# Views in the Enterprise Domain

Richard Martin
Tinwisle Corporation
Bloomington, Indiana

Edward Robertson
Computer Science/Informatics
Indiana University

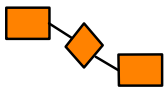
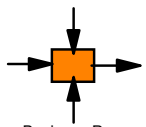
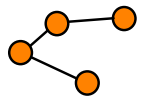
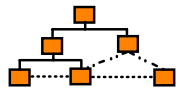
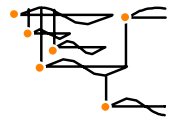
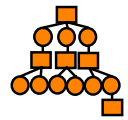
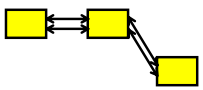
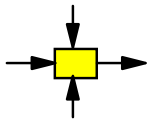
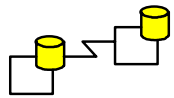
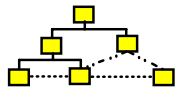
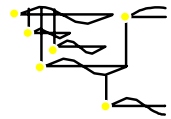
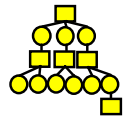
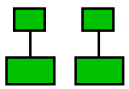
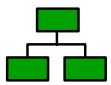
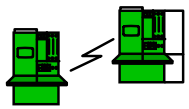
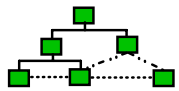
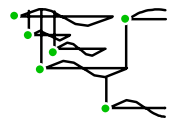
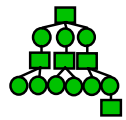
# Background: Our Motivations

- understand use of views in Enterprise Architecture Frameworks (EAF - example follows) and related standards
- facilitate formalization & implementation
- manage confusion caused by multiple views of "views"

# Background: Our Experience

- developing and teaching about information systems
- formal, top-down orientation
  - "Nothing is as practical as a good theory."
  - EAF organizes concepts, models, & activities
- involvement in International Standards yielding EAF for industrial processes

# ENTERPRISE ARCHITECTURE - A FRAMEWORK ™

| | DATA *What* | FUNCTION *How* | NETWORK *Where* | PEOPLE *Who* | TIME *When* | MOTIVATION *Why* | |
|---|---|---|---|---|---|---|---|
| SCOPE (CONTEXTUAL) *Planner* | List of Things Important to the Business  ENTITY = Class of Business Thing | List of Processes the Business Performs  Function = Class of Business Process | List of Locations in which the Business Operates  Node = Major Business Location | List of Organizations Important to the Business  People = Major Organizations | List of Events Significant to the Business  Time = Major Business Event | List of Business Goals/Strat  Ends/Means=Major Bus. Goal/ Critical Success Factor | SCOPE (CONTEXTUAL) *Planner* |
| ENTERPRISE MODEL (CONCEPTUAL) *Owner* | e.g. Semantic Model  Ent = Business Entity Reln = Business Relationship | e.g. Business Process Model  Proc. = Business Process I/O = Business Resources | e.g. Logistics Network  Node = Business Location Link = Business Linkage | e.g. Work Flow Model  People = Organization Unit Work = Work Product | e.g. Master Schedule  Time = Business Event Cycle = Business Cycle | e.g. Business Plan  End = Business Objective Means = Business Strategy | ENTERPRISE MODEL (CONCEPTUAL) *Owner* |
| SYSTEM MODEL (LOGICAL) *Designer* | e.g. Logical Data Model  Ent = Data Entity Reln = Data Relationship | e.g. "Application Architecture"  Proc .= Application Function I/O = User Views | e.g. "Distributed System Architecture"  Node = I/S Function (Processor, Storage, etc) Link = Line Characteristics | e.g. Human Interface Architecture  People = Role Work = Deliverable | e.g. Processing Structure  Time = System Event Cycle = Processing Cycle | e.g., Business Rule Model  End = Structural Assertion Means =Action Assertion | SYSTEM MODEL (LOGICAL) *Designer* |
| TECHNOLOGY MODEL (PHYSICAL) *Builder* | e.g. Physical Data Model  Ent = Segment/Table/etc. Reln = Pointer/Key/etc. | e.g. "System Design"  Proc.= Computer Function I/O = Screen/Device Formats | e.g. "System Architecture"  Node = Hardware/System Software Link = Line Specifications | e.g. Presentation Architecture  People = User Work = Screen Format | e.g. Control Structure  Time = Execute Cycle = Component Cycle | e.g. Rule Design  End = Condition Means = Action | TECHNOLOGY CONSTRAINED MODEL (PHYSICAL) *Builder* |
| DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor* | e.g. Data Definition  Ent = Field Reln = Address | e.g. "Program"  Proc.= Language Stmt I/O = Control Block | e.g. "Network Architecture"  Node = Addresses Link = Protocols | e.g. Security Architecture  People = Identity Work = Job | e.g. Timing Definition  Time = Interrupt Cycle = Machine Cycle | e.g. Rule Specification  End = Sub-condition Means = Step | DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor* |
| FUNCTIONING ENTERPRISE | e.g. DATA | e.g. FUNCTION | e.g. NETWORK | e.g. ORGANIZATION | e.g. SCHEDULE | e.g. STRATEGY | FUNCTIONING ENTERPRISE |




