Advanced SW Engineering: Architecture Evolution and Reconstruction

Uwe Zdun

Software Architecture

Faculty of Computer Science

University of Vienna

http://cs.univie.ac.at/swa

ARCHITECTURE EVOLUTION

Software Architecture Evolution

One frequently accompanying property of evolution is an increasing brittleness of the system – that is, an increasing resistance to change, or at least to changing gracefully.

– Perry & Wolf, 92.

- Architectural Drift
- Architectural Erosion

Software systems must evolve or become obsolete

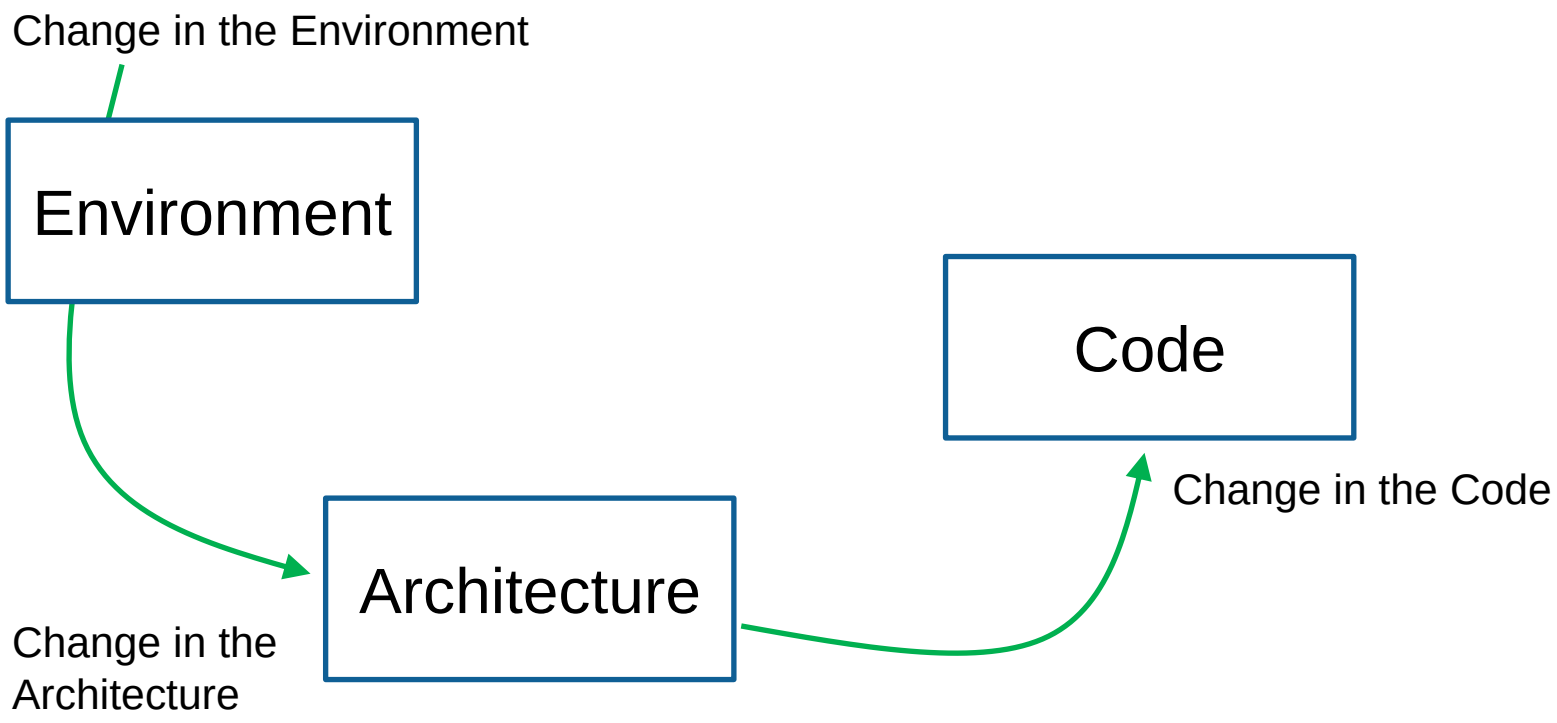
– Lehmann'80.

Software Architecture Erosion

Refers to the gap observed between the planned and actual architecture of a software system as realized in its implementation

