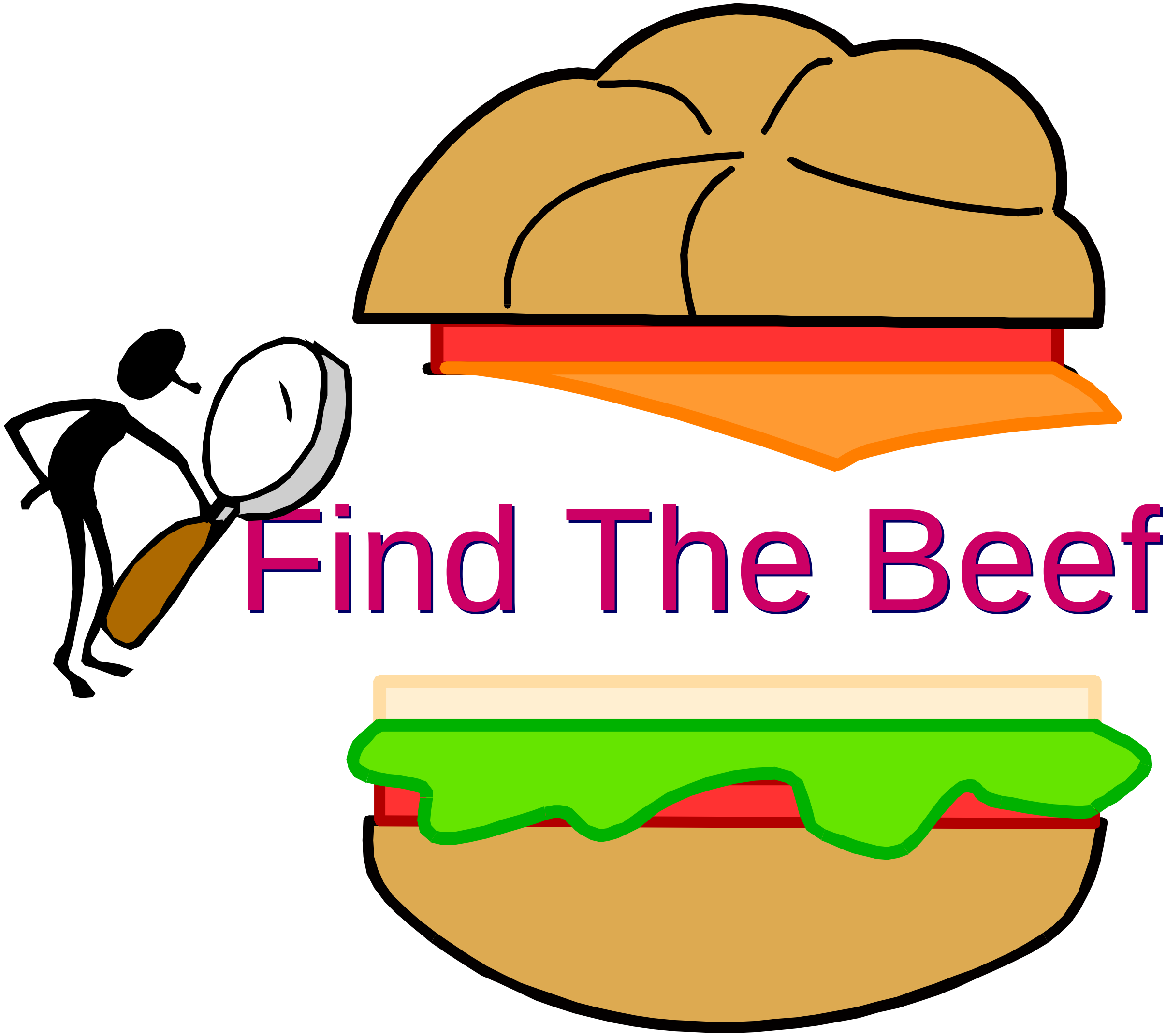
Architecture in UML



Find The Beef?