Views in the Enterprise Domain

# Goals for Studying Views

- explicit characterization for all facets of views and viewing
- accommodate wide range of views and view uses
- facilitate use of views in design
  - particularly with multiple parties
- formalisms suitable for application and implementation

# Reasons for Having Views

- accommodate multiple users
  - examining content
  - defining content
- expose content to enable interoperability
- mask apparent complexity
- provide focus
- enable modularity of process
- enable “need to know” restrictions
- move toward particular domain knowledge
- enable interoperation with larger knowledge sets

# Views in the Enterprise Domain
## Outline

- Distinctions in views and models
- Meta-levels and views
- Usage of views in standards
- Technology of views

# Sources of Confusion



- "view" and "model" both noun and verb
- different reasons for viewing
- "meta" matters

- in international standards, word translation is not one-to-one

# View and Model: nouns



- view and model have different intentions
  - model ≅ something constructed
  - view ≅ something derived, observed
- extension may be the same
- model (noun) is a special kind of view (noun) specified not by content or structure but rather by the medium (wood, plastic, paper, ER, DFD, UML, etc.) of its representation

# View and Model: verbs



- view (verb) is different than model (verb)
- view (verb) is to observe from the perspective of an individual
- model (verb) is to construct a model to overcome limited perspective of individual participants
- view (verb) is a process of interpreting a view (noun)
- model (verb) is a process of synthesis resulting in a model (noun)

# View and Viewpoint

- view is the observation
- viewpoint is observational perspective
  - makes features of a model more or less significant
- viewpoint is characterized by intent
  - concerns
  - responsibilities
  - some things must be believed to be seen
- viewpoints often associated with “roles”
- standards sometimes specify a view using a viewpoint

# User vs. Modeler Views



- user view (as is)
  - (noun) extracted content
  - (verb) specification of extraction process, e.g., RDB view, report financials
  - manifestation may be updatable
- modeler view (to be)
  - (noun) spectrum of usage viewpoints
  - always updatable and reversible
  - many meta-levels
- both may cross multiple models

# Necessity



- prescribed views
  - fundamental perspectives for model generation
  - domain specific
  - often required by standard, contract, etc.
- possible views
  - arrangements of content
  - permissible
  - consistent

# Incidental distinctions



- single model view vs. multiple model view
- incomplete partial model (view) vs. complete parts of whole model (view)
- enterprise view (model) vs. constituent views (models) of enterprise
- view (model) driven by function vs. view (model) driven by information (process vs. data)
- model view vs. object view (CIMOSA)

