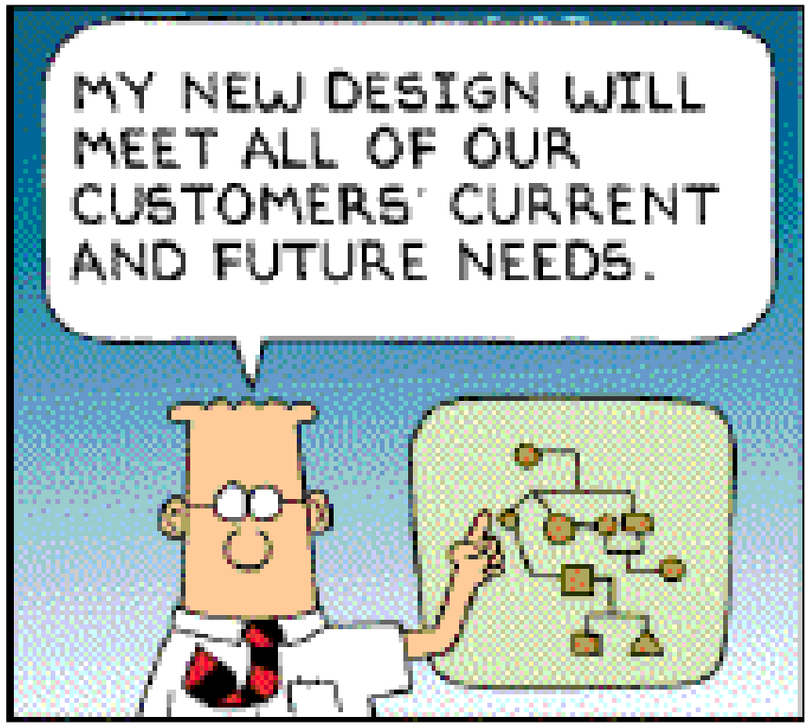


Security and reverse engineering: a prospective vision

http://stephane.ducasse.free.fr
Stéphane Ducasse

Me in a Nutshell: not a security expert

Head of RMOD team (7 permanents, 20 people)

4 years scientific deputee of Inria Lille (300 people)

Wrote several open-source books / ~ 300 articles

~ 15 K citations / H-index~56

One of the leader of the Pharo community

- http://www.pharo.org

Past core dev of Moose data and code analysis platform

- http://moosetechnology.org

Co-founder of http://www.synectique.eu

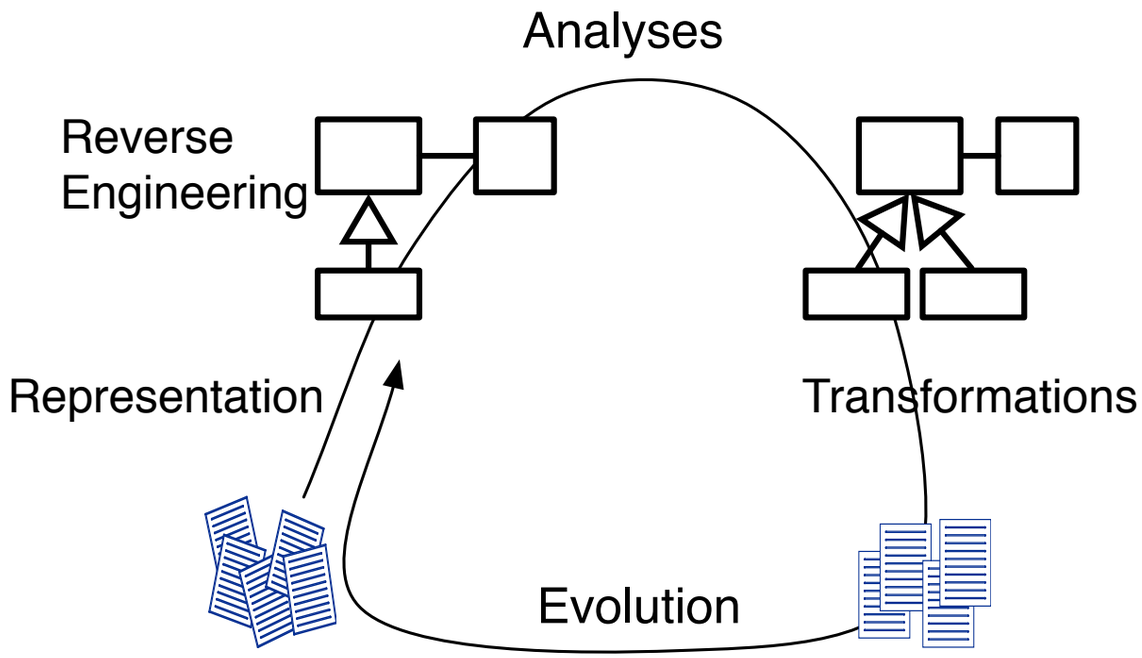
synectique
Inventive Analysis

Inria

RMoD

Bottom up team: interested in problems

code analysis, metamodeling, software metrics, program understanding, ***program visualization***, ***reverse engineering***, evolution analysis, refactorings, quality, changes analysis, commit, dependencies, merging support rule and bug assessment ***semi-automatic migration*** example-based transformations test selection, rearchitecturing blockchains, ***ui-migration***



Collaborations

IMT Douai, Soft (VUB), ENSTA (Bretagne)

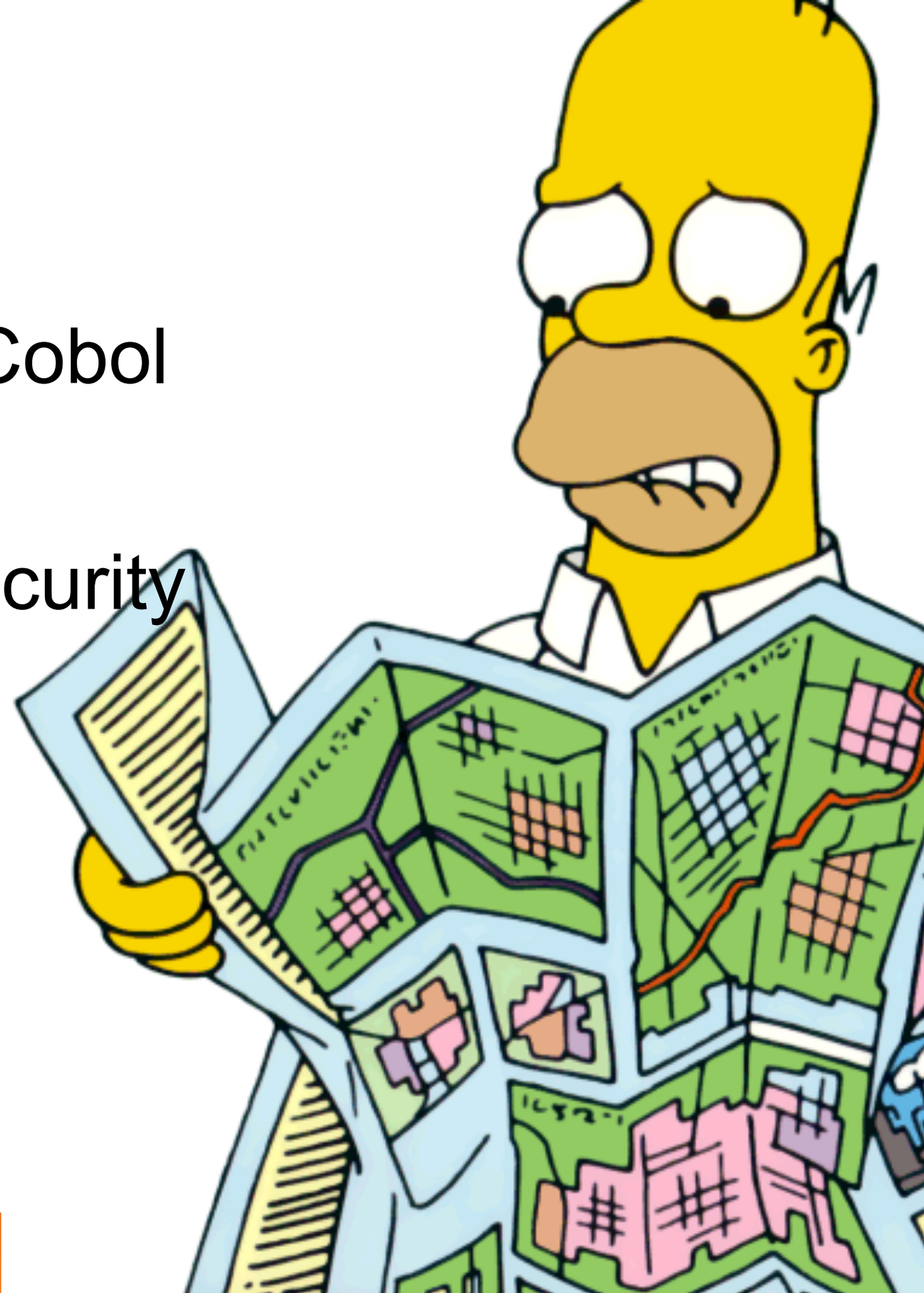
Berger-Levrault, Siemens, Thales, CIM, Arolla, Lifeware, WordLine/ATOS

Roadmap

Legacy is not just Cobol

Software Maps

Dreaming about security



Software is Complex

Laws of software evolution

Continuing change

- A program that is used in a real-world environment must change, or become progressively less useful in that environment.

Increasing complexity

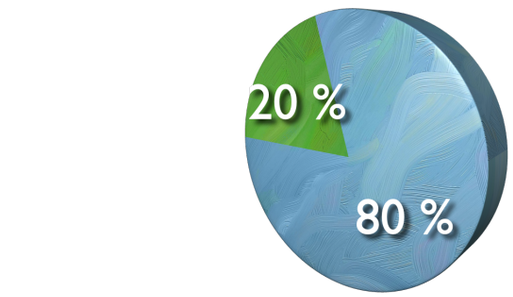
- As a program evolves, it becomes more complex, and extra resources are needed to preserve and simplify its structure.

Software is a living entity...

