Notes on
Constraint Transformations

Enrico Malavasi - malavasi@cadence.com

cadence

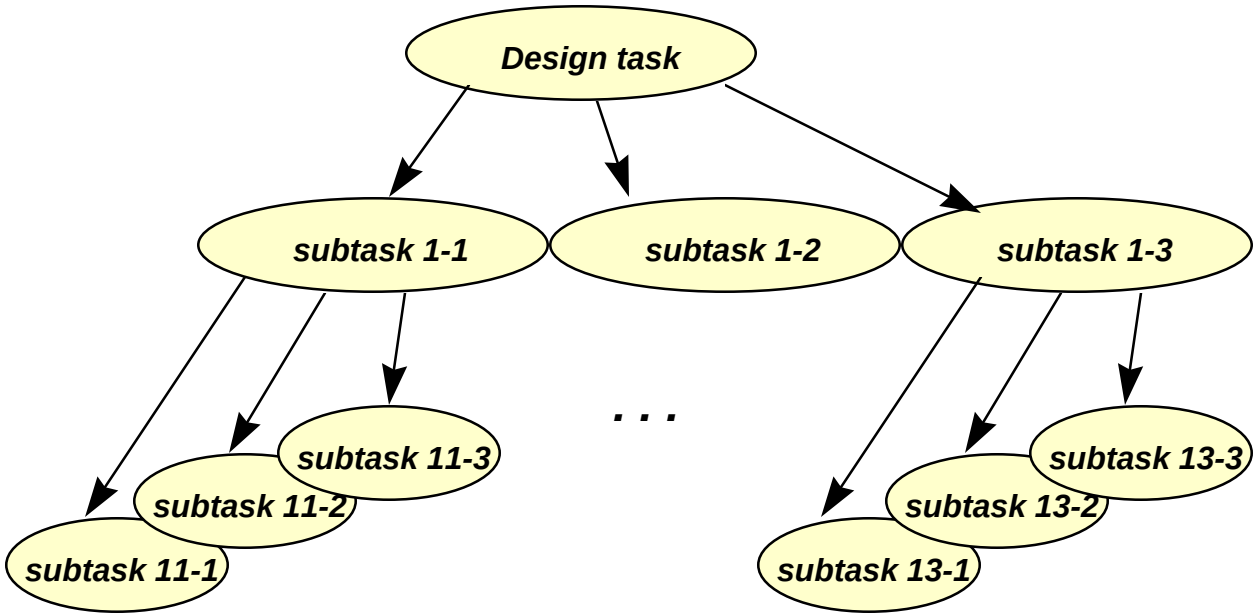
Outline

- Levels of Abstraction and Transitions
- Constraint Transformations
- Architectural Considerations
- Constraint Structure