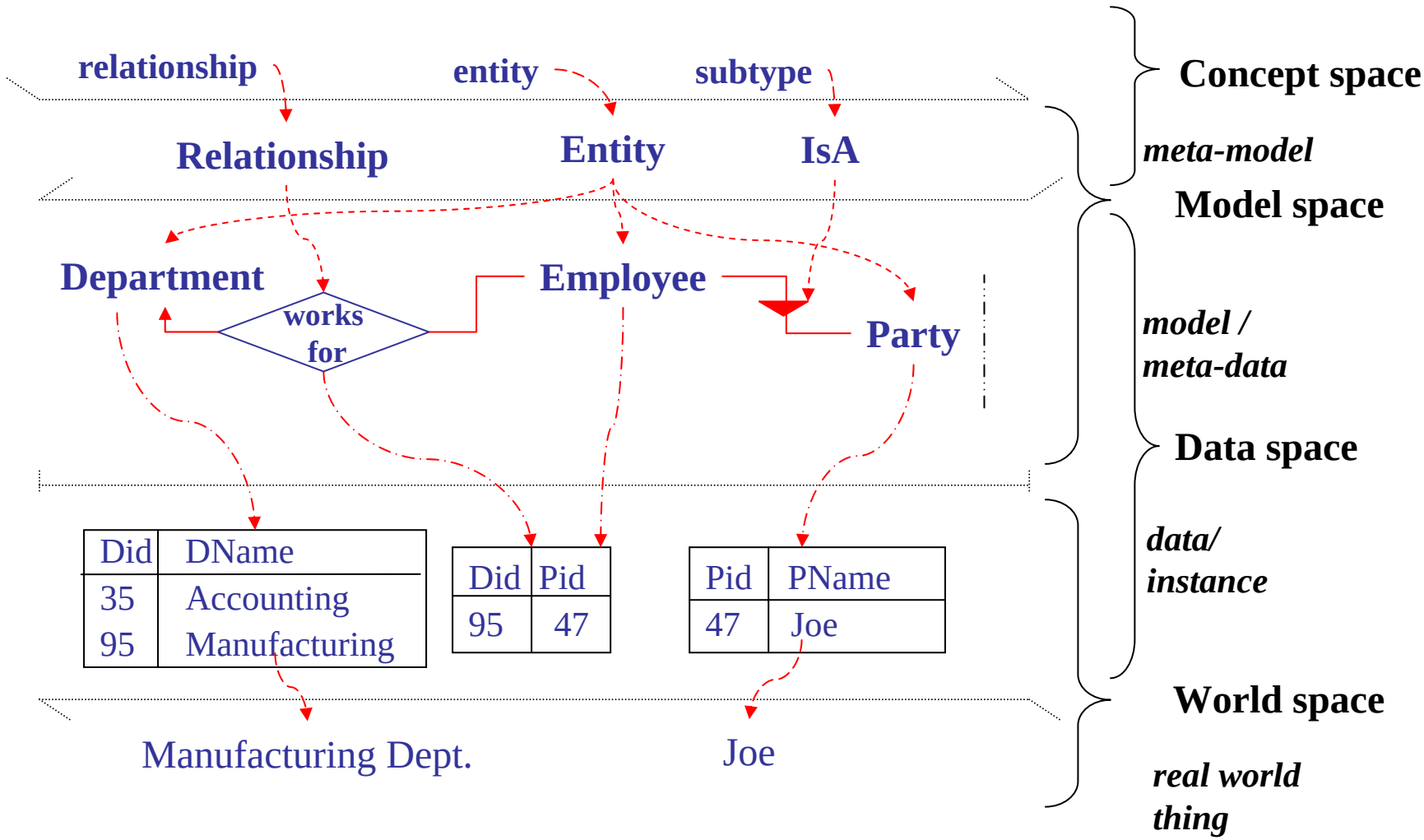
# Scope of View



- view of whole *vs.* view of piece *vs.* ???
- ISO 14439: “view” is of whole
- ISO 14440: “view” is of piece
  - “object view”