Desmond D’Souza

www.kinetium.com

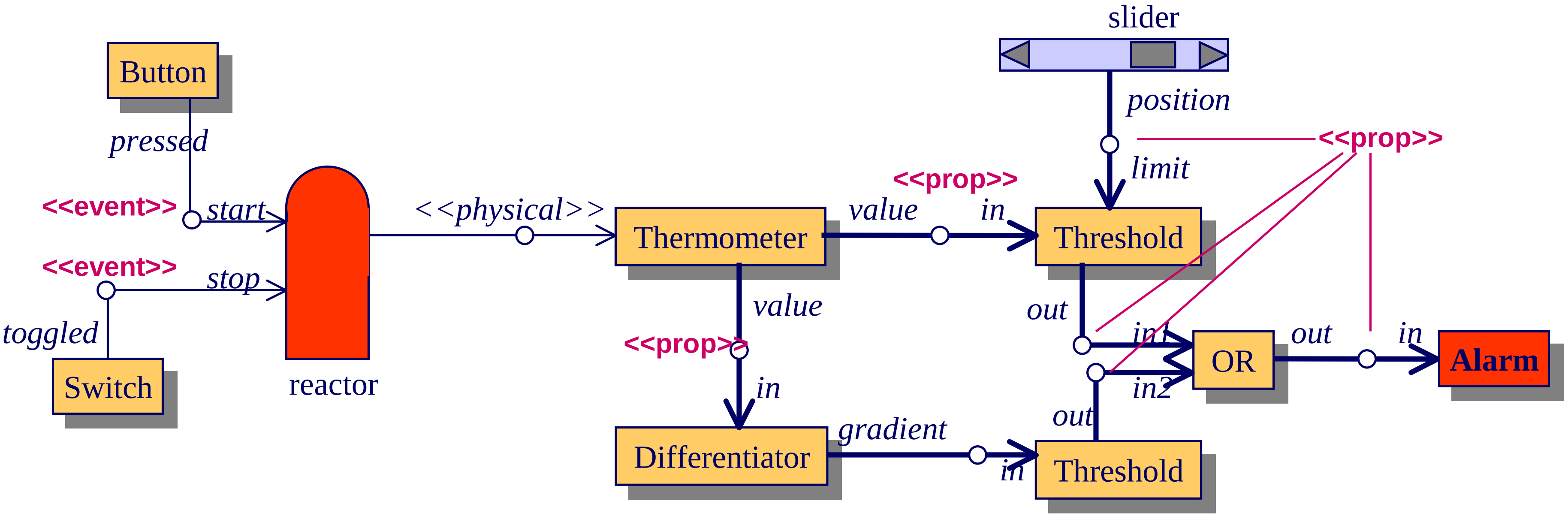
The Question

- Is UML an **architecture description language**?

- We have always distinguished …
  - ✓ Can you do objects in C ?
  - ✓ Is C an object-oriented language ?