cadence

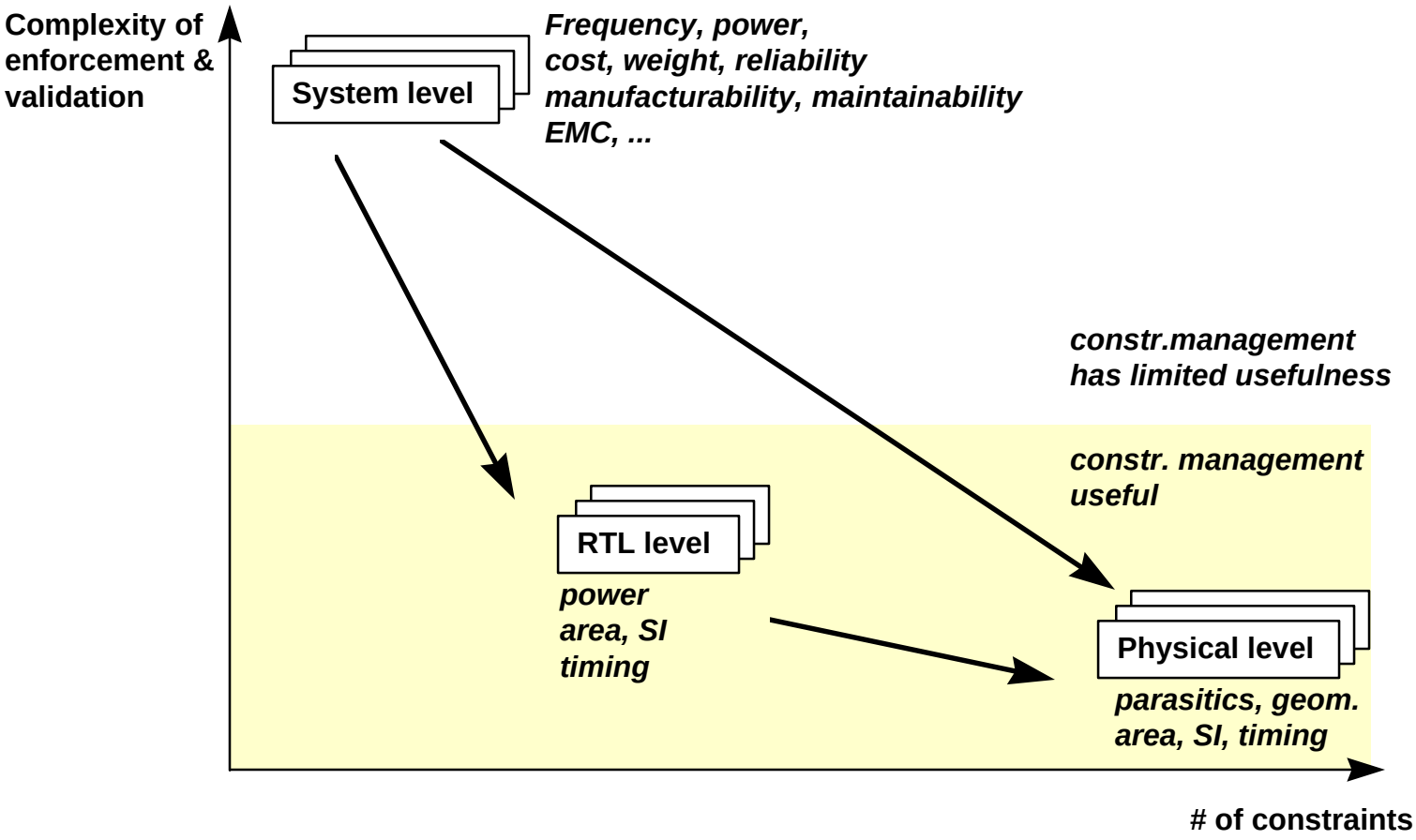
Top-down design methodology

Each partition corresponds to
- a step down in abstraction
- a constraint transformation