- Early decisions were certainly good at that time
- But the context changes
- Customers change
- Technology changes
- People change

We only maintain useful successful software

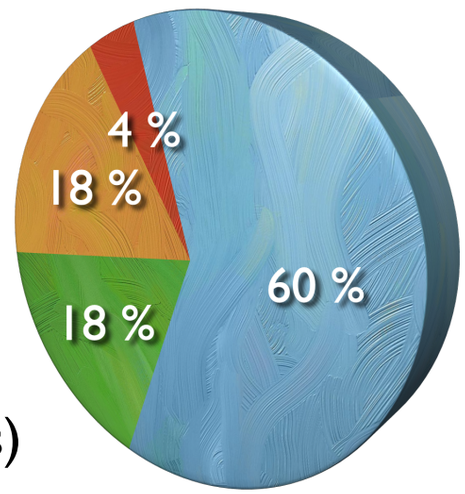
Maintenance is *continuous* Development



Between **70**% and **90**% of ***global*** effort is spent on "maintenance" !

18% Adaptive
(new platforms or OS)

18% Corrective
(fixing reported errors)

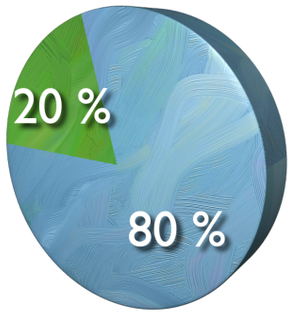


4% Other

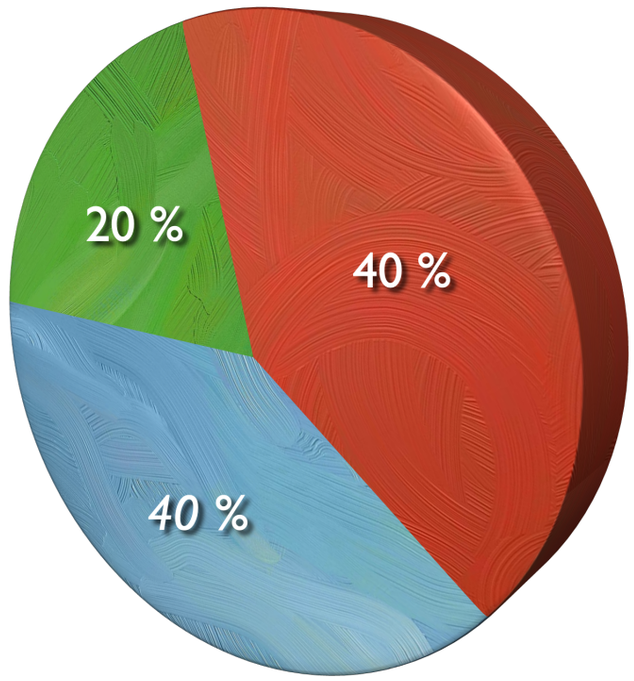
60% Perfective
(new functionality)

"Maintenance"

50% of development time is lost trying to understand code !



