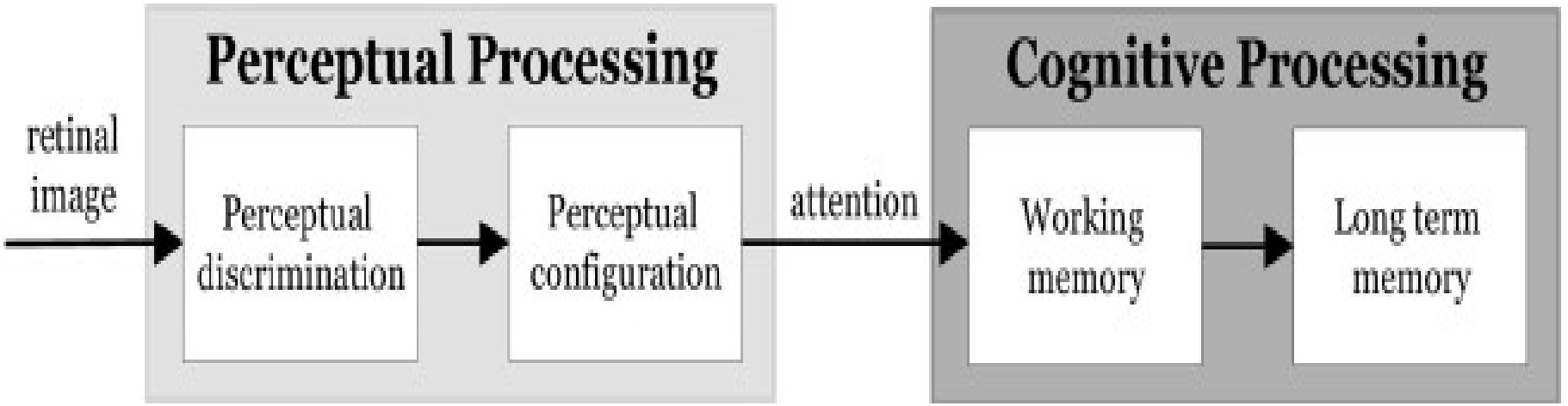
Communication Theory



Encoding: 8 visual variables to (graphically) encode information



Decoding



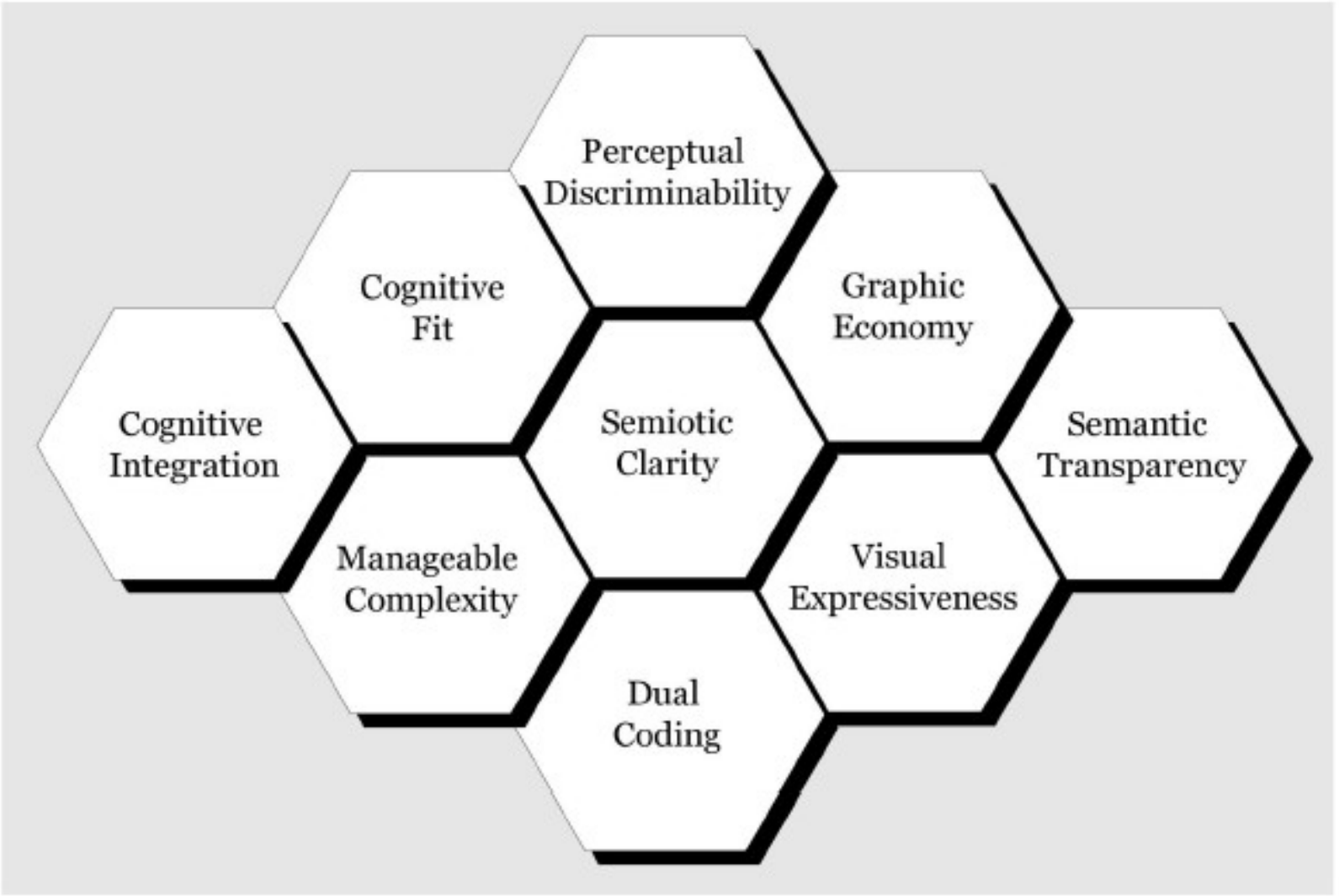
automatic, fast, parallel

