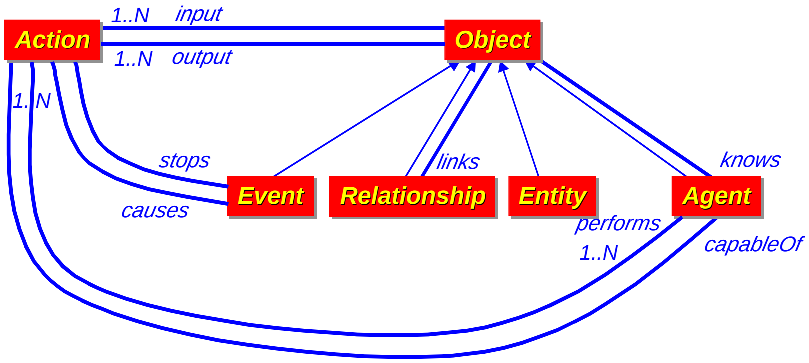
Telos: Representing Knowledge About Information Systems

History
Propositions
Attributes as Objects
Knowledge Base Operations
Metaclasses

History

- ***RML:Requirements Modeling Language*** *[Greenspan84] -- structuring facilities, assertions, time, influenced by SADT™; never implemented, but widely referenced.*
- ***CML:Conceptual Modeling Language*** *[Stanley86, Koubarakis88] -- attributes as objects for extensibility.*
- ***Telos*** *[Koubarakis89, Mylopoulos90] -- language for modeling requirements, design, implementation, design decisions, etc.; used in the DAIDA project; cleaned up version of CML, including a tractable model of time.*
- ***Objective****: Define a modeling language which is expressive enough for defining other modeling languages, e.g., EER, KAOS, RML,…*

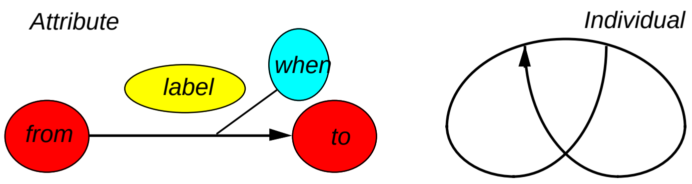
Part of the KAOS Metamodel



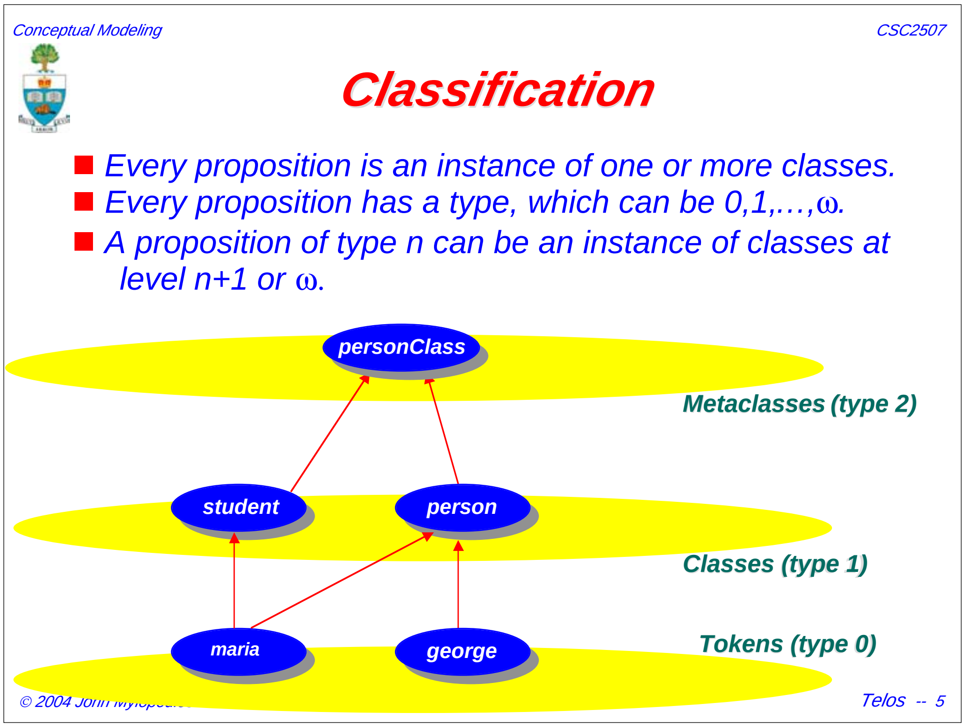
But, where do we define "Action", "Object", etc.?
What about "causes", "stops",...?