Between **50**% and **80**% of the ***overall cost is spent in the evolution***



We lose a lot of time with inappropriate and ineffective practices

Legacy systems exist in ***any*** language

Berger-Levrault by example

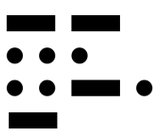
+150 produits

Plusieurs langages

Maintenabilité sur plusieurs décennies

Temps de migration

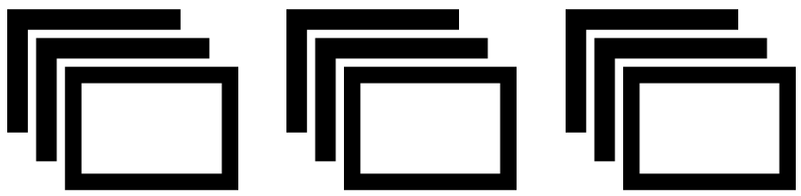
One case



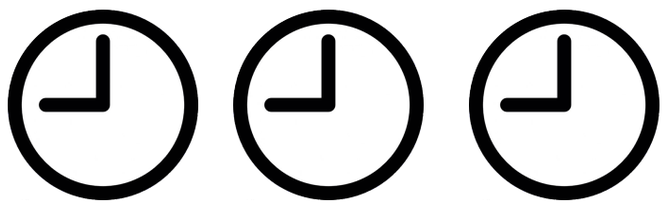
1 MLOCS

21 433 classes

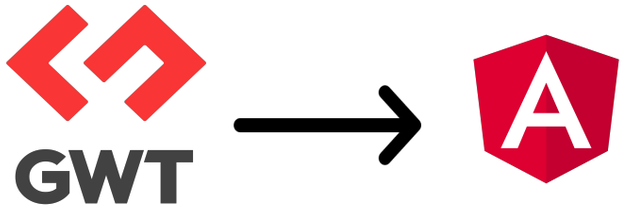
95 164 méthodes



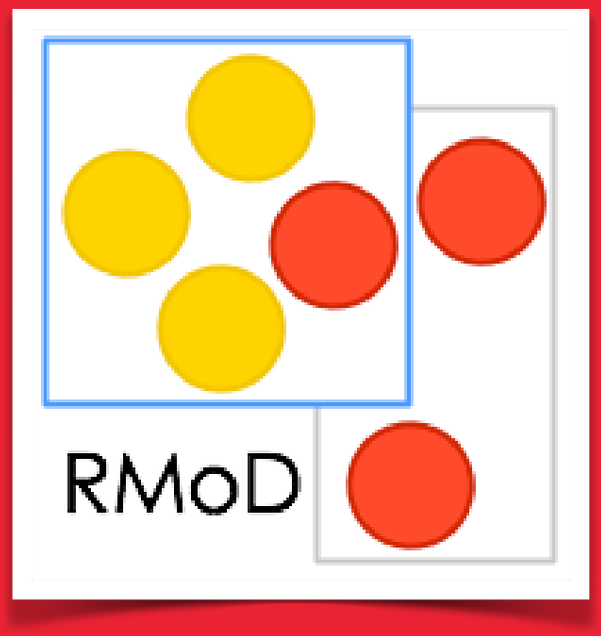
500 pages web



36 ans/homme
de migration

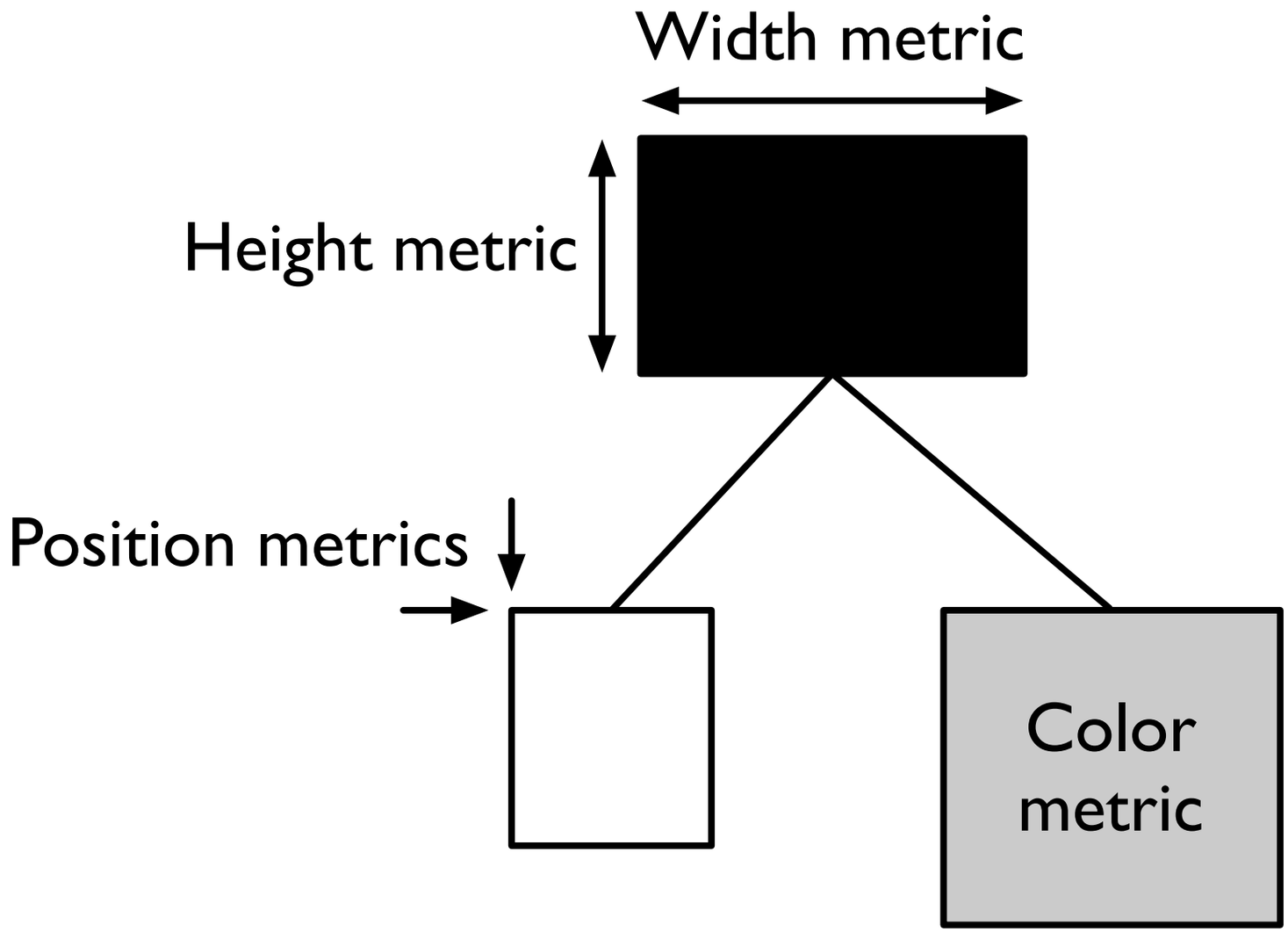


Depuis GWT vers
Angular

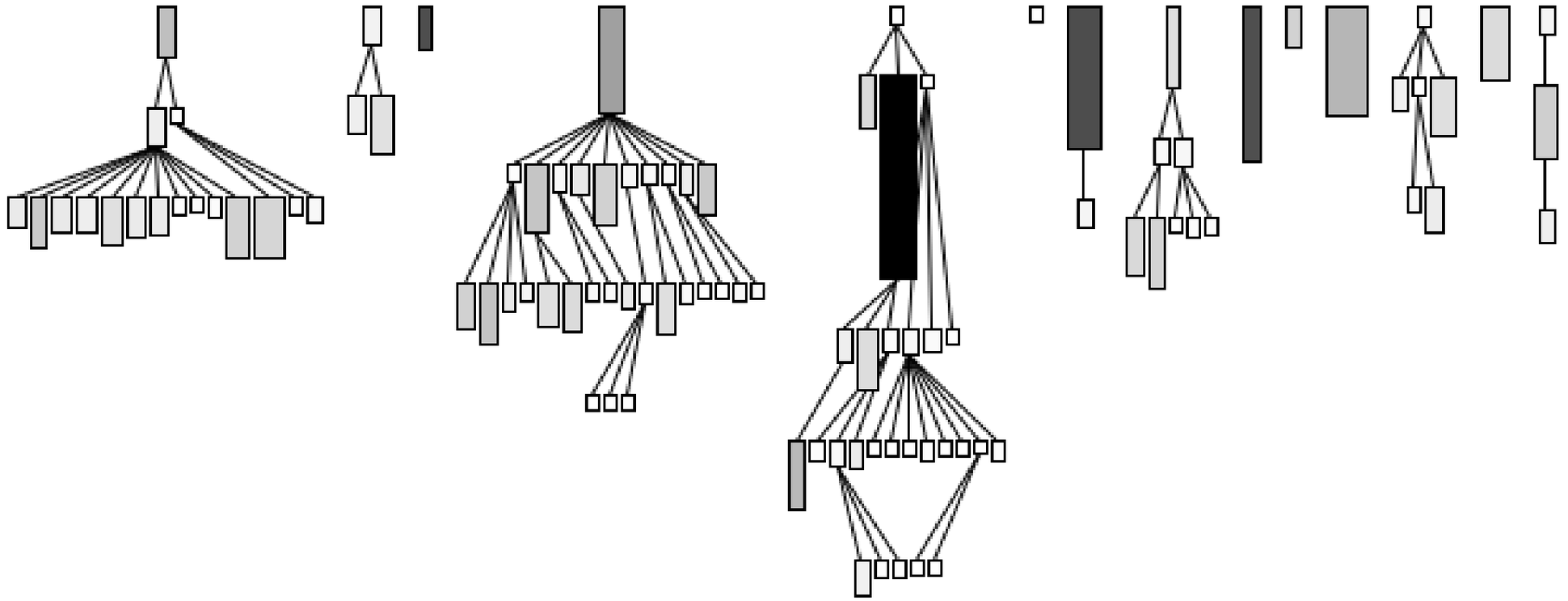


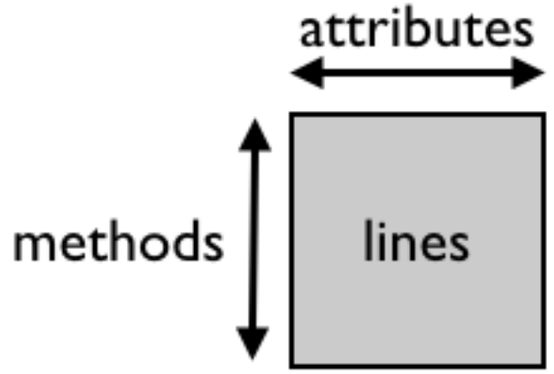
Some selected software maps — to build **yourselves** at home

First glance at large systems: Polymetric views [PhD Lanza]

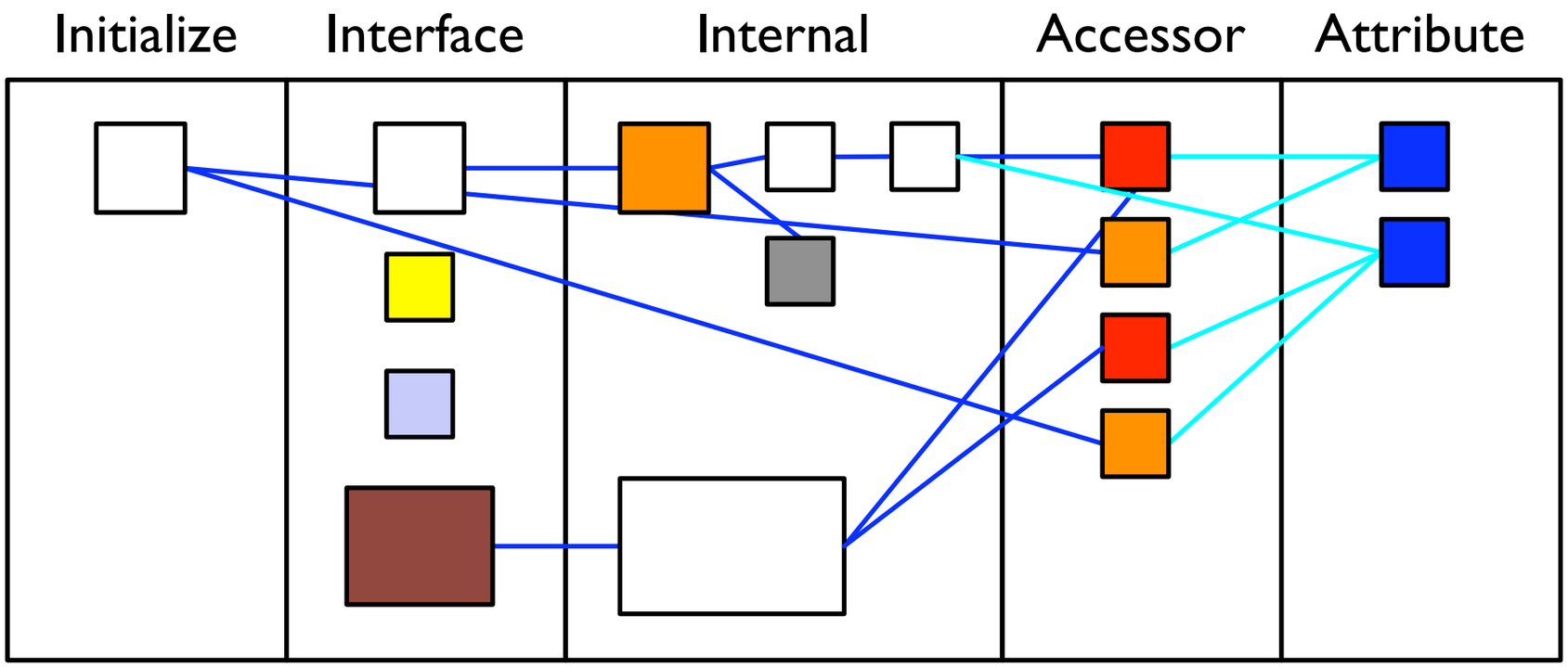


Understanding systems [PhD M. Lanza]

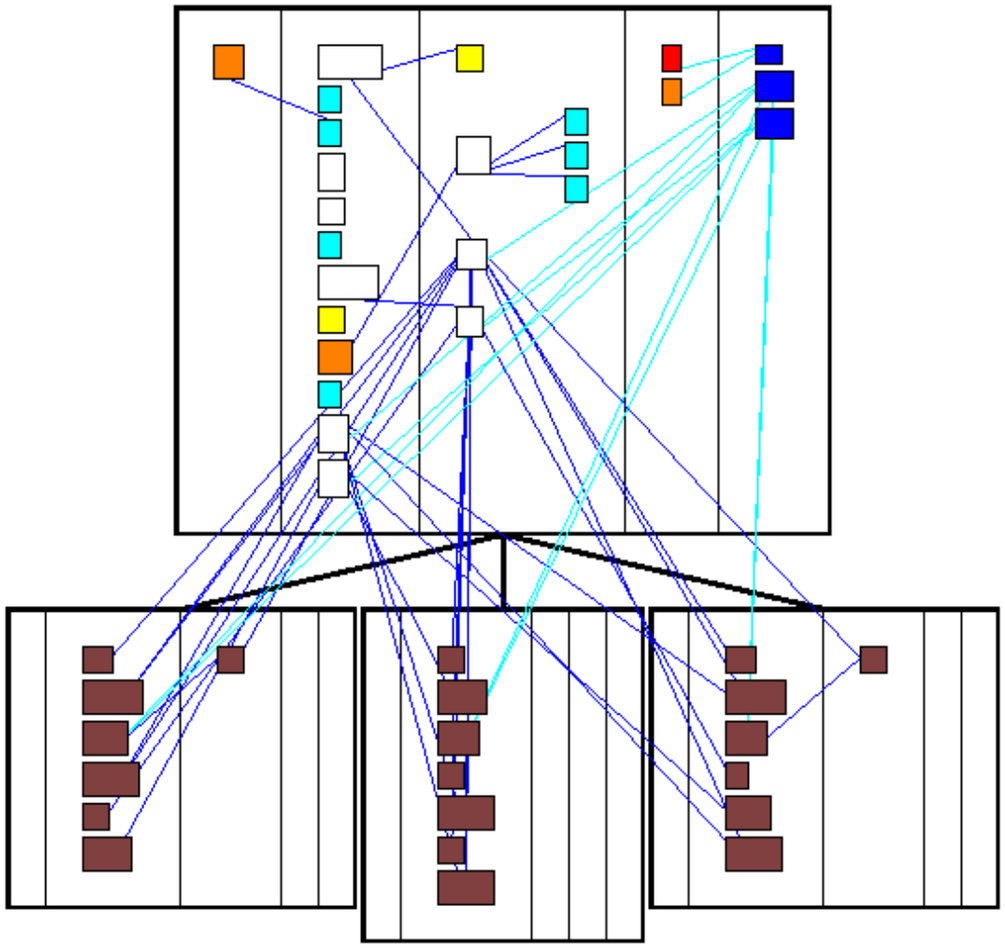


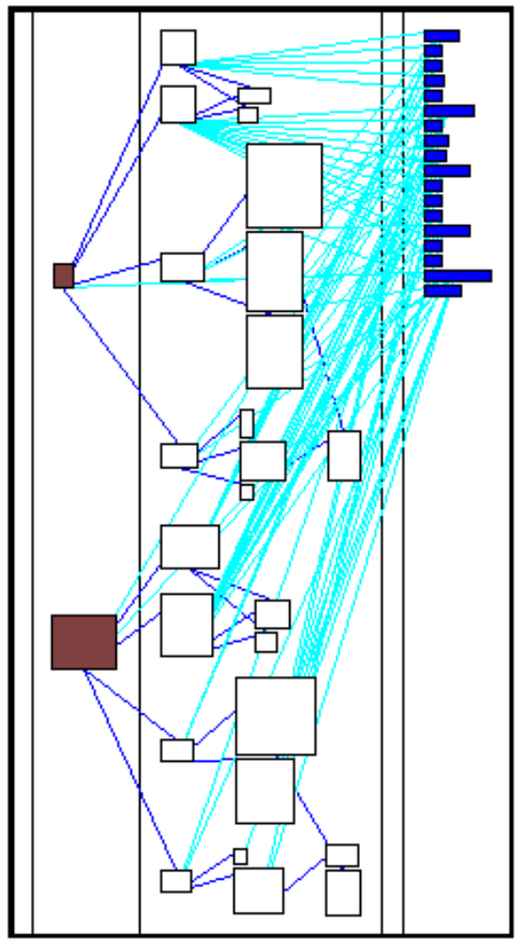


Understanding a single class [PhD M. Lanza]