slow, large effort, sequential

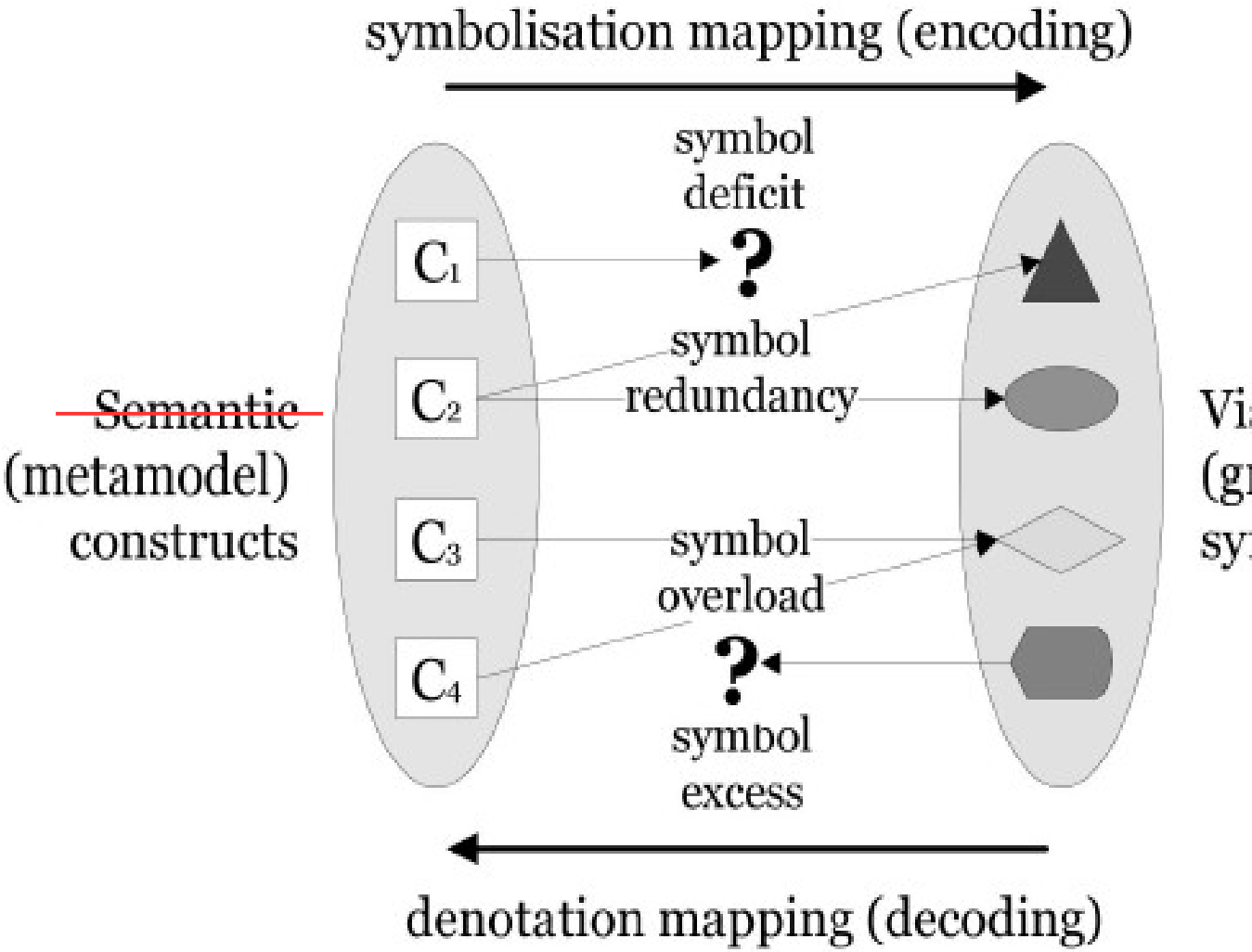
Appropriate notations » offload some of the burden from cognitive to perceptual

Note: “dual channel theory”: brain processes textual/audio in parallel with visual data