Propositions

- Everything is a **proposition** (***unit*** might have been a better name); there are two kinds of propositions: individuals and attributes
- **Individuals (units/objects)** -- John, 7, Person,…
- **Attributes (links/roles/binary relationships)** --
[John, homeAddr, '42 Elm Street', 23/12/75-04/08/98]
[Person, address, GeographicLocation, AllTime]



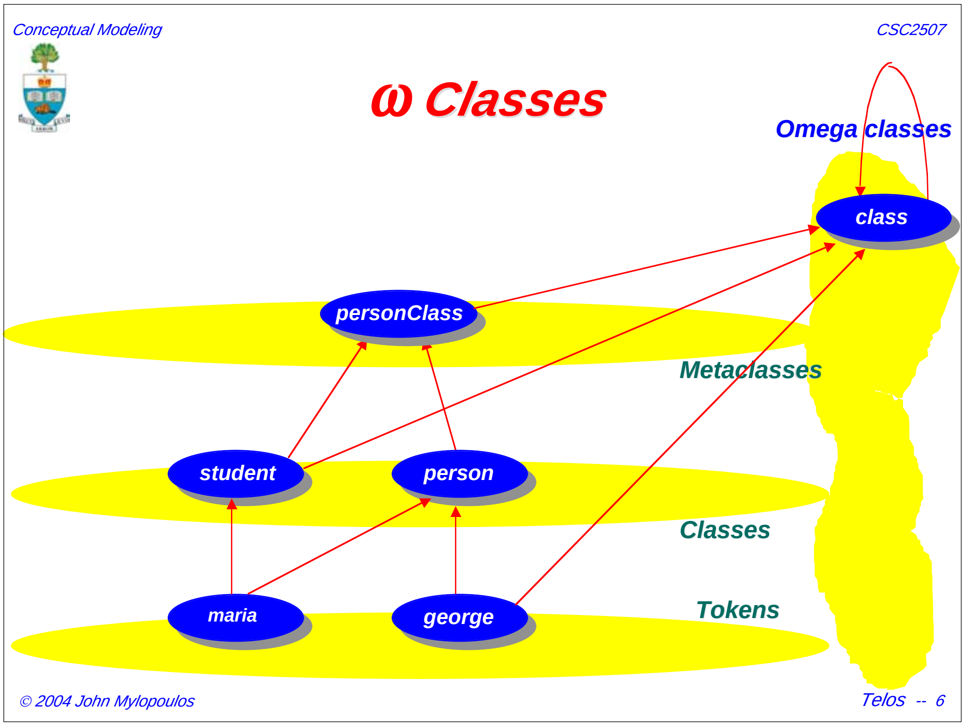
[Note: Time will be ignored in the rest of the discussion]