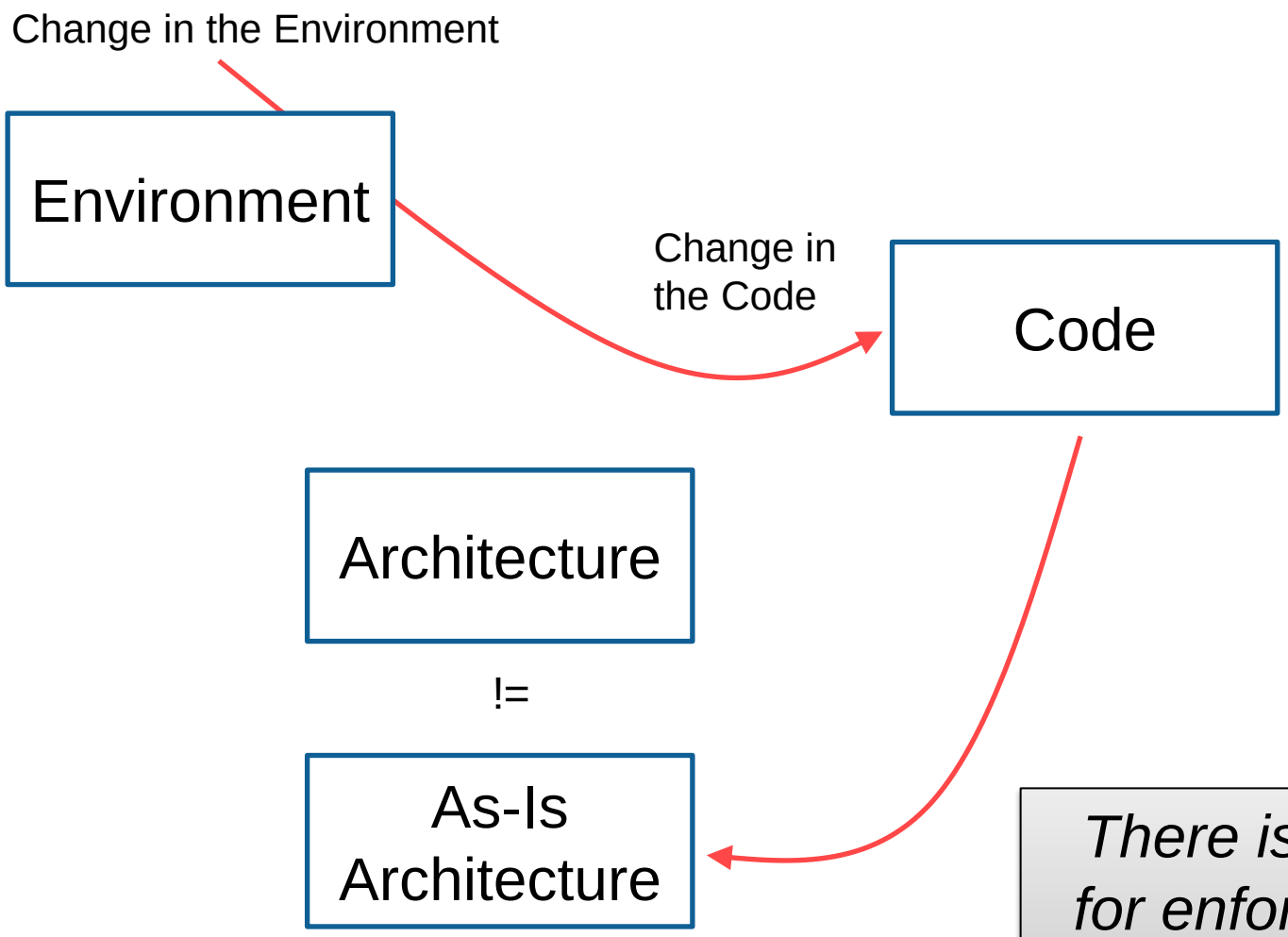
– Terra, Valente, Czarnecki, Bigonha, 2012

Two types of changes



Adapted from Mircea Lungu, SW Architecture Recovery, 2008.

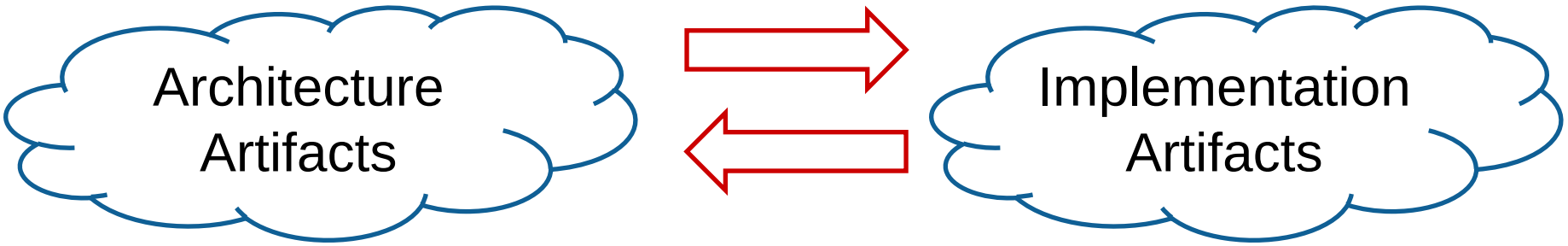
Two types of changes



There is a need for enforcing the architecture of a system

Adapted from Mircea Lungu, SW Architecture Recovery, 2008.