Principles for Designing Efficient and Effective Visual Notations

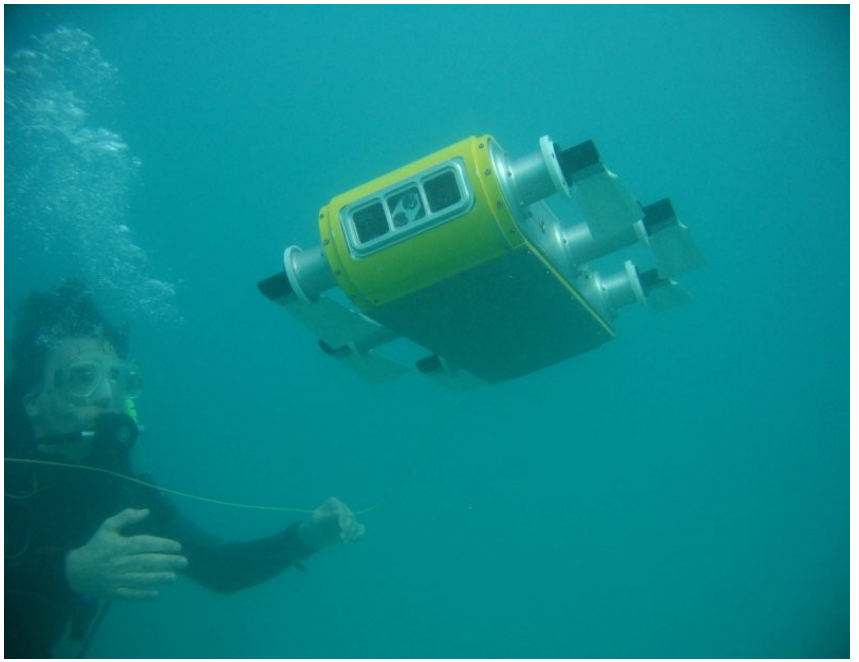


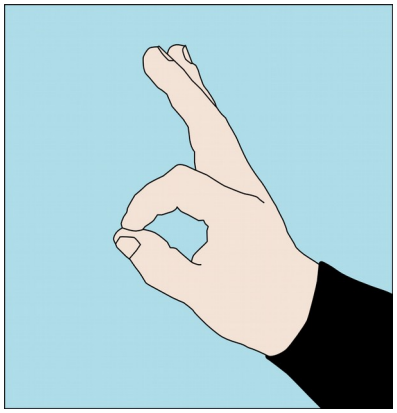
Semiotic Clarity (semiotics = study of signs and sign processes)

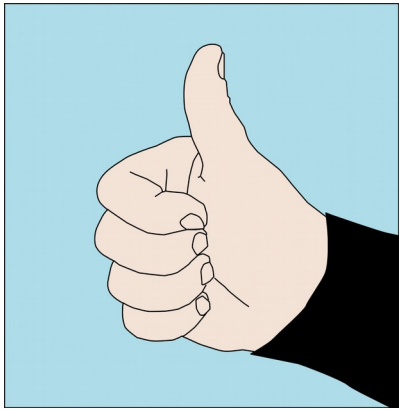


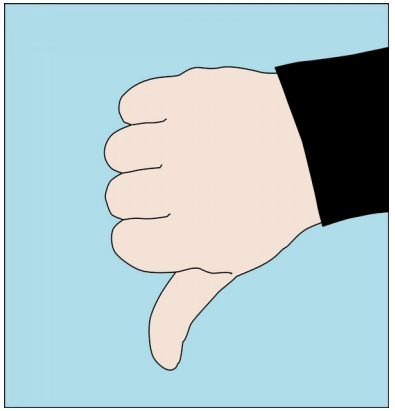
1 to 1

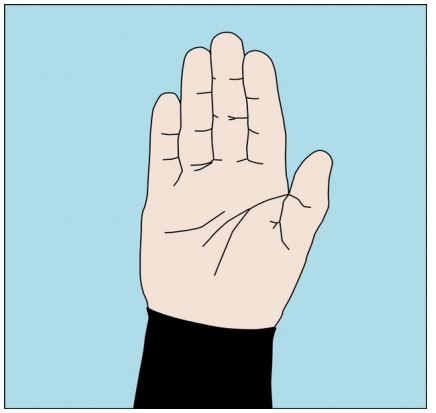
Perceptual Discriminability









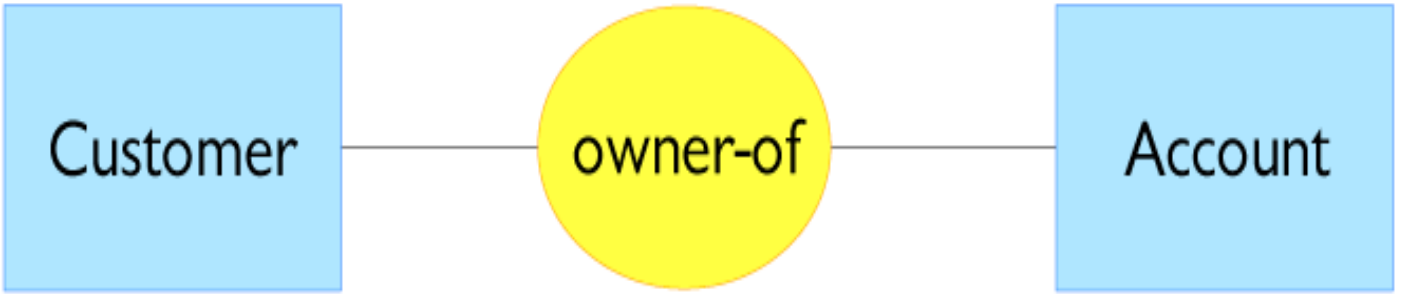


Perceptual Discriminability

should be easy to **distinguish** visual symbols

ability to distinguish is determined by **visual distance**
larger visual distance » faster, more accurate recognition

- number of visual variables on which they differ and the size of the differences
- shape is the main visual variable



Perceptual Discriminability

Software Enginering notations mostly use rectangle variants

Use redundant visual encoding to increase distance (e.g., textual + visual)

