Referências


BRANDÃO, A. A. F., SILVA, V. e LUCENA, C. A Knowledge-based Approach to the Verification of Multi-Agent Systems Design. The 17th International Conference on Software Engineering and Knowledge Engineering. Taipei,
An Ontology-based Method for Structuring Multi-Agent Systems


CHAVEZ, C.V.F.C Uma abordagem orientada a modelos para design orientado a aspectos, Tese de Doutorado. Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ, Brasil, 2004


ICSE. International Conference on Software Engineering.


KALFOGLOU, Y. e D. ROBERTSON. A case study in applying Ontologies to augment and reason about the correctness of Specifications. 11th International Conference on Software Engineering and Knowledge Engineering (SEKE99). Kaiserslautern, Germany 1999.


The role of roles in designing effective agent organizations. Berlin: Springer-Verlag, v.2603. 2003 (LNCS).


Apêndice A

A Ontologia Ont1 para MAS-ML

Neste apêndice encontra-se o código da Ontologia Ont1, construída durante a aplicação do Observed-MAS para a linguagem MAS-ML.

```
(in-tbox MASML-TBox)
(signature
 :atomic-concepts
 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
 entity
class
citizen-class
agent-class
organization-class
main-organization-class
sub-organization-class
object-class
environment-class
active-environment-class
passive-environment-class
role-class
agent-role-class
object-role-class
class-instance
agent
a-environment
p-environment
organization
main-organization
sub-organization
object
agent-role
object-role
features
goal
belief
action
plan
duty
right
protocol
axiom
attribute
method
condition
relationship
inhabit
play
ownership
specialization
association
aggregation
control
dependency
msg
agent-msg
object-msg
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
 MAS-ML concepts
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
 model
 static-model
 class-model
 organization-model
 role-model
 sequence-model
 stereotyped-msg
 create-msg
 destroy-msg
 role-commit-msg
```

role-cancel-msg
role-activate-msg
role-deactivate-msg
role-change-msg
path
class-path
agent-class-path
organization-class-path
main-organization-class-path
sub-organization-class-path
object-class-path
active-environment-class-path
passive-environment-class-path
agent-role-class-path
object-role-class-path
instance-path
agent-instance-path
organization-instance-path
main-organization-instance-path
sub-organization-instance-path
object-instance-path
a-environment-instance-path
p-environment-instance-path
agent-role-instance-path
object-role-instance-path
sequence
seq-element

:roles ( ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; TAO properties
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(has-goal :domain (or agent-class organization-class agent-role-class
active-environment-class)
:range goal)
(has-belief :domain (or agent-class organization-class
agent-role-class active-environment-class)
:range belief)
(has-axiom :domain organization-class
:range axiom)
(has-duty :domain agent-role-class
:range duty)
(has-right :domain agent-role-class
:range right)
(has-action :range action
:parent has-element)
(has-precond :domain action
:range condition)
(has-poscond :domain action
:range condition)
(has-plan :range plan)
(has-protocol :range protocol)
(has-end1 :domain relationship
:range class
:inverse is-end1)
(has-end2 :domain relationship
:range class
:inverse is-end2)
(is-end1 :domain class
:range relationship
:inverse has-end1)
(is-end2 :domain class
:range relationship
:inverse has-end2)
(has-msg :range msg
:parent has-element)
(has-msg-end :domain msg
:inverse is-msg-end)
(is-msg-end :inverse has-msg-end)
(has-msg-receiver :parent has-msg-end)
(has-msg-sender :parent has-msg-receiver)
(has-end1 :domain class-instance
:range class)
(has-attribute :domain (or object-class object-role-class passive-
environment-class)
:range attribute)
(has-method :domain (or object-class object-role-class passive-environment-
class)
:range method)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; MAS-ML properties
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(has-class :domain static-model
:range class
:inverse is-in-static-model)
(is-in-static-model :inverse has-class)
(has-relationship :domain static-model
:inverse is-in-relationship)
(is-relationship-of :inverse has-relationship)
(has-path :domain sequence-model :range path)
(has-head :domain path :range (or class-instance class))
(has-tail :domain path :range path)

(ordering properties)
(has-structure :range sequence)
(has-first :domain sequence :range seq-element
 :parent has-element :inverse is-first)
(has-last :domain sequence :range seq-element
 :parent has-element :inverse is-last)
(is-first :inverse has-first)
(is-last :inverse has-last)
(is-before :domain seq-element :range seq-element
 :transitive t :inverse is-after)
(is-after :domain seq-element :range seq-element
 :transitive t :inverse is-before)

(implies class entity)
(implies class-instance entity)
(implies environment-class class)
(implies role-class class)
(implies citizen-class class)
(implies active-environment-class environment-class)
(implies passive-environment-class environment-class)
(implies agent-role-class role-class)
(implies object-role-class role-class)
(implies agent-class citizen-class)
(implies organization-class citizen-class)
(implies object-class citizen-class)
(implies main-organization-class organization-class)
(implies sub-organization-class organization-class)
(implies inhabit relationship)
(implies play relationship)
(implies ownership relationship)
(implies specialization relationship)
(implies association relationship)
(implies aggregation relationship)
(implies control relationship)
(implies dependency relationship)
(implies goal features)
(implies belief features)
(implies action features)
(implies duty features)
(implies right features)
(implies duty action)
(implies right action)
(implies plan features)
(implies protocol features)
(implies axiom features)
(implies attribute features)
(implies method features)
(implies condition features)
(implies agent class-instance)
(implies a-environment class-instance)
(implies p-environment class-instance)
(implies organization class-instance)
(implies main-organization organization)
(implies sub-organization organization)
(implies object class-instance)
(implies agent-role class-instance)
(implies object-role class-instance)

(implies agent-msg msg)
(implies object-msg msg)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;; MAS-ML concepts taxonomy (it reuses some TAO concepts)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies stereotyped-msg msg)

(implies static-model model)
(implies sequence-model model)

(implies class-model static-model)
(implies organization-model static-model)
(implies role-model static-model)

(implies create-msg stereotyped-msg)
(implies destroy-msg stereotyped-msg)
(implies role-commit-msg stereotyped-msg)
(implies role-cancel-msg stereotyped-msg)
(implies role-activate-msg stereotyped-msg)
(implies role-deactivate-msg stereotyped-msg)
(implies role-change-msg stereotyped-msg)

(equivalent path (or class-path instance-path))

(implies citizen-class-path class-path)
(implies role-class-path class-path)
(implies environment-class-path class-path)
(implies agent-class-path citizen-class-path)
(implies organization-class-path citizen-class-path)
(implies object-class-path citizen-class-path)
(implies agent-role-class-path role-class-path)
(implies object-role-class-path role-class-path)
(implies main-organization-class-path organization-class-path)
(implies sub-organization-class-path organization-class-path)
(implies active-environment-class-path environment-class-path)
(implies passive-environment-class-path environment-class-path)
(implies agent-instance-path instance-path)
(implies role-instance-path instance-path)
(implies environment-instance-path instance-path)

;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;; ontology axioms
;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;               TAO disjoint restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint class class-instance features msg relationship)
(disjoint citizen-class environment-class object-class)
(disjoint agent-class organization-class object-class)
(disjoint main-organization-class sub-organization-class)
(disjoint active-environment-class passive-environment-class)
(disjoint agent-role-class object-role-class)
(disjoint agent a-environment p-environment main-organization
sub-organization object agent-role object-role)
(disjoint agent-msg object-msg)

(disjoint goal belief protocol attribute)
(disjoint inhabit play ownership specialization aggregation
control dependency)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(disjoint static-model sequence-model)
(disjoint class-model organization-model role-model)
(disjoint create-msg destroy-msg role-commit-msg
  role-cancel-msg role-activate-msg role-deactivate-msg
  role-change-msg)
(disjoint class-path instance-path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;                                              ;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;      Instances-related axioms                ;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(implies agent (all is-instanceOf agent-class))
(implies organization (all is-instanceOf organization-class))
(implies main-organization (all is-instanceOf main-organization-class))
(implies object (all is-instanceOf object-class))
(implies a-environment (all is-instanceOf active-environment-class))
(implies p-environment (all is-instanceOf passive-environment-class))
(implies agent-role (all is-instanceOf agent-role-class))
(implies object-role (all is-instanceOf object-role-class))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; path axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; pathnames are defined recursively