Keeping Architecture and Implementation in Sync



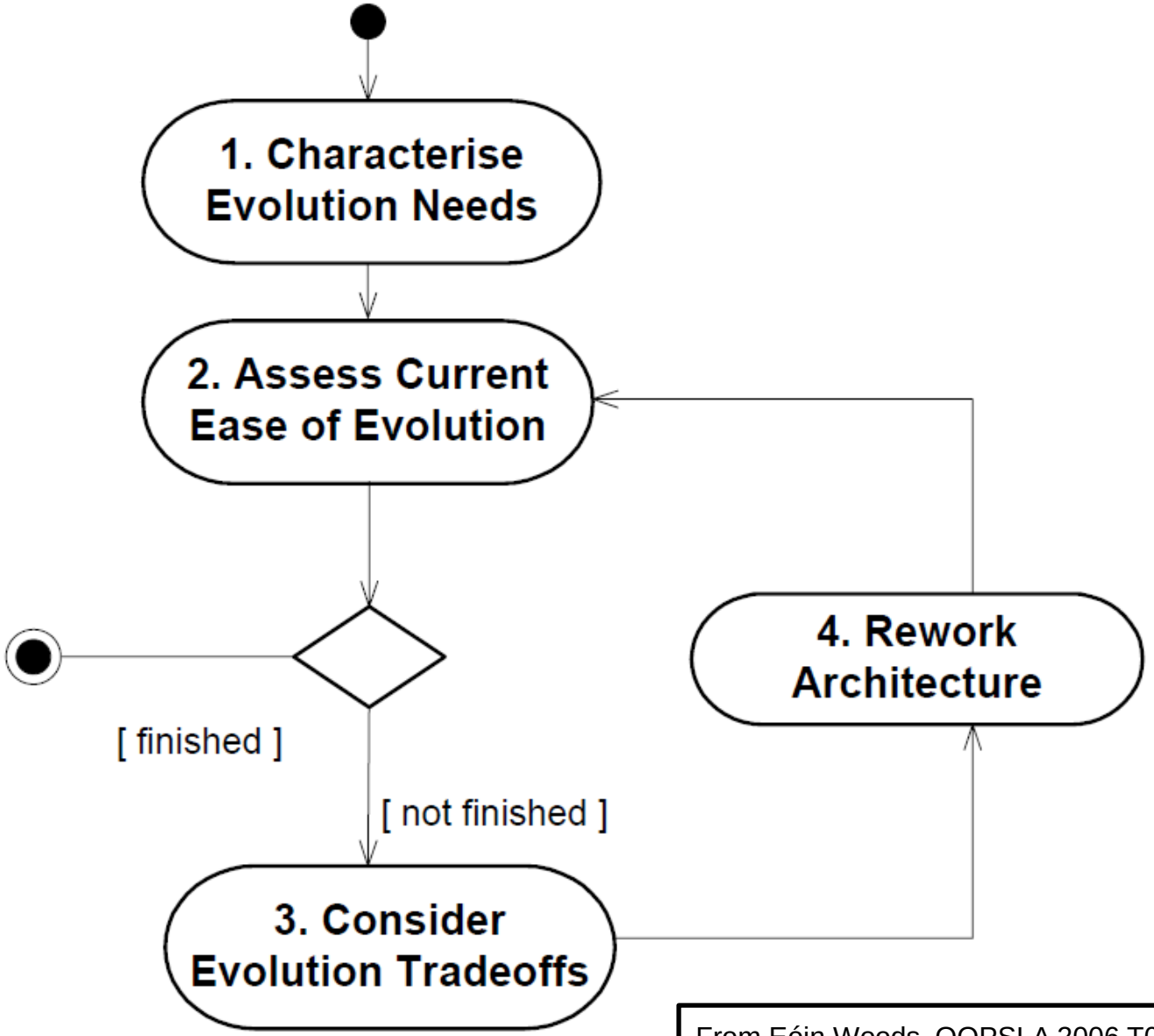
- Keeping architecture and implementation **in sync** is a difficult technical and organizational problem
- Possible approaches to keep them in sync:
  - Create and maintain **traceability links** between architecture and implementation artifacts
  - Make the **architectural model part of the implementation**
  - **Generate** some or all of the implementation from the architecture

SW Architecture Evolution Perspective by Rozanski and Woods: Quality, Applicability, and Concerns

Quality	The ability of the system to be flexible in the face of the inevitable change that all systems experience after deployment, balanced against the costs of providing such flexibility
Applicability	Important for all systems to some extent; more important for longer-lived and more widely used systems
Concerns	• product management • magnitude of change • dimensions of change • likelihood of change • timescale for change • when to pay for change • changes driven by external factors • development complexity • preservation of knowledge • reliability of change

From http://www.viewpoints-and-perspectives.info/home/perspectives/evolution/

Actitives of the Evolution Perspective



From Eóin Woods. OOPSLA 2006 T05 Tutorial Slides

SW Architecture Evolution Perspective by Rozanski and Woods: Tactics and Pitfalls

Tactics	• contain change • create extensible interfaces • apply design techniques that facilitate change • apply metamodel-based architectural styles • build variation points into the software • use standard extension points • achieve reliable change • preserve development environments
Pitfalls	• prioritization of the wrong dimensions • changes that never happen • impacts of evolution on critical quality properties • overreliance on specific hardware or software • lost development environments • ad hoc release management

From http://www.viewpoints-and-perspectives.info/home/perspectives/evolution/

SOFTWARE ARCHITECTURE RECONSTRUCTION

Missing Architectural Specifications

Frequently we are asked to analyze a system's software architecture and are given only its code and the (limited) time of a designer.

– Kazman et al. 1999

- For many systems the architecture is not documented or not sufficiently documented

Software Architecture Reconstruction (also Recovery)

Techniques and processes used to uncover a system's architecture from available information

– Jazayeri, Ran, van der Linde, 2000

Relation to Architecture Views

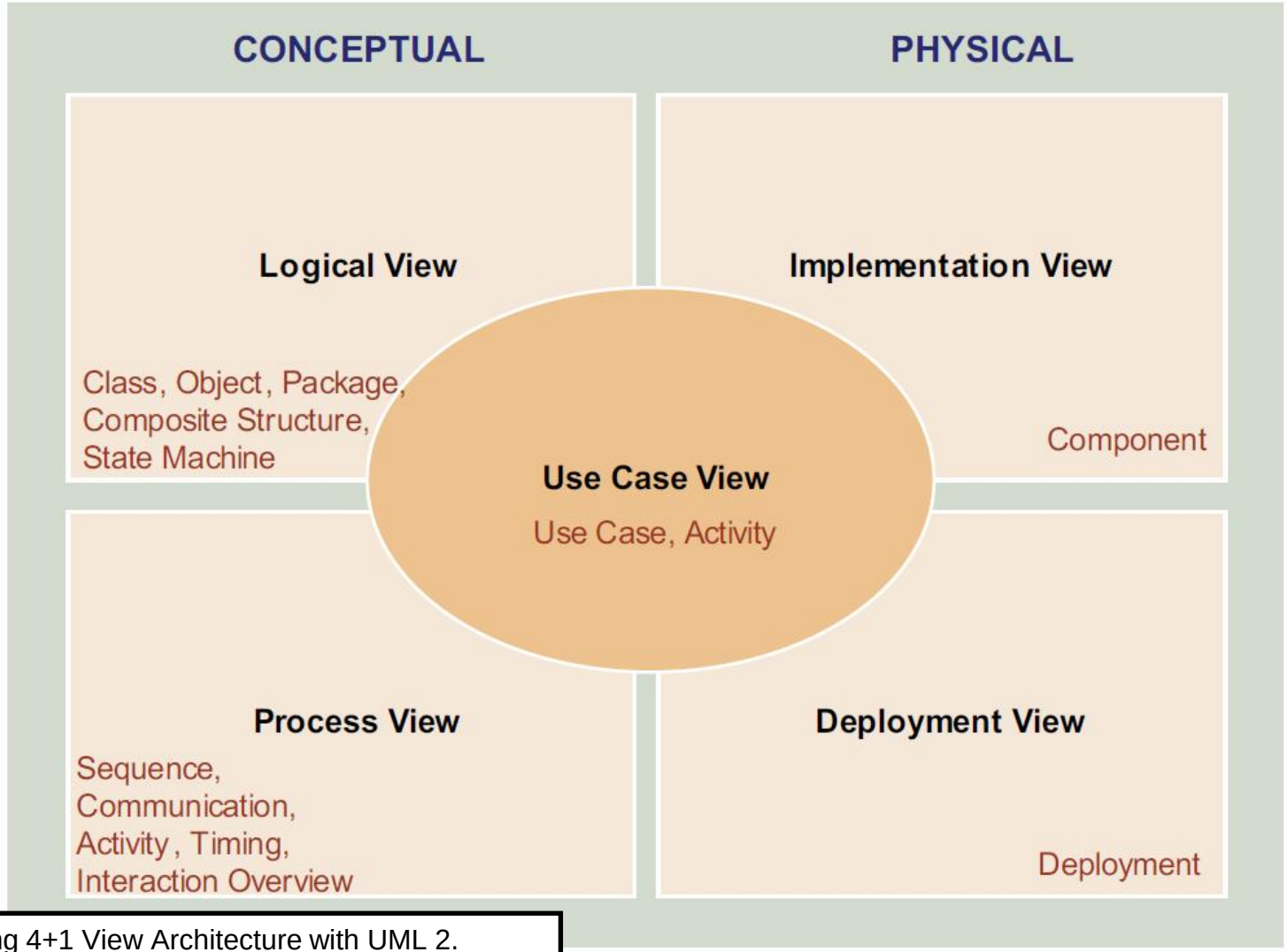
- Architecture recovery addresses the extraction of architectural views that represent the system