Semantic Transparency

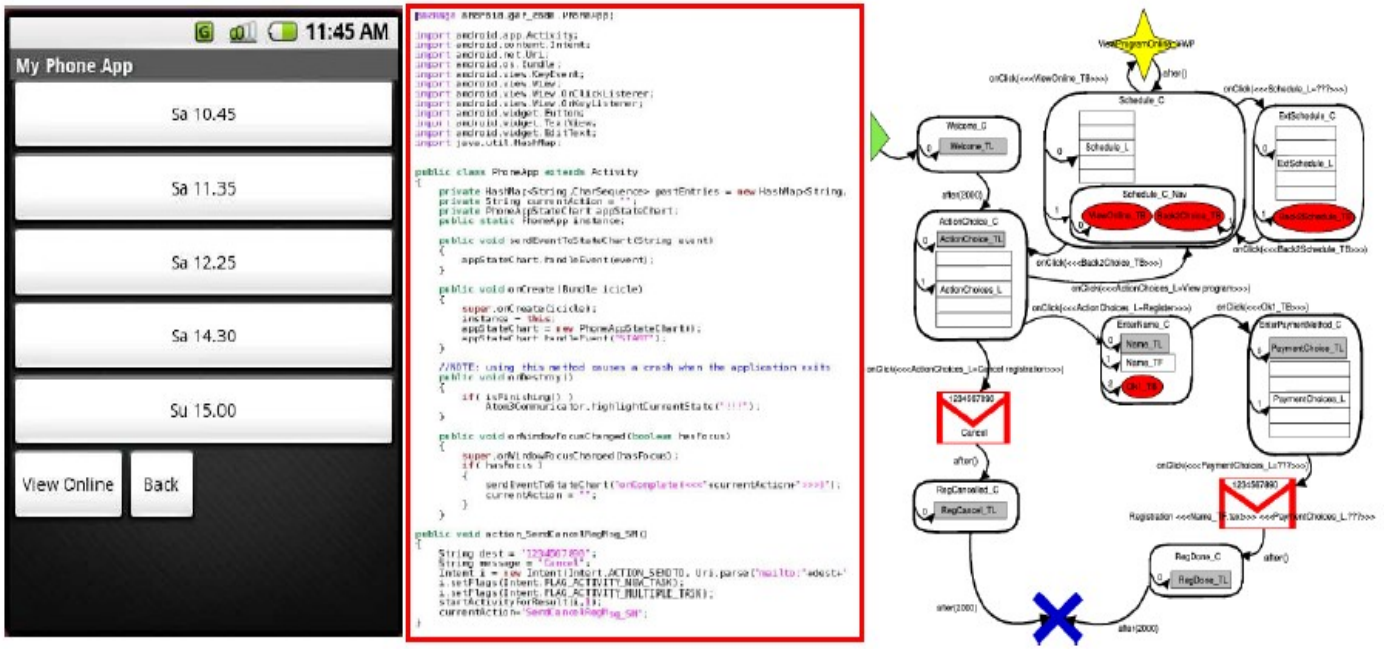
The meaning of a symbol can be inferred from its appearance (intuitive)

Symbols can be:

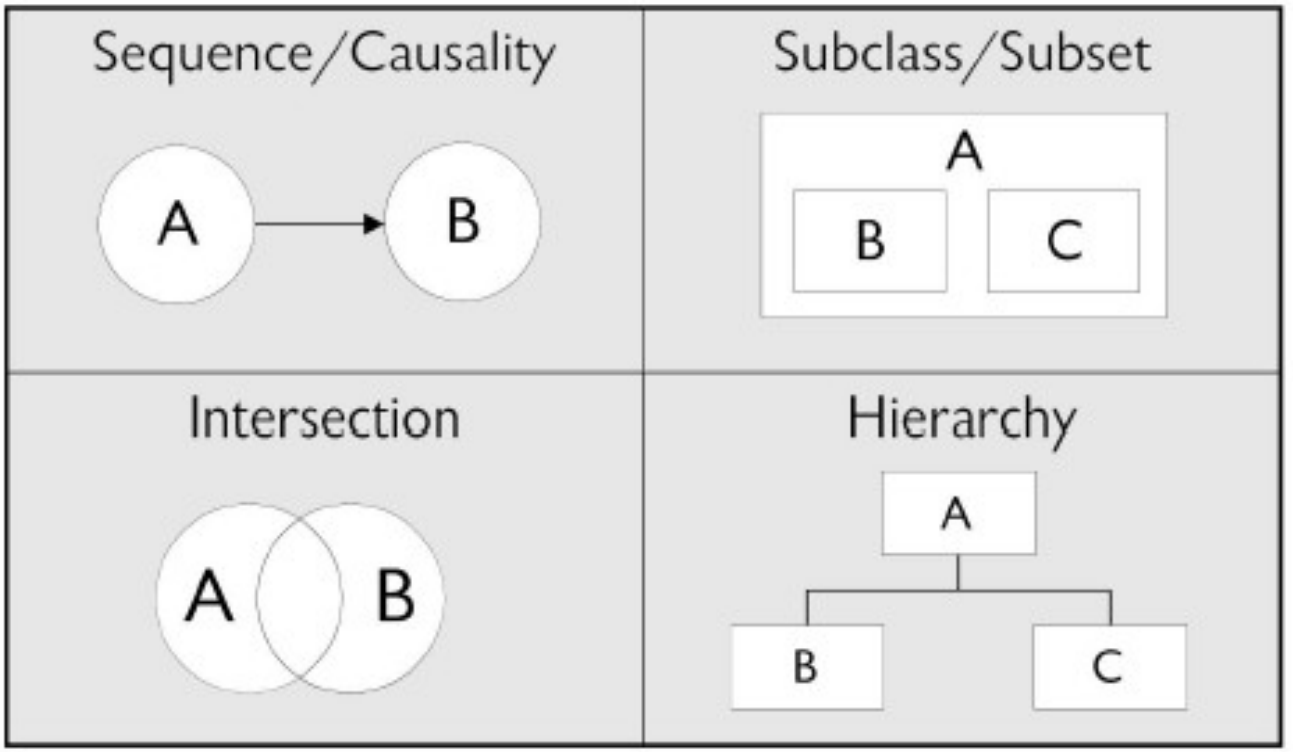
- Semantically Immediate
- Semantically Opaque
- Semantically Perverse