# Meta-levels of design





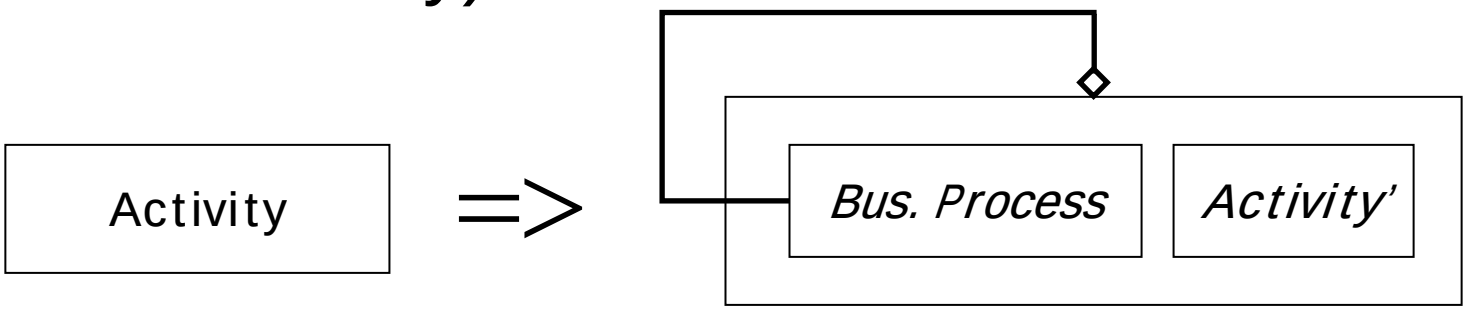
# Views and Meta-levels

- views exist at all meta-levels
  - prominent in IS
    - model level – construct, populate
    - instance level – subset, extract
- view of structure is meta with respect to view of data (e.g. SQL)
- view definition
  - typically at one meta-level
  - should propagate to lower meta-levels
- view update often crosses meta-levels

# View and context dependency



- 'activity' must be view dependent
  - your "assemble activity" (a step in process) may be my "assemble process" ( a sequence of activities to accomplish your assemble activity)




- to achieve a consistent context a view may need to abstract

# View use



- view as image (noun)
- view generation as computation of image (verb)

---

- views to aid user understanding
- views as means for consistency, completeness and interoperability

---

- view of a model as expression of content
- view as a means to add new content

# View use



- view interoperability
  vs. model interoperability
- number of necessary views
  vs. enterprise scope
- what we can view from a model
  vs. what must change in a model
  to satisfy a view