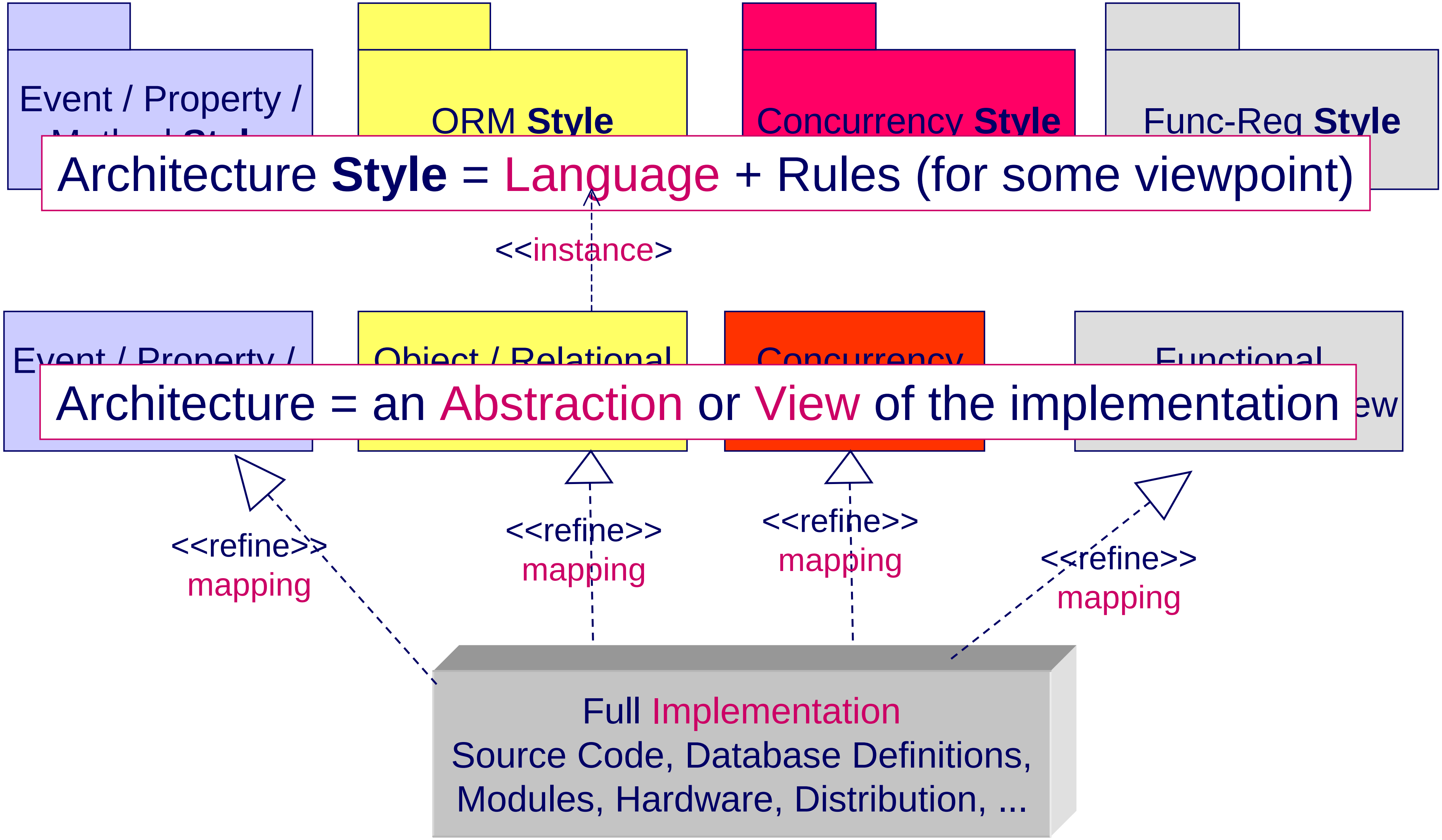
- We can do some architecture descriptions in UML
  - ✓ That does not mean it **is** an ADL

Is This an “Architecture”?



- This is an abstract view of the implementation
  - ✓ It uses the language of properties, events, methods
    - ✓ … and of connectors between these “connection points”
  - ✓ It has a mapping to Java code patterns i.e. a refinement
- This design is an instance of the Java Beans style: design + code
- These cannot be effectively and precisely described in UML

Architecture as **View** based on **Style**



Event / Property /

ORM **Style**

Concurrency **Style**

Func-Req **Style**

Architecture **Style** = Language + Rules (for some viewpoint)

<<instance>>

Event / Property /

Object / Relational

Concurrency

Functional

ew

Architecture = an Abstraction or View of the implementation

<<refine>> mapping

<<refine>> mapping

<<refine>> mapping

<<refine>> mapping

Full Implementation
Source Code, Database Definitions,
Modules, Hardware, Distribution, ...

Arch Style as Constraint on Refinement

- Style keeps designer from “needless creativity”
  - ✓ Spectrum of 0% to 100% constraining