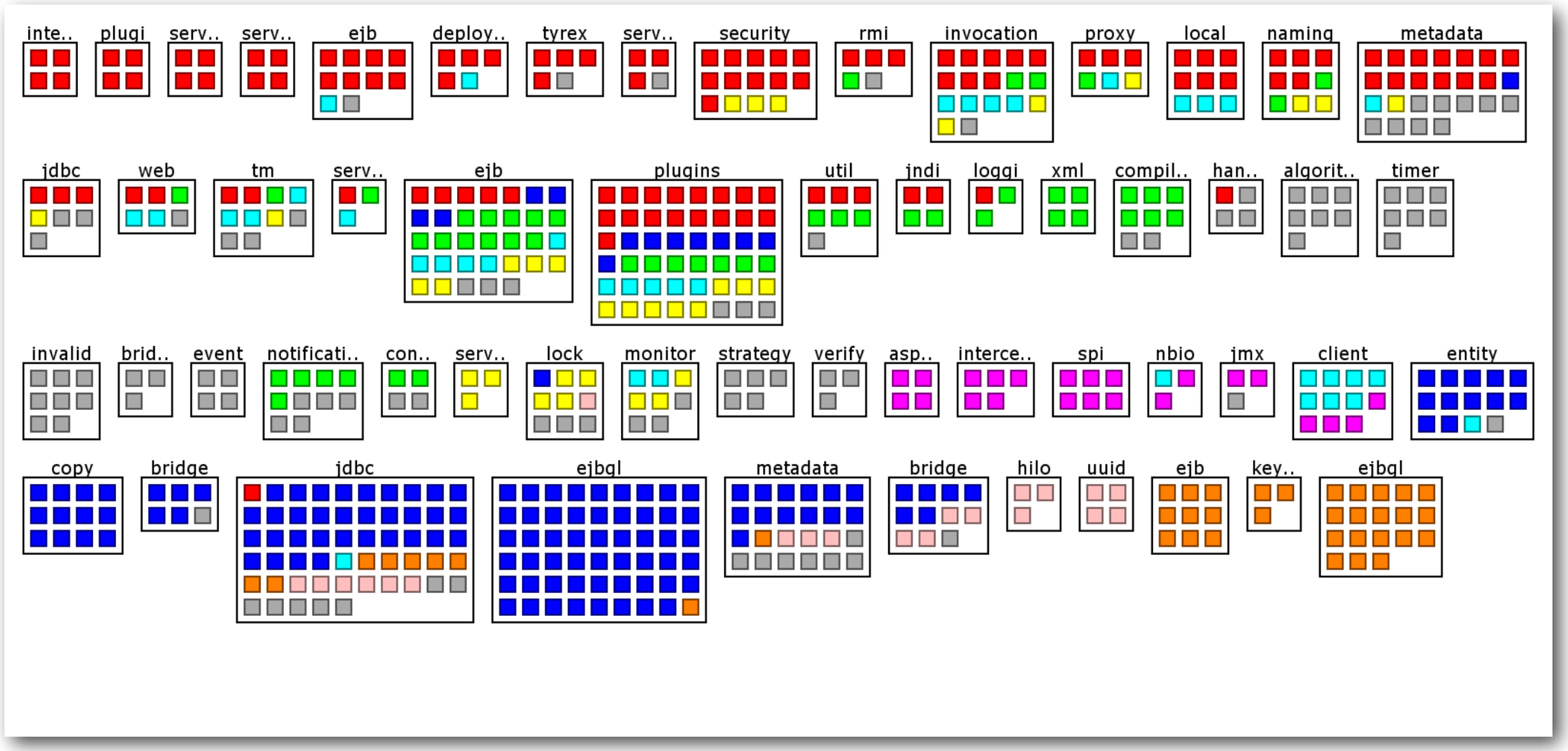


Understanding classes [PhD M. Lanza]

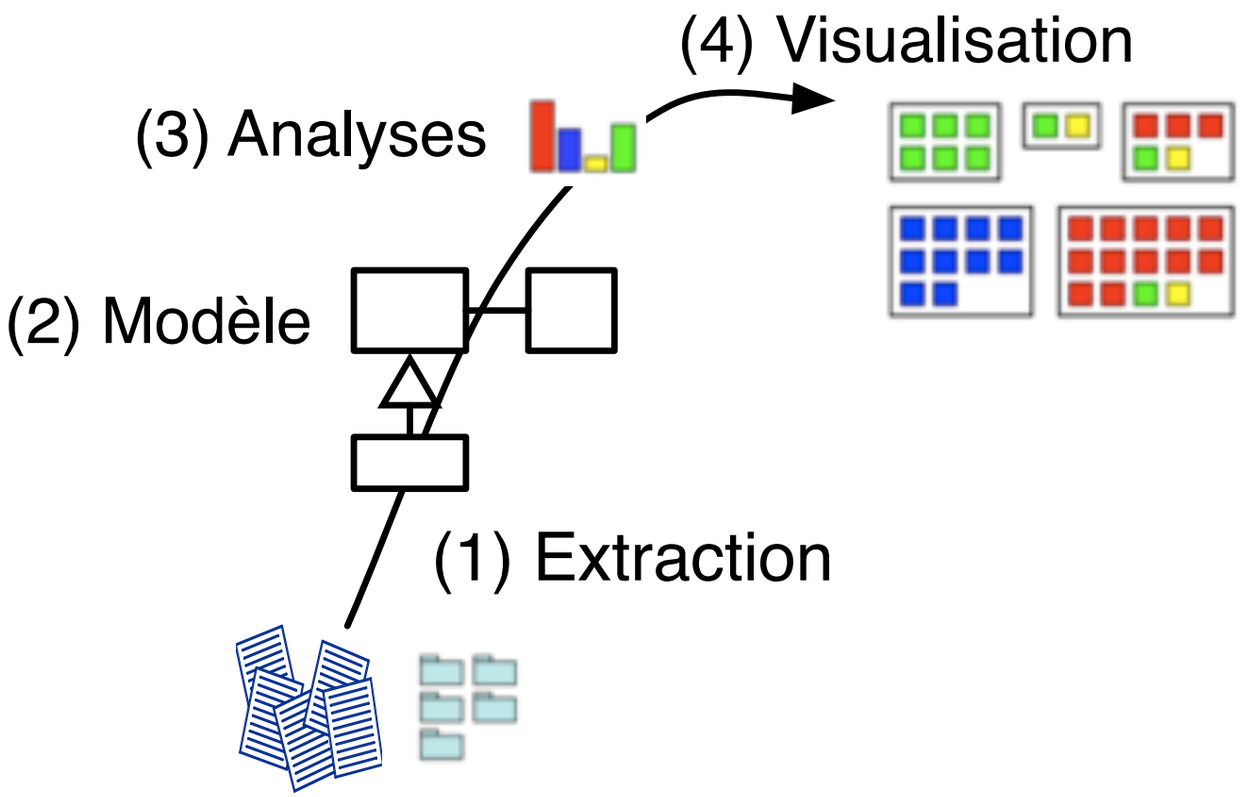




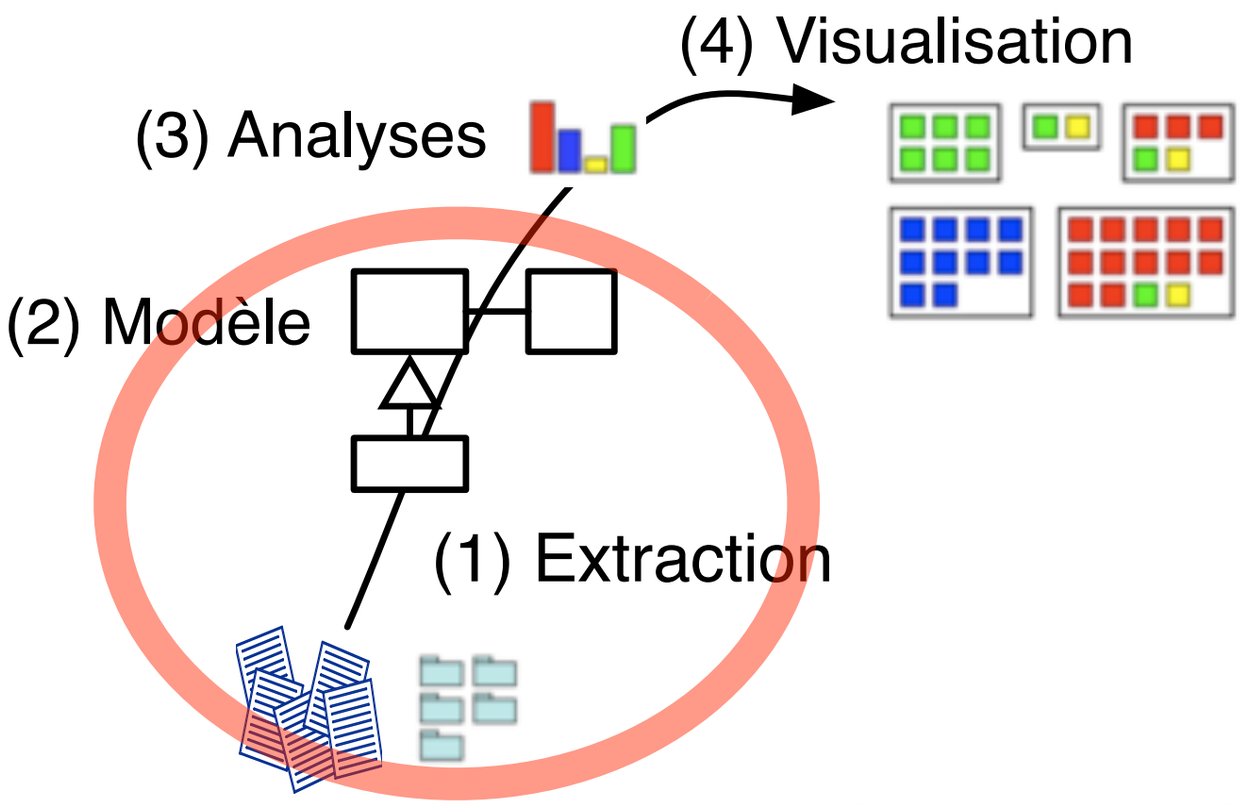
How a property spread on a system?