(implies agent-class-path
  (and (some has-head agent-class)
       (some has-tail agent-role-class-path)))
(implies object-class-path
  (and (some has-head object-class)
       (some has-tail object-role-class-path)))
(implies main-organization-class-path
  (and (all has-head main-organization-class)
       (some has-tail environment-class-path)))
(implies sub-organization-class-path
  (and (all has-head sub-organization-class)
       (some has-tail sub-organization-class-path)))
(implies agent-role-class-path
  (and (all has-head agent-role-class)
       (some has-tail agent-role-instance-path)))
(implies object-role-class-path
  (and (all has-head object-role-class)
       (some has-tail object-role-instance-path)))
(implies active-environment-class-path
  (and (all has-head active-environment-class)
       (all has-tail "bottom")))
(implies passive-environment-class-path
  (and (all has-head passive-environment-class)
       (all has-tail "bottom")))
(implies agent-instance-path
  (and (all has-head agent-instance-path)
       (some has-tail agent-role-instance-path)))
(implies object-instance-path
  (and (all has-head object-instance-path)
       (some has-tail object-role-instance-path)))
(implies main-organization-instance-path
  (and (all has-head main-organization-instance-path)
       (some has-tail environment-instance-path)))
(implies sub-organization-instance-path
  (and (all has-head sub-organization-instance-path)
       (some has-tail sub-organization-instance-path)))
(implies agent-role-instance-path
  (and (all has-head agent-role-instance-path)
       (some has-tail agent-role-instance-path)))
(implies object-role-instance-path
  (and (all has-head object-role-instance-path)
       (some has-tail object-role-instance-path)))
(implies a-environment-instance-path
  (and (all has-head a-environment-instance-path)
       (all has-tail "bottom")))
(implies p-environment-instance-path
  (and (all has-head p-environment-instance-path)
       (all has-tail "bottom")))

; ; order axioms

; ; semantics of the first element of a sequence
(implies (some is-before "top") (some is-first "top"))
(implies (some is-after "top") (some is-first "top"))

; ; semantics of the last element of a sequence
(implies (some is-first msg) (some has-msg msg))
(implies (some is-last msg) (some has-msg msg))
(implies (some has-first action) (some has-action action))
(implies (some has-last action) (some has-action action))

; ; end of tbox
(classify-tbox)
(tbox-classified-p)
(tbox-coherent-p)
(tbox-cyclic-p)
Apêndice B

QV1 para o método aplicado a MAS-ML

Neste apêndice encontram-se as consultas referentes à fase F1 do método Observed-MAS aplicado a MAS-ML.

Consultas referentes ao domínio de SMAs

```
;Q1 checks relationships that are not in any static-diagram
(retrieve (?relation)
  (and (?relation relationship)
    (nil ?relation has-relationship)))

;Q2 checks bad-defined inhabit relationships
(retrieve (?end1 ?end2 ?inh)
  (and (?inh inhabit)
    (?end1 ?inh is-end1)
    (?end2 ?inh is-end2)
    (or (?end1 (or environment-class role-class))
      (?end2 (or citizen-class role-class)))))

;Q3 checks bad-defined play relationships
(retrieve ($?end1 $?end2 ?play)
  (and (?play play)
    ($?end1 ?play is-end1)
    ($?end2 ?play is-end2)
    (or ($?end1 (or agent-class sub-organization-class))
      ($?end2 agent-role-class))
    (not (and ($?end1 (or agent-class sub-organization-class))
      (not ($?end2 agent-role-class)))))

;Q4 checks bad-defined ownership relationships
(retrieve ($?end1 $?end2 ?own)
  (and (?own ownership)
    ($?end1 ?own is-end1)
    ($?end2 ?own is-end2)
    (or ($?end1 (or role-class environment-class main-organization-class))
      ($?end2 (or citizen-class environment-class))
      (and ($?end1 (or agent-class sub-organization-class))
        (not ($?end2 agent-role-class))))
    (and ($?end1 object-class) (not ($?end2 object-role-class)))))

;Q5 checks bad-defined specialization relationships
(retrieve (?end1 ?end2 ?spec)
  (and (?spec specialization)
    (?end1 ?spec is-end1)
    (?end2 ?spec is-end2)
    (or (?end1 object-class)
      (?end2 (or organization-class agent-class environment-class role-class))
      (and (?end1 agent-class)
        (?end2 (or organization-class object-class environment-class role-class)))))
```
(and (?end1 main-organization-class)
  (?end2 (or agent-class object-class sub-organization-class
          environment-class role-class))))
(and (?end1 sub-organization-class)
  (?end2 (or agent-class object-class main-organization-class
          environment-class role-class))))
(and (?end1 passive-environment-class)
  (?end2 (or agent-class organization-class object-class
          active-environment-class role-class))))
(and (?end1 agent-role-class)
  (?end2 (or agent-class environment-class object-role-class))
  (?end2 (or citizen-class environment-class agent-role-class))
  (?end2 (or citizen-class environment-class agent-role-class))
  (?end1 (or agent-class organization-class environment-class))))

;; Q6 checks bad-defined aggregation relationships
(retrieve (?end1 ?end2 ?agg)
  (and (?agg aggregation)
       (?end1 ?agg is-end1)
       (?end2 ?agg is-end2)
       (or (and (?end1 object-class)
                (?end2 (or organization-class agent-class environment-class
                        role-class)))
            (and (?end1 agent-role-class)
                 (?end2 (or citizen-class environment-class object-role-class))
                 (?end2 (or citizen-class environment-class agent-role-class)))
            (and (?end1 object-role-class)
                 (?end2 (or citizen-class environment-class agent-role-class))))))

;; Q7 checks bad-defined control relationships
(retrieve (?end1 ?end2 ?ctrl)
  (and (?ctrl control)
       (?end1 ?ctrl is-end1)
       (?end2 ?ctrl is-end2)
       (or (?end1 (or citizen-class environment-class object-role-class))
           (and (?end1 agent-role-class)
                (?end2 (or citizen-class environment-class object-role-class)))))

;; Q8 checks bad-defined dependency relationships
(retrieve (?end1 ?end2 ?dep)
  (and (?dep dependency)
       (?end1 ?dep is-end1)
       (?end2 ?dep is-end2)
       (or (?end1 (or agent-class organization-class environment-class))
           (and (?end1 object-class)
                (?end2 (or organization-class agent-class environment-class
                        role-class)))
           (and (?end1 agent-role-class)
                (?end2 (or citizen-class environment-class object-role-class)))
           (and (?end1 object-role-class)
                (?end2 (or citizen-class environment-class agent-role-class)))))

;; Q10 checks classes that are not in any static-diagram
(retrieve (?class-alone)
  (and (?class-alone class)
       (NIL ?class-alone has-class)))

;; Q11-A checks agent-classes that don't have assigned goals
(retrieve (?agentwithoutgoal)
  (and (?agentwithoutgoal agent-class)
       (?agentwithoutgoal NIL has-goal)))

;; Q11-B checks organization-classes that don't have assigned goals
(retrieve (?orgwithoutgoal)
  (and (?orgwithoutgoal organization-class)
       (?orgwithoutgoal NIL has-goal)))

;; Q11-C checks agent-role-classes that don't have assigned goals
(retrieve (?rolewithoutgoal)
  (and (?rolewithoutgoal agent-role-class)
       (?rolewithoutgoal NIL has-goal)))

;; Q12 checks citizens that don’t inhabit any environment
(retrieve (?no-citizen)
  (and (?no-citizen citizen-class)
       (not (?no-citizen (some is-end1
                           (and inhabit (some has-end2 environment-class))))))