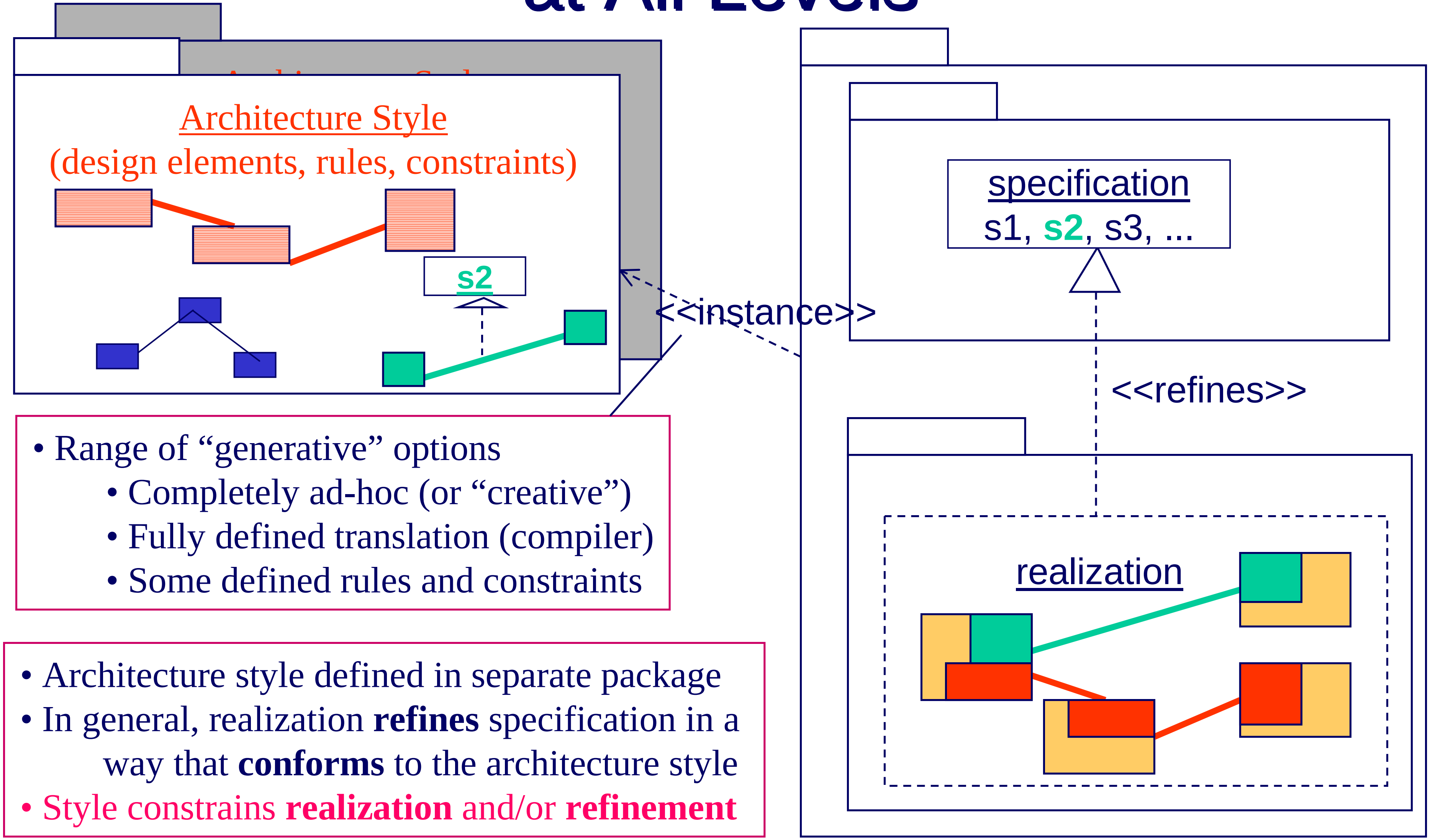
- In general, a style will
  - ✓ Focus on a viewpoint or set of concerns
  - ✓ Define a language for expressing a design
  - ✓ Specify a set of conforming spec / design pairs

- Three sample data points
  - ✓ Totally no constraints on design
  - ✓ Design language but no refinement constraints
  - ✓ Full translation scheme from spec to design

Varying Degree of Generative Style

- Architectural styles to keep 2 attributes in sync
  - ✓ Style 0: “The Cowboy” - do it any way you want
  - ✓ Style 1: “2 copies + update protocol” construct
  - ✓ Style 2: “1 copy in shared memory” construct
  - ✓ Style 3: both Style 1 and Style 2 available
  - ✓ Style 4: *Whenever* you have a **requirement** to keep 2 attributes in sync with each other, *use* the “2 copies + update protocol” **design**