Architecture View: Work product expressing the architecture of a system from the perspective of specific system concerns.

Architecture Viewpoint: Work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns.

– ISO/IEC/IEEE 42010:2011: Systems and software engineering — Architecture description

Example: 4+1 View Model and Related UML Diagrams



From: Applying 4+1 View Architecture with UML 2. http://www.sparxsystems.com.au/downloads/whitepapers/FCGSS_US_WP_Applying_4+1_w_UML2.pdf

Architectural Components

A software component is an architectural entity that

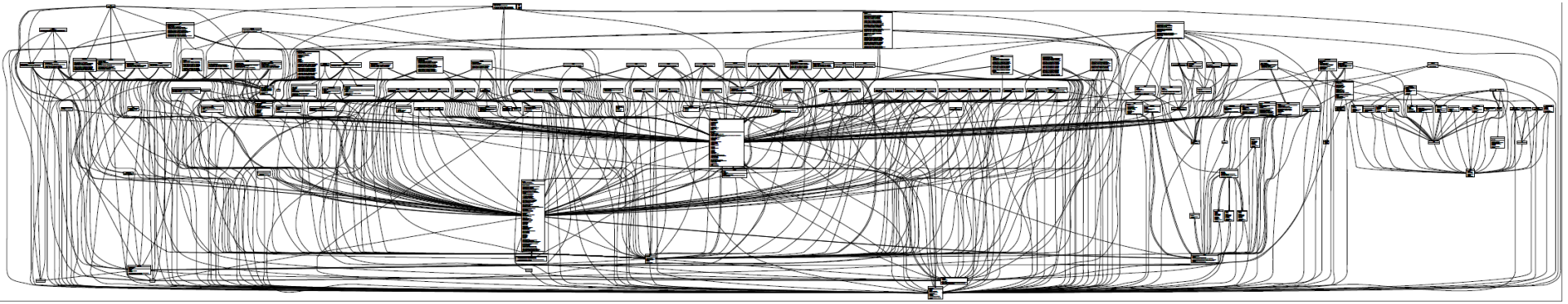
- *encapsulates a subset of the system's functionality and/or data*
- *restricts access to that subset via an explicitly defined interface*
- *has explicitly defined dependencies on its required execution context.*

– Taylor, Medvidovic, Dashofy, Software Architecture: Foundations, Theory, and Practice

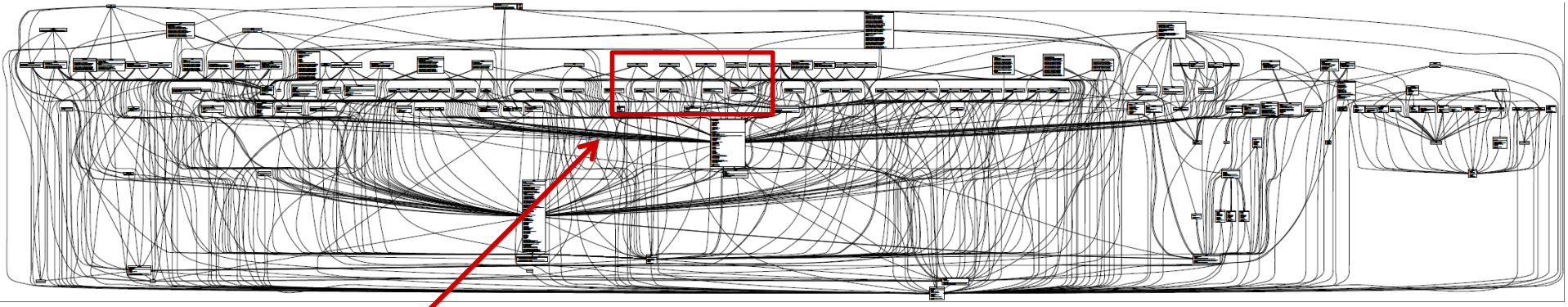
Components: Key System Concerns

- Components address different **key system concerns**:
  - **Processing** (aka functionality, behavior)
  - **State** (aka information, data)
  - **Interaction** (aka interconnection, communication, coordination, mediation)
- Interaction is typically handled by **connectors**