# Relevant standards

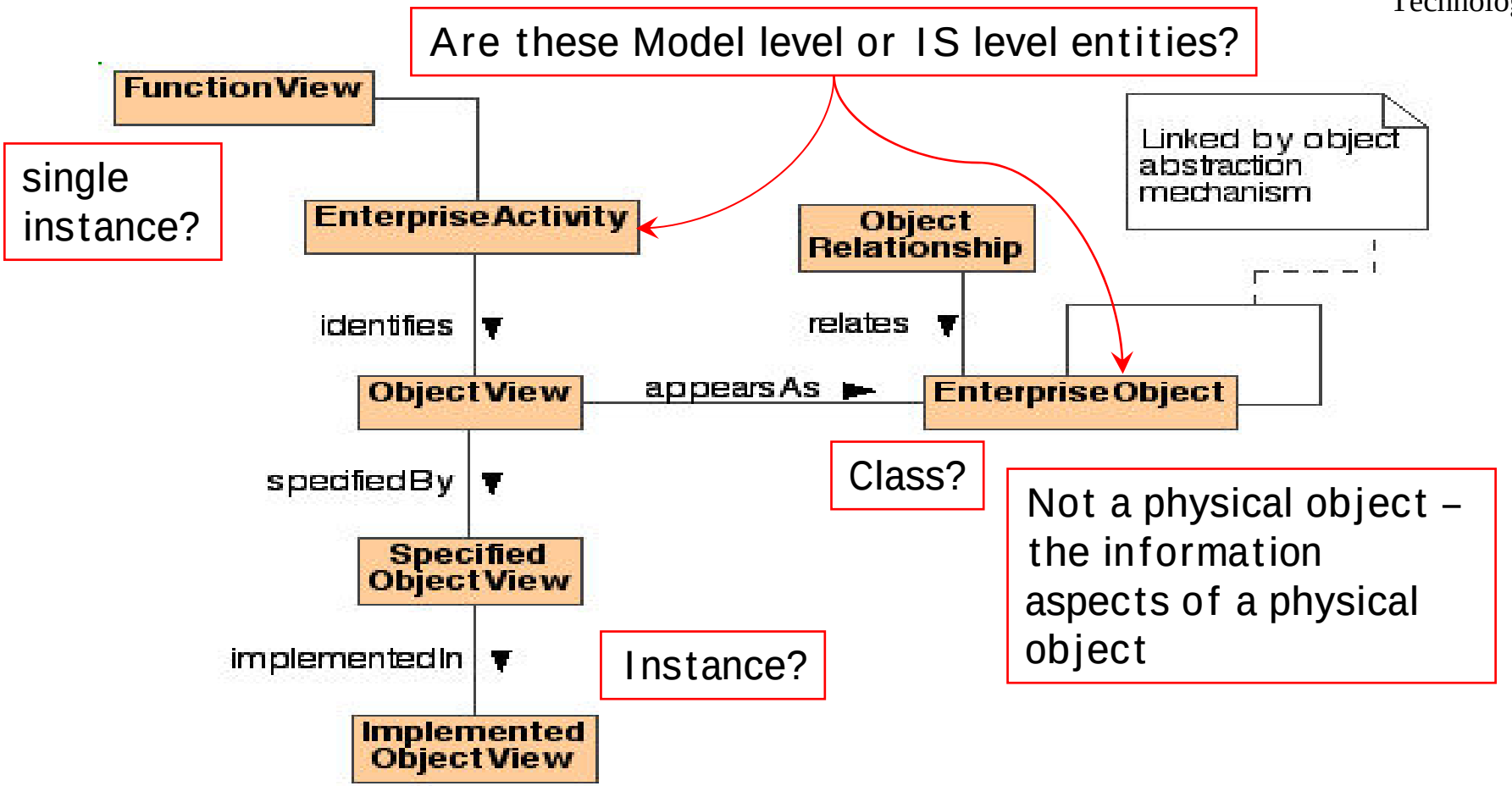
- ISO 14258: Concepts and Rules for Enterprise Models
- ISO 19439: Enterprise Integration – Framework for Enterprise Modeling
- ISO 19440: Enterprise Integration – Constructs for Enterprise Modeling
- IEEE 1471: Recommended Practice for Architectural Description of Software-Intensive Systems

# Uses of "View" in Standards

(don't expect consistency)