Catalysis - Architecture and Style in UML at All Levels



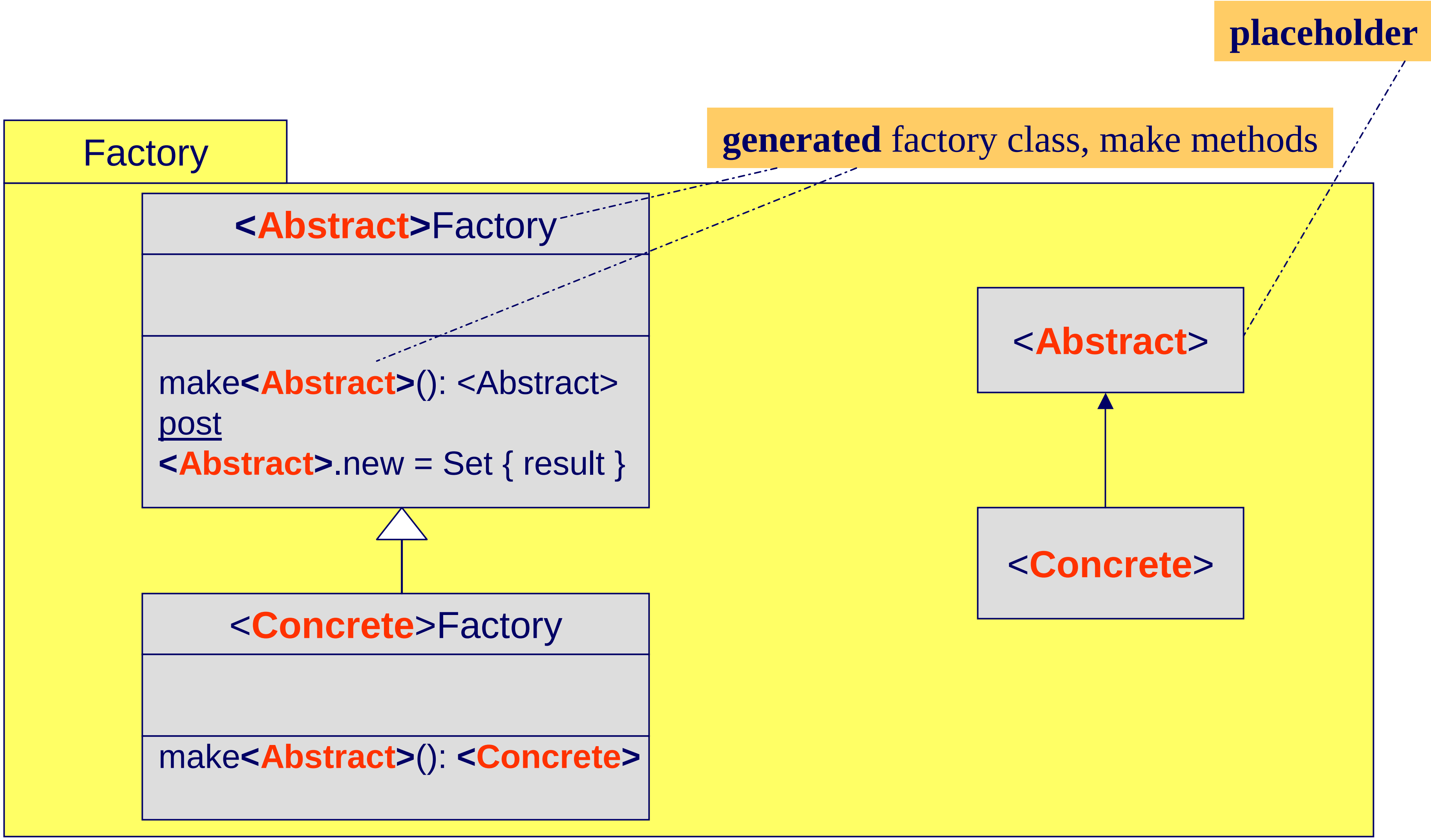
- Range of "generative" options
  - Completely ad-hoc (or "creative")
  - Fully defined translation (compiler)
  - Some defined rules and constraints

- Architecture style defined in separate package
- In general, realization **refines** specification in a way that **conforms** to the architecture style
- Style constrains **realization** and/or **refinement**