Example: Implementation-Level View of the Frag Scripting Language

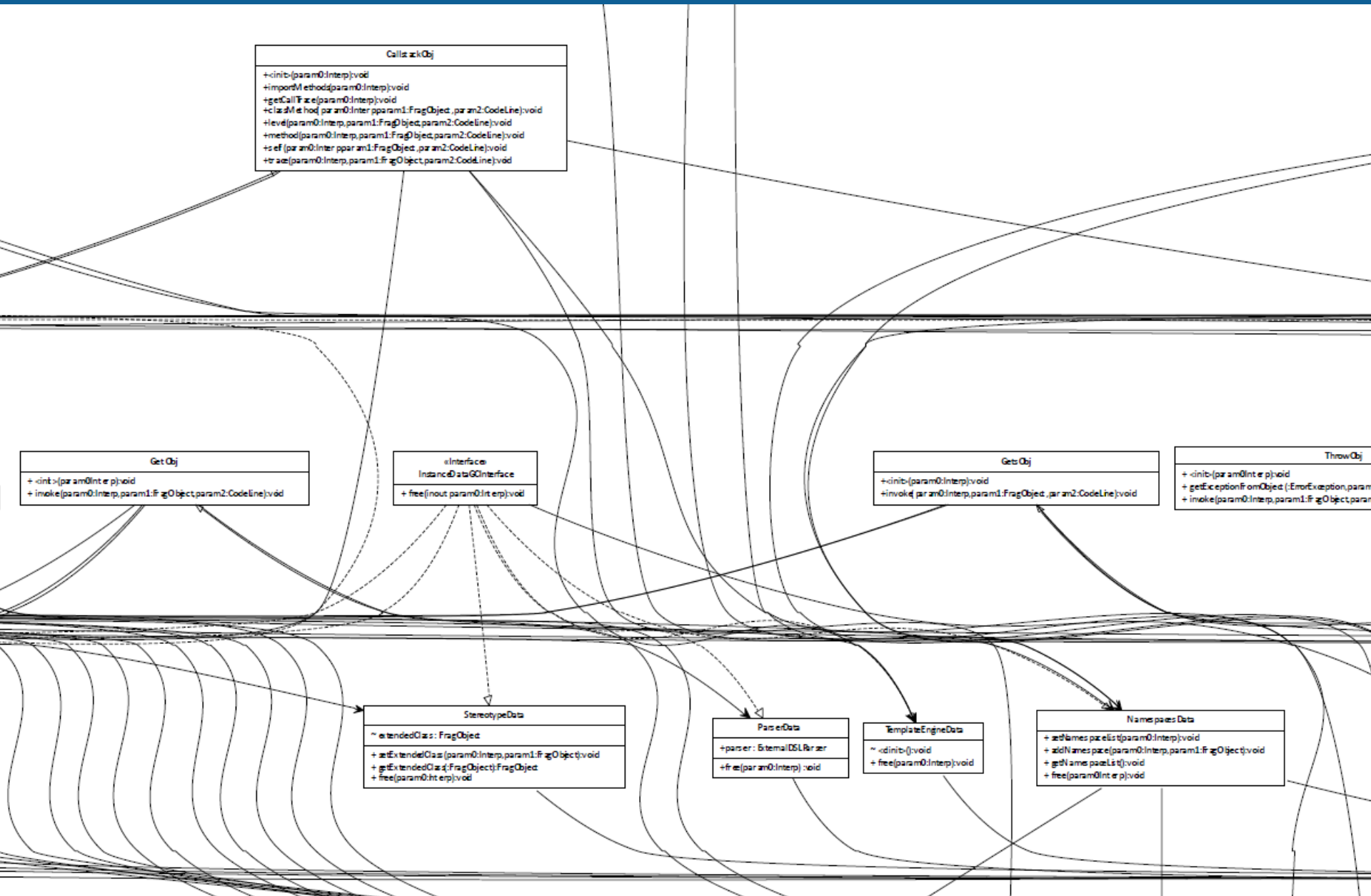


Example: Implementation-Level View of the Frag Scripting Language

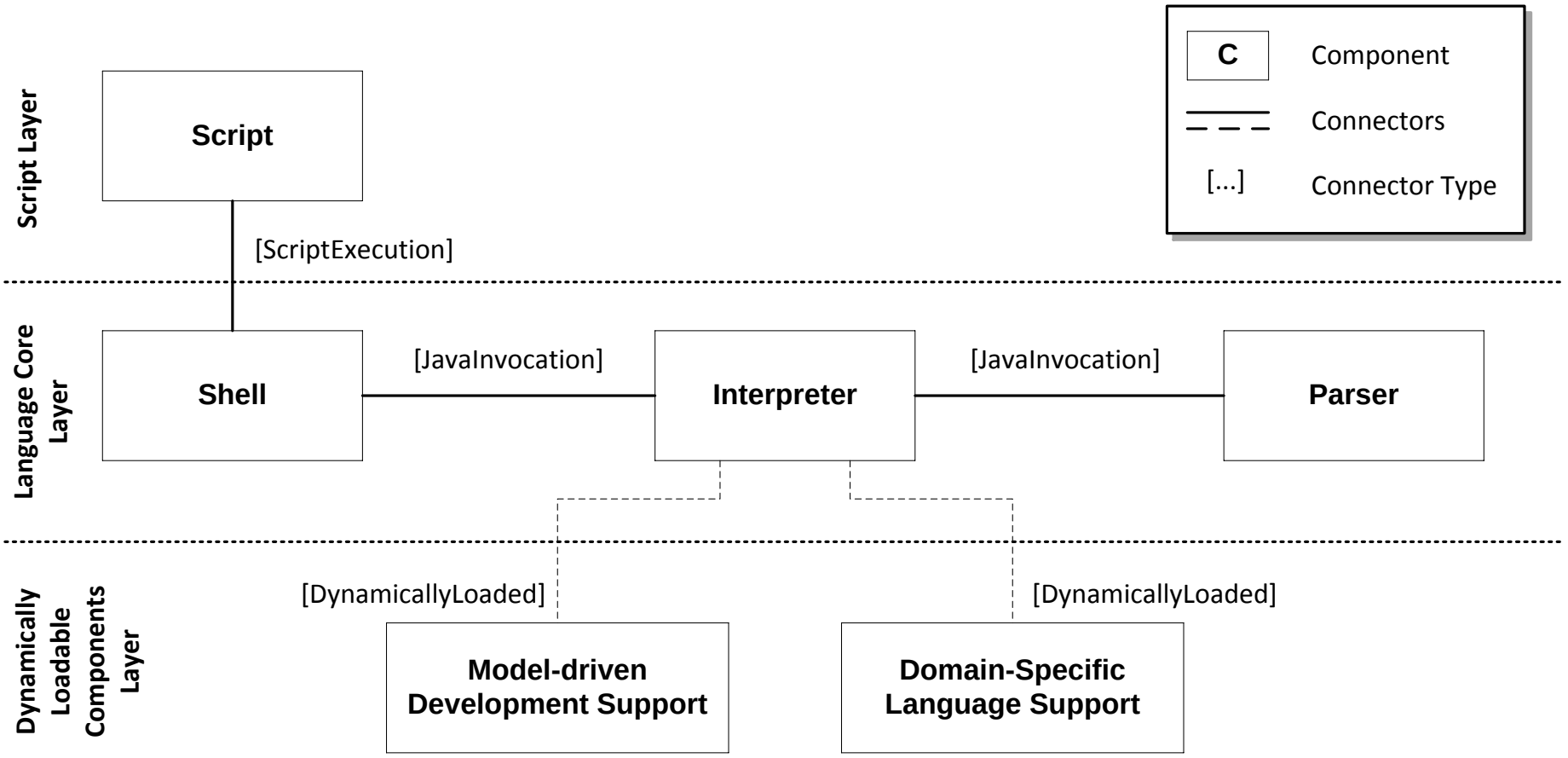


Lets zoom in here.

Example: Implementation-Level View of the Frag Scripting Language