Software Engineering notations are usually abstract (non-intuitive)

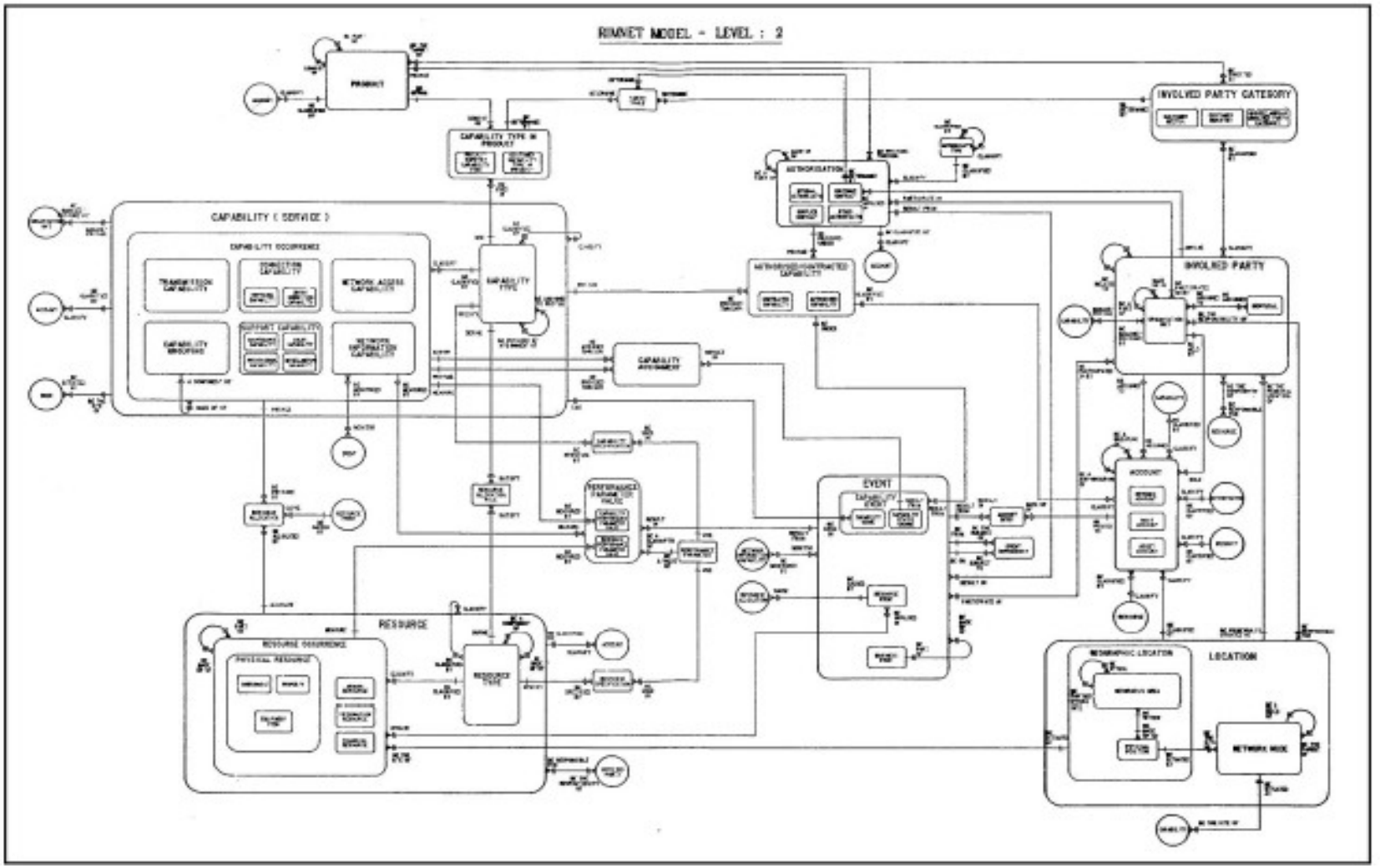
Domain-specific icons are intuitive