;; Q12A checks environments that aren't the habitat of any citizen
(retrieve (?no-habitat)
  (and (?no-habitat environment-class)
       (not (?no-habitat (some is-end2 (and inhabit (some has-end1 citizen-class))))))

;; Q13 checks agents that don't have assigned roles
(retrieve (?agentwithoutrole)
  (and (?agentwithoutrole agent-class)
       (not (?agentwithoutrole (some is-end1
                               (and play (some has-end2 agent-role-class)))))))
;; Q14 checks organizations that don't own roles
(retrieve (?org-no-owner)
  (and (?org-no-owner organization-class)
      (not (?org-no-owner (some is-end1 (and ownership (some has-end2 agent-role-class)))))))

;; Q15 checks main-organizations that play roles
(retrieve (?org 7play)
  (and (?org main-organization-class)
      (?play (and play (some has-end2 agent-role-class))
            (?org 7play is-end1)))

;; Q16 checks sub-organizations that don't have assigned roles
(retrieve (?org-no-play)
  (and (?org-no-play sub-organization-class)
      (not (?org-no-play (some is-end1 play))))))

;; Q17 checks organization-classes that don't have assigned axioms
(retrieve (?org)
  (and (?org organization-class)
      (?org NIL has-axiom)))

;; Q18 checks role-classes that aren't owned by any organization
(retrieve (?role-no-owned)
  (and (?role-no-owned role-class)
      (not (?role-no-owned (some is-end2 ownership))))))

;; Q19 checks paths that aren't part of any sequence-model
(retrieve (?path-without-model)
  (and (?path-without-model path)
      (nil ?path-without-model has-path)))

;; Q20 checks for bad-defined object-msgs
(retrieve (?bad-obj-msg)
  (and (?bad-obj-msg object-msg)
      ($?sender ?bad-obj-msg is-msg-sender)
      ($?receiver ?bad-obj-msg is-msg-receiver)
      (not (and ($?sender (or citizen-class-path citizen-instance-path
                          environment-class-path environment-instance-path))
               ($?receiver (object-class-path object-instance-path
                             passive-environment-class-path p-environment-instance-path)))))

;; Q20A same object as sender and receiver
(retrieve (?bad-obj-msg)
  (and (?bad-obj-msg object-msg)
      (?sender-receiver ?bad-obj-msg is-msg-sender)
      (?sender-receiver ?bad-obj-msg is-msg-receiver)
      (?sender-receiver (object-class-path object-instance-path
                          passive-environment-class-path p-environment-instance-path)))

;; Q21 checks for bad-defined agent-msgs
(retrieve ($?sender $?receiver ?ag-msg)
  (and (?ag-msg agent-msg)
      ($?sender ?ag-msg is-msg-sender)
      ($?receiver ?ag-msg is-msg-receiver)
      (or (and ($?sender (or agent-role-class-path agent-role-instance-path
                          active-environment-class-path a-environment-instance-path
                          object-class-path object-instance-path
                          object-role-class-path object-role-instance-path
                          main-organization-class-path main-organization-instance-path))
          ($?receiver (agent-class-path agent-instance-path
                        passive-environment-class-path p-environment-instance-path
                        object-role-class-path object-role-instance-path
                        main-organization-class-path main-organization-instance-path))
          (not ($?sender (agent-role-class-path agent-role-instance-path
                          active-environment-class-path a-environment-instance-path
                          main-organization-class-path main-organization-instance-path)))
          (not ($?receiver (agent-role-class-path agent-role-instance-path
                          active-environment-class-path a-environment-instance-path
                          main-organization-class-path main-organization-instance-path)))))

;; Q22 checks for bad-defined create-msgs
(retrieve (?sender ?receiver ?create)
  (and (?create create-msg)
      (sender ?create is-msg-sender)
      (receiver ?create is-msg-receiver)
      (or (and ($?sender (main-organization-class-path main-organization-instance-path
                          agent-class-path agent-instance-path
                          role-class-path role-instance-path))
            (receiver (sub-organization-class-path sub-organization-instance-path))
            (receiver (main-organization-class-path main-organization-instance-path
                       role-class-path role-instance-path)))
      (and ($?sender (environment-class-path environment-instance-path))
            (receiver (environment-class-path environment-instance-path))))
(7receiver (or role-class-path role-instance-path))
(and (7sender (or object-class-path object-instance-path))
(7receiver (or agent-class-path agent-instance-path
organization-class-path organization-instance-path
environment-class-path environment-instance-path
role-class-path role-instance-path))
(7sender (or role-class-path role-instance-path))))

;; Q22-A checks for bad-defined create-msgs (auto-creation)
(retrieve (7sender ?create)
(and (7create create-msg)
(7sender ?create is-msg-sender)
(7sender ?create is-msg-receiver)))

;; Q23 checks for bad-defined destroy-msgs
(retrieve (7sender ?receiver ?destroy)
(and (7destroy destroy-msg)
(7sender ?destroy is-msg-sender)
(7receiver ?destroy is-msg-receiver)
(or (and (7sender (or main-organization-class-path main-organization-instance-path
agent-class-path agent-instance-path))
(7receiver (or role-class-path role-instance-path)))
(and (7sender (or sub-organization-class-path sub-organization-instance-path))
(7receiver (or main-organization-class-path main-organization-instance-path
role-class-path role-instance-path)))
(and (7sender (or environment-class-path environment-instance-path))
(7receiver (or role-class-path role-instance-path)))
(and (7sender (or object-class-path object-instance-path))
(7receiver (or agent-class-path agent-instance-path
organization-class-path organization-instance-path
environment-class-path environment-instance-path
role-class-path role-instance-path)))
(7sender (or role-class-path role-instance-path))))

;; Q24 checks for bad-defined role-commit-msgs
;; 24A wrong sender
(retrieve (7bad-commit-ag-role ?wrong-sender)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?wrong-sender has-msg-sender)
(not (?wrong-sender (or agent-class-path agent-instance-path)))))

;; 24B sender ok, but wrong receivers (agents)
(retrieve (7bad-commit-ag-role ?sender ?wrong-receiver)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?sender has-msg-sender)
(?bad-commit-ag-role ?wrong-receiver has-msg-receiver)
(?sender (or agent-class-path agent-instance-path))
(not (?wrong-receiver (or agent-class-path agent-instance-path)))))

;; 24B sender ok, but wrong receivers (sub-orgs)
(retrieve (7bad-commit-ag-role ?sender ?wrong-receiver)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?sender has-msg-sender)
(?bad-commit-ag-role ?wrong-receiver has-msg-receiver)
(?sender (or sub-organization-class-path sub-organization-instance-path))
(not (?wrong-receiver (or sub-organization-class-path sub-organization-instance-path))))

;; 24C sender and receiver ok but representing different agents
(retrieve (7bad-commit-ag-role ?sender ?receiver)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?sender has-msg-sender)
(?bad-commit-ag-role ?receiver has-msg-receiver)
(?sender (or agent-class-path agent-instance-path))
(?receiver (or agent-class-path agent-instance-path))))

;; 24C sender and receiver ok but representing different sub-orgs
(retrieve (7bad-commit-ag-role ?sender ?receiver)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?sender has-msg-sender)
(?bad-commit-ag-role ?receiver has-msg-receiver)
(?sender (or sub-organization-class-path sub-organization-instance-path))
(?receiver (or sub-organization-class-path sub-organization-instance-path))))

;; 24D sender and receiver are the same agent playing the role they’re trying to commit
(retrieve (7bad-commit-ag-role ?sender ?receiver)
(and (7bad-commit-ag-role role-commit-msg)
(?bad-commit-ag-role ?sender has-msg-sender)
(?bad-commit-ag-role ?receiver has-msg-receiver)
(?sender (or agent-class-path agent-instance-path))
(?receiver (or agent-class-path agent-instance-path))
(?sender ?head has-head) (?receiver ?head has-head)
(?role agent-role-instance-path)
(?sender ?role has-tail) (?receiver ?role has-tail)))