Example : Who is behind package X ?



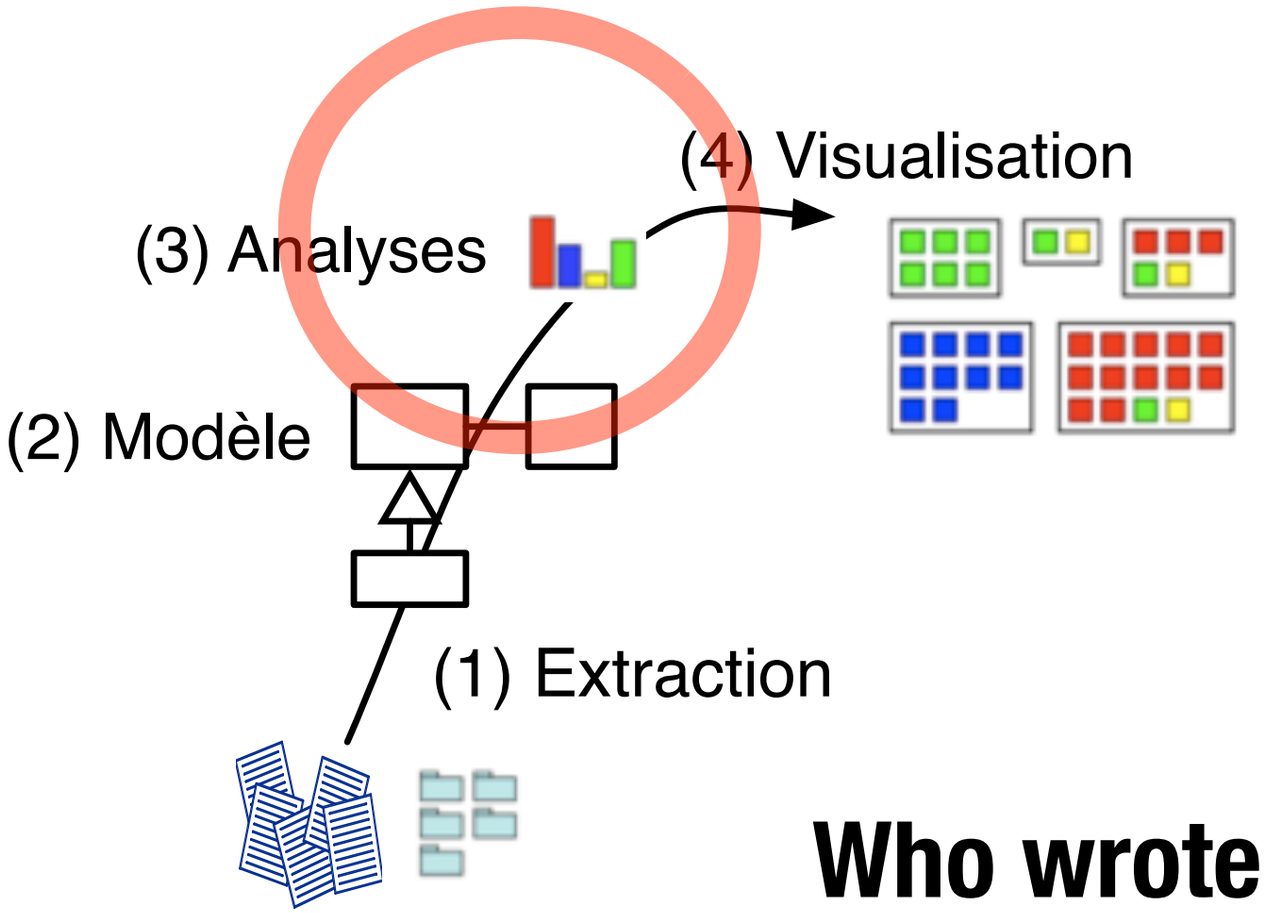
Step 1 - Model Creation/Import



Definition of a model to represent entities

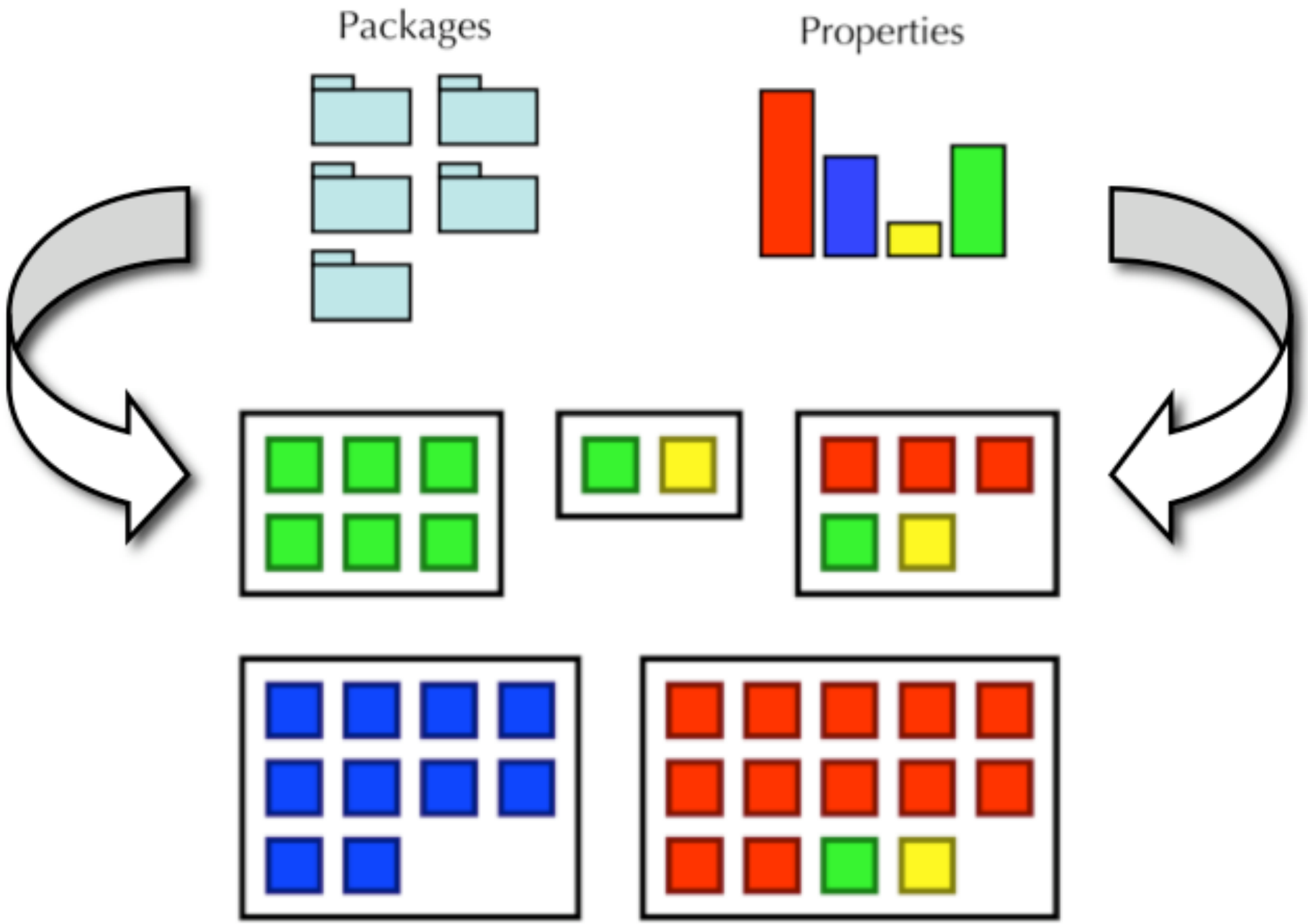
Data Extraction (CVS...)

Step 2 - Analyses



Who wrote how many lines of code?