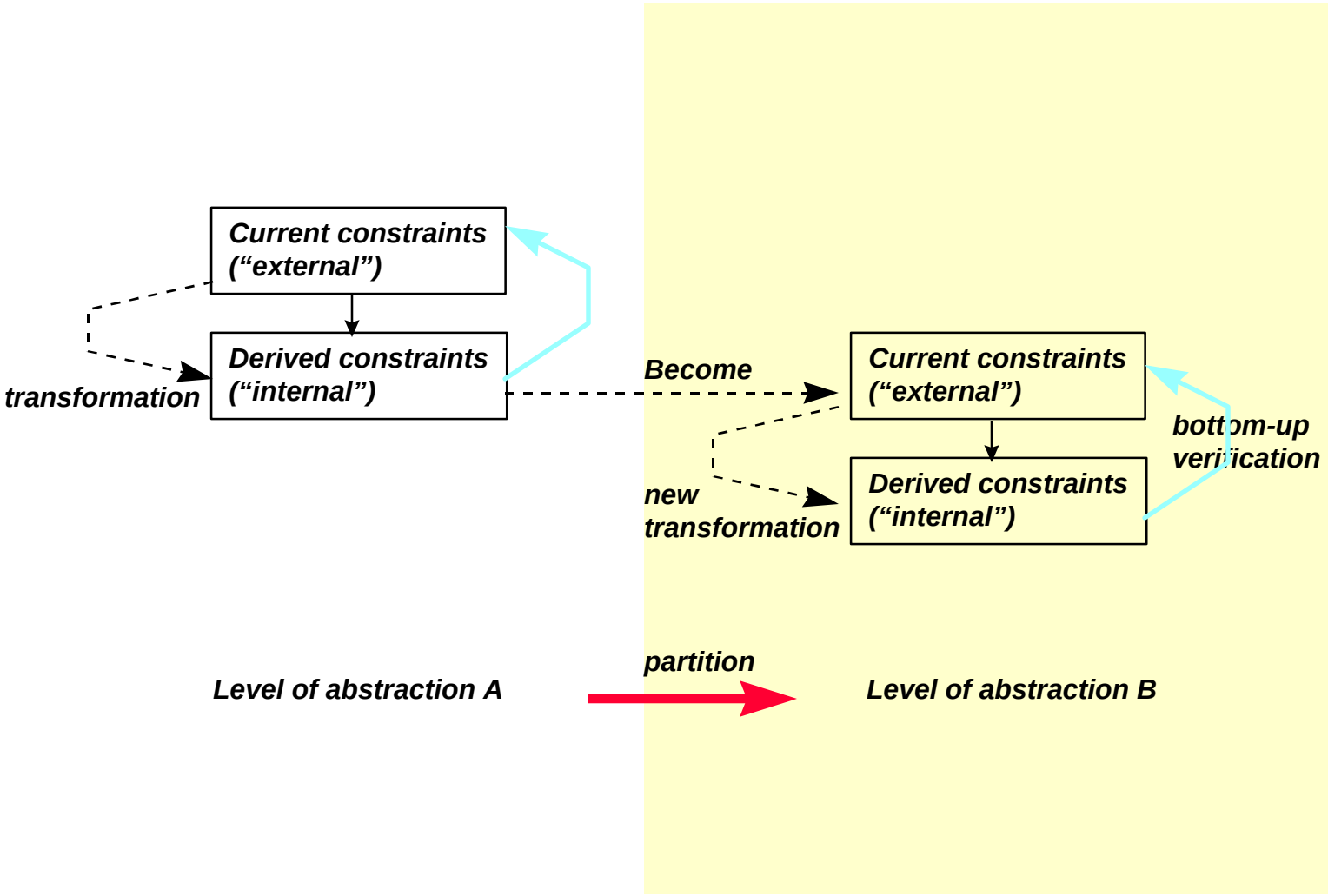
cadence

Constraints and levels of abstraction



cadence

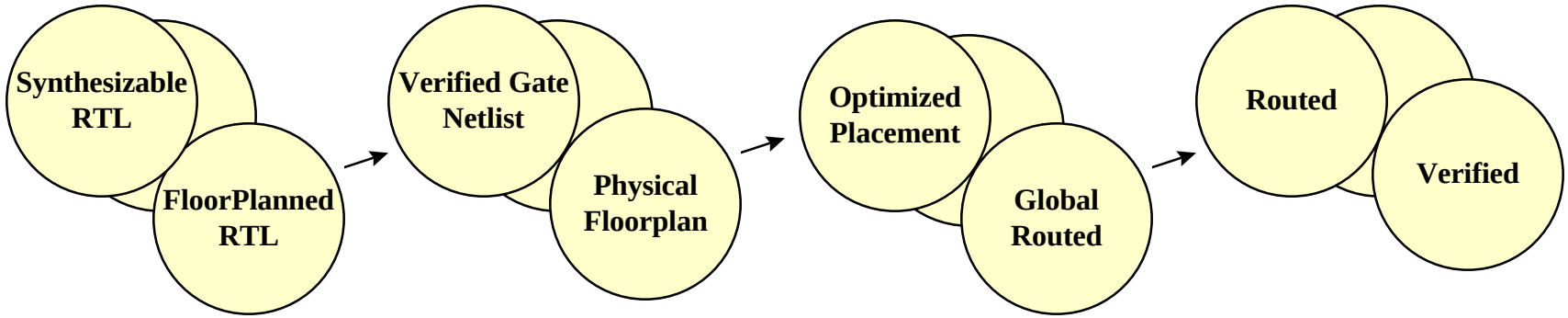
Internal vs. External Constraints



cadence

Levels of Abstraction

- Similar to data states
  - characterized by *sign-off points*
  - need well-defined *transitions* between “states”