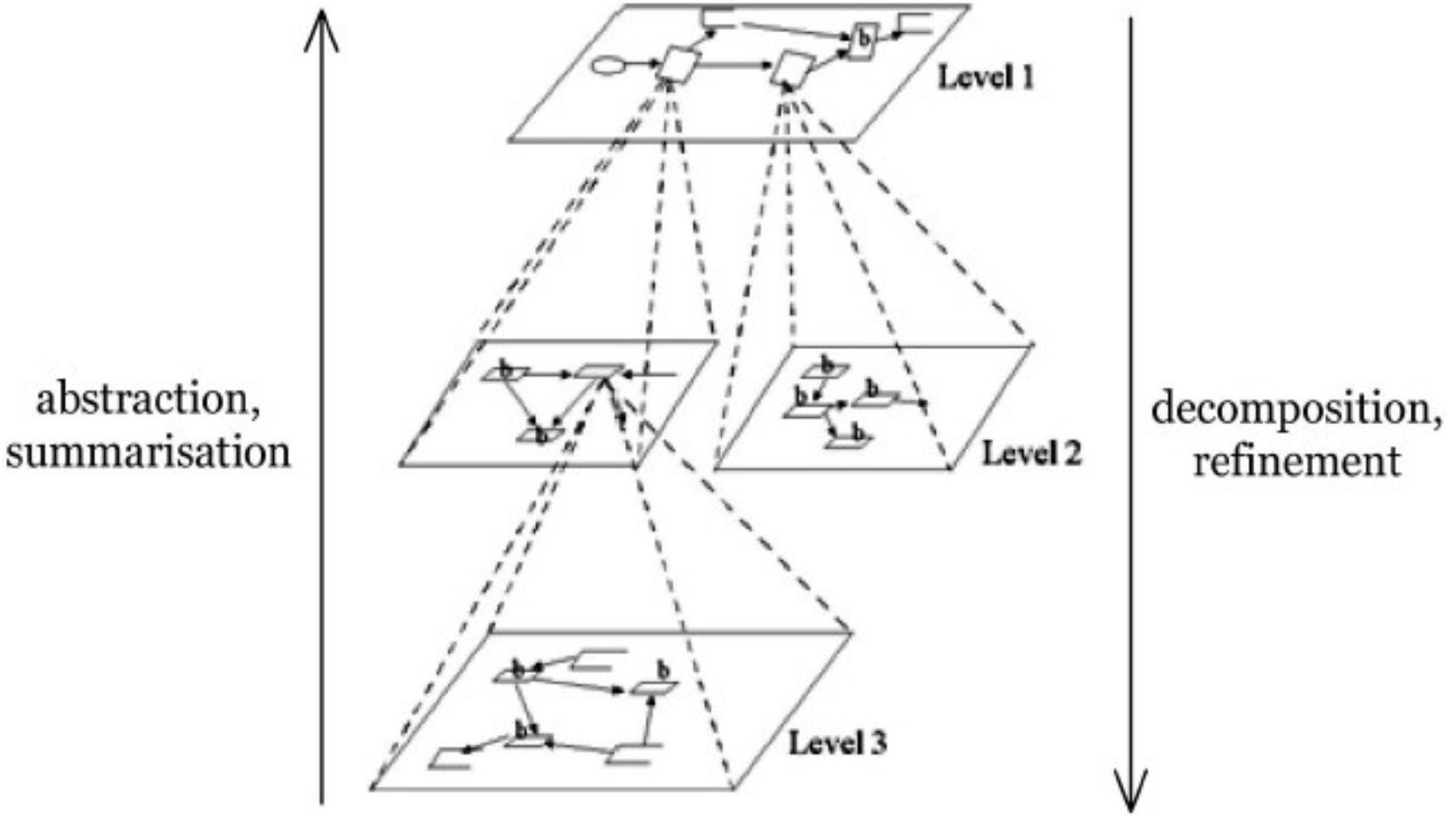
Semantic Transparency



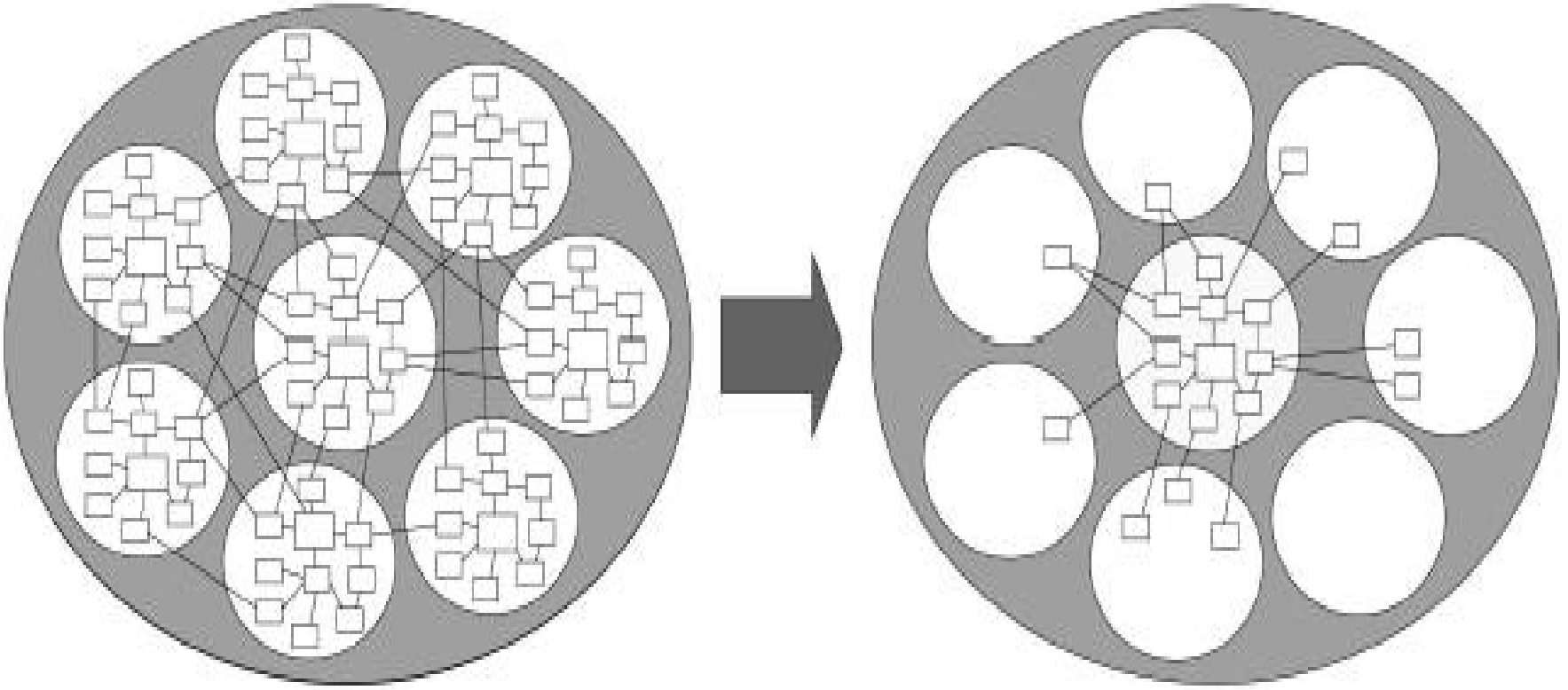
Complexity management (# elements in diagram » cognitive overload)