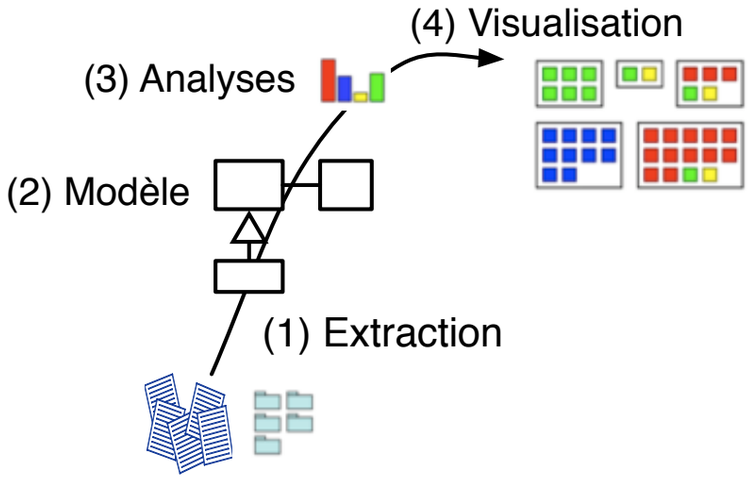
Step : 3 - Creating the Map

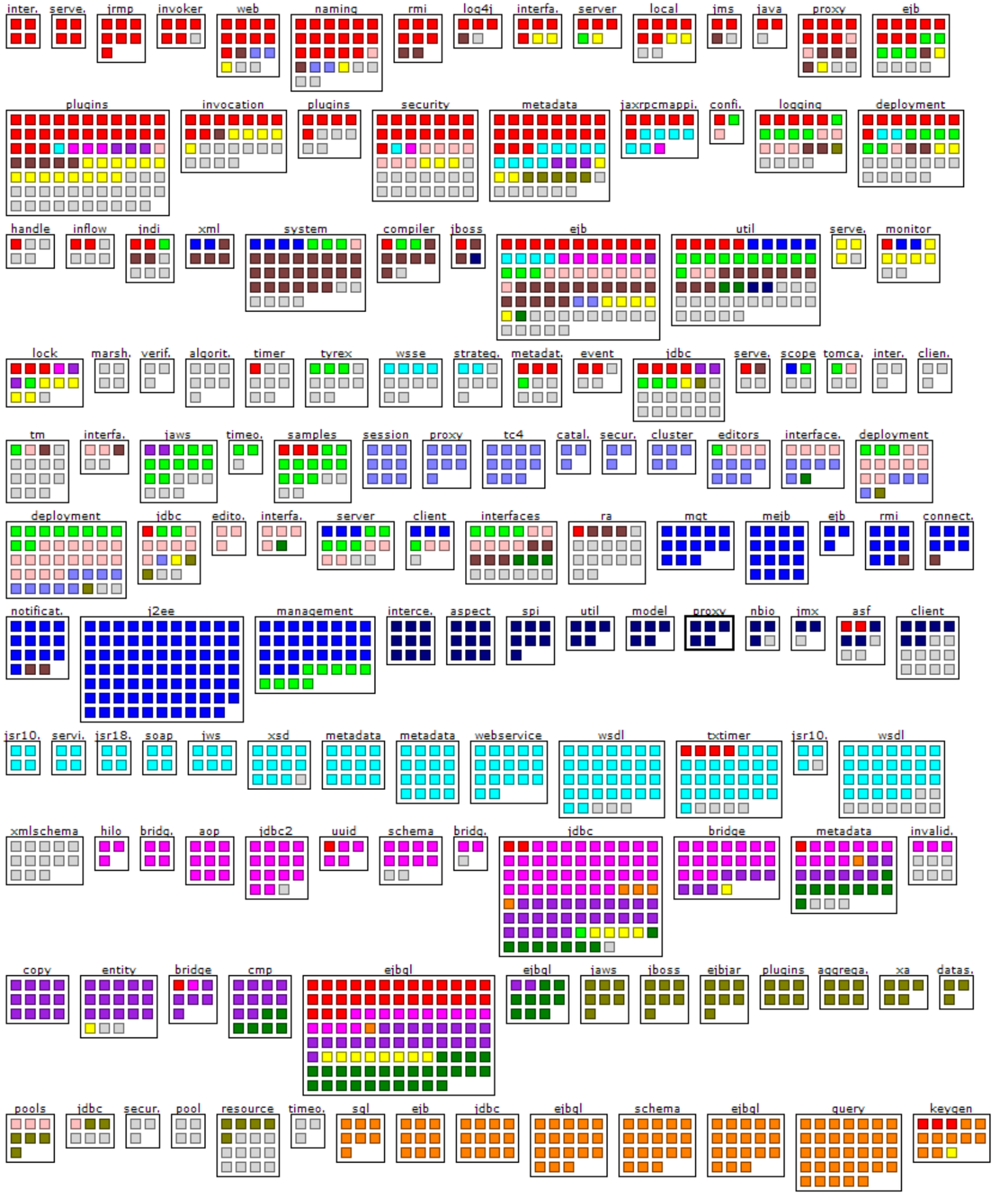


JBoss at a glance

Interactive tool

Data in perspective

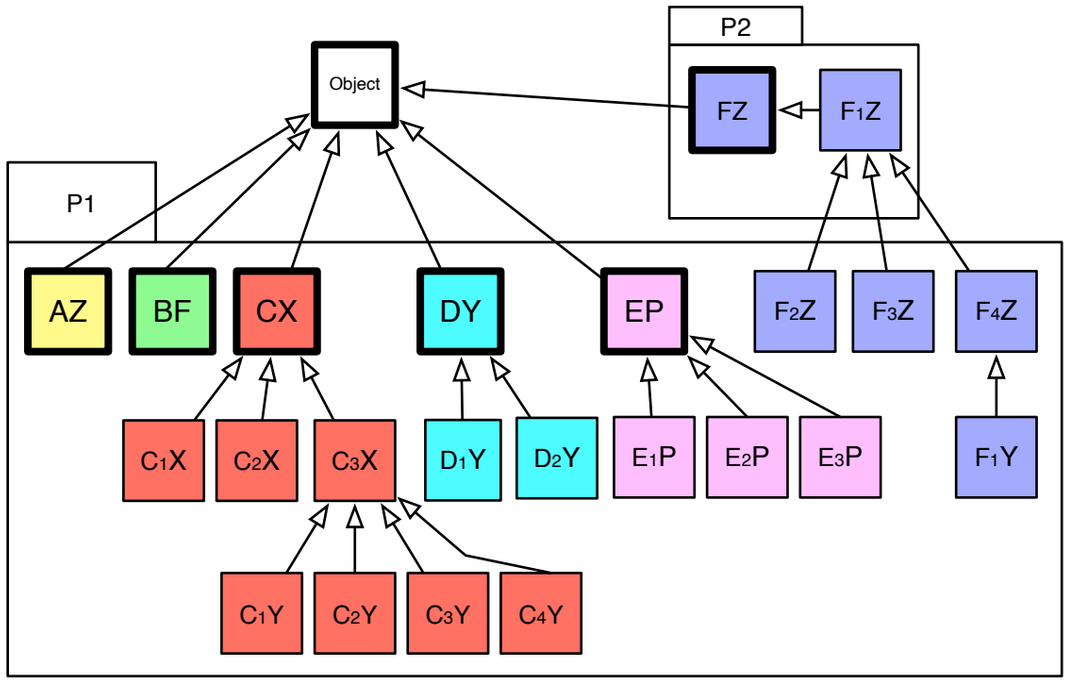




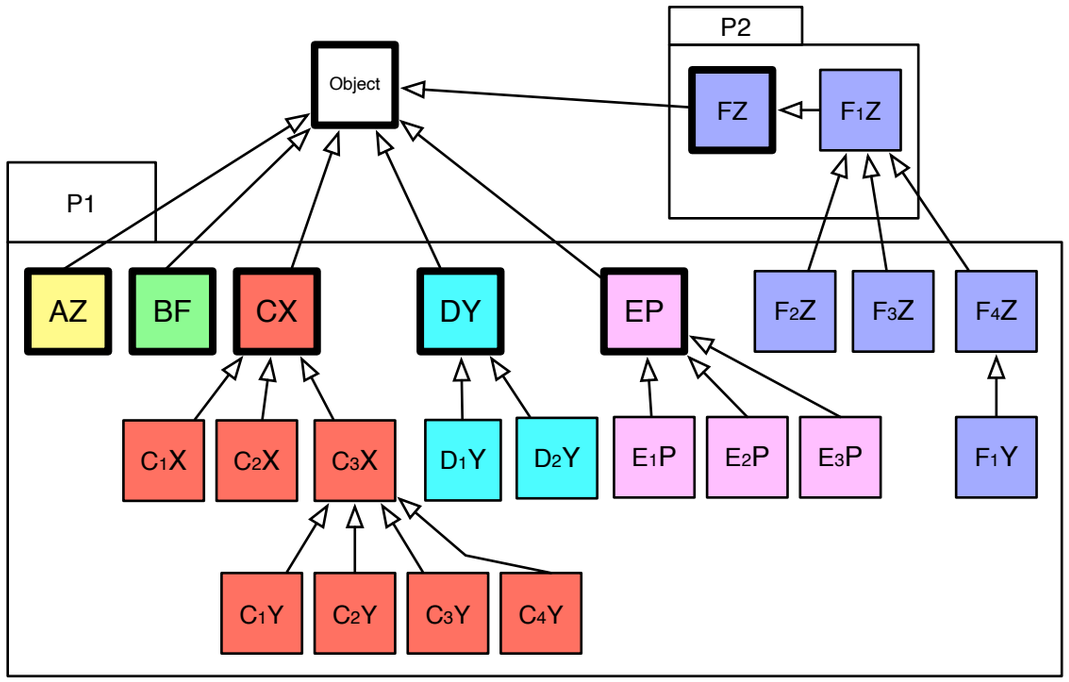
How to support understand classnames? [PhD N.-J. Agouf]

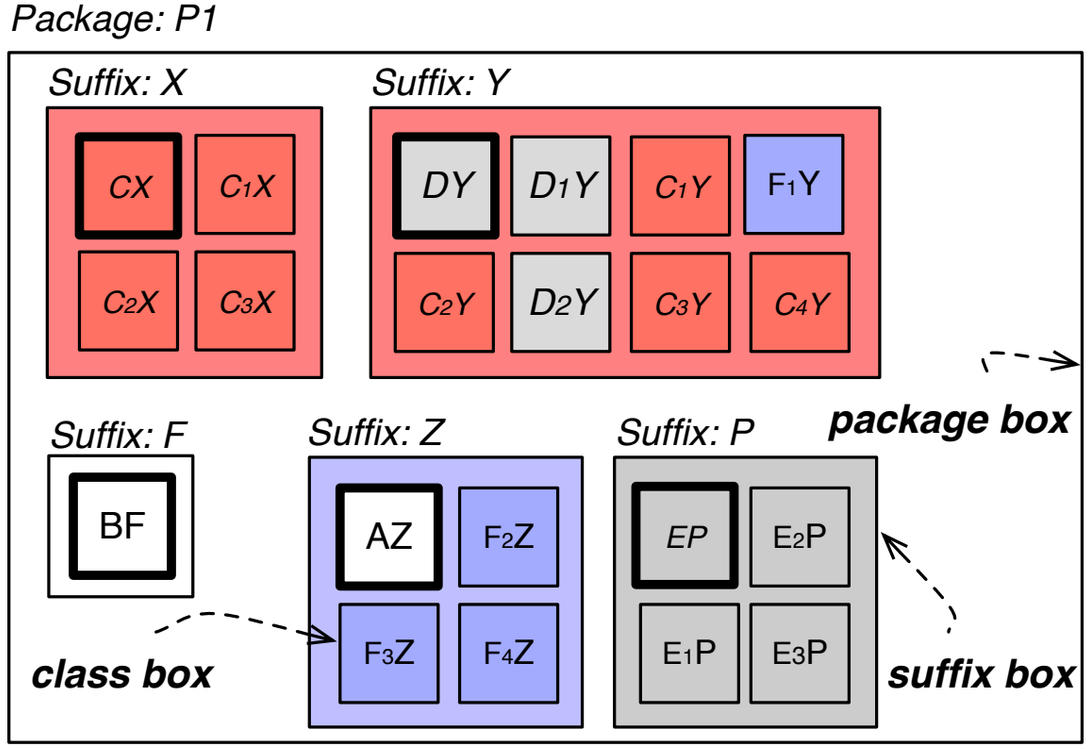
- How class are named?
  - is inheritance conveyed through names
- Is naming consistent?