Classification

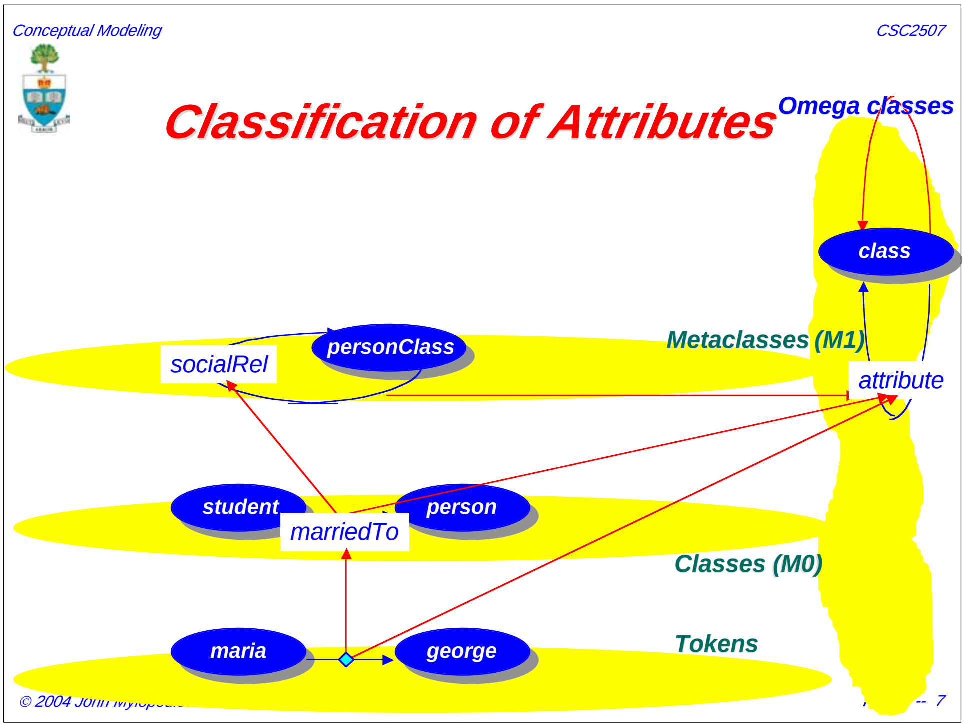
- Every proposition is an instance of one or more classes.
- Every proposition has a type, which can be 0,1,…,ω.
- A proposition of type n can be an instance of classes at level n+1 or ω.

personClass
Metaclasses (type 2)
student
person
Classes (type 1)
maria
george
Tokens (type 0)

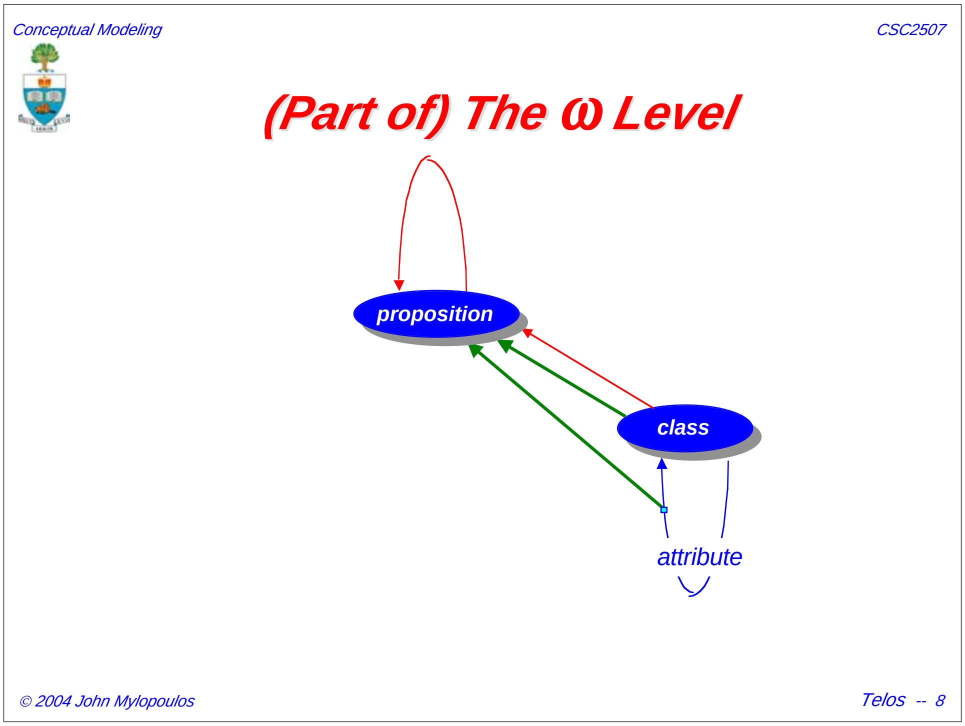


ω Classes

Omega classes
class
personClass
Metaclasses
student
person
Classes
maria
george
Tokens