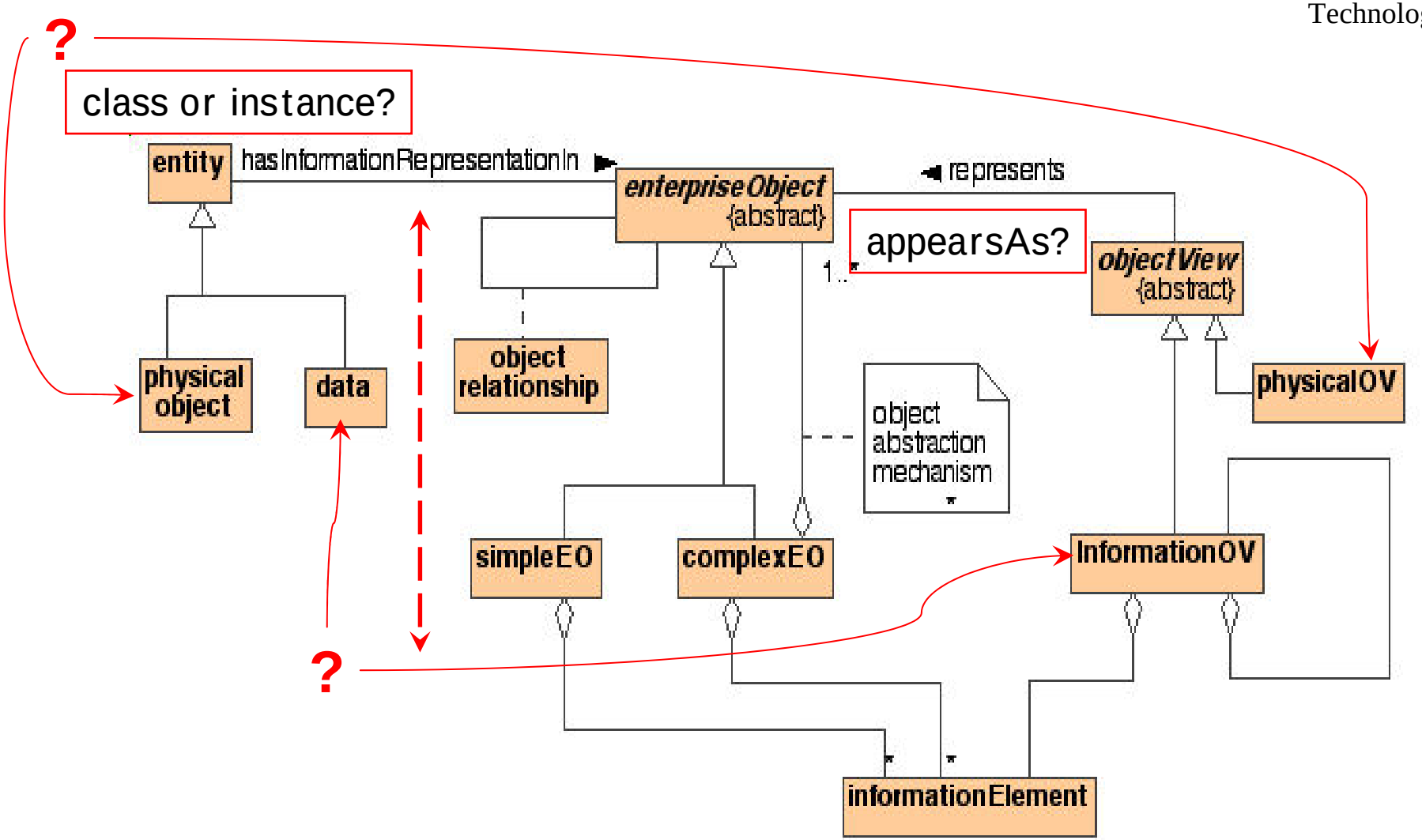
- prescribed modeler views (19439)
- "Object View" (19440)
  - not objects or views in OO sense
  - cannot view an Object View
  - instances are transient
  - instances shift representation
  - instances support processes
  - e.g.: shipping order → pick list