How to support understand classnames? [PhD N.-J. Agouf]

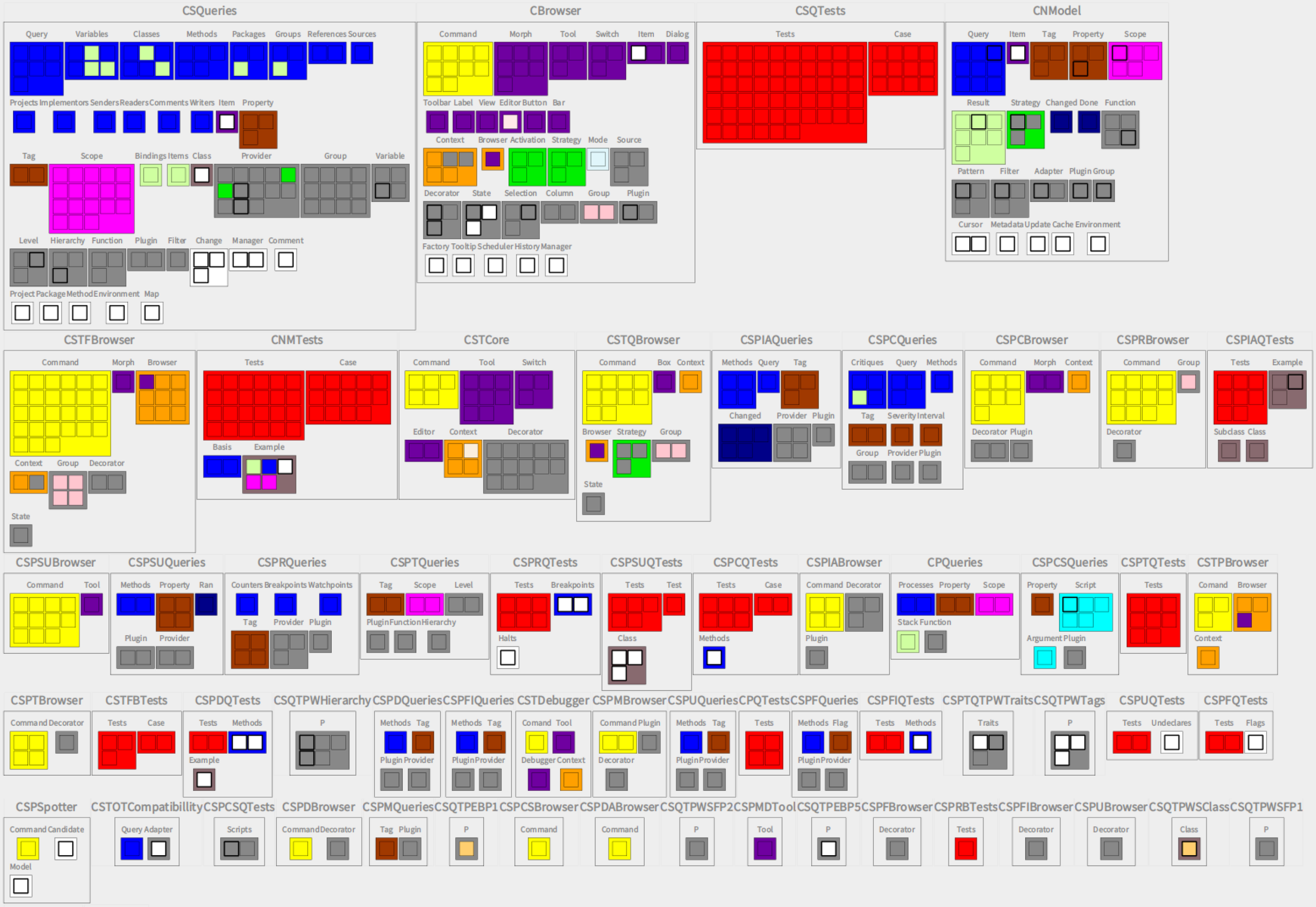


How to support understand classnames? [PhD N.-J. Agouf]