cadence

Transitions between abstraction levels

- Two types of transitions:
  - Major transitions
    - Well-defined sequence/progression or reconvergence point reaching a sign-off point
  - Minor transitions
    - Incremental changes in data
    - May be done in any order

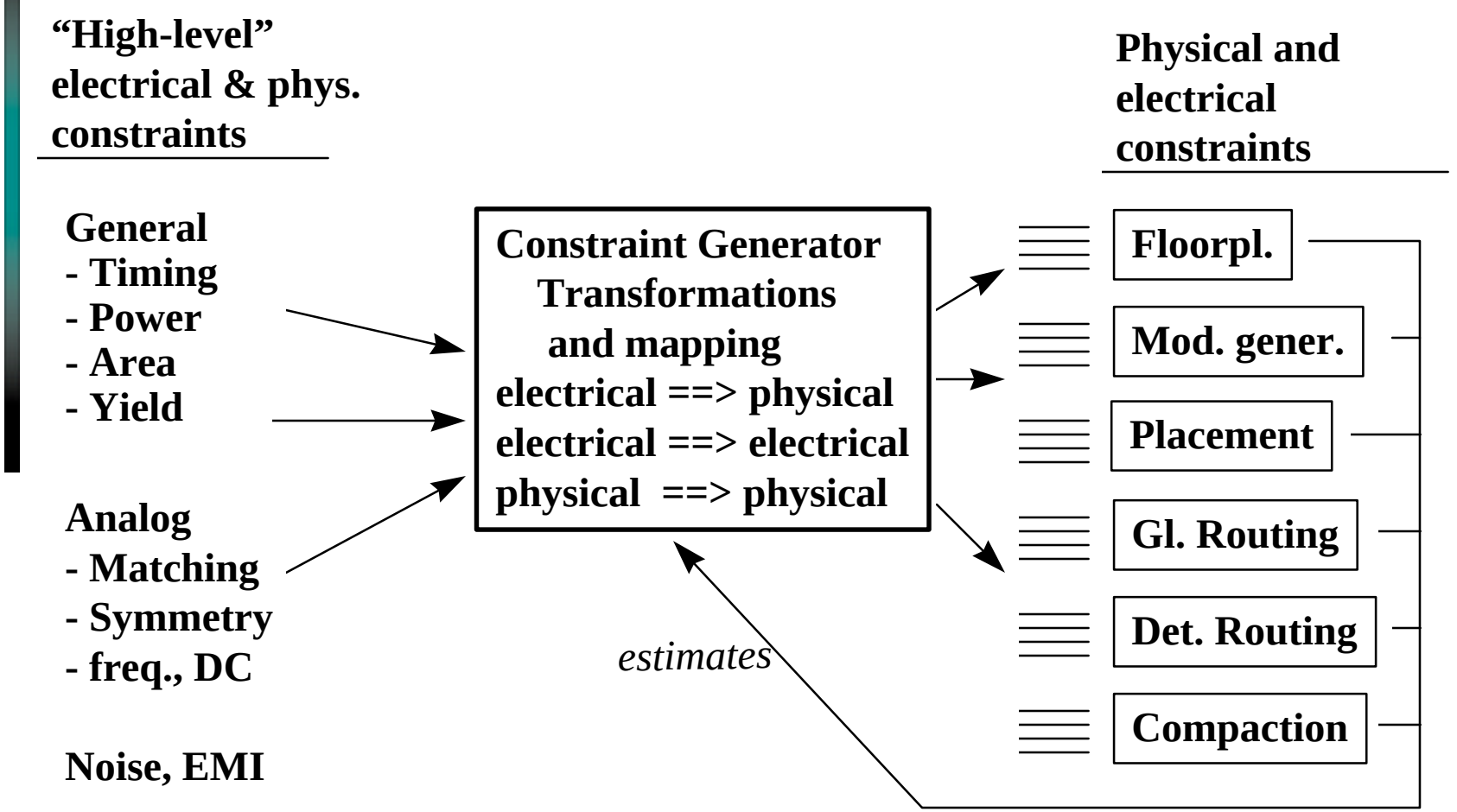
cadence

Concepts: Transitions

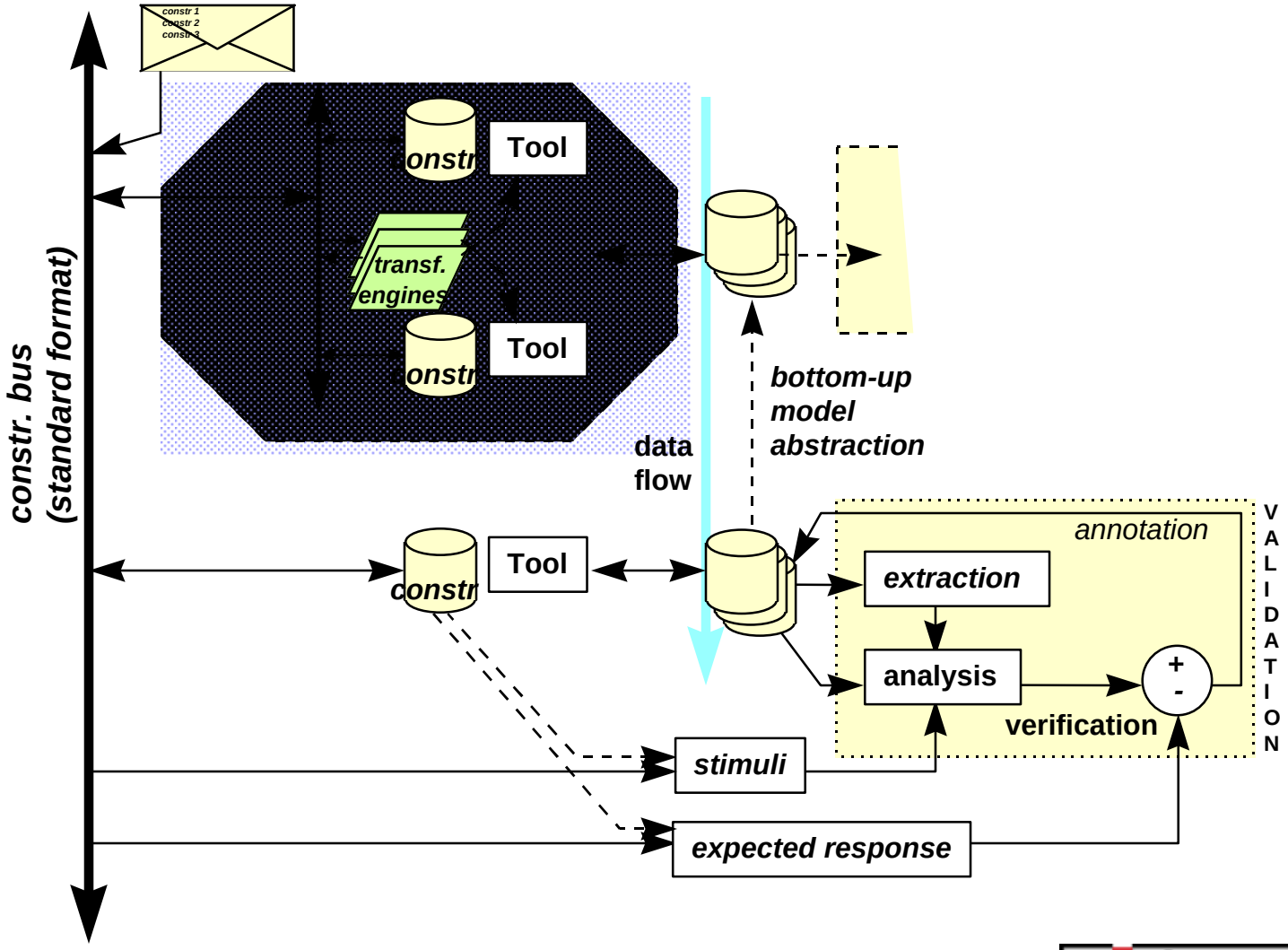
- Most applications require a design object to have reached a certain sign-off point
  - ☛e.g., final cross-talk verification may require parasitics from a 2.5d extractor
- If the object is not in the required state, data can be derived by either:
  - **abstracting** from data at a **later state** or
  - **estimating** from data at an **earlier state**
- A *confidence factor* can be derived based on how much estimation or abstraction was required to provide the data to the application.