Modularization/Hierarchy



Cognitive Integration (different notations)

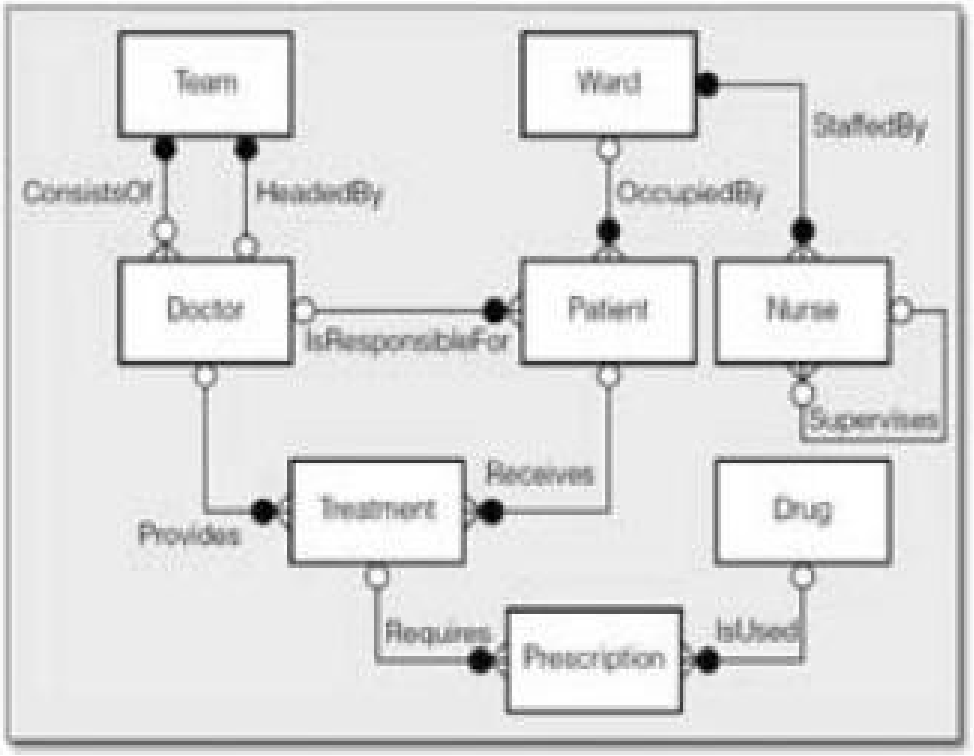


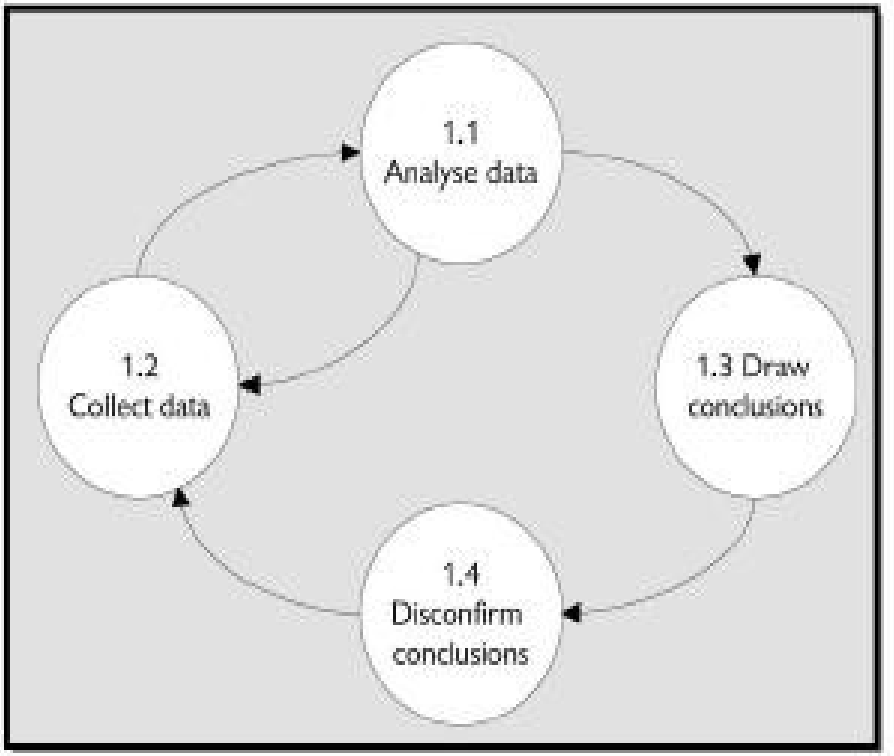
- Conceptual integration (coherent mental model)
- Enable navigation and transition between notations