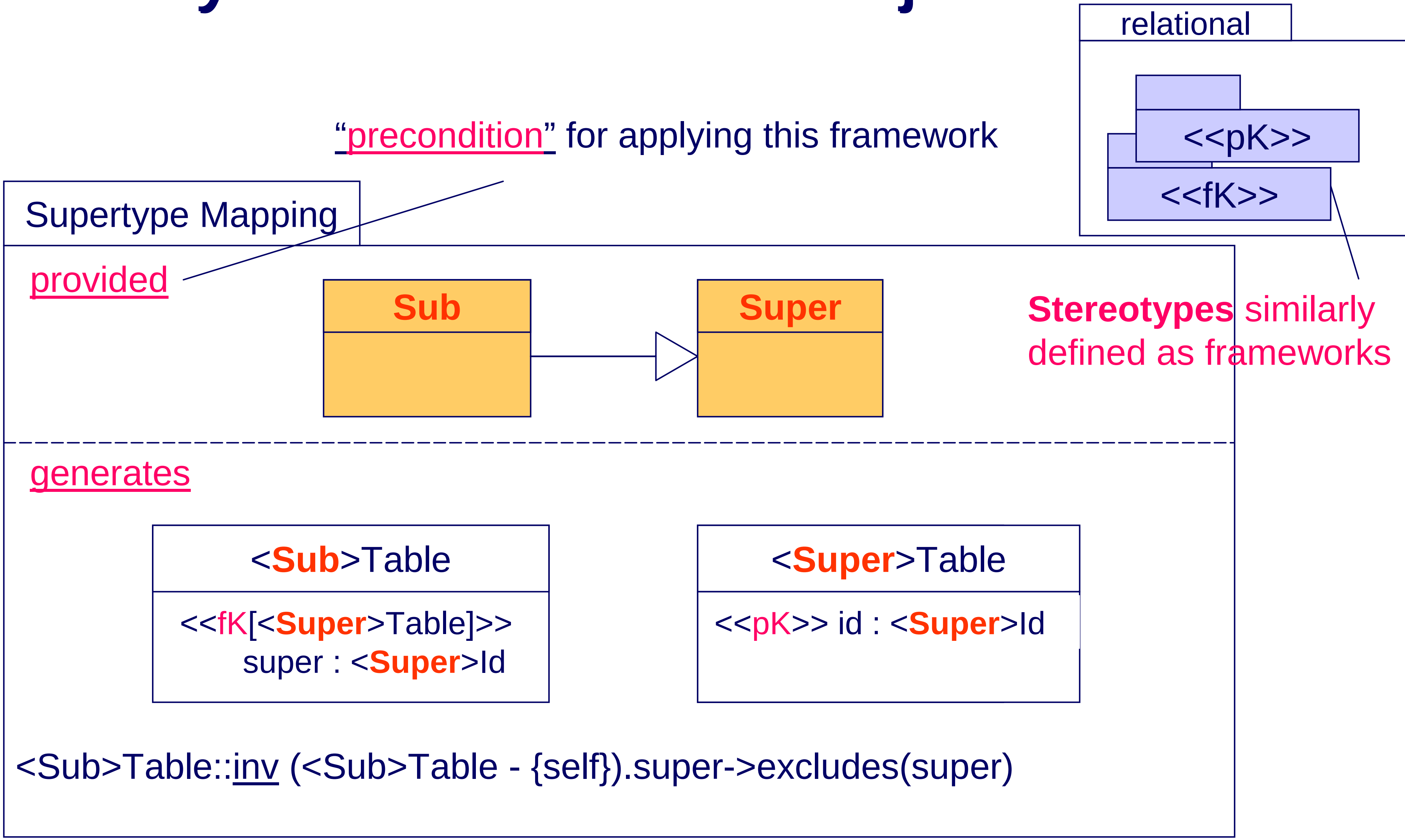
Architecture - All Sizes, Scales, Domains

- Use 3-Tier Client Server
- UI style: do 1-* association with master-slave list box
- Tiered component packaging: domain object + UI
- Java Beans: event, property, method, connectors
- Design patterns: all object creation via factories
- Coding rules: get / set method names

Catalysis Generative Framework - Factory


placeholder
generated factory class, make methods
Factory
<Abstract>Factory
make<Abstract>(): <Abstract>
post
<Abstract>.new = Set { result }
<Concrete>Factory
make<Abstract>(): <Concrete>
<Abstract>
<Concrete>