# CIMOSA "object view"



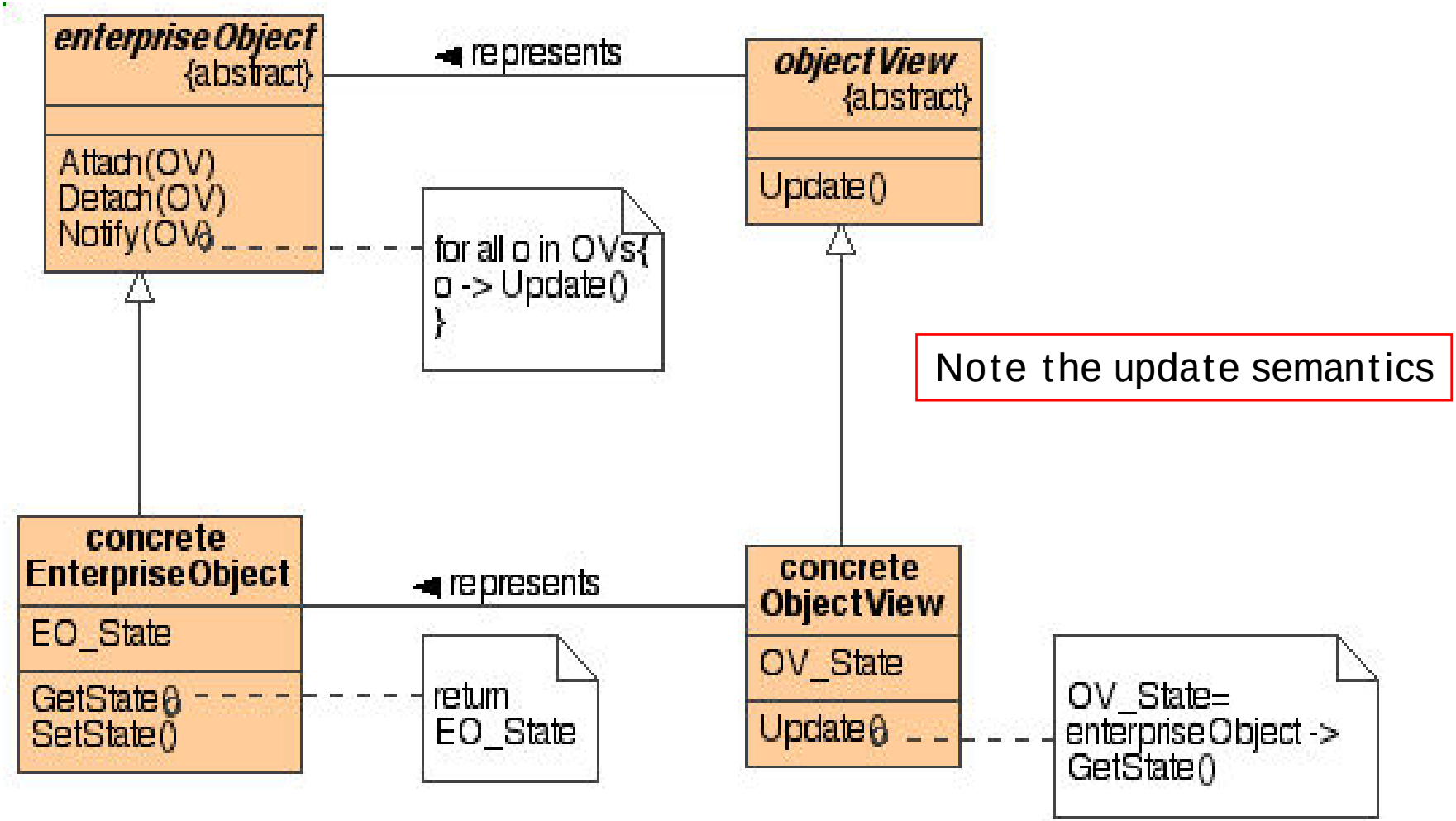

Not about model (noun) but rather about model (verb)

# Complex "object view"



# Object view as observer pattern





# Technical aspects of view



- current standards (19439/19440 in particular) have no general principle for mapping modeling constructs in views
- if communication channels are used to assure consistency among views then any unified view is limited by those messages
- if ontology exists then it brokers the model and any views

# Technical aspects of view

- means to accommodate views without relaxing constraints
  - encapsulation barrier
  - dependency retention
- means to integrate multiple selective views
- means to examine dependency relationships, existence
- means to make a selective view address a particular perception

# View role in change management

- content affected
- relationships affected
- model versions significant
- assessment of responsibility
- access control authorization
- cognitive space / domain examination
- threshold detection
- update constraints

# Views in RDBs

- views for reading ≈ manifest queries
- updating through a view has pitfalls
  - e.g. a class roster is a view but deleting a student from a class should not remove her from the university
- appropriate view updates leave the "complement" unchanged
- equivalently, appropriate updates are those reversible within the view
  - J. Lecthenbörger, PODS 03

# Views in trees/XML

- navigational access – XPATH . . .
  - XQuery analog of SQL view
- results expressed as lists or tree transformations (XSLT)
- trees have order within paths; transformations rearrange that order and may confound navigation
- formal models: tree automata, attribute grammars

# Views in XML



- XML more than trees
  - non-branch associations (XPOINTER)
- views along links – many open issues
- “schema aware” rewriting may facilitate views
  - vs. “schema unaware”
- is there a comparable “navigation aware” notion that would facilitate updates?