;; 24D sender and receiver are the same agent playing the role they’re trying to commit
(retrieve (7bad-commit-ag-role ?sender ?receiver)
(and (7bad-commit-ag-role role-commit-msg))
(bad-commit-ag-role $sender has-msg-sender)
(bad-commit-ag-role $receiver has-msg-receiver)
($sender sub-organization-instance-path)
($receiver sub-organization-instance-path)
($sender ?head has-head) ($receiver ?head has-head)
(role agent-role-instance-path)
($sender ?role has-tail) ($receiver ?role has-tail))

;; 24E COMMIT-TO-OBJECT-ROLE WRONG SENDER
(retrieve (bad-commit-obj-role $sender $receiver)
(and (bad-commit-obj-role role-commit-msg)
(bad-commit-obj-role $sender has-msg-sender)
(bad-commit-obj-role $receiver has-msg-receiver)
($sender (or object-class-path object-instance-path
environment-class-path environment-instance-path
role-class-path role-instance-path)))

;; 24F COMMIT-TO-OBJECT-ROLE SENDER OK AND WRONG RECEIVER
(retrieve (bad-commit-obj-role $sender $receiver)
(and (bad-commit-obj-role role-commit-msg)
(bad-commit-obj-role $sender has-msg-sender)
(bad-commit-obj-role $receiver has-msg-receiver)
?sender (or agent-class-path agent-instance-path
organization-class-path organization-instance-path)
(not (?receiver (or object-class-path object-instance-path))))

;; Q25 checks for bad-defined role-cancel-msgs
(retrieve (bad-cancel-ag-role $sender $receiver)
(and (bad-cancel-ag-role role-cancel-msg)
(bad-cancel-ag-role $sender has-msg-sender)
(bad-cancel-ag-role $receiver has-msg-receiver)
(or (?sender (or object-class-path object-instance-path
environment-class-path environment-instance-path
role-class-path role-instance-path
main-organization-class-path main-organization-instance-path))
(and (?sender (or agent-class-path agent-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))
(and (?sender (or sub-organization-class-path sub-organization-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))))

(retrieve (bad-cancel-obj-role $sender $receiver)
(and (bad-cancel-obj-role role-cancel-msg)
(bad-cancel-obj-role $sender has-msg-sender)
(bad-cancel-obj-role $receiver has-msg-receiver)
(or (?sender (or object-class-path object-instance-path
environment-class-path environment-instance-path
role-class-path role-instance-path
main-organization-class-path main-organization-instance-path))
(and (?sender (or agent-class-path agent-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))
(and (?sender (or sub-organization-class-path sub-organization-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))))

;; Q26 checks for bad-defined role-activate-msgs
(retrieve ($sender $receiver $activate)
(and ($activate role-activate-msg)
($sender $activate is-msg-sender)
($receiver $activate is-msg-receiver)
(or (?sender (or main-organization-class-path main-organization-instance-path
role-class-path role-instance-path
environment-class-path environment-instance-path
main-organization-class-path main-organization-instance-path))
(and (?sender (or agent-class-path agent-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))
(and (?sender (or organization-class-path organization-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))))

;; Q27 checks for bad-defined role-deactivate-msgs
(retrieve ($sender $receiver $deactivate)
(and ($deactivate role-deactivate-msg)
($sender $deactivate is-msg-sender)
($receiver $deactivate is-msg-receiver)
(or (?sender (or main-organization-class-path main-organization-instance-path
role-class-path role-instance-path
environment-class-path environment-instance-path
main-organization-class-path main-organization-instance-path)
(and (?sender (or agent-class-path agent-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))
(and (?sender (or sub-organization-class-path sub-organization-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))))

;; Q28 checks for bad-defined role-change-msgs
(retrieve ($sender $receiver $change)
(and ($change role-change-msg)
($sender $change is-msg-sender)
($receiver $change is-msg-receiver)
(or (?sender (or main-organization-class-path main-organization-instance-path
role-class-path role-instance-path
environment-class-path environment-instance-path
main-organization-class-path main-organization-instance-path)
(and (?sender (or agent-class-path agent-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))
(and (?sender (or sub-organization-class-path sub-organization-instance-path)
(role-class-path role-instance-path)
(main-organization-class-path main-organization-instance-path)
(7sender ?head-s has-head) (?receiver ?head-r has-head))))

;; Q29 checks for bad-defined role-cancel-msgs
(receiver ?change is-msg-receiver)
(or (?sender (or main-organization-class-path main-organization-instance-path
role-class-path role-instance-path
environment-class-path environment-instance-path
object-class-path object-instance-path))
(and (?sender (or agent-class-path agent-instance-path
sub-organization-class-path sub-organization-instance-path
environment-class-path environment-instance-path
object-class-path object-instance-path
object-role-class-path object-role-instance-path)))

;; Paths identify instances of MAS-ML entities, therefore there is the need of
;; check if all instances have an associated id.
;; Q29 checks for class-instances without paths
(retrieve (?path-without-source)
(and (?path-without-source instance-path)
(?path-without-source ?head has-head)
(?head nil is-instanceOf)))

;; Messages must be sent and received by paths
;; Q30 checks for bad-defined msgs
(retrieve (?msg-bad-def ?sender ?receiver)
(and (?msg-bad-def msg)
(?msg-bad-def ?sender has-msg-sender)
(?msg-bad-def ?receiver has-msg-receiver)
(or (not (?sender path)) (not (?receiver path)))))

;; Q31 checks agent-msgs that are in an agent-role-class but aren't in any protocol
(retrieve (?ag-msg ?ag-role-class)
(and (?ag-msg agent-msg)
(?ag-role-class agent-role-class)
(?ag-role-class ?ag-msg has-msg)
(not (?ag-msg (some (inv has-msg) protocol)))))

;; Q32 checks protocols that aren't in any agent-role-class
(retrieve (?prtcl-alone)
(and (?prtcl-alone protocol)
(not (?prtcl-alone (some (inv has-protocol) agent-role-class))))))

;; Q33 checks plans that aren't in any agent-class, organization-class or
;; active-environment-class
(retrieve (?plan-alone)
(and (?plan-alone plan)
(not (?plan-alone (some (inv has-plan)
(or agent-class organization-class
active-environment-class)))))

;; Q34 checks actions that aren't in any agent-class, organization-class or
;; active-environment-class or plan.
(retrieve (?action-alone)
(and (?action-alone action)
(not (?action-alone (some (inv has-action)
(or agent-class organization-class
active-environment-class)))))

;; Q35 checks agent-msgs that aren't in any agent-role-class,
;; main-organization-class or active-environment-class
(retrieve (?ag-msg-alone)
(and (?ag-msg-alone agent-msg)
(not (?ag-msg-alone (some (inv has-msg)
(or agent-role-class
main-organization-class
active-environment-class)))))

;; Q36 checks paths without heads (their id)
(retrieve (?path-without-head)
(and (?path-without-head path)
(?path-without-head nil has-head)))

;; Q37 checks protocols without structure
(retrieve (?prtcl-without-struct)
(and (?prtcl-without-struct protocol)
(?prtcl-without-struct nil has-structure))

;; Q38 checks plans without structure
(retrieve (?plan-without-struct)
(and (?plan-without-struct plan)
(?plan-without-struct nil has-structure))

;; Q39 checks bad-defined protocol structures
;; it shows a triple containing the protocol, its structure and the elements which
;; must have be in both and are not.
(retrieve (?bad-struct ?prtcl ?element)
(and (?bad-struct sequence)
(?prtcl protocol)
(?prtcl ?bad-struct has-structure)
(or (and (?bad-struct ?element has-element)
(not (?prtcl ?element has-msg)))
(and (?prtcl ?element has-msg)))
Consultas referentes as propriedades intra-diagramas MAS-ML