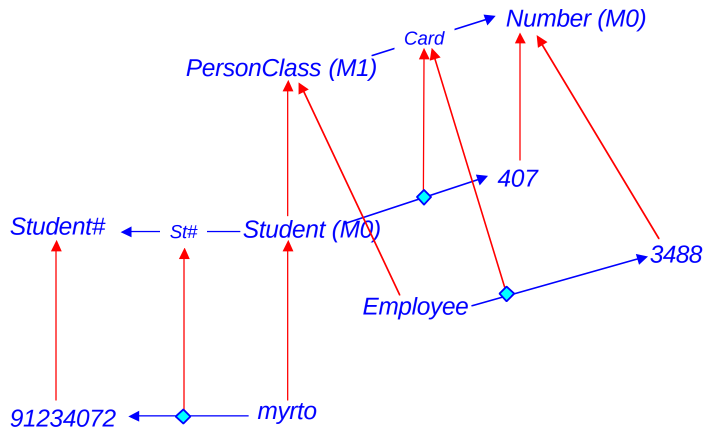


Classification of Attributes



(Part of) The ω Level

Attributes



Operations

- We need operations for creating, updating and deleting propositions.
- We'd like such operations to be at a level that is higher than that of single propositions.
- Every proposition can be thought of as a **composite object** consisting of itself and all attributes that have it as source

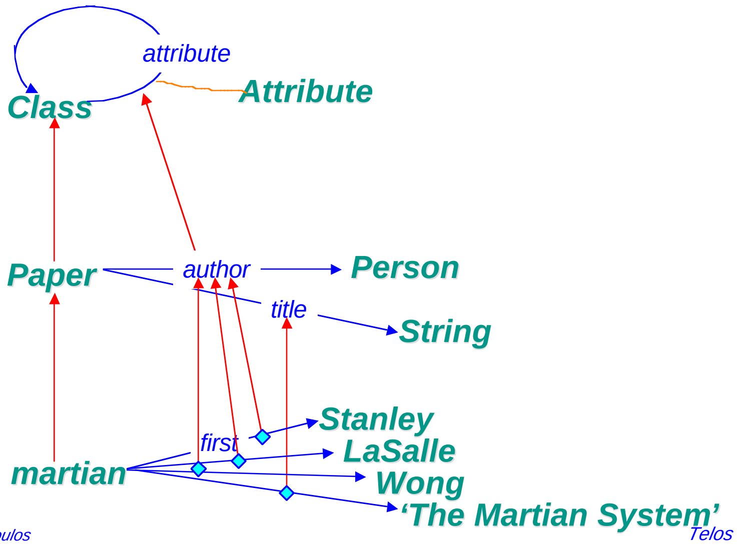
  e.g., {Maria, [Maria, homePhone#, 463-2375],
  [Maria, sal, $65,000] }
  {Person, [Person, phone, Phone#],
  [Person, sal, $-Value]}

Defining Composite Objects

```
TELL CLASS Paper
    IN SimpleClass
WITH
    attribute
      author: Person;
      referee: Person;
      title: String;
      pages: 1..100;
      conf: Conference
END Paper
```

```
TELL TOKEN martian IN Paper
   WITH
        author
             first: Stanley;
                  : LaSalle : Wong
        title *: ‘The MARTIAN System’
        session *: ‘Applications’
        nationality *: Canada
   END martian
```

What Do We Have So Far?