cadence

Constraint transformations (phys.des.)

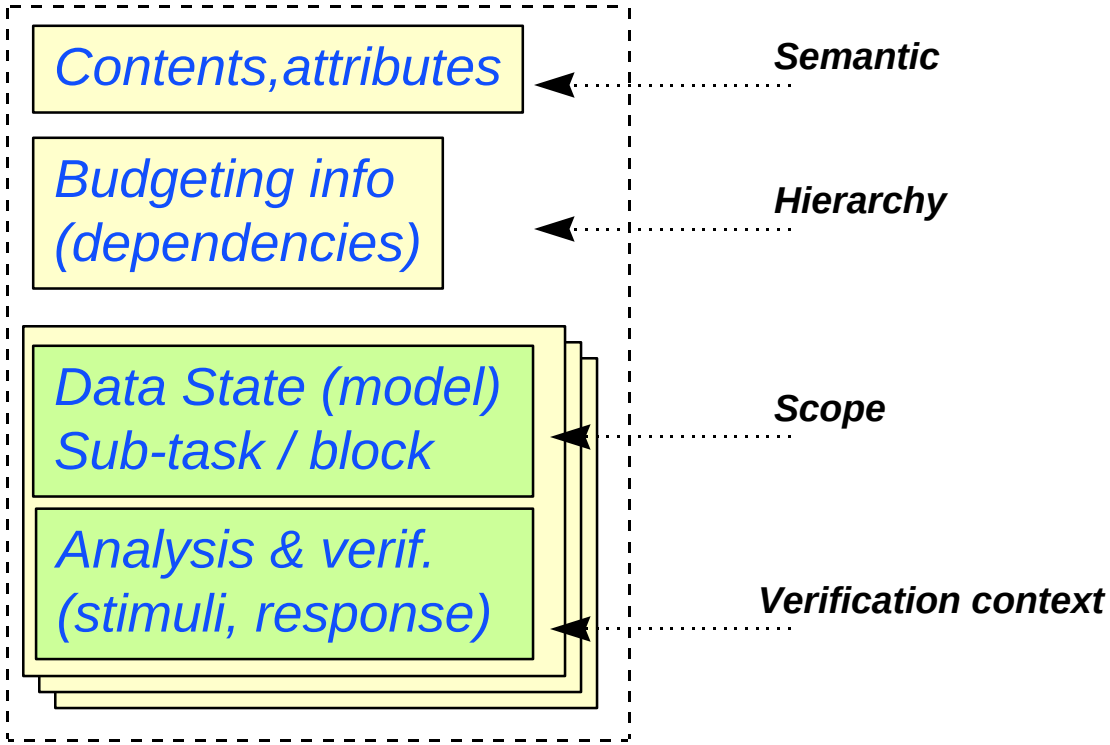


Architectural considerations



Constraint structure

- Verification stimuli & expected response are part of definition for constraints



cadence

Constraint structure

- Budgeting info is defined by constraint dependencies / transformation
- Scope is defined by design data hierarchy
- Multiple scopes (models) can be available with same constraint semantic
- Verification contexts can vary with scope
- Version control / multiple constraint sets

cadence