;; These queries analyze the class diagrams structures and their results are the inconsistencies that were found in the diagram

;; Q1 checks for role-classes definition in a class-diagram
(retrieve (?rolecl ?clmd)
  (and (?rolecl role-class)
   (?clmd class-model)
   (?clmd ?rolecl has-class)
   )
)

;; Q2 checks for play relationships definition in a class-diagram
(retrieve (?pl ?clmd)
  (and (?pl play)
   (?clmd class-model)
   (?clmd ?pl has-relationship)
   )
)

;; Q3 checks for ownership relationships definition in a class-diagram
(retrieve (?own ?clmd)
  (and (?own ownership)
   (?clmd class-model)
   (?clmd ?own has-relationship)
   )
)

;; Q4 checks for control relationships definition in a class-diagram
(retrieve (?ctrl ?clmd)
  (and (?ctrl control)
   (?clmd class-model)
   (?clmd ?ctrl has-relationship)
   )
)

;; Q5 checks for bad-defined inhabit relationships in a class-diagram
(retrieve (?inh ?clmd)
  (and (?inh inhabit)
       (?clmd class-model)
       (?clmd ?inh has-relationship)
       (?end1 ?inh is-end1)
       (?end2 ?inh is-end2)
       (or (?end1 (or agent-class organization-class role-class environment-class))
           (?end2 (or citizen-class role-class))))
)

;;Q6 checks for bad-defined association relationships in a class-diagram
(retrieve (?ass ?clmd)
  (and (?ass association)
       (?clmd class-model)
       (?clmd ?ass has-relationship)
       (?end1 ?ass is-end1)
       (?end2 ?ass is-end2)
       (or (and (?end1 (or agent-class organization-class))
             (?end2 (or agent-class organization-class environment-class)))
           (and (?end1 environment-class)
                (?end2 citizen-class))
           (and (?end1 role-class)
                (?end2 role-class))))
)

;;Q7 checks for bad-defined specialization relationships in a class-diagram
(retrieve (?spec ?clmd)
  (and (?spec specialization)
       (?clmd class-model)
       (?clmd ?spec has-relationship)
       (?end1 ?spec is-end1)
       (?end2 ?spec is-end2)
       (or (?end1 role-class) (?end2 role-class)
           (and (?end1 agent-class)
                (?end2 (or object-class organization-class environment-class)))
           (and (?end1 main-organization-class)
                (?end2 (or agent-class organization-class environment-class)))
           (and (?end1 sub-organization-class)
                (?end2 (or agent-class main-organization-class environment-class)))
           (and (?end1 active-environment-class)
                (?end2 (or citizen-class passive-environment-class)))
           (and (?end1 passive-environment-class)
                (?end2 (or citizen-class active-environment-class))))
)

;;Q8 checks for bad-defined aggregation relationships in a class-diagram
(retrieve (?agg ?clmd)
  (and (?agg aggregation)
       (?clmd class-model)
       (?clmd ?agg has-relationship)
       (?end1 ?agg is-end1)
       (?end2 ?agg is-end2)
       (or (?end1 (or role-class agent-class organization-class environment-class))
           (?end2 (or role-class agent-class organization-class environment-class)))
)

;;Q9 checks for bad-defined dependency relationships in a class-diagram
(retrieve (?dep ?clmd)
  (and (?dep dependency)
       (?clmd class-model)
       (?clmd ?dep has-relationship)
       (?end1 ?dep is-end1)
       (?end2 ?dep is-end2)
       (or (?end1 (or role-class agent-class organization-class environment-class))
           (?end2 (or role-class agent-class organization-class environment-class)))
)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
 ;; These queries analyze the organization diagrams structures and their
 ;; results are the inconsistencies that were found in the diagram
 ;; Organization diagrams can have all class types defined in the MAS-ML metamodel
 ;; and only the relationships inhabit, play and ownership

;; Q1 checks for specialization relationships definition in an organization-diagram
(retrieve (?spec ?orgmd)
  (and (?spec specialization)
       (?orgmd organization-model)))
(and (?assoc association)  
  (?orgmd organization-model)  
  (?orgmd ?assoc has-relationship) 
  )

;; Q3 checks for aggregation relationships definition in an organization-diagram
(retrieve (?agg ?orgmd)  
  (and (?agg aggregation)  
    (?orgmd organization-model)  
    (?orgmd ?agg has-relationship) 
    )
  )

;; Q4 checks for control relationships definition in an organization-diagram
(retrieve (?ctrl ?orgmd)  
  (and (?ctrl control)  
    (?orgmd organization-model)  
    (?orgmd ?ctrl has-relationship) 
    )
  )

;; Q5 checks for dependency relationships definition in an organization-diagram
(retrieve (?dep ?orgmd)  
  (and (?dep dependency)  
    (?orgmd organization-model)  
    (?orgmd ?dep has-relationship) 
    )
  )

;; Q6 checks for relationships defined between roles
(retrieve (?rel ?orgmd)  
  (and (?rel relationship)  
    (?orgmd organization-model)  
    (?orgmd ?rel has-relationship)  
    (?end1 ?rel is-end1)  
    (?end2 ?rel is-end2)  
    (?end1 role-class) (?end2 role-class) 
    )
  )

;; Q7 checks for relationships defined between agents
(retrieve (?rel ?orgmd)  
  (and (?rel relationship)  
    (?orgmd organization-model)  
    (?orgmd ?rel has-relationship)  
    (?end1 ?rel is-end1)  
    (?end2 ?rel is-end2)  
    (?end1 agent-class) (?end2 agent-class) 
    )
  )

;; Q8 checks for relationships defined between environment
(retrieve (?rel ?orgmd)  
  (and (?rel relationship)  
    (?orgmd organization-model)  
    (?orgmd ?rel has-relationship)  
    (?end1 ?rel is-end1)  
    (?end2 ?rel is-end2)  
    (?end1 environment-class) (?end2 environment-class) 
    )
  )

;; Q9 checks for bad-formed organization diagrams
(retrieve (?rel ?orgmd-bad)  
  (and (?rel relationship)  
    (?orgmd-bad organization-model)  
    (?orgmd-bad ?rel has-relationship)  
    (?rel (or association aggregation specialization control dependency))  
    )
  )

;; Q10 checks organization-models which model main-organizations and  
;; didn't have inhabit relationship
(retrieve (?orgmd-without-env ?main-org)  
  (and (?orgmd-without-env organization-model)  
    (?main-org main-organization-class)  
    (?orgmd-without-env ?main-org has-class)  
    (not (?orgmd-without-env (some is-end1 inhabit)))
    (not (?orgmd-without-env (some has-relationship inhabit)))
    )
  )
In an organization diagram the internal properties of the organization
which is being modeled as well as the internal properties of the roles
it defines are considered in isolation and, therefore, are in the
GeneralQueriesFinal.racer

These queries analyze the role diagrams structures and their
results are the inconsistencies that were found in the diagram
Role diagrams can have role-class and object-class
and the relationships control, dependency, association, aggregation
and specialization

Q1 checks for agent-classes definition in a role-diagram
(retrieve (?agcl ?rlmd)
  (and (?agcl agent-class)
  (?rlmd role-model)
  (?rlmd ?agcl has-class)
  )
)

Q2 checks for organization-classes definition in a role-diagram
(retrieve (?orgcl ?rlmd)
  (and (?orgcl organization-class)
  (?rlmd role-model)
  (?rlmd ?orgcl has-class)
  )
)

Q3 checks for environment-classes definition in a role-diagram
(retrieve (?envcl ?rlmd)
  (and (?envcl environment-class)
  (?rlmd role-model)
  (?rlmd ?envcl has-class)
  )
)

Q4 checks for play relationships definition in a role-diagram
(retrieve (?pl ?rlmd)
  (and (?pl play)
  (?rlmd role-model)
  (?rlmd ?pl has-relationship)
  )
)