More Composite Objects

TELL CLASS AcceptedPaper
IN SimpleClass
ISA Paper
WITH
    attribute
        sess: ConfProSession
END AcceptedPaper

TELL CLASS InvitedPaper
IN SimpleClass
ISA Paper
WITH
    attribute
        sess: ConfSession;
        nationality: Country
END InvitedPaper

Accessing the Telos Graph Structure

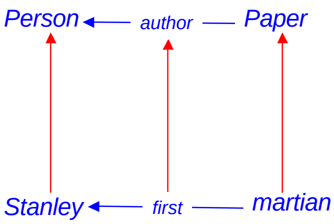
- There are four basic selectors for accessing the Telos graph structure:

    martian.author = {Stanley, LaSalle, Wong}

    martian|author = {[martian,first,Stanley],[martian,-,LaSalle],[martian,-,Wong]}

    martian^first = {Stanley}

    martian:first = {[martian,first,Stanley]}



Metaclasses

- This is perhaps the most distinctive feature of Telos.
- Metaclasses can be defined for individuals and attributes and allow the definition of “models”

  e.g., relation (or table) for the relational model

  entity, relationship for E-R models

  object, message for O-O models

Metaclasses for a Document World

Suppose we want to model a world of documents. A document model will offer built-in terms such as Document, Form, each with an appropriate internal structure. In Telos you can define these as metaclasses:

```
TELL CLASS DocumentClass
IN MetaClass WITH
   attribute
     source: AgentClass;
     content: SimpleClass;
     audience: AgentClass;
     responseTime: Time in SimpleClass
END DocumentClass
```

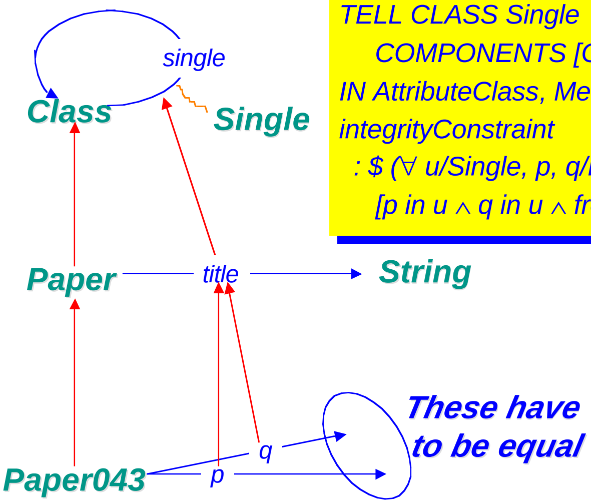
Categorizing Attributes

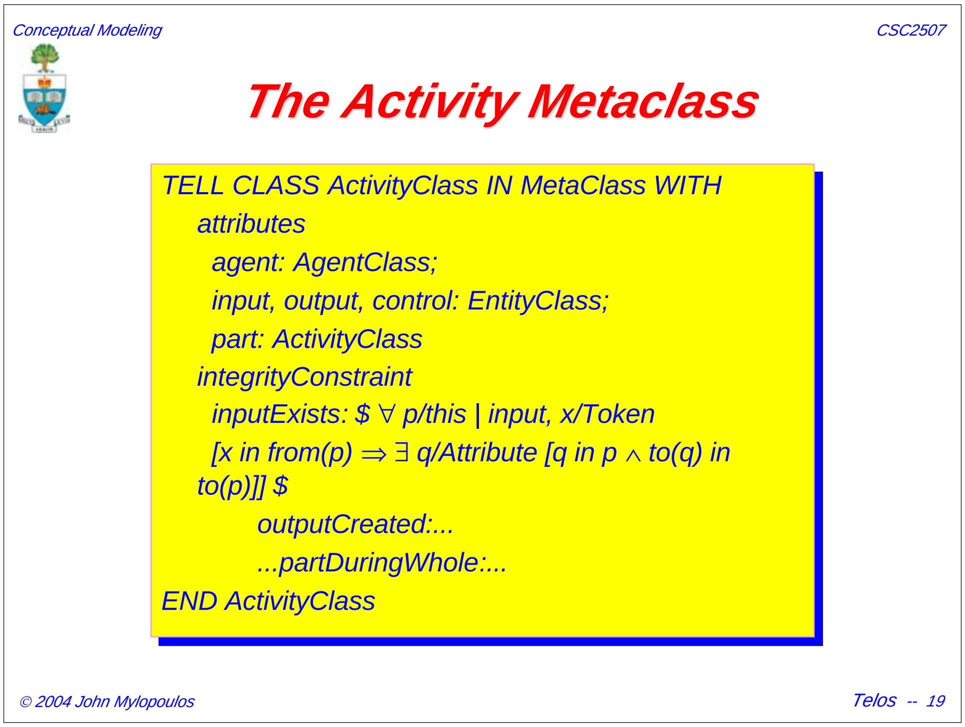
```
TELL Paper IN DocumentClass WITH
    source
      author: Person;  replyAddress:
    Address
    audience
      conf: Conference
    contents
      title: String;  pages: 1..100
    attribute
      referee: Person
      responseTime *: 4wks
END Paper
```

```
TELL CLASS Paper
IN SimpleClass
WITH
   ...
   single, contents
        title: String
   ...
END Paper
```

Single-Valued Attributes

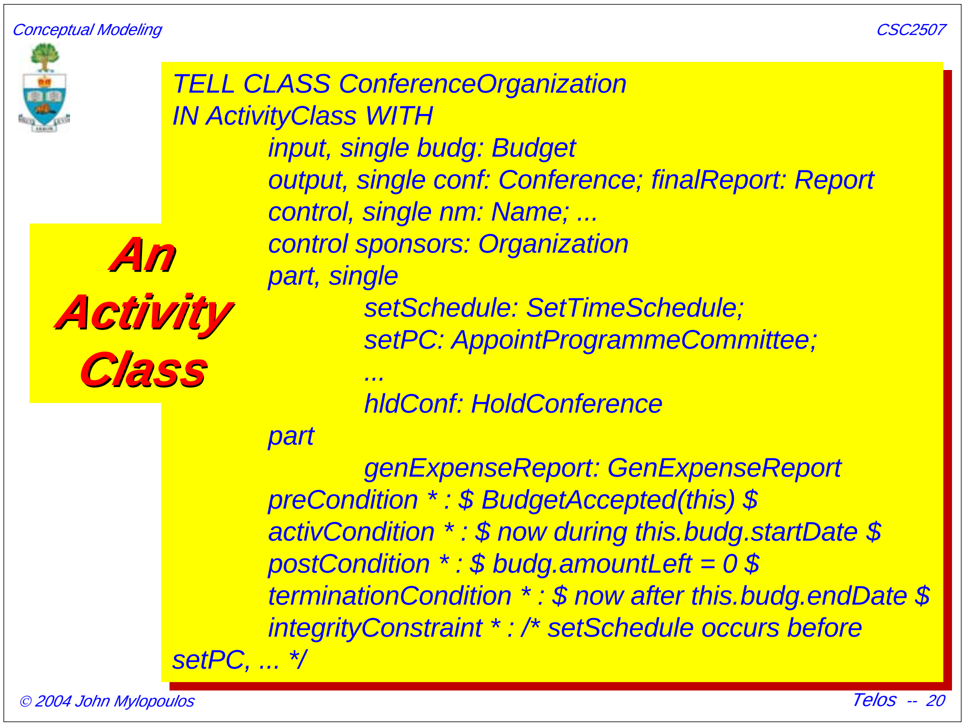
```
TELL CLASS Single
   COMPONENTS [Class, single, Class]
IN AttributeClass, MetaClass  WITH
integrityConstraint
 : $ (∀ u/Single, p, q/Proposition)
   [p in u ∧ q in u ∧ from(p)=from(q) ⇒ p = q]$
```





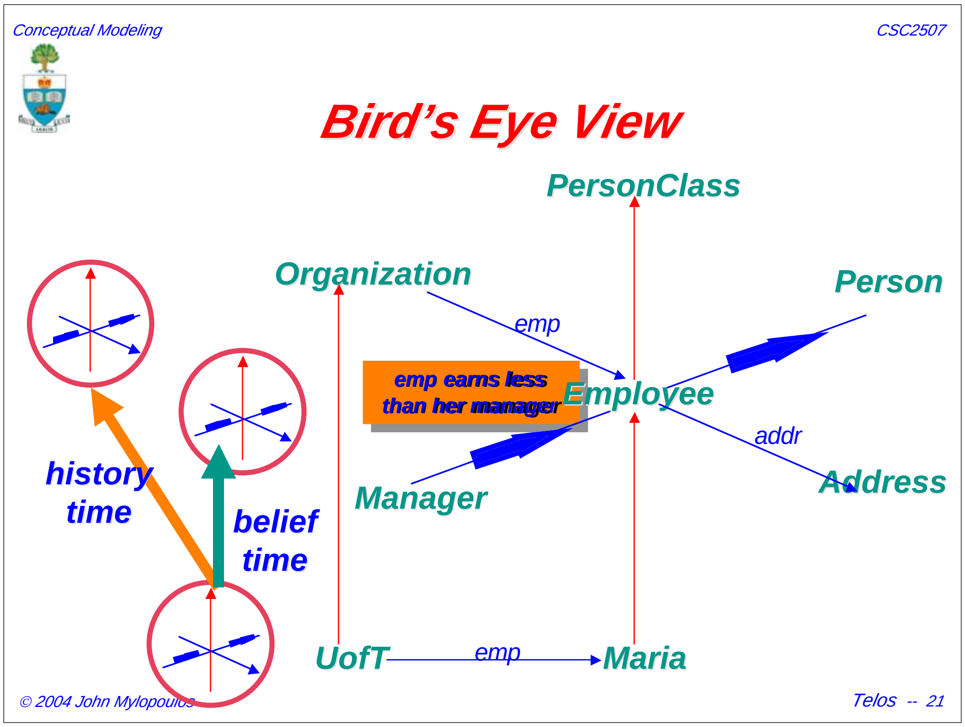
The Activity Metaclass

```
TELL CLASS ActivityClass IN MetaClass WITH
   attributes
     agent: AgentClass;
     input, output, control: EntityClass;
     part: ActivityClass
   integrityConstraint
     inputExists: $ ∀ p/this | input, x/Token
     [x in from(p) ⇒ ∃ q/Attribute [q in p ∧ to(q) in
   to(p)]] $
         outputCreated:...
         ...partDuringWhole:...
END ActivityClass
```



An Activity Class

```
TELL CLASS ConferenceOrganization
IN ActivityClass WITH
        input, single budg: Budget
        output, single conf: Conference; finalReport: Report
        control, single nm: Name; ...
        control sponsors: Organization
        part, single
                setSchedule: SetTimeSchedule;
                setPC: AppointProgrammeCommittee;
                ...
                hldConf: HoldConference
        part
                genExpenseReport: GenExpenseReport
        preCondition * : $ BudgetAccepted(this) $
        activCondition * : $ now during this.budg.startDate $
        postCondition * : $ budg.amountLeft = 0 $
        terminationCondition * : $ now after this.budg.endDate $
        integrityConstraint * : /* setSchedule occurs before
setPC, ... */
```



Bird's Eye View

PersonClass

Organization

Person

emp

emp earns less than her manager

Employee

addr

Manager

Address

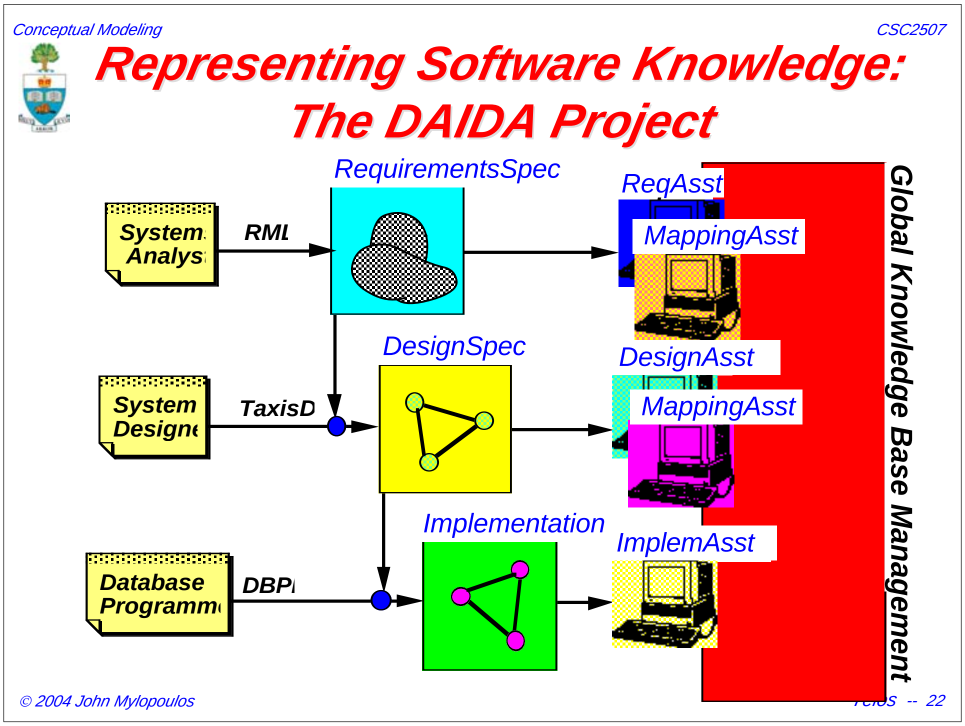
history time

belief time

UofT

emp

Maria



Representing Software Knowledge: The DAIDA Project

RequirementsSpec

ReqAsst

MappingAsst

System Analys

RML

DesignSpec

DesignAsst

MappingAsst

System Designe

TaxisD

Implementation

ImplemAsst

Database Programm

DBPl

Global Knowledge Base Management

Postscript

- Telos has seen several implementations, including ConceptBase (Technical University of Aachen) and SIS (University of Crete). It has also been used in a number of case studies. Some of these will appear in a forthcoming book:

Manfred Jeusfeld, Matthias Jarke and John Mylopoulos (eds.) *The Method Engineering Textbook*, MIT Press (to appear.)

References

- [Constantopoulos91] Constantopoulos, P., Jarke, M., Mylopoulos, J. and Vassiliou, Y., "The Software Information Base: A Server for Reuse", *VLDB Journal*, 1994.
- [Greenspan84] Greenspan, S., *Requirements Modelling: A Knowledge Representation Approach to Requirements Definition*, Ph.D. thesis, Department of Computer Science, University of Toronto, 1984.
- [Koubarakis88] Koubarakis, M., *An Implementation of CML*, M.Sc. thesis, Department of Computer Science, University of Toronto, 1988.
- [Koubarakis89] Koubarakis, M., Mylopoulos, J., Stanley, M. and Jarke, M., "Telos: Features and Formalization", KRR-TR-89-4, Department of Computer Science, University of Toronto, 1989.
- [Mylopoulos90] Mylopoulos, J., Borgida, A., Jarke, M. and Koubarakis, M., "Telos: Representing Knowledge About Information Systems", *ACM Transactions on Information Systems*, October 1990.
- [Plexousakis90] Plexousakis, D., *The Semantics of Telos: A Language for Knowledge Representation*, M.Sc. thesis, Department of Computer Science, University of Toronto, 1990.
- [Stanley86] Stanley, M., *CML: A Knowledge Representation Language with Applications to Requirements Modelling*, M.Sc. thesis, Department of Computer Science, University of Toronto, 1986.