Example: Architectural Component Model of the Frag Scripting Language

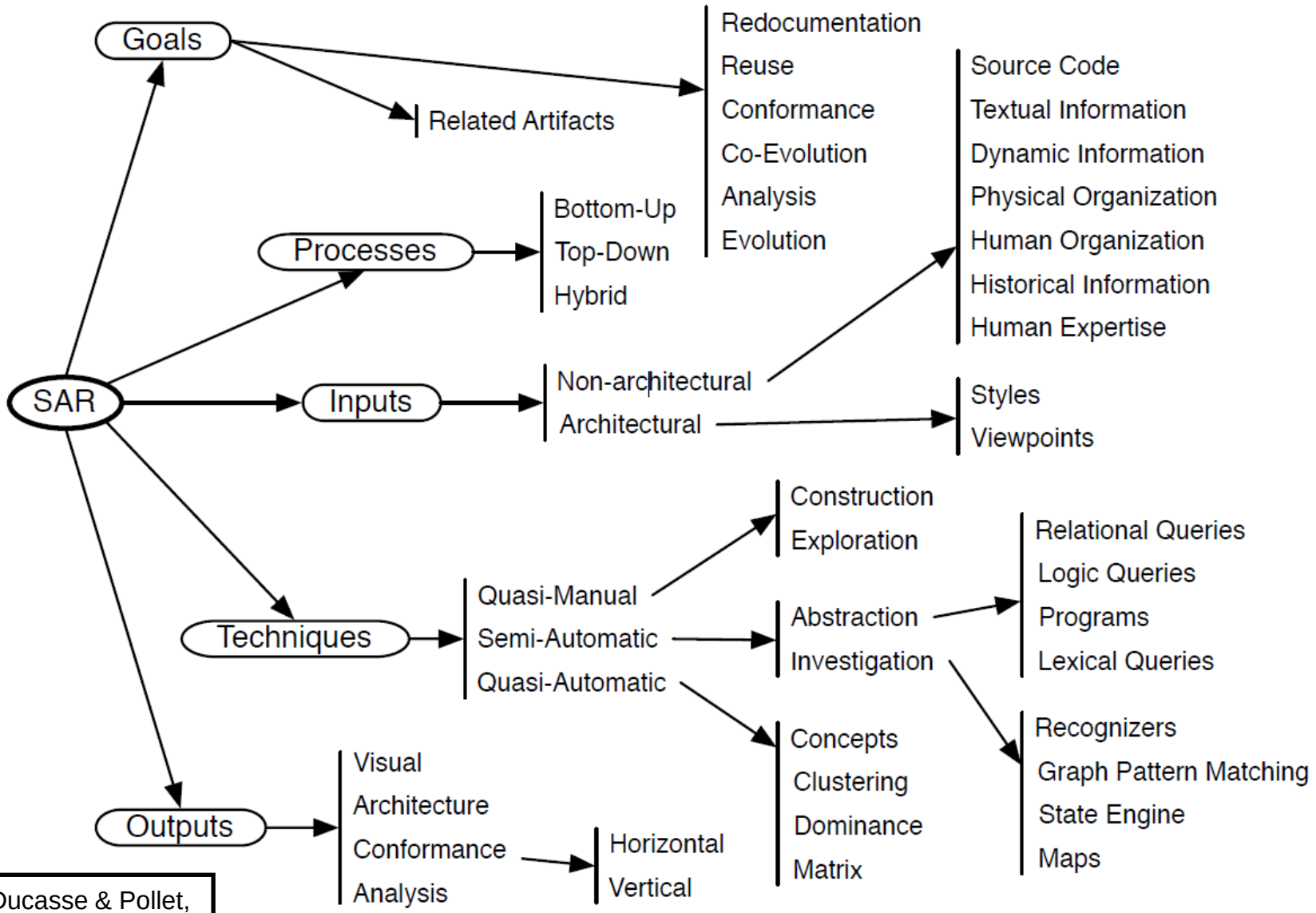


SOFTWARE ARCHITECTURE RECONSTRUCTION: MAJOR APPROACHES

Sources for Data Extraction

- Source code (static analyis)
- Historical information
- Human expertize
- Runtime behavior (dynamic analysis)
- Physical organization
- Social organization