Q5 checks for ownership relationships definition in a role-diagram
(retrieve (?own ?rlmd)
  (and (?own ownership)
  (?rlmd role-model)
  (?rlmd ?own has-relationship)
  )
)

Q6 checks for inhabit relationships definition in a role-diagram
(retrieve (?inh ?rlmd)
  (and (?inh inhabit)
  (?rlmd role-model)
  (?rlmd ?inh has-relationship)
  )
)

The control relationship was previously analyzed by the general queries

Q7 checks for bad-defined dependency relationships in role diagrams
(retrieve (?dep ?end1 ?end2 ?rlmd)
  (and (?dep dependency)
  (?rlmd role-model)
  (?rlmd ?dep has-relationship)
  (?end1 ?dep is-end1)
  (?end2 ?dep is-end2)
  (or (?end1 (or citizen-class environment-class))
   (?end2 (or citizen-class environment-class))
   (and (?end1 agent-role-class) (?end2 (or citizen-class environment-class)))
   (and (?end1 object-role-class)
    (?end2 (or agent-role-class citizen-class environment-class)))
  )
  )
)

Q8 checks for bad-defined aggregation relationships
(retrieve (?agg ?end1 ?end2 ?rlmd)
  (and (?agg aggregation)
  (?rlmd role-model)
  (?rlmd ?agg has-relationship)
  (?end1 ?agg is-end1)
  (?end2 ?agg is-end2)
  )
;;Q9 checks for bad-defined specialization relationships
(retrieve (?spec ?end1 ?end2 ?rlmd)
  (and (?spec specialization)
    (?rlmd role-model)
    (?rlmd ?spec has-relationship)
    (?end1 ?spec is-end1)
    (?end2 ?spec is-end2)
    (or (?end1 (or citizen-class environment-class))
      (?end2 (or citizen-class environment-class))
    (and (?end1 agent-role-class) (not (?end2 agent-role-class)))
    (and (?end1 object-role-class) (not (?end2 object-role-class))))
)

;;Q10 checks for bad-defined association relationships
(retrieve (?assoc ?end1 ?end2 ?rlmd)
  (and (?assoc association)
    (?rlmd role-model)
    (?rlmd ?assoc has-relationship)
    (?end1 ?assoc is-end1)
    (?end2 ?assoc is-end2)
    (or (?end1 (or agent-class organization-class environment-class))
      (?end2 (or agent-class organization-class environment-class))
    (and (?end1 object-class) (not (?end2 object-class))))
)

; This file contains queries related to the msgs from MAS-ML, which are described
; in sequence diagrams.
; Sequence diagrams also describe the execution of plans and protocols. In this
; sense, a sequence diagram is composed of class-paths (of agents, organizations
; and active-environments) when we are modeling plans, and agent-role-class-paths
; when we are modeling protocols.

;; Q1 check stereotyped and object-msgs that aren’t part of any sequence-model
(retrieve (?msg-without-model)
  (and (?msg-without-model (or stereotyped-msg object-msg))
    (not (?msg-without-model (some (inv has-msg) sequence-model)))
)

;; Q2 check agent-msgs that are in a sequence-model and are not in an
;; agent-role-class or main-organization-class or active-environment-class
;; ps: the inclusion of agent-msgs in main-org and act-env are new
(retrieve (?msg-without-owner ?seq-md ?sender)
  (and (?msg-without-owner agent-msg)
    (?seq-md sequence-model)
    (?seq-md ?msg-without-owner has-msg)
    (?msg-without-owner ?sender has-msg-sender)
    (not (?msg-without-owner
      (some (inv has-msg) (or agent-role-class main-organization-class
        active-environment-class))))
)

;; Q3 check agent-msgs that are in a sequence-model, which sender is an agent-role
;; and which didn’t belong to ANY protocol
(retrieve (?msg-without-prtcl ?ag-role-sender ?seq-md)
  (and (?msg-without-prtcl agent-msg)
    (?seq-md sequence-model)
    (?seq-md ?msg-without-prtcl has-msg)
    (?ag-role-sender ?msg-without-prtcl is-msg-sender)
    (not (?msg-without-prtcl (some (inv has-msg) protocol)))
)

;; Q3A check agent-msgs that are in a sequence-model, which sender is an agent-role
;; but the msg didn’t belong to the sender protocols
;; This query also return the Q3 queries results!
(retrieve (?msg-without-prtcl ?ag-role-sender ?seq-md)
  (and (?msg-without-prtcl agent-msg)
    (?ag-role-sender agent-role-class-path)
    (?seq-md sequence-model)
    (?prtcl protocol)
    (?seq-md ?msg-without-prtcl has-msg)
    (?seq-md ?ag-role-sender has-path)
(??ag-role-sender ??msg-without-prtcl is-msg-sender)
(??ag-role-sender ??ag-role has-head)
(??ag-role ??prtcl has-protocol)
(not (??prtcl ??msg-without-prtcl has-msg))
)

;;; Q4 check seq-models which have plans and don't have any agent/org or
;;; act-env-class-paths
(retrieve (?seq-md ?plan-without-owner)
  (and (?seq-md sequence-model)
    (?plan-without-owner plan)
    (?seq-md ?plan-without-owner has-plan)
    (not (?seq-model (some (inv has-path) (or agent-class-path
      organization-class-path active-environment-class-path)))))
)

;;; Q5 check paths that aren't part of any sequence-model
(retrieve (?path-without-model)
  (and (?path-without-model path)
    (not (?path-without-model (some (inv has-path) sequence-model))))
)

;;; Q6 check sequence-models without structure
(retrieve (?seq-without-struct)
  (and (?seq-without-struct sequence-model)
    (?seq-without-struct nil has-structure))
)

;;; Q7 check bad-structured sequences in sequence-models
(retrieve (?seq-md-loop ?element1 ?element2)
  (and (?seq-md sequence-model)
    (?seq sequence)
    (?seq-md-loop ?seq has-structure)
    (?seq ?element1 has-first)
    (?seq ?element2 has-last)
    (?element2 ?element1 is-before)
)

;;; Q8 check bad-defined sequence-model which models a protocol
(retrieve (?seqmd ?role-without-prtcl ?prtcl)
  (and (?seqmd sequence-model)
    (?role-without-prtcl agent-role-class)
    (?prtcl protocol)
    (?role-path agent-role-class-path)
    (?seqmd ?role-path has-path)
    (?role-path ?role-without-prtcl has-head)
    (?struct sequence)
    (?seqmd ?struct has-structure)
    (?prtcl ?struct has-structure)
    (not (?role-without-prtcl ?prtcl has-protocol))
)
)
Apêndice C

A ontologia Ont2 para MAS-ML

Neste apêndice encontra-se o código da ontologia Ont2 construída durante a aplicação do Observed-MAS para MAS-ML.

```lisp
(in-tbox MASML-2-TBox)
(signature
 :atomic-concepts
  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
  ;;;;;;;  TAO concepts
  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
  class
  citizen-class
  agent-class
  organization-class
  main-organization-class
  sub-organization-class
  object-class
  environment-class
  active-environment-class
  passive-environment-class
  role-class
  agent-role-class
  object-role-class
  class-instance
  agent
  a-environment
  p-environment
  organization
  main-organization
  sub-organization
  object
  agent-role
  object-role
  features
  goal
  belief
  action
  plan
  duty
  right
  protocol
  axiom
  attribute
  method
  relationship
  inhabit
  play
  ownership
  specialization
  association
  aggregation
  control
  dependency
  play-in
  msg
  agent-msg
  object-msg
  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
  ;;;;;;;  MAS-ML concepts
  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
  model
  static-model
  class-model
  organization-model
  role-model
  sequence-model
  stereotyped-msg
  create-msg
  destroy-msg
  role-commit-msg
```
role-cancel-msg
role-activate-msg
role-deactivate-msg
role-change-msg
path
class-path
citizen-class-path
agent-class-path
organization-class-path
main-organization-class-path
sub-organization-class-path
object-class-path
environment-class-path
active-environment-class-path
passive-environment-class-path
role-class-path
agent-role-class-path
object-role-class-path
instance-path
citizen-instance-path
agent-instance-path
organization-instance-path
main-organization-instance-path
sub-organization-instance-path
object-instance-path
environment-instance-path
active-environment-instance-path
passive-environment-instance-path
role-instance-path
agent-role-instance-path
object-role-instance-path
sequence
seq-element

:roles (;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(has-goal :range goal)
(has-belief :range belief)
(has-axiom :range axiom
(has-duty :domain agent-role-class
:range duty)
(has-right :domain agent-role-class
:range right)
(has-action :range action
:parent has-element)
(has-plan :range plan)
(has-protocol :range protocol)
(has-end :domain relationship
:range class)
(has-end1 :parent has-end
:inverse is-end1)
(has-end2 :parent has-end
:inverse is-end2)
(is-end :domain class
:range relationship
:inverse has-end)
(is-end1 :domain class
:range relationship
:inverse has-end1)
(is-end2 :domain class
:range relationship
:inverse has-end2)
(has-msg :range msg
:parent has-element)
(has-msg-end :domain msg
:inverse is-msg-end)
(has-msg-end :inverse has-msg-end)
(has-msg-sender :parent has-msg-end
(has-msg-receiver :parent has-msg-end)
(has-msg-sender :inverse has-msg-sender)
(has-msg-receiver :inverse has-msg-receiver)
(is-instanceOf :domain class-instance
:range class)
(has-attribute :range attribute)
(has-method :range method)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(has-class :domain static-model
:range class
:inverse is-in-static-model)
(is-in-static-model :inverse has-class)
(has-relationship :domain static-model
:range relationship
:inverse is-relationship-of)
(is-relationship-of :inverse has-relationship)
(has-path :range path)
(has-head :domain path
   :range (or class-instance class))
(has-tail :domain path
   :range path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;; ordering properties
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(has-structure :range sequence)
(has-element :range seq-element)
(has-first :domain sequence
   :range seq-element
   :parent has-element
   :inverse is-first)
(is-first :inverse has-first)
(has-last :domain sequence
   :range seq-element
   :parent has-element
   :inverse is-last)
(is-last :inverse has-last)
(is-before :domain seq-element
   :range seq-element
   :transitive t
   :inverse is-after)
(is-after :domain seq-element
   :range seq-element
   :transitive t
   :inverse is-before)
)
)

;; concepts restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;; TAO concepts taxonomy
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies environment-class class)
(implies role-class class)
(implies citizen-class class)
(implies active-environment-class environment-class)
(implies passive-environment-class environment-class)
(implies agent-role-class role-class)
(implies object-role-class role-class)
(implies agent-class citizen-class)
(implies organization-class citizen-class)
(implies object-class citizen-class)
(implies main-organization-class organization-class)
(implies sub-organization-class organization-class)
(implies inhabit relationship)
(implies play relationship)
(implies ownership relationship)
(implies specialization relationship)
(implies association relationship)
(implies aggregation relationship)
(implies control relationship)
(implies dependency relationship)
(implies play-in relationship)
(implies goal features)
(implies belief features)
(implies action features)
(implies plan features)
(implies duty features)
(implies right features)
(implies protocol features)
(implies axiom features)
(implies attribute features)
(implies method features)
(implies agent class-instance)
(implies a-environment class-instance)
(implies p-environment class-instance)
(implies organization class-instance)
(implies main-organization organization)
(implies sub-organization organization)
(implies agent-role class-instance)
(implies object-role class-instance)
(implies agent-msg msg)
(implies object-msg msg)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;; MAS-ML concepts taxonomy (it reuses some TAO concepts)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies stereotyped-msg msg)
(implies static-model model)
(implies sequence-model model)
(implies class-model static-model)
(implies organization-model static-model)
(implies role-model static-model)
(implies create-msg stereotyped-msg)
(implies destroy-msg stereotyped-msg)
(implies role-commit-msg stereotyped-msg)
(implies role-cancel-msg stereotyped-msg)
(implies role-activate-msg stereotyped-msg)
(implies role-deactivate-msg stereotyped-msg)
(implies role-change-msg stereotyped-msg)
(equivalent path (or class-path instance-path))
(implies citizen-class-path class-path)
(implies role-class-path class-path)
(implies environment-class-path class-path)
(implies agent-class-path citizen-class-path)
(implies organization-class-path citizen-class-path)
(implies object-class-path citizen-class-path)
(implies agent-role-class-path role-class-path)
(implies object-role-class-path role-class-path)
(implies main-organization-class-path organization-class-path)
(implies sub-organization-class-path organization-class-path)
(implies active-environment-class-path environment-class-path)
(implies passive-environment-class-path environment-class-path)
(implies citizen-instance-path instance-path)
(implies role-instance-path instance-path)
(implies environment-instance-path instance-path)
(implies agent-instance-path citizen-instance-path)
(implies organization-instance-path citizen-instance-path)
(implies object-instance-path citizen-instance-path)
(implies main-organization-instance-path organization-instance-path)
(implies sub-organization-instance-path organization-instance-path)
(implies agent-role-instance-path role-instance-path)
(implies object-role-instance-path role-instance-path)
(implies a-environment-instance-path environment-instance-path)
(implies p-environment-instance-path environment-instance-path)

;; axiom that guarantee the possibility of the dynamic structure definition
(implies msg seq-element)
(implies action seq-element)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; ontology axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
// TAO disjoint restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint class class-instance features msg relationship)
(disjoint citizen-class environment-class role-class)
(disjoint agent-class organization-class object-class)
(disjoint main-organization-class sub-organization-class)
(disjoint active-environment-class passive-environment-class)
(disjoint agent-role-class object-role-class)
(disjoint agent a-environment p-environment main-organization
   sub-organization object agent-role object-role)
(disjoint agent-msg object-msg)
(disjoint goal belief protocol attribute)
(disjoint inhabit play ownership specialization association aggregation
   control dependency)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; MAS-ML disjoint restrictions
(disjoint static-model sequence-model)
(disjoint class-model organization-model role-model)
(disjoint create-msg destroy-msg role-commit-msg
  role-cancel-msg role-activate-msg role-deactivate-msg
  role-change-msg)
(disjoint class-path instance-path)

;; instances-related axioms
(implies agent (all is-instanceOf agent-class))
(implies organization (all is-instanceOf organization-class))
(implies main-organization (all is-instanceOf main-organization-class))
(implies sub-organization (all is-instanceOf sub-organization-class))
(implies object (all is-instanceOf object-class))
(implies a-environment (all is-instanceOf active-environment-class))
(implies p-environment (all is-instanceOf passive-environment-class))
(implies agent-role (all is-instanceOf agent-role-class))
(implies object-role (all is-instanceOf object-role-class))

;; path axioms
(implies agent-class-path (and (all has-head agent-class)
  (some has-tail agent-role-class-path)))
(implies object-class-path (and (all has-head object-class)
  (some has-tail object-role-class-path)))
(implies main-organization-class-path (and (all has-head main-organization-class)
  (some has-tail environment-class-path)))
(implies sub-organization-class-path
  (and (all has-head sub-organization-class)
  (some has-tail (or environment-class-path organization-class-path-path))))
(implies agent-role-class-path (and (all has-head agent-role-class)
  (some has-tail organization-class-path)))
(implies object-role-class-path (and (all has-head object-role-class)
  (some has-tail organization-class-path)))
(implies active-environment-class-path (and (all has-head active-environment-class)
  (all has-tail 'bottom*)))
(implies passive-environment-class-path (and (all has-head passive-environment-class)
  (all has-tail 'bottom*)))
(implies agent-instance-path (and (all has-head agent)
  (some has-tail agent-role-instance-path)))
(implies object-instance-path (and (all has-head object)
  (some has-tail object-role-instance-path)))
(implies main-organization-instance-path (and (all has-head main-organization)
  (some has-tail environment-instance-path)))
(implies sub-organization-instance-path
  (and (all has-head sub-organization)
  (some has-tail (or organization-instance-path environment-instance-path-path))))
(implies agent-role-instance-path (and (all has-head agent-role)
  (some has-tail organization-instance-path)))
(implies object-role-instance-path (and (all has-head object-role)
  (some has-tail organization-instance-path)))
(implies a-environment-instance-path (and (all has-head a-environment)
  (all has-tail 'bottom*)))
(implies p-environment-instance-path (and (all has-head p-environment)
  (all has-tail 'bottom*)))

;; order axioms
(implies (some has-first msg) (some has-msg msg))
(implies (some has-last msg) (some has-msg msg))

;; semantics of the first element of a sequence
(implies (and (some is-before *top*) (some is-after *bottom*)
  (some is-first *top*)))

;; semantics of the last element of a sequence
(implies (and (some is-after *top*) (some is-before *bottom*)
  (some is-last *top*)))

;; as has-first and has-msg are children of has-element, this relation must be declared
;; a first-msg and a last-msg are msgs
(implies (some has-first msg) (some has-msg msg))
(implies (some has-last msg) (some has-msg msg))
;;; the first element is always before the last element
(implies (some is-first *top*) (some is-before (some is-last *top*)))

Relationships semantics

;;; inhabit ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies inhabit (and
  (some has-end1 citizen-class)
  (some has-end2 environment-class)
))

;;; play ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies play
  (or (and (some has-end1 agent-class) (some has-end2 agent-role-class))
      (and (some has-end1 sub-organization-class) (some has-end2 agent-role-class))
      (and (some has-end1 object-class) (some has-end2 object-role-class))
))

;;; ownership ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies ownership (and
  (some has-end1 organization-class)
  (some has-end2 role-class)
))

;;; specialization ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies specialization
  (or (and (some has-end1 organization-class) (all has-end2 organization-class))
      (and (some has-end1 agent-class) (all has-end2 agent-class))
      (and (some has-end1 object-class) (all has-end2 object-class))
      (and (some has-end1 environment-class) (all has-end2 environment-class))
      (and (some has-end1 agent-role-class) (all has-end2 agent-role-class))
      (and (some has-end1 object-role-class) (all has-end2 object-role-class))
))

;;; association is commutative ;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies association
  (or
    (and (some has-end1 object-class) (some has-end2 citizen-class))
    (and (some has-end1 citizen-class) (some has-end2 object-class))
    (and (some has-end1 object-class) (some has-end2 role-class))
    (and (some has-end1 role-class) (some has-end2 object-class))
    (and (some has-end1 role-class) (some has-end2 role-class))
))

;;; aggregation ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies aggregation
  (or
    (and (some has-end1 object-class) (all has-end2 object-class))
    (and (some has-end1 agent-role-class) (all has-end2 agent-role-class))
    (and (some has-end1 object-role-class) (all has-end2 object-role-class))
))

;;; control
(implies control (and
  (some has-end1 agent-role-class)
  (all has-end2 agent-role-class)
))

;;; dependency ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies dependency
  (or (and (some has-end1 object-class) (all has-end2 object-class))
      (and (some has-end1 agent-role-class) (all has-end2 agent-role-class))
      (and (some has-end1 object-role-class) (all has-end2 object-role-class))
))
play-in is an implicit relationship between agent or sub-organization-classes and organization-classes.

\[ \text{play-in} \Rightarrow \begin{cases} 
\text{(or (and (some has-end1 agent-class) (all has-end2 organization-class)))} \\
\text{(and (some has-end1 sub-organization-class) (all has-end2 organization-class)))} 
\end{cases} \]

---

a class-model is a diagram composed of citizen classes and environment classes.

\[ \text{class-model} \Rightarrow \begin{cases} 
\text{(all has-class (or citizen-class environment-class)))} 
\end{cases} \]

---

association is used in class-models between object classes and citizen classes.

\[ \text{association} \Rightarrow \begin{cases} 
\text{(and association (or (and (some has-end1 (or citizen-class environment-class))) (some has-end2 object-class)))} \\
\text{(and (some has-end1 object-class) (some has-end2 (or citizen-class environment-class)))} 
\end{cases} \]

---

specialization is used between classes of the same type that take part in the diagram.

\[ \text{specialization} \Rightarrow \begin{cases} 
\text{(some has-relationship (or (and specialization (some has-end1 organization-class) (all has-end2 organization-class)))} \\
\text{(and specialization (some has-end1 agent-class) (all has-end2 agent-class)))} \\
\text{(and specialization (some has-end1 object-class) (all has-end2 object-class)))} \\
\text{(and specialization (some has-end1 environment-class) (all has-end2 environment-class)))} 
\end{cases} \]

---

aggregation is used between object-classes.

\[ \text{aggregation} \Rightarrow \begin{cases} 
\text{(some has-relationship (and aggregation (some has-end1 object-class) (all has-end2 object-class)))} 
\end{cases} \]

---

dependency is used between object-classes.

\[ \text{dependency} \Rightarrow \begin{cases} 
\text{(some has-relationship (and dependency (some has-end1 object-class) (all has-end2 object-class)))} 
\end{cases} \]
;; role-model axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; a role model is a diagram composed of role classes and object classes
(implies role-model (all has-class (or object-class role-class)))
;; the allowed relationships in this diagram are control, dependency, aggregation,
;; association and specialization
(implies role-model (some has-relationship control))
(implies role-model (some has-relationship
   (and dependency (some has-end1 agent-role-class) (all has-end2 role-class))
   (and dependency (some has-end1 object-role-class) (all has-end2 object-role-class))))
(implies role-model (some has-relationship
   (and aggregation (some has-end1 agent-role-class) (all has-end2 agent-role-class))
   (and aggregation (some has-end1 object-role-class) (all has-end2 object-role-class))))
;; association is used between agent-role classes and role classes,
;; between object-role classes, between object classes and role classes
(implies role-model (some has-relationship
   (and association
      (and (some has-end1 role-class)
           (some has-end2 object-class))
      (and (some has-end1 object-class)
           (some has-end2 role-class))
      (and (some has-end1 role-class)
           (some has-end2 role-class))
   )))
;; specialization is used between agent-role classes and between object-role classes
(implies role-model (some has-relationship
   (and specialization
      (and specialization (some has-end1 agent-role-class) (all has-end2 agent-role-class))
      (and specialization (some has-end1 object-role-class) (all has-end2 object-role-class))
   )))
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Sequence-model structure semantics
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies sequence-model (some has-path path))
(implies sequence-model (some has-msg msg))
(implies sequence-model (some has-structure sequence))
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; semantics of message definition
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; agent-msgs must be sent by agents-roles (through protocols), main-organizations or
;; active environments and they cannot be received by objects and object-roles
(implies agent-msg
   (or (and (some has-msg-sender (or agent-role-class-path agent-role-instance-path
      main-organization-class-path main-organization-instance-path
      active-environment-class-path a-environment-instance-path)
      (all has-msg-receiver (or agent-role-class-path agent-role-instance-path
         main-organization-class-path main-organization-instance-path
      environment-class-path environment-instance-path)))))
)c
;; agents, organizations and sub-organizations can create/destroy agents, objects, sub-
;; organizations
;; and environments that were different from the one they live in.
;; environments can create/destroy citizens and environments
;; objects can create/destroy only objects
(implies create-msg
   (or (and (some has-msg-sender (or agent-role-class-path agent-role-instance-path
      main-organization-class-path main-organization-instance-path
      active-environment-class-path a-environment-instance-path)
      (all has-msg-receiver (or agent-role-class-path agent-role-instance-path
         main-organization-class-path main-organization-instance-path
      environment-class-path environment-instance-path)))))
)
(or (and (some has-msg-sender (or main-organization-class-path main-organization-instance-path)
  (all