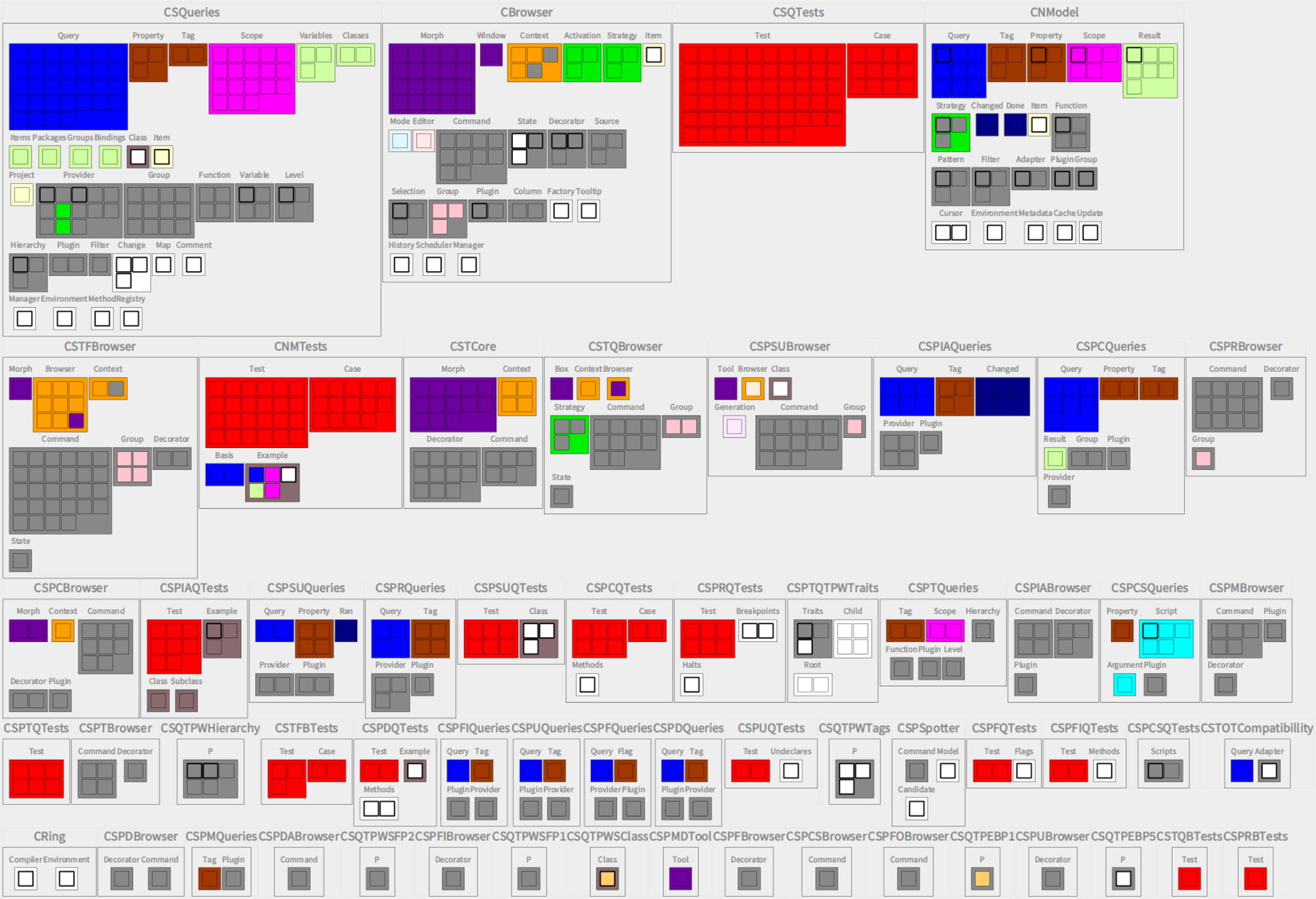




V6

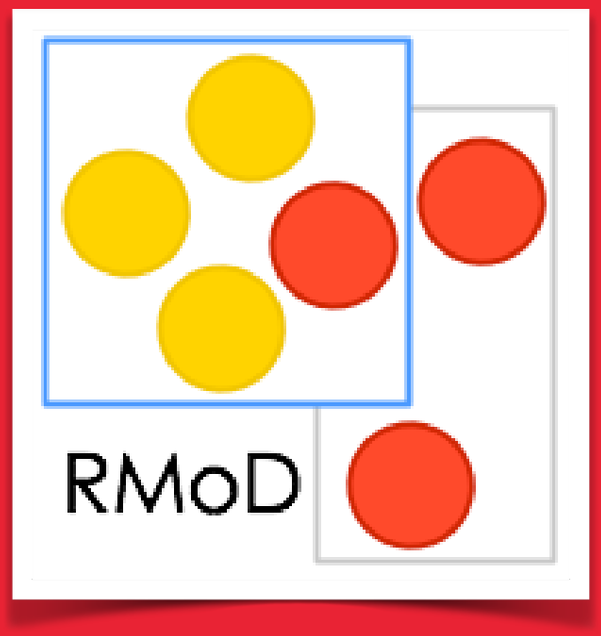

CSQueries
Query Variables Classes Methods Packages Groups References Sources
Projects Implementors Senders Readers Comments Writers Item Property
Tag Scope Bindings Items Class Provider Group Variable
Level Hierarchy Function Plugin Filter Change Manager Comment
Project Package Method Environment Map
CBrowser
Command Morph Tool Switch Item Dialog
Toolbar Label View Editor Button Bar
Context Browser Activation Strategy Mode Source
Decorator State Selection Column Group Plugin
Factory Tooltip Scheduler History Manager
CSQTests
Tests Case
CNModel
Query Item Tag Property Scope
Result Strategy Changed Done Function
Pattern Filter Adapter Plugin Group
Cursor Metadata Update Cache Environment
CSTFBrowser
Command Morph Browser
Context Group Decorator
State
CNMTests
Tests Case
Basis Example
CSTCore
Command Tool Switch
Editor Context Decorator
CSTQBrowser
Command Box Context
Browser Strategy Group
State
CSPIAQueries
Methods Query Tag
Changed Provider Plugin
CSPCQueries
Critiques Query Methods
Tag Severity Interval
Group Provider Plugin
CSPCBrowser
Command Morph Context
Decorator Plugin
CSPRBrowser
Command Group
Decorator
CSPIAQTests
Tests Example
Subclass Class
CSPSUBrowser
Command Tool
CSPSUQueries
Methods Property Ran
Plugin Provider
CSPRQueries
Counters Breakpoints Watchpoints
Tag Provider Plugin
CSPTQueries
Tag Scope Level
Plugin Function Hierarchy
CSPRQTests
Tests Breakpoints
Halts
CSPSUQTests
Tests Test
Class
CSPCQTests
Tests Case
Methods
CSPIABrowser
Command Decorator
Plugin
CPQueries
Processes Property Scope
Stack Function
CSPCSQueries
Property Script
Argument Plugin
CSPTQTests
Tests
CSTPBrowser
Comand Browser
Context
CSPTBrowser
Command Decorator
CSTFBTests
Tests Case
CSPDQTests
Tests Methods
Example
CSQTPWHierarchy
P
CSPDQueries
Methods Tag
Plugin Provider
CSPFIQueries
Methods Tag
Plugin Provider
CSTDebugger
Comand Tool
Debugger Context
CSPMBrowser
Command Plugin
Decorator
CSPUQueries
Methods Tag
Plugin Provider
CPQTests
Tests
CSPFQueries
Methods Flag
Plugin Provider
CSPFIQTests
Tests Methods
CSPTQTPWTraits
Traits
CSQTPWTags
P
CSPUQTests
Tests Undeclares
CSPFQTests
Tests Flags
CSPSpotter
Command Candidate
Model
CSTOTCompatibillity
Query Adapter
CSPCSQTests
Scripts
CSPDBrowser
Command Decorator
CSPMQueries
Tag Plugin
CSQTPEBP1
P
CSPCSBrowser
Command
CSPDABrowser
Command
CSQTPWSFP2
P
CSPMDTool
Tool
CSQTPEBP5
P
CSPFBrowser
Decorator
CSPRBTests
Tests
CSPFIBrowser
Decorator
CSPUBrowser
Decorator
CSQTPWSClass
Class
CSQTPWSFP1
P

V8


CSQueries
Query
Property
Tag
Scope
Variables
Classes
Items
Packages
Groups
Bindings
Class
Item
Project
Provider
Group
Function
Variable
Level
Hierarchy
Plugin
Filter
Change
Map
Comment
Manager
Environment
Method
Registry
CBrowser
Morph
Window
Context
Activation
Strategy
Item
Mode
Editor
Command
State
Decorator
Source
Selection
Group
Plugin
Column
Factory
Tooltip
History
Scheduler
Manager
CSQTests
Test
Case
CNModel
Query
Tag
Property
Scope
Result
Strategy
Changed
Done
Item
Function
Pattern
Filter
Adapter
Plugin
Group
Cursor
Environment
Metadata
Cache
Update
CSTFBrowser
Morph
Browser
Context
Command
Group
Decorator
State
CNMTests
Test
Case
Basis
Example
CSTCore
Morph
Context
Decorator
Command
CSTQBrowser
Box
Context
Browser
Strategy
Command
Group
State
CSPSUBrowser
Tool
Browser
Class
Generation
Command
Group
CSPIAQueries
Query
Tag
Changed
Provider
Plugin
CSPCQueries
Query
Property
Tag
Result
Group
Plugin
Provider
CSPRBrowser
Command
Decorator
Group
CSPCBrowser
Morph
Context
Command
Decorator
Plugin
CSPIAQTests
Test
Example
Class
Subclass
CSPSUQueries
Query
Property
Ran
Provider
Plugin
CSPRQueries
Query
Tag
Provider
Plugin
CSPSUQTests
Test
Class
CSPCQTests
Test
Case
Methods
CSPRQTests
Test
Breakpoints
Halts
CSPTQTPWTraits
Traits
Child
Root
CSPTQueries
Tag
Scope
Hierarchy
Function
Plugin
Level
CSPIABrowser
Command
Decorator
Plugin
CSPCSQueries
Property
Script
Argument
Plugin
CSPMBrowser
Command
Plugin
Decorator
CSPTQTests
Test
CSPTBrowser
Command
Decorator
CSQTPWHierarchy
P
CSTFBTests
Test
Case
CSPDQTests
Test
Example
Methods
CSPFIQueries
Query
Tag
Plugin
Provider
CSPUQueries
Query
Tag
Plugin
Provider
CSPFQueries
Query
Flag
Provider
Plugin
CSPDQueries
Query
Tag
Plugin
Provider
CSPUQTests
Test
Undeclares
CSQTPWTags
P
CSPSpotter
Command
Model
Candidate
CSPFQTests
Test
Flags
CSPFIQTests
Test
Methods
CSPCSQTests
Scripts
CSTOTCompatibillity
Query
Adapter
CRing
Compiler
Environment
CSPDBrowser
Decorator
Command
CSPMQueries
Tag
Plugin
CSPDABrowser
Command
CSQTPWSFP2
P
CSPFIBrowser
Decorator
CSQTPWSFP1
P
CSQTPWSClass
Class
CSPMDTool
Tool
CSPFBrowser
Decorator
CSPCSBrowser
Command
CSPFOBrowser
Command
CSQTPEBP1
P
CSPUBrowser
Decorator
CSQTPEBP5
P
CSTQBTests
Test
CSPRBTests
Test

V9


CSQueries
Property Tag Result Classes Annotation Item Project
Query Scope Group Provider Function Variable Level
Hierarchy Plugin Filter Change Registry Map Manager
Class Method Environment Comment
CBrowser
Morph Window Activation Annotation Presenter Item
Mode Editor Command Context Decorator Group
Selection State Source Column Plugin Strategy
Manager Factory Scheduler History Action Tooltip
CSQTests
Test Case
CNModel
Property Tag Result Done Changed
Item Query Scope Function Filter
Strategy Pattern Adapter Plugin Group
Cursor Update Cache Environment Metadata
CSTFBrowser
Morph Command Context
Group Decorator State
CNMTests
Test Case Result
Query Scope Example
CSTCore
Morph Decorator
Command Context
CSPRBrowser
Presenter Browser Command
Decorator Group
CSTQBrowser
Morph Command Strategy
Group State Context
CSPSUBrowser
Tool Generation Browser
Command Group Class
CSPIAQueries
Tag Changed Query
Provider Plugin
CSPCQueries
Tag Property Result
Query Group Plugin
Provider
CSPCBrowser
Morph Command Decorator
Context Plugin
CSPIAQTests
Test Mock
CSPRQueries
Tag Query
Provider Plugin
CSPSUQueries
Property Ran Query
Provider Plugin
CSPRQTests
Test Mock
Breakpoints
CSPCQTests
Test Methods
CSPTQTPWTraits
Traits Child
Root
CSPSUQTests
Test Mock
CSPTQueries
Tag Scope Level
Hierarchy Function Plugin
CSPCSQueries
Property Script
Argument Plugin
CSPIABrowser
Command Decorator
Plugin
CSPMBrowser
Command Decorator
Plugin
CSPTBrowser
Command Decorator
CSPTQTests
Test
CSQTPWHierarchy
Mock
CSTFBTests
Test
CSTQBTests
Test Mock
CSPDQTests
Test Methods
Example
CSPDQueries
Tag Plugin
Query Provider
CSPUQueries
Tag Plugin
Provider Query
CSPFQueries
Flag Provider
Query Plugin
CSPFIQueries
Tag Provider
Query Plugin
CSPFQTests
Test Flags
CSPUQTests
Test Undeclares
CSPFIQTests
Test Methods
CSPSpotter
Command Model
Candidate
CSQTPWTags
Mock
CSPDBrowser
Decorator Command
CSPMQueries
Tag Plugin
CSTOTCompatibillity
Adapter Query
CRing
Environment Compiler
CSPRBTests
Test
CSPCSQTests
Scripts
CSPFIBrowser
Decorator
CSQTPEBP1
Mock
CSPUBrowser
Decorator
CSQTPWSFP1
Mock
CSPFOBrowser
Command
CSQTPEBP5
Mock
CSQTPWSClass
Mock
CSPMDTool
Tool
CSPCSBrowser
Command
CSQTPWSFP2
Mock
CSPDABrowser
Command
CSPFBrowser
Decorator
CSPFOQueries
CSPTQTPETrait



What about security?

No solution yet but we are interested …

by your wishes

by your ideas

What are the maps you want to see?

- constructs maps
- “dangerous” expressions?

What are the maps you want to see?

- input places?

What are the maps you want to see?

- previous bugs?
- places not covered by test?
- buggy places covered by tests?

What are the maps you want to see?

- domains?
- symbols used

Code as a database

What are the queries you would like to do to spot problem?

What properties such query engine should have?

Let us dream a bit more…

Can we have security aware refactorings?

Can I refactor a piece of code without breaking a non-functional requirement?

- concurrency
- speed
- security

Ready to collaborate

Interested

- Software Maps for security
- ANR proposal around “qualisecure” (we wrote one already)
- Security-aware refactoring
- And your problems