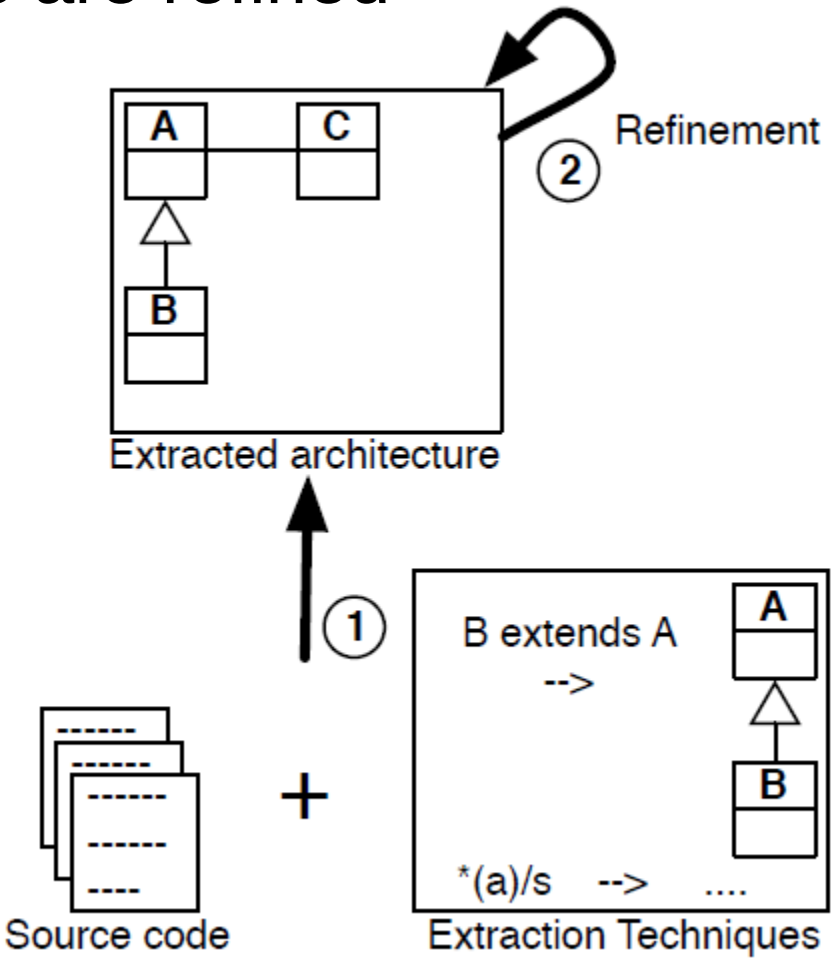
A process-oriented taxonomy for SAR



From: Ducasse & Pollet, TSE, 99

Bottom-up Processess

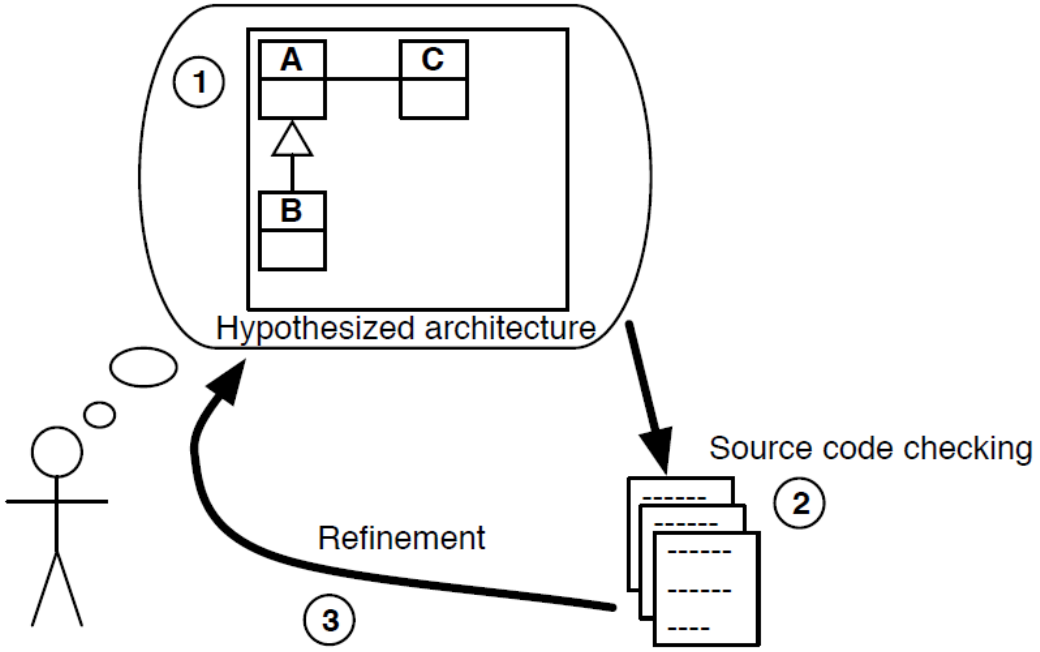
1. From the source code, views are extracted
2. The views are refined