Catalysis Frameworks - Object-Relational



relational

<<pK>>

<<fK>>

"precondition" for applying this framework

Supertype Mapping

provided

Sub

Super

Stereotypes similarly defined as frameworks

generates

<**Sub**>Table

<<fK[<**Super**>Table]>>
super : <**Super**>Id

<**Super**>Table

<<pK>> id : <**Super**>Id

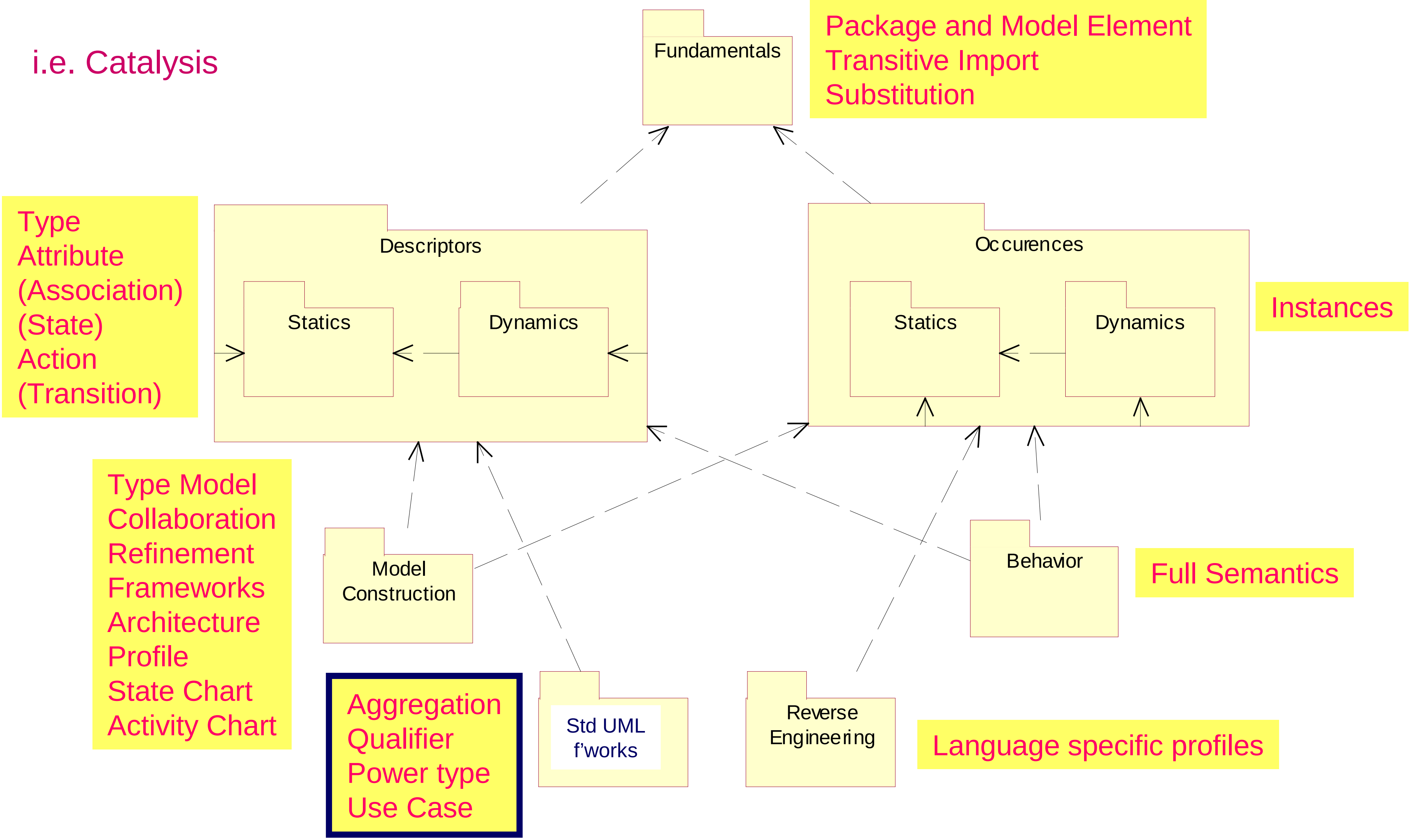
<Sub>Table::inv (<Sub>Table - {self}).super->excludes(super)

Summary

- Is UML an Architecture Description Language?
  - ✓ Not quite
  - ✓ Not designed as composable family of languages
  - ✓ Particular points to be fixed in UML 2.0
    - ✓ Extensibility: package, stereotype, pattern, profile
    - ✓ Refinement: separate and re-relate abstractions
    - ✓ Fractal: granularity of object, interaction, aspect
    - ✓ Composition: precisely relating different views
    - ✓ Components and Component Architectures
    - ✓ Reliable building blocks: impl, design, specs

What could UML [fixed] look like?

i.e. Catalysis



Package and Model Element
Transitive Import
Substitution

Type
Attribute
(Association)
(State)
Action
(Transition)

Instances

Type Model
Collaboration
Refinement
Frameworks
Architecture
Profile
State Chart
Activity Chart

Full Semantics

Aggregation
Qualifier
Power type
Use Case

Language specific profiles