Visual Expressiveness

Number of visual variables used (UML, mostly shape, no colour)

8 degrees of visual freedom (0 = non-visual – 8 = visually saturated)





Visual Expressiveness

Different visual variables have different capacity to encode information

Variable	Power	Capacity
Horizontal position (x)	Interval	10-15
Vertical position (y)	Interval	10-15
Size	Interval	20
Brightness	Ordinal	6-7
Colour	Nominal	7-10
Texture	Nominal	2-5
Shape	Nominal	Unlimited
Orientation	Nominal	4

Dual Encoding

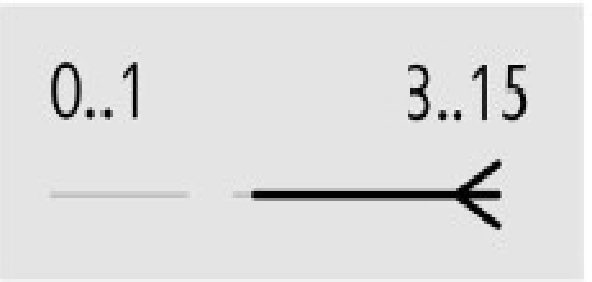
Combine Textual and Visual

Supplement rather than duplicate (e.g., cardinality values)
Reinforce meaning

Graphical encoding

Textual encoding

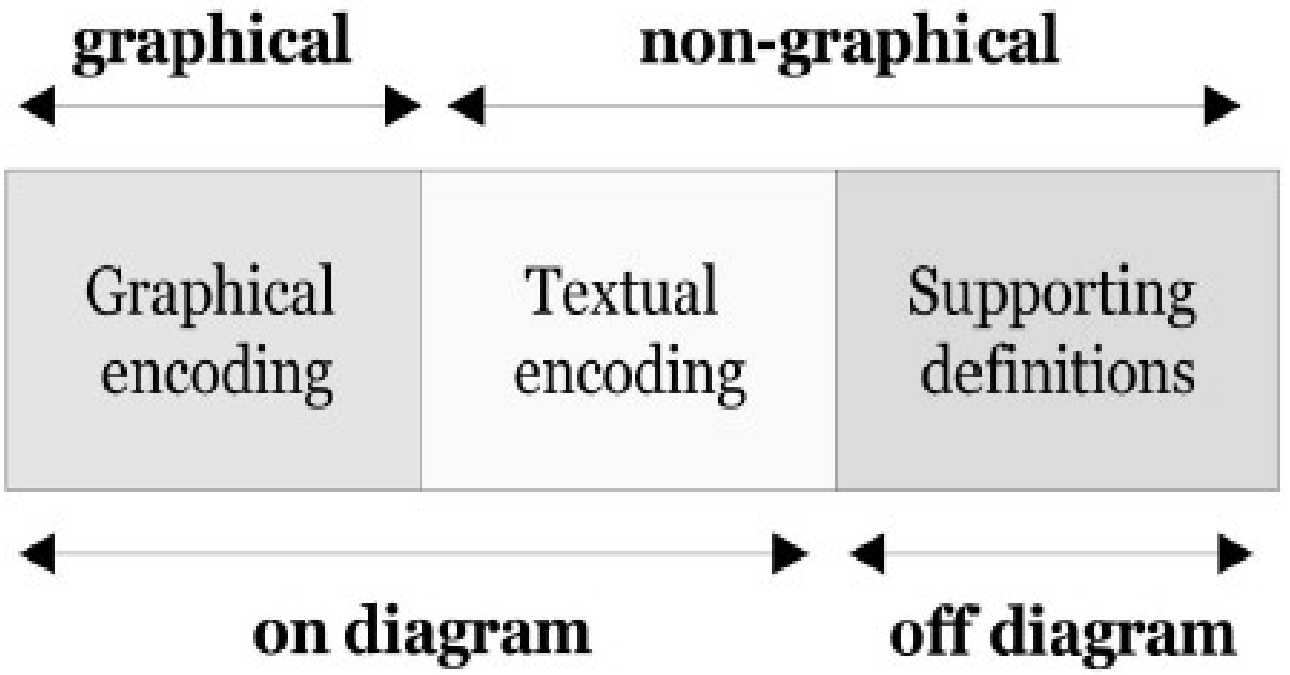
Dual coding (graphics+ text)



Graphic Economy

- Not too many symbols. If many, provide legend
- Limit on human discrimination capability (6 levels per variable)
- Upper limit on graphic complexity

How?



Cognitive Fit

Adapt choice of visual notation to
- Task
- Audience novices and experts

Adaptation may be dynamic (“learn” about Task/User proficiency)

Representation medium

Interactions among principles

	Semiotic Clarity	Perceptual Discriminability	Semantic Transparency	Complexity Management	Cognitive Integration	Visual Expressiveness	Dual Coding	Graphic Economy	Cognitive Fit
Semiotic Clarity							±		
Perceptual Discriminability						+			+
Semantic Transparency		+							±
Complexity Management							−		+
Cognitive Integration	−			+				−	
Visual Expressiveness		+						+	±
Dual Coding									+
Graphic Economy		+		+		−			+
Cognitive Fit									