From: Ducasse & Pollet, TSE, 99

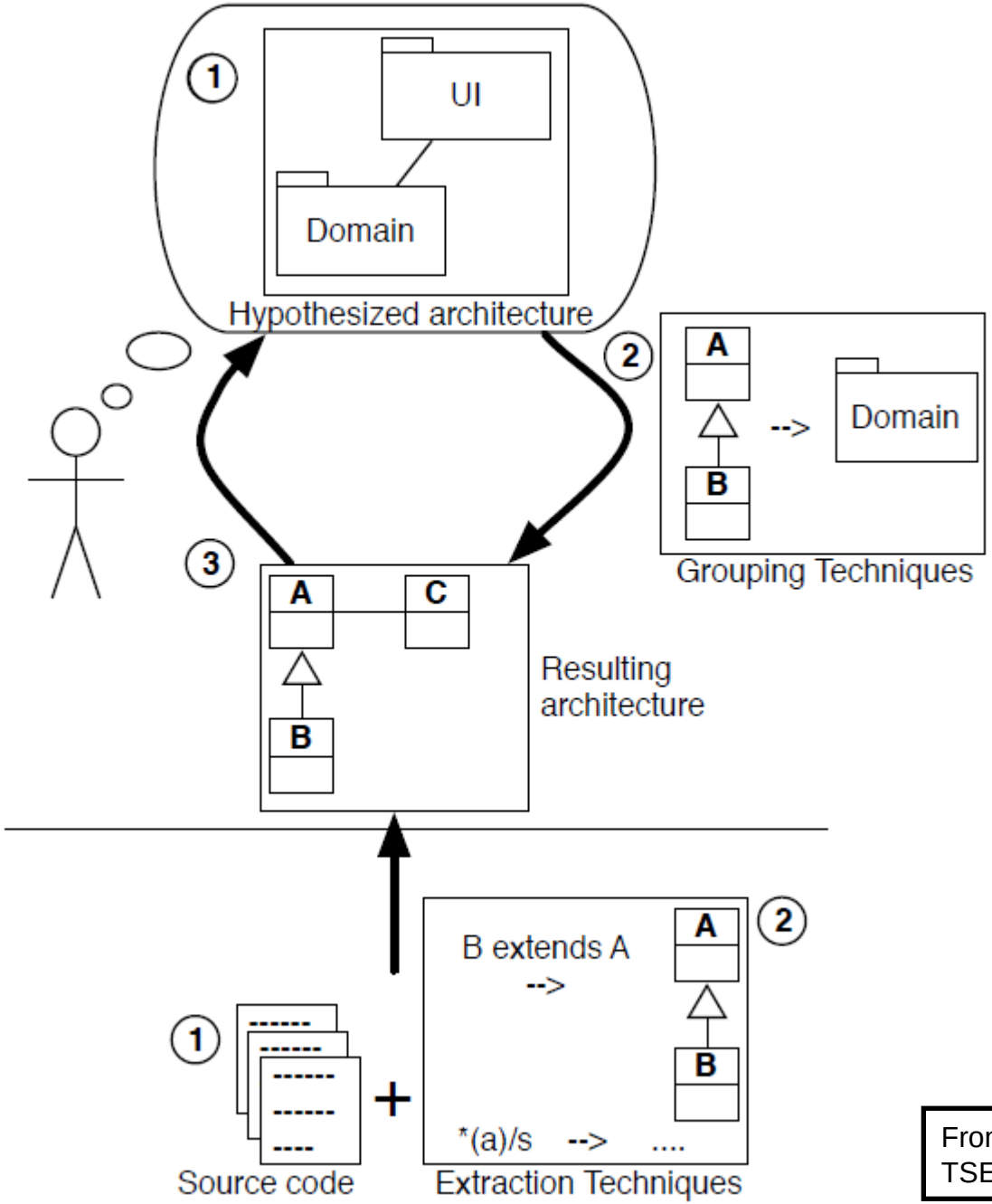
Top-down Processess

1. An hypothesized architecture is defined
2. The architecture is checked against the source code
3. The architecture is refined



From: Ducasse & Pollet, TSE, 99

Hybrid Processess



From: Ducasse & Pollet, TSE, 99

Existing Approaches and the Process They Use

Alborz [139]			hybrid
ArchView [126]	bottom-up		
ArchVis [62]	bottom-up		
ARES [36]	bottom-up		
ARM [57]			hybrid
ARMIN [79, 120]	bottom-up		
ART [43]			hybrid
Bauhaus [20, 35, 84]			hybrid
Bunch [100, 112]	bottom-up		
Cacophony [39]			hybrid
Dali [74, 78]	bottom-up		
DiscoTect [180]			hybrid
Focus [24, 104]			hybrid
Gupro [33]	bottom-up		
Intensive [109, 179]	bottom-up		
ManSART [60, 181]			hybrid
MAP [149]			hybrid
PBS/SBS [12, 42, 67, 144]			hybrid
PuLSE/SAVE [83]		top-down	
QADSAR [150, 151]			hybrid
Revealer [127, 128]	bottom-up		
RMTool [114, 115]		top-down	
SARTool [41, 86]	bottom-up		
SAVE [111, 116]		top-down	
Softwarenaut [97, 98]	bottom-up		
Symphony,Nimeta [134, 165]			hybrid
URCA [10]	bottom-up		
W4 [61]		top-down	
X-Ray [107]			hybrid
— [11]			hybrid
— [69]			hybrid
— [96]	bottom-up		
— [123]			hybrid
— [162]			hybrid

From: Ducasse & Pollet, TSE, 99

Relevant Literature and Sources

- Stéphane Ducasse, Damien Pollet: Software Architecture Reconstruction: A Process-Oriented Taxonomy. IEEE Trans. Software Eng. 35(4): 573-591 (2009)
- Mircea Lungu. Software Architecture Recovery. Oct, 2008. http://de.slideshare.net/mircea.lungu/software-architecture-recovery-in-five-questions-presentation
- Nick Rozanski and Eóin Woods. 2005. Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives. Addison-Wesley Professional.
- Eóin Woods. OOPSLA 2006 T05 Tutorial Slides. http://www.viewpoints-and-perspectives.info/vpandp/wp-content/themes/secondedition/doc/oopsla2006_vpandp_tutorial.pdf

Many thanks for your attention!

Uwe Zdun

Software Architecture
Faculty of Computer Science
University of Vienna
http://cs.univie.ac.at/swa