Trinity: A Language for Multi-View Architecture Description and Control

Maddie Kirwin kirwinma@grinnell.edu

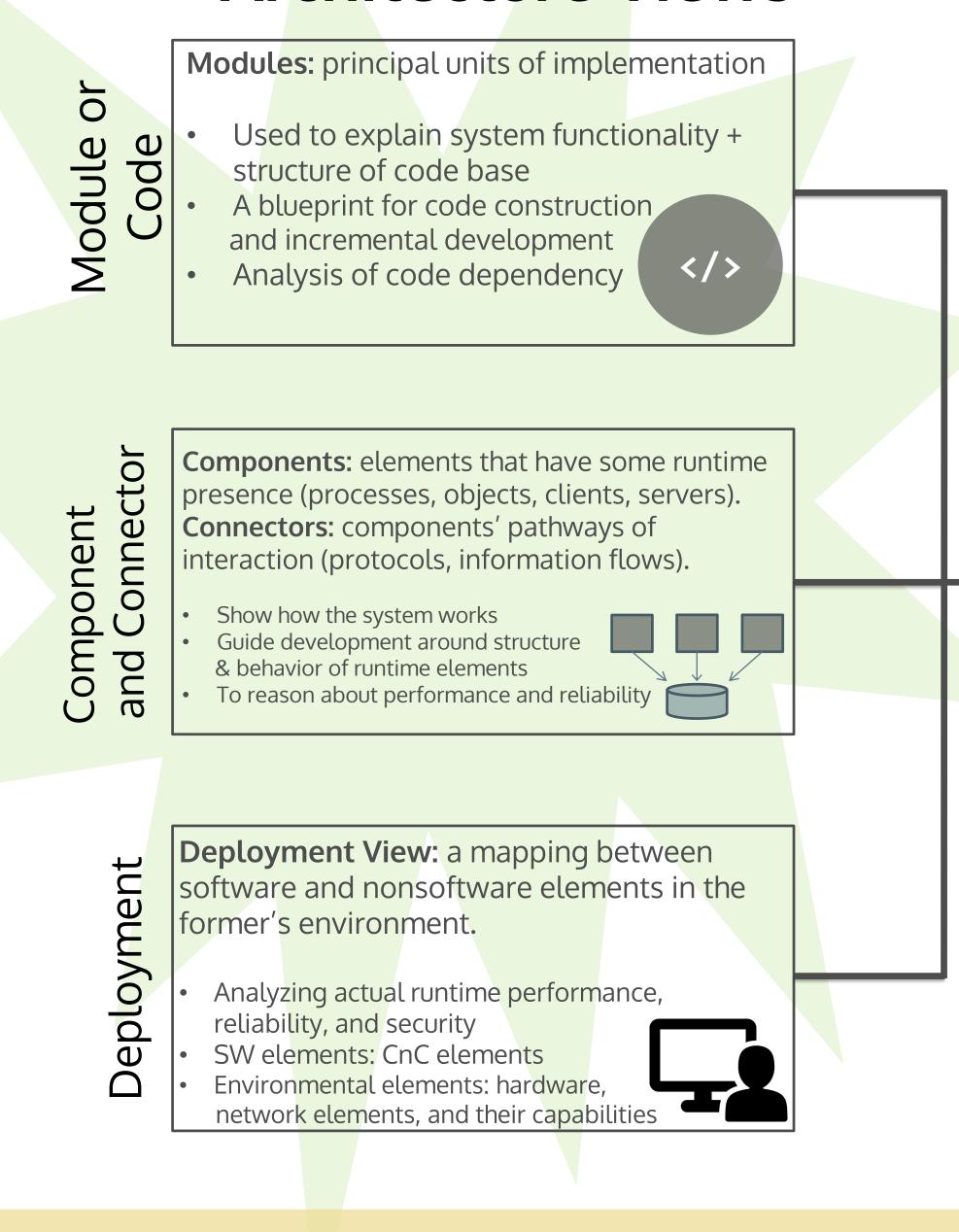
Selva Samuel ssamuel@cs.cmu.edu

Jonathan Aldrich Jonathan.Aldrich@cs.cmu.edu

Software Architecture

the "fundamental organization of a system embodied in its components, their relations to each other, and the environment"

Architecture Views



The Problem

It is hard to determine whether the logical relationships between entities in architecture diagrams are present in system implementations.

Previous Solutions

Architecture Description Languages (ADLs)

- (-) *Description:* Inferred by the name, ADLs only describe software architectures; they do not prescribe, or **enforce conformance** to them
- (+) Analysis: ADLs are focused on system analyses
- (+) Formal Notation: Currently, ADLs are the most formal mainstream architecture tools available

ArchJava Java extension unifying SWA and implementation

- (-) *Application:* Does not do anything interesting with SWA (i.e. checks)
- (-) *Distributed Systems:* No support for distributed systems
- (-) Multiple SWA Views: Lacks support for multiple architecture views; focuses only on Component-and-Connector view.

Trinity's Approach

- Makes software architecture a "live" component of Trinity systems
- Trinity enforced architecture conformance complements ADL analyses
- Directly translate the conceptual entities from SWA views into code-enforced constructs
- Support for all three software architecture views (module or code, CnC, and deployment).
- Support for architecture conformance in **distributed systems**.

Design

Example