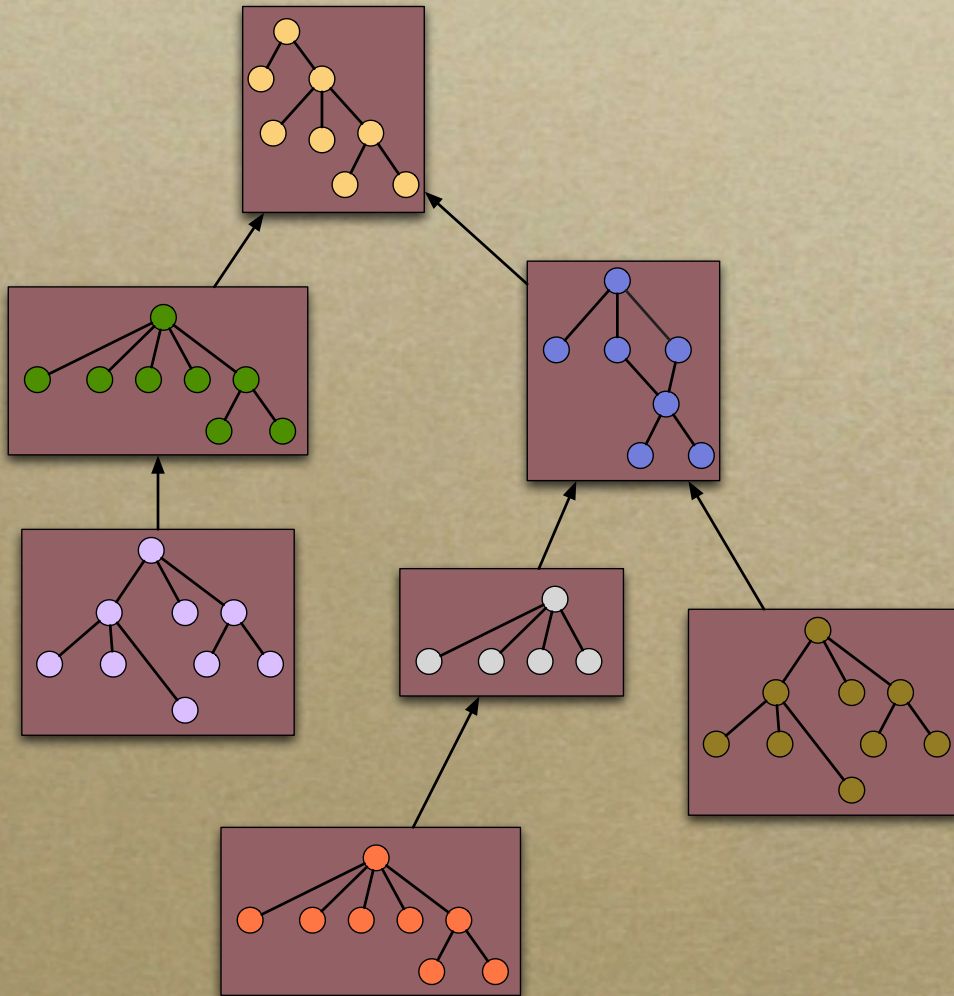


Ontology Management with the Prompt Plugin

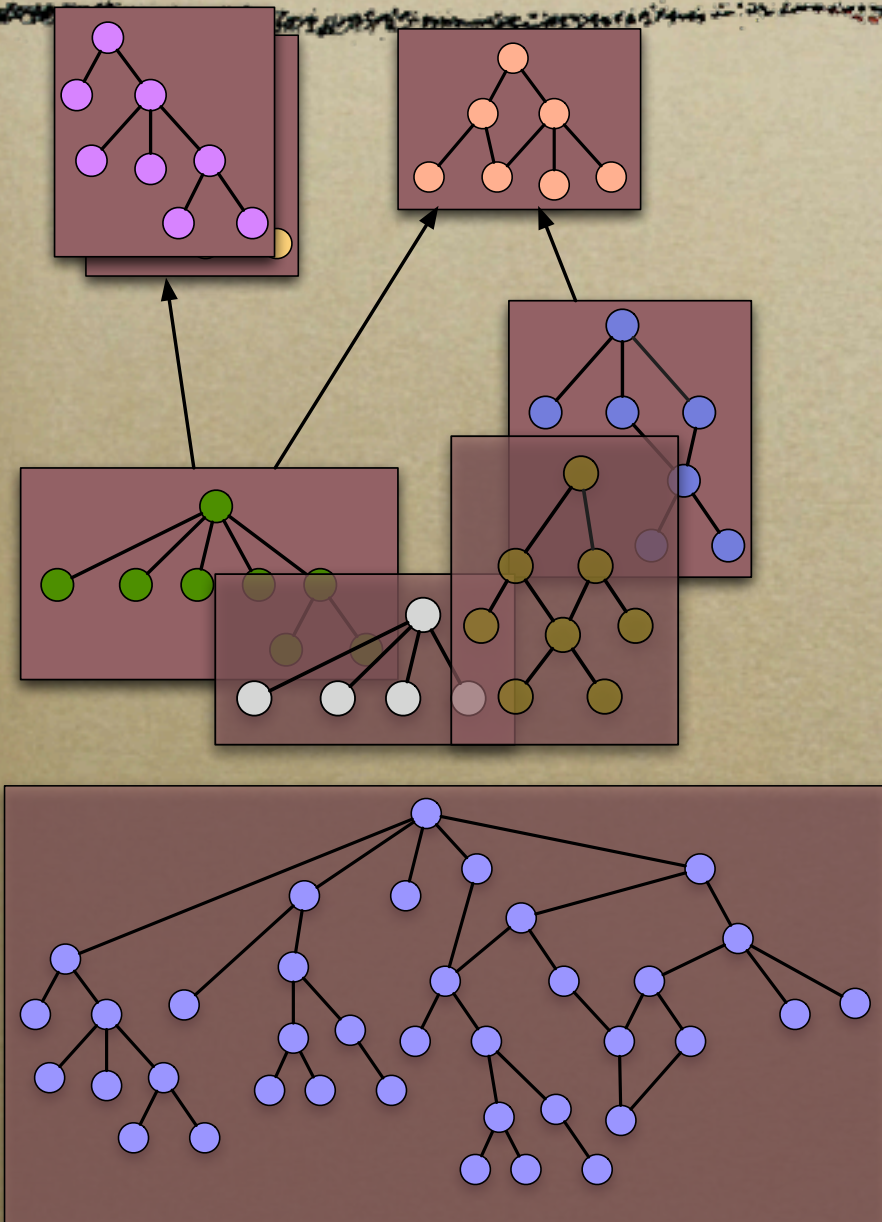
Natasha Noy
Stanford University

The Ideal World



- The same *language*
- No overlap in *coverage*
- No new *versions*
- A single *extension tree*
- Small reusable *modules*

The Real World



- ~~The same language~~
- ~~No overlap in coverage~~
- ~~No new versions~~
- ~~A single extension tree~~
- ~~Small reusable modules~~

What We Need

- *Find similarities and differences between ontologies*

ontology mapping and merging

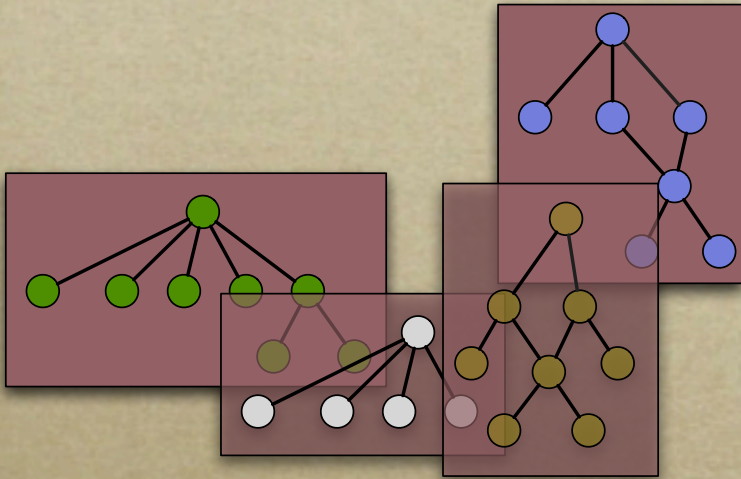
- *Compare versions of ontologies*

ontology evolution

- *Extract meaningful portions of ontologies*

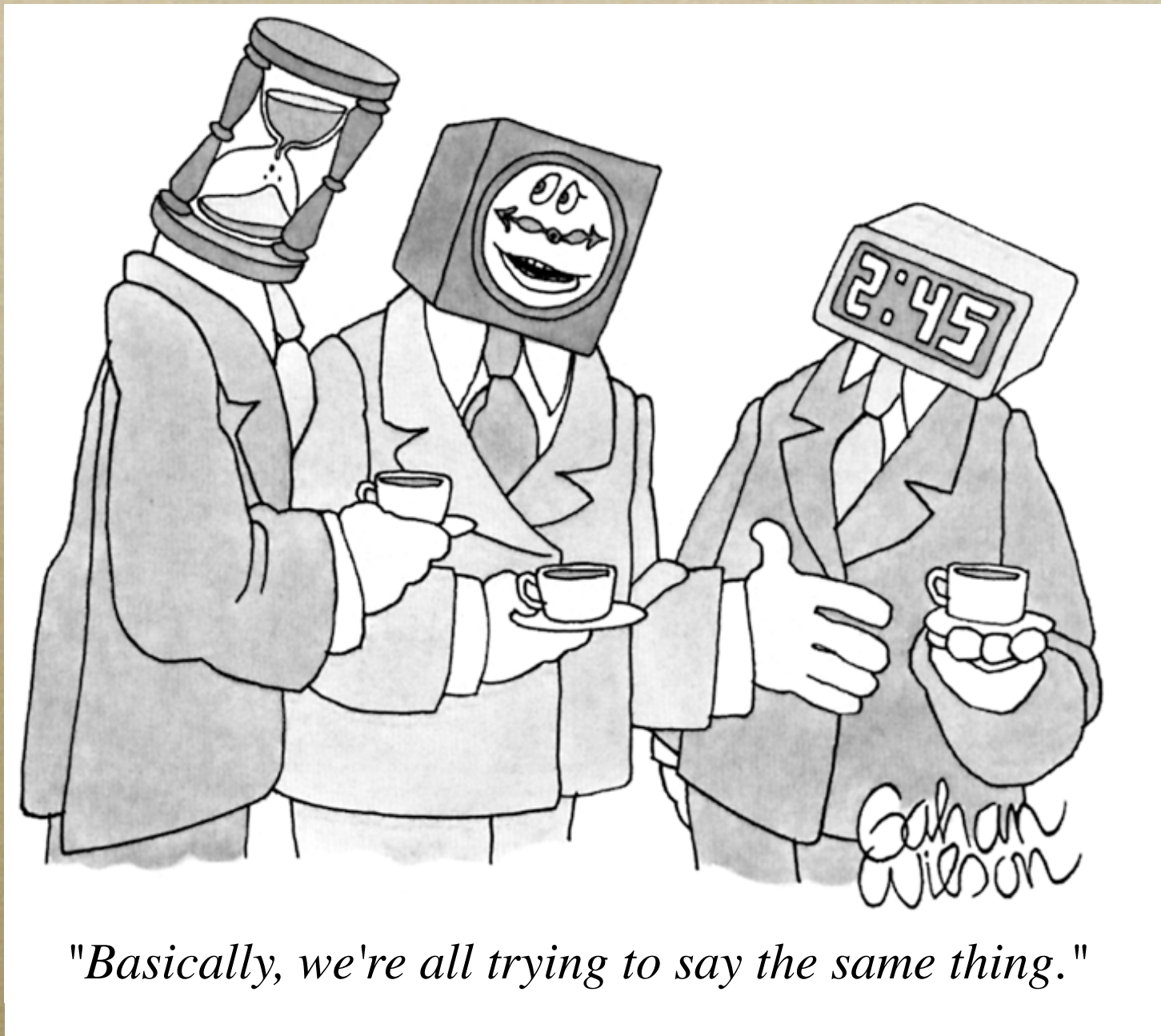
ontology views

Mapping and Merging



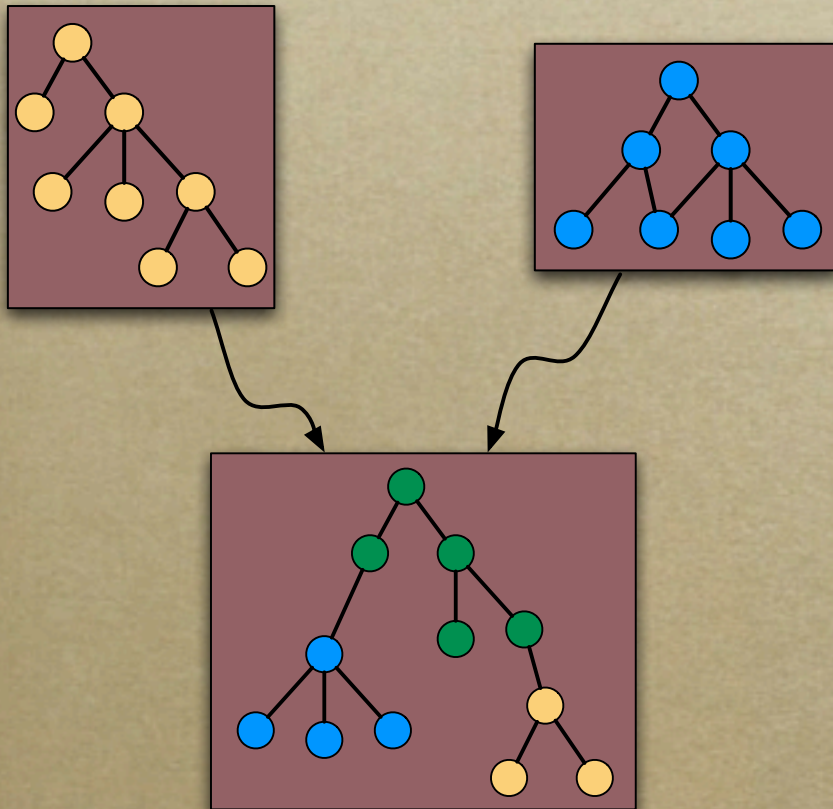
iPrompt
AnchorPrompt

- *Existing ontologies*
 - *cover overlapping domains*
 - *use the same terms with different meaning*
 - *use different terms for the same concept*
 - *have different definitions for the same concept*



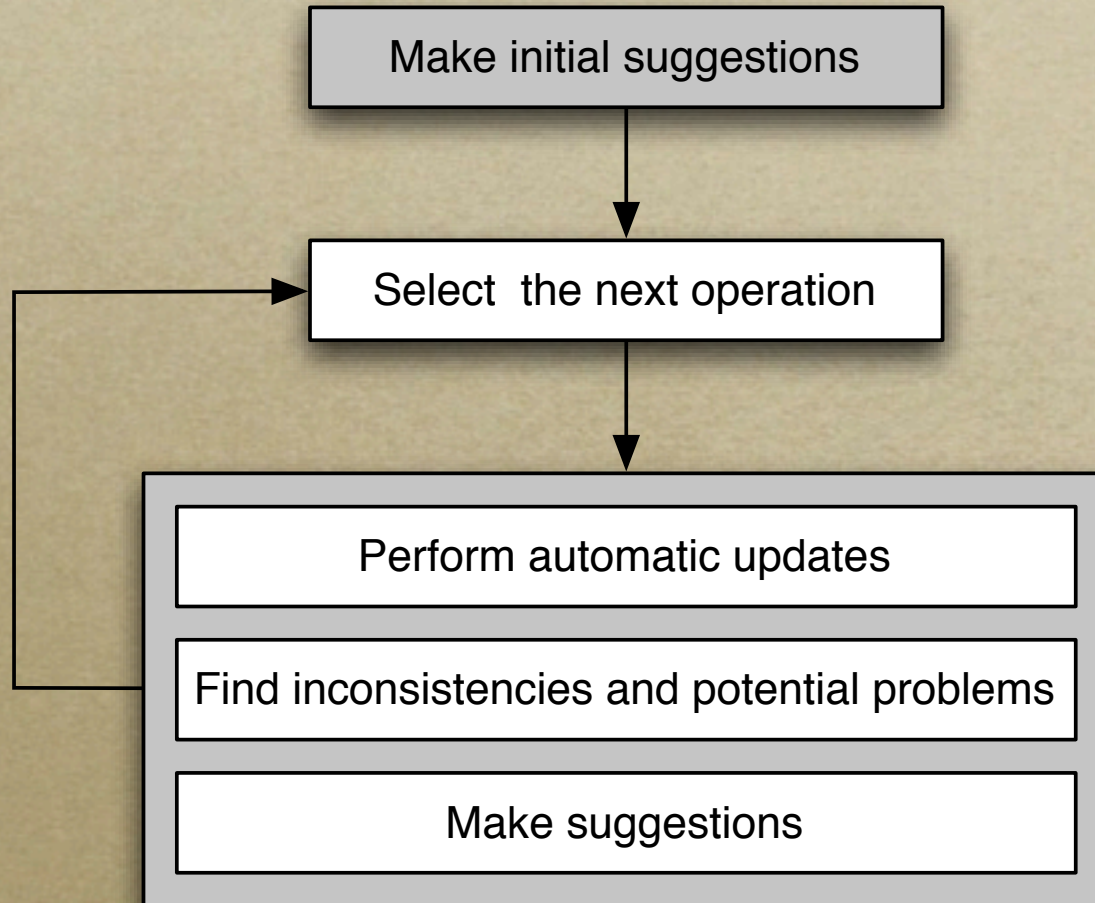
"Basically, we're all trying to say the same thing."

iPrompt: An Interactive Ontology-Merging Tool



- *iPrompt provides*
 - *Partial automation*
 - *Algorithm based on*
 - concept-representation **structure**
 - **relations** between concepts
 - user's **actions**
- *iPrompt does not provide*
 - *complete automation*

iPrompt Algorithm



iPrompt: Initial Suggestions

The screenshot displays the iPrompt interface with several panels:

- Top Navigation:** Classes, Slots, Forms, Instances, Queries, Prompt.
- Sub-panels:** Suggestions, Conflicts, New operations, Result classes.
- To Do list:** A table of merge suggestions with columns for Name, Arg1, Arg2, and a status column.
- Reason for selected suggestion:** A text box containing "frames have identical names".
- Do It:** A button with a green checkmark.
- Result classes:** A list showing the current result classes: :THING and :SYSTEM-CLASS.

Name	Arg1	Arg2	P
merge	Publication cmu	Publication umd	
merge	Employee cmu	Employee umd	
merge	Proceedings cmu	Proceedings umd	
merge	Organisation cmu	Organization umd	
merge	Research_Group cmu	ResearchGroup umd	
merge	Email cmu	Email umd	
merge	Article cmu	Article umd	
merge	Book cmu	Book umd	
merge	Thesis cmu	Thesis umd	
merge	MastersThesis cmu	MastersThesis umd	
merge	PhdThesis cmu	Thesis umd	
merge	Person cmu	Person umd	

Suggestions Conflicts New operations

To Do list

V C X S

Name	Arg1	Arg2	...
copy	Research_Project cmu		
merge	Sex cmu	Gender umd	
copy	Image umd		
merge	age--cmu	age--umd	
copy	Agent umd		
copy	Sex cmu		
merge	Research cmu		
copy	Address umd		
copy	Organism umd		
copy	Gender umd		
merge	Organisation cmu		
merge	Employee cmu		
merge	Research_Group cmu		

Result classes

- current
- :THING
- :SYSTEM-CLASS
- Person

Template Slots

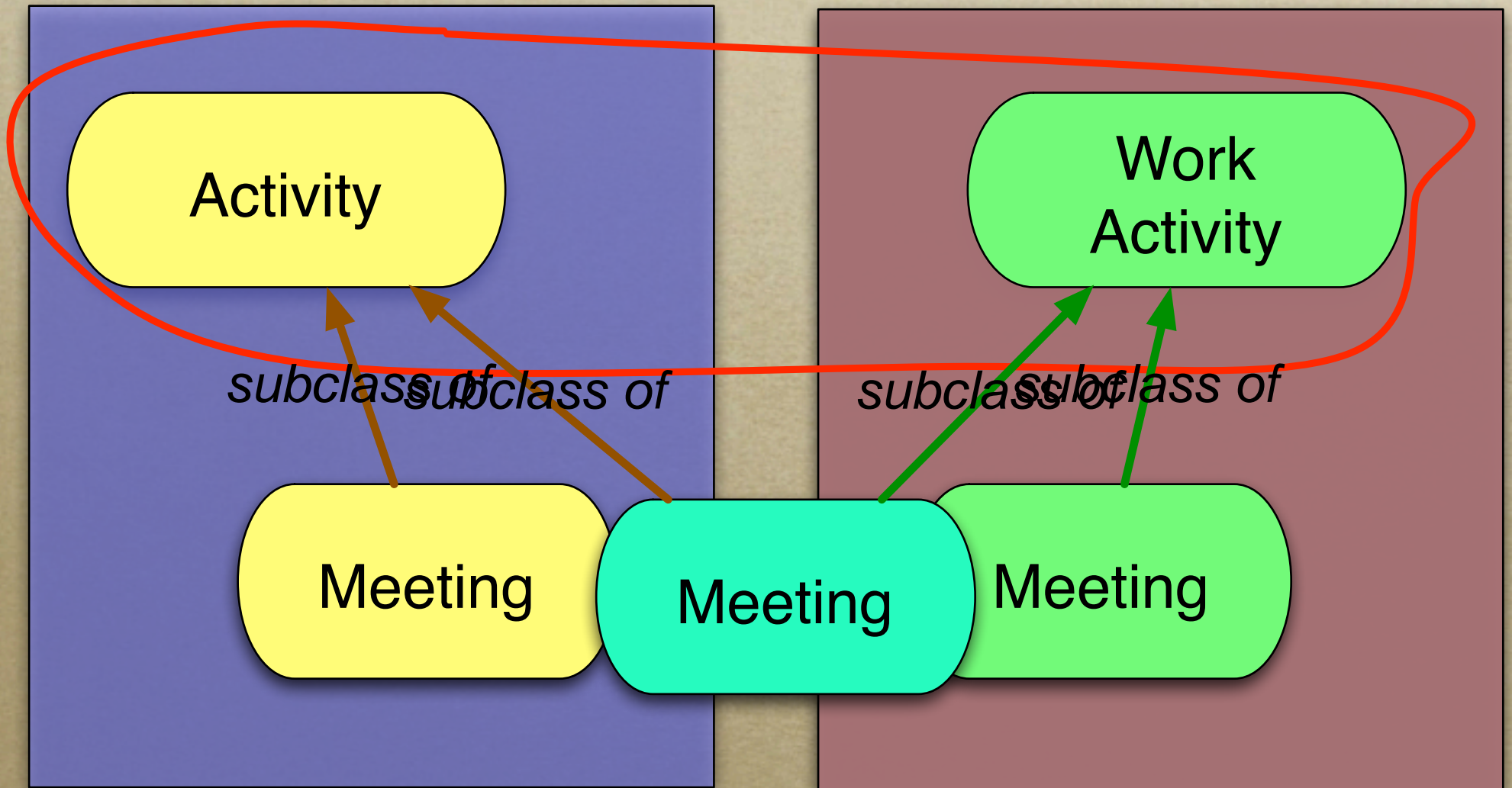
Name	Type	Cardinality	Other Fa
has_employes--cmu	String	multiple	
age--cmu	Integer	multiple	
name--cmu	String	multiple	
friend--umd	Instance	multiple	classes={Person}
father--umd	Instance	multiple	classes={Person}
doctoralDegreeFrom--u...	Instance	multiple	classes={University-

Reason for selected suggestion

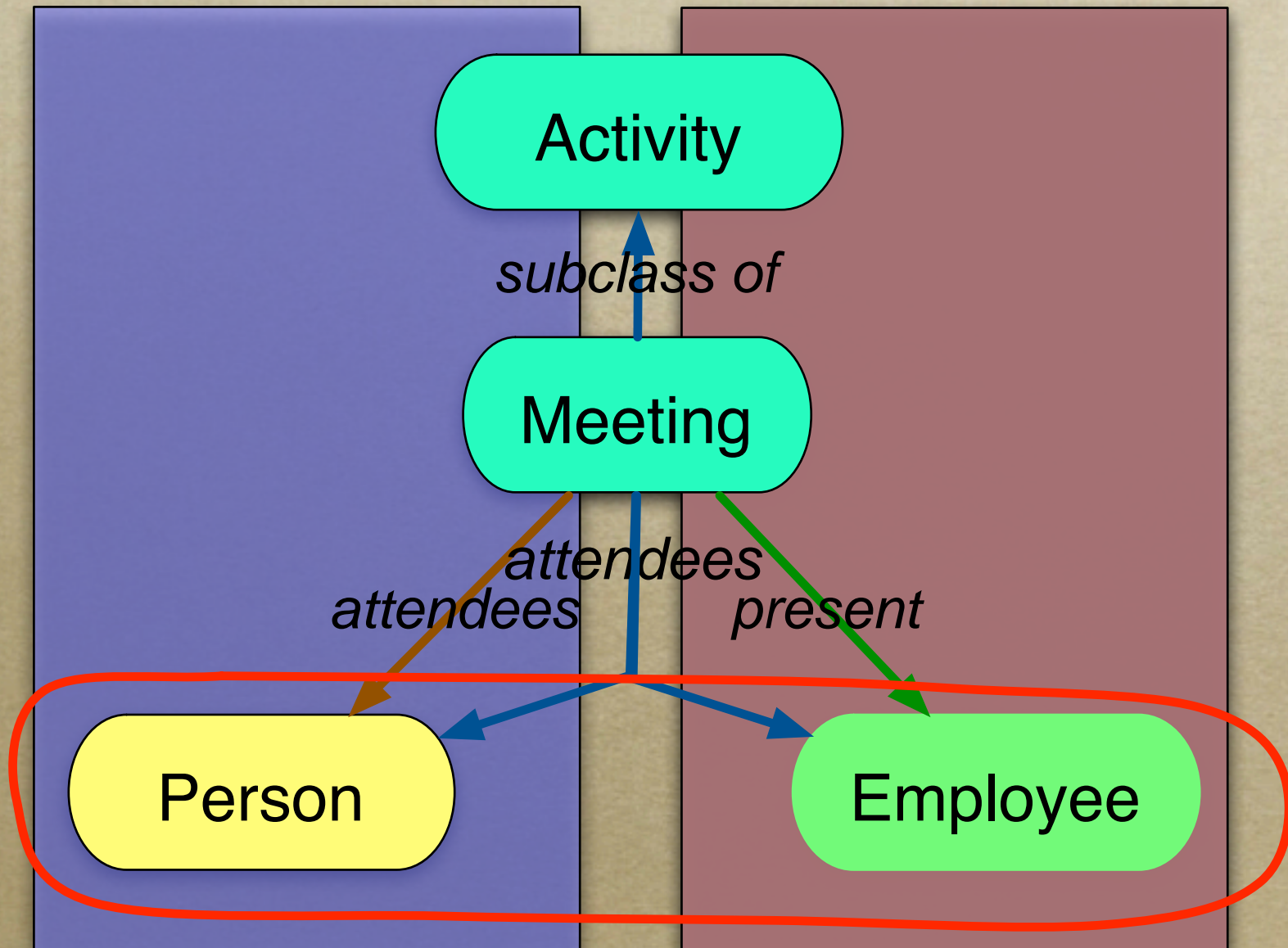
Frame were value types for merged slots that are now sex

Do It

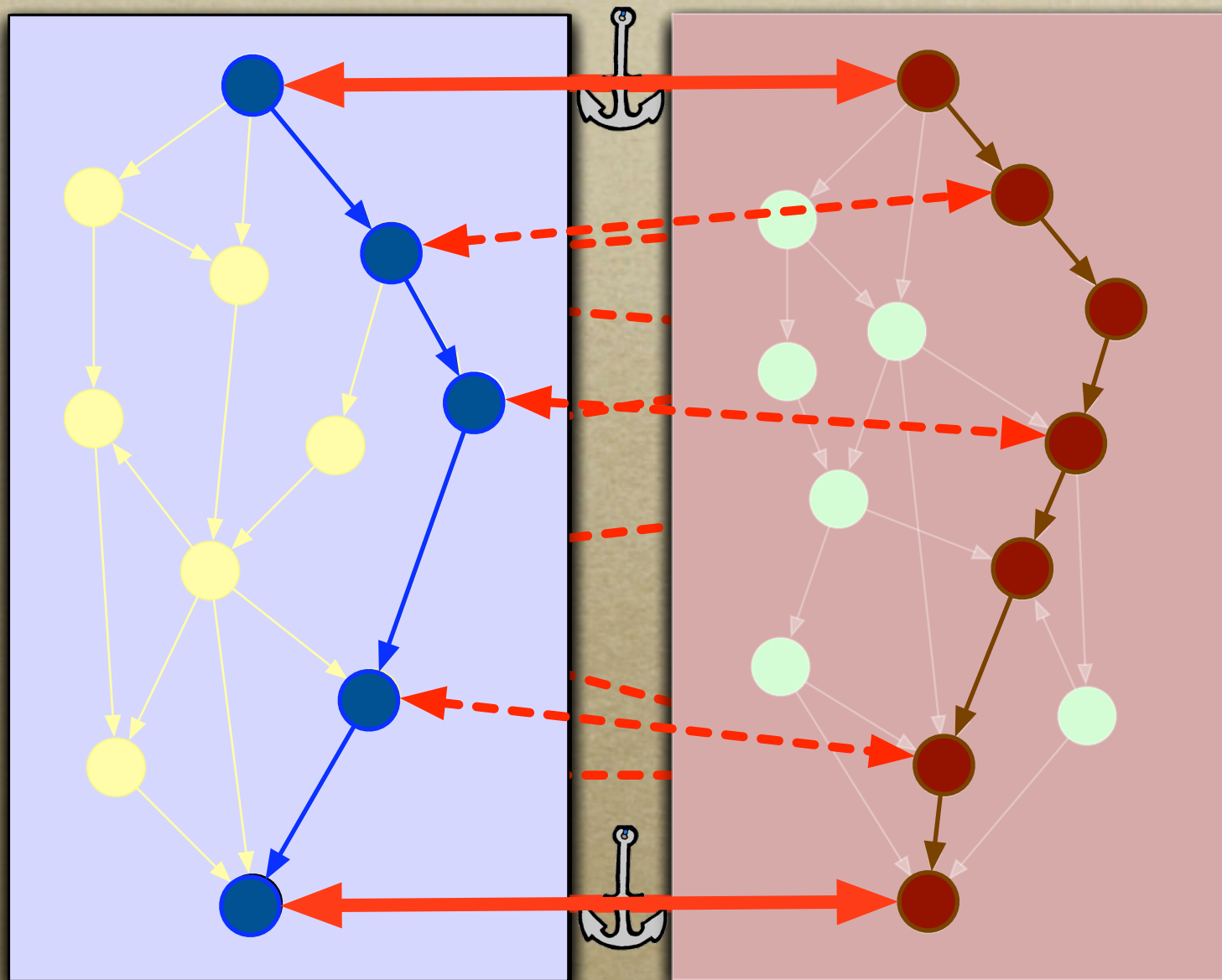
Example: Merge Classes



Example: Merge Classes (II)



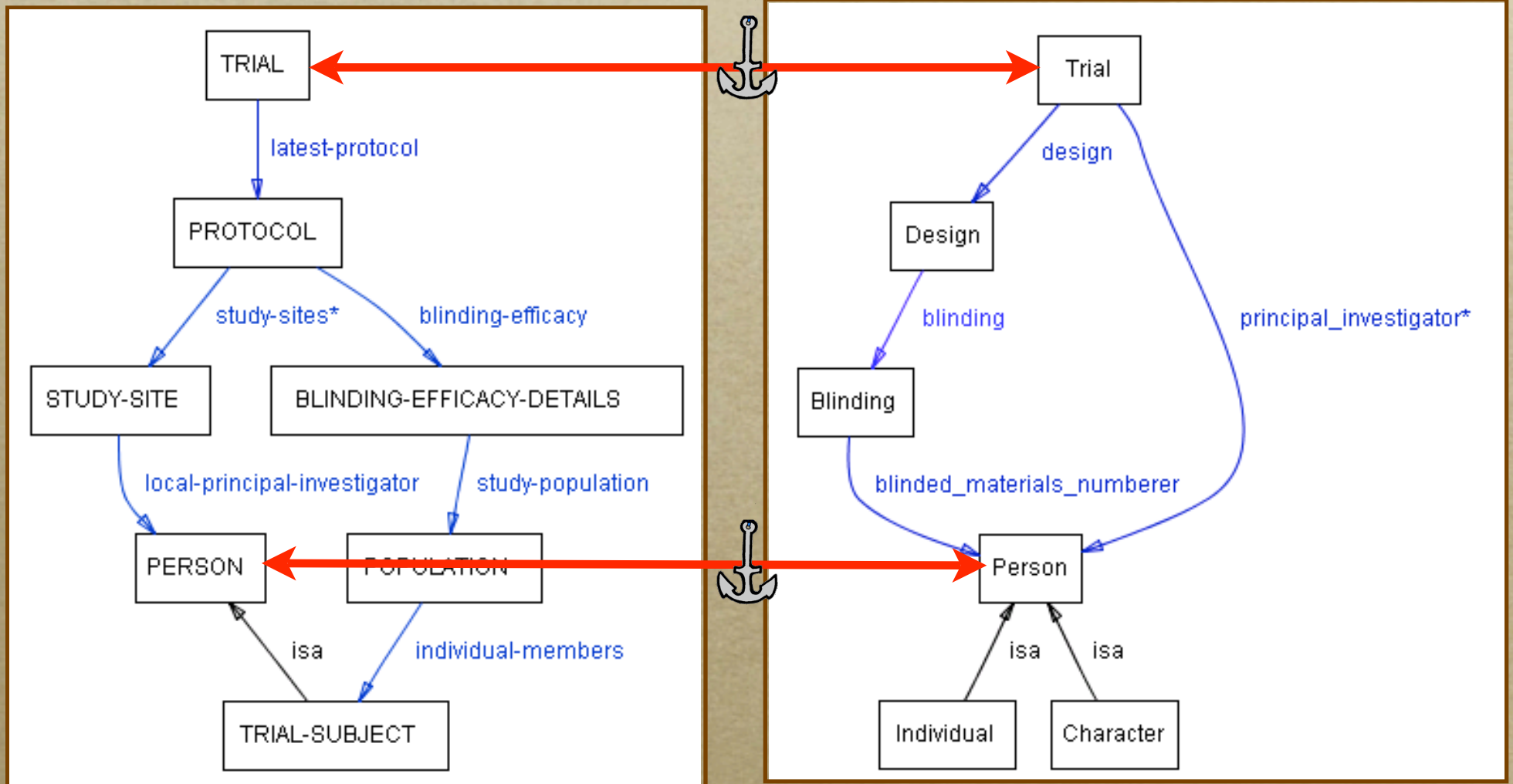
AnchorPrompt: Analyzing Graph Structure



Similarity Score

- *Generate a set of all paths (of length $< L$)*
- *Generate a set of all possible pairs of paths of equal length*
- *For each pair of paths and for each pair of nodes in the identical positions in the paths, increment the similarity score*
- *Combine the similarity score for all the paths*

AnchorPrompt: Example



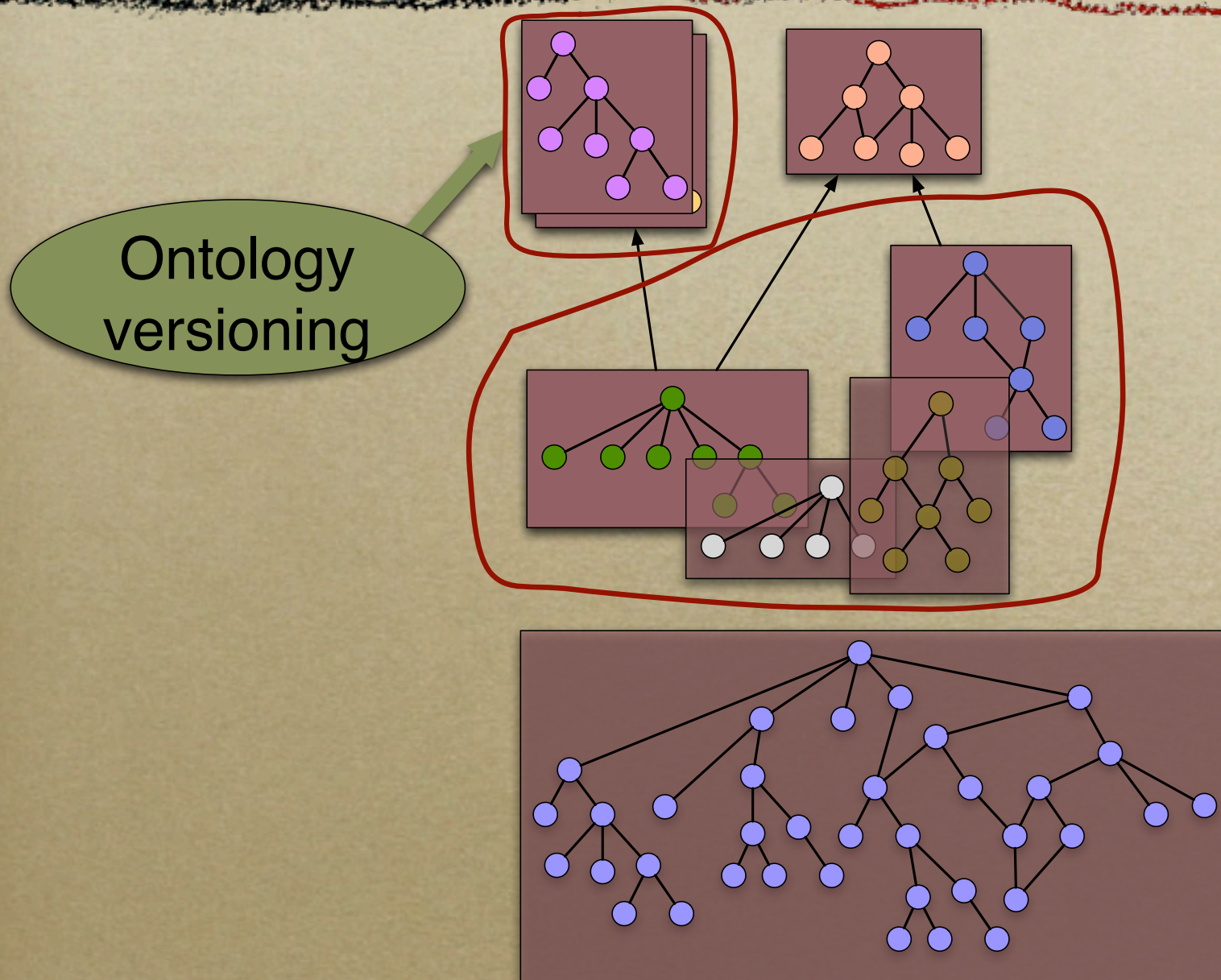
AnchorPrompt: Example

TRIAL	Trial
PERSON	Person
CROSSOVER	Crossover



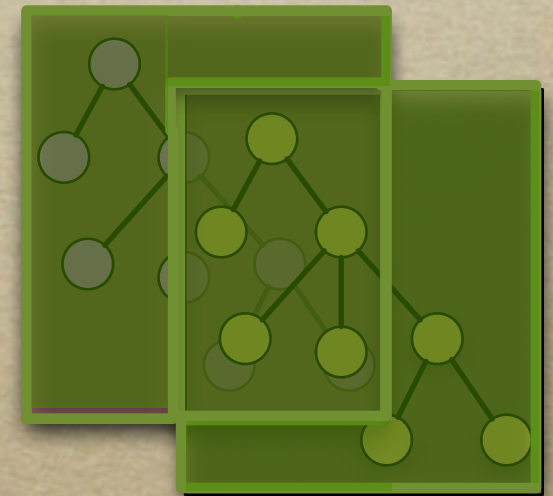
PROTOCOL	Design
TRIAL-SUBJECT	Person
INVESTIGATORS	Person
POPULATION	Action_Spec
PERSON	Character
TREATMENT-POPULATION	Crossover_arm

The Messy Picture



General Problem: Ontology Matching

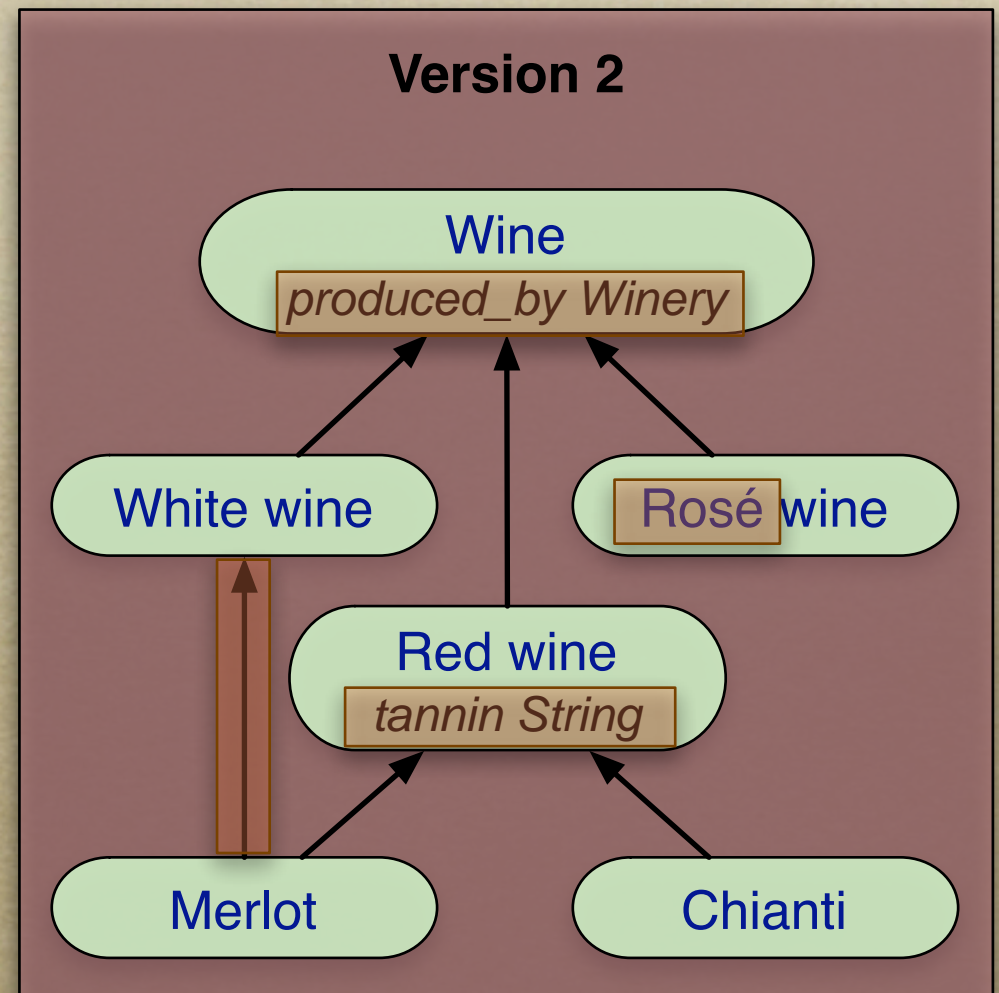
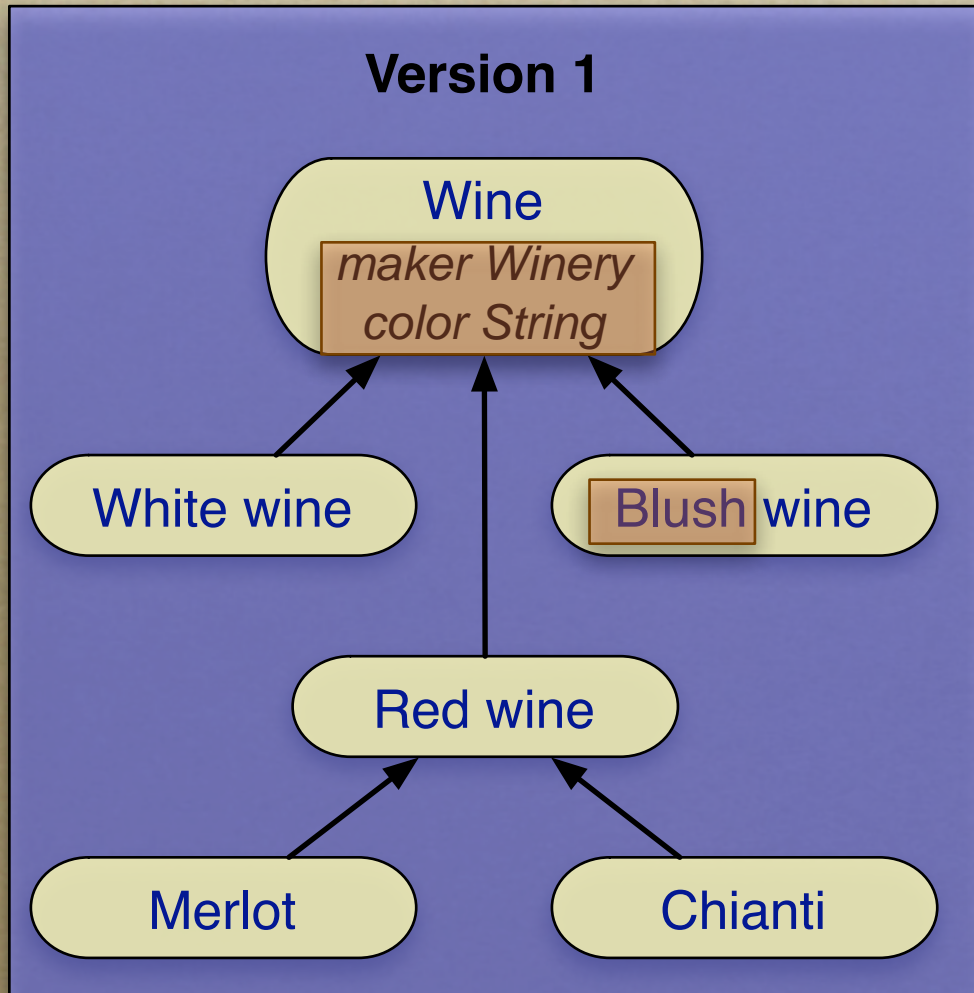
- *Compare ontologies*
- *Find similarities and differences*
 - *Merging: similarities*
 - *Mapping: similarities*
 - *Versioning: differences*
- *Ontology Versioning*
 - *If things look similar, they probably are*
 - *A large fraction of ontologies remains unchanged from version to version*



Ontology Versioning

- *Ontology development became a **dynamic, collaborative** process*
 - *Need to maintain different ontology versions*
- *CVS-type systems*
 - ✓ *Repository of versions*
 - ✓ *Check-in/check-out mechanisms*
 - ✗ *Version comparison (diff)*

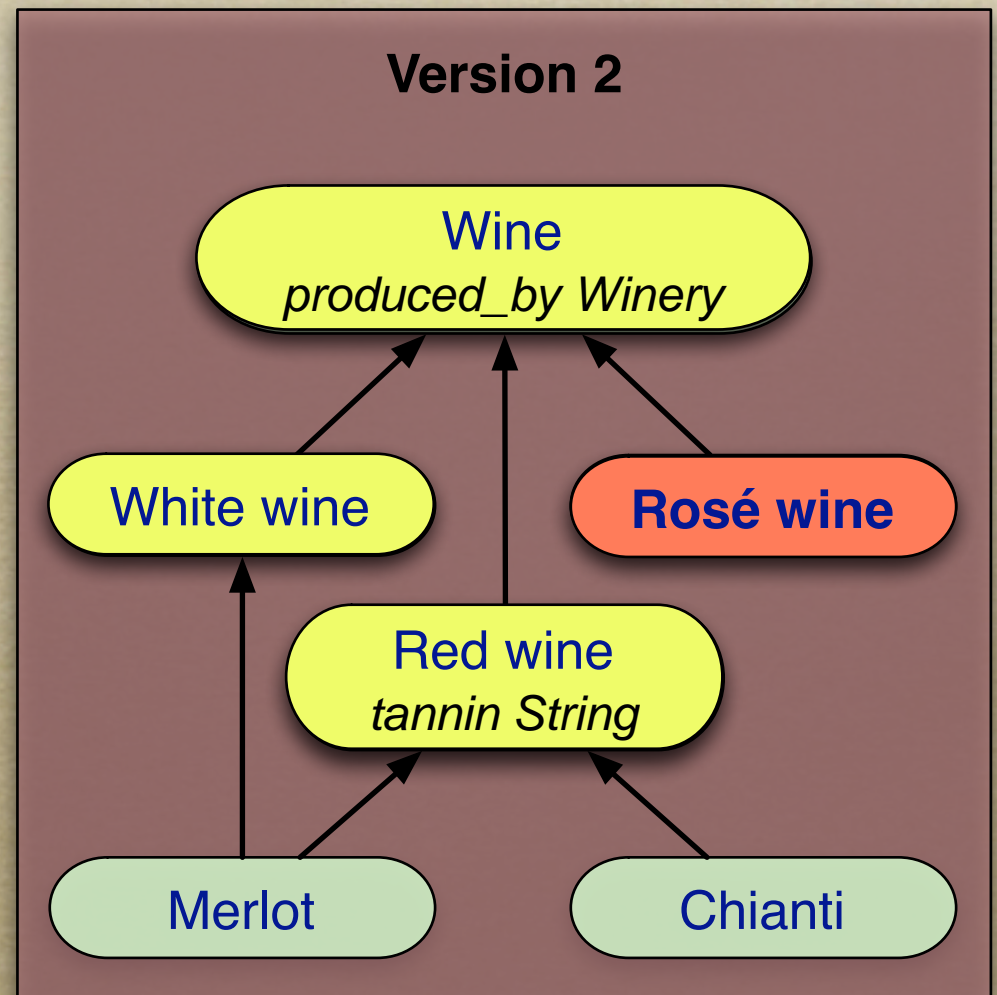
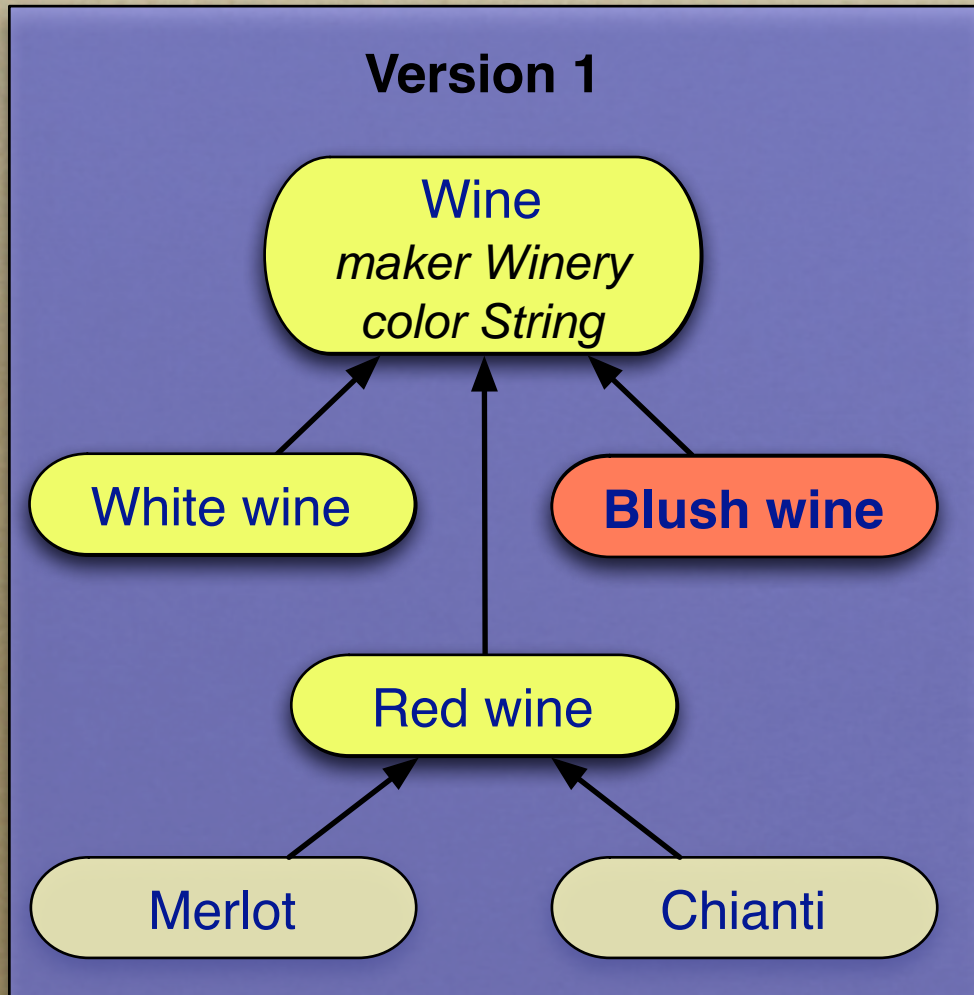
Structural Diff



PromptDiff Algorithm

- *Consists of two parts*
 - *A set of heuristic matchers*
 - *A fixed-point algorithm to combine the results of the matchers*
- *Can be extended with any number of matchers*

PromptDiff Heuristic Matchers



PromptDiff Interface

The interface is divided into three main sections:

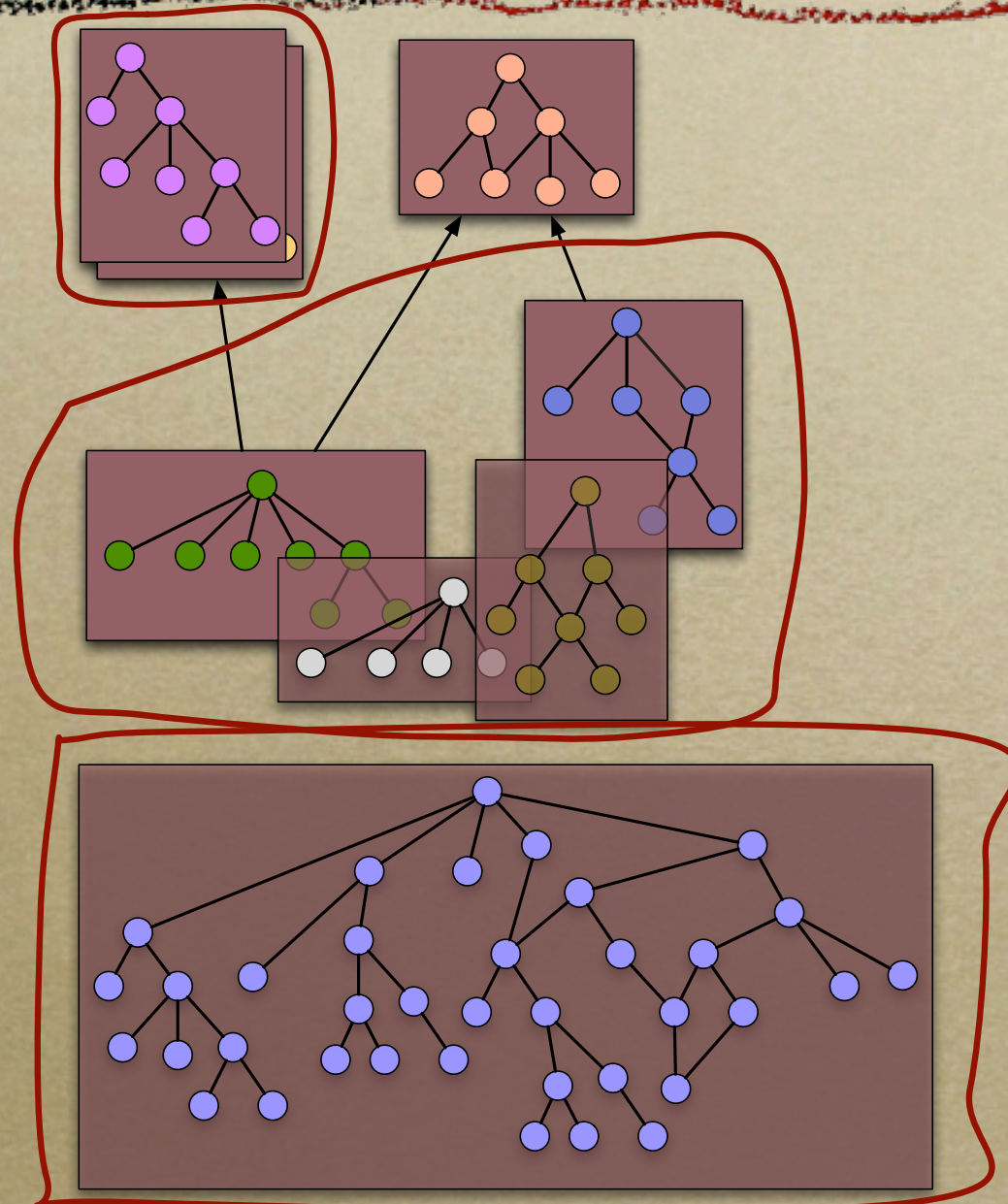
- Left Panel (Class Hierarchy):** A tree view showing a hierarchy of classes. The root is `:THING`, followed by `:SYSTEM-CLASS`, `Wine`, `Red wine`, `White wine`, `Riesling`, `Dry Riesling`, `Sweet Riesling`, `Rosé wine` (highlighted), `Winery`, and `Meal-course`.
- Center Panel (Class Details):** A detailed view of the selected `Rosé wine` class (type=:STANDARD-CLASS). It includes:
 - Name:** A text field containing "Rosé wine" (highlighted with a blue box).
 - Role:** A dropdown menu set to "Concrete".
 - Template Slots:** A table with columns: Name, Type, Cardinality, and Other. It contains one slot: "S produced by" with Type "Instance", Cardinality "single", and Other "classes={Winery}".
 - Differences:** A table showing changes to the class's slots.
- Bottom Panel (Superclasses):** A list of superclasses, currently showing `Wine`.

Differences Table:

Operation	Slot	Face	Old Value	New Value
own slot value changed	S Name		Blush wine	Rosé wine

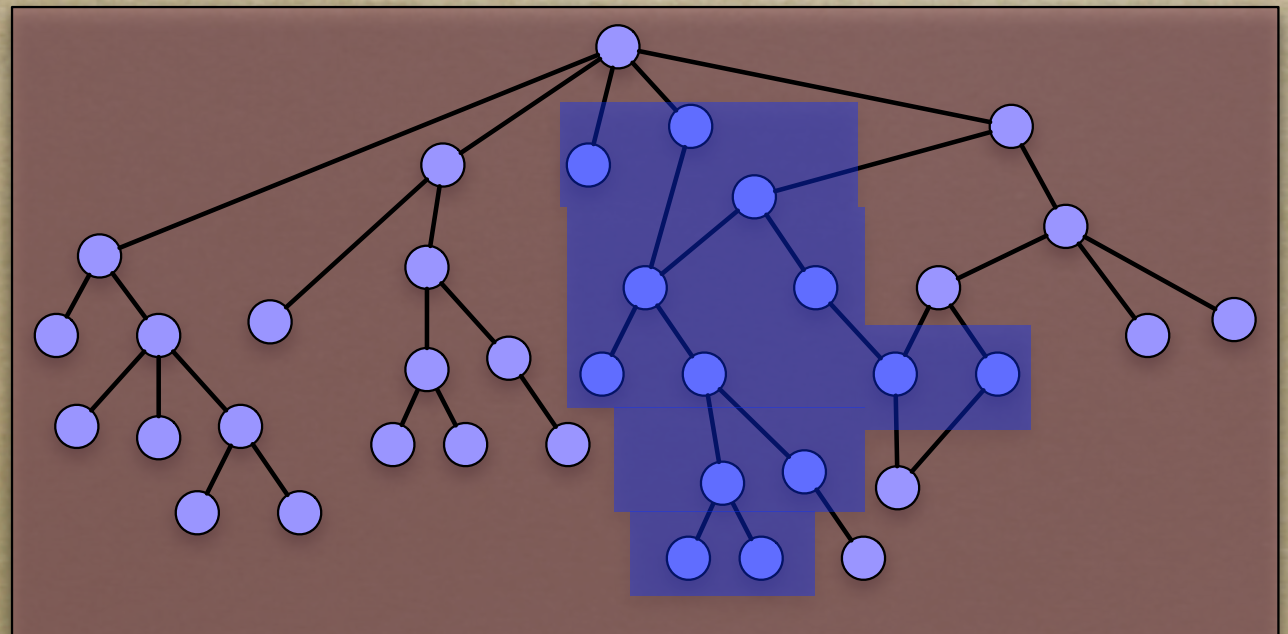
Joint work with Michel Klein and Sandhya Kunnatur

The Messy Picture



Ontology Views

- *Extract a self-contained subset of an ontology*
- *Ensure that all the necessary concepts are defined in the sub-ontology*
- *Specify the depth of transitive closure of relations*



Defining a View

The image shows a software interface with two main windows. The background window displays a class hierarchy for 'fm_localJuly28'. The hierarchy is as follows:

- fm_localJuly28
 - Cavitated organ
 - Organ with organ
 - Organ with cavita
 - Heart
 - Bone (organ)
 - Long bone
 - Short bone
 - Flat bone
 - Irregular bone

The 'Heart' class is selected. Below the hierarchy, there is a 'copy class' section with a dropdown menu set to 'fm_localJuly28'. There are buttons for 'Choose class', 'View', 'List', 'Add', and 'Remove'. Below these are checkboxes for 'subclass', 'instance', and 'superclass', with 'superclass' checked. A 'Do It' button is at the bottom.

The foreground window is titled 'Slot traversal depth'. It contains the following text: 'Number of levels to copy for specific slots (checking the box and leaving the field blank sets levels to "unlimited")'. Below this text is a list of slots with checkboxes and input fields:

- every slot
- S connected to
- S connecting part
- S connection type
- S constitutional part
- S constitutional part of 4
- S contained in 4
- S contains
- S continuous with

At the bottom of the dialog are 'Select All' and 'Deselect All' buttons, and 'OK' and 'Cancel' buttons at the very bottom.

Saving a View

- *Save a view as a Protégé ontology*
- *Replay the view on a new version*
- *Determine if a view is “dirty”*

Starter Concept
Heart

Subclasses Instances Superclasses

Everything Number Of Levels

Slot Directives

slot	depth
constitutional part of	4
contained in	4

Dealing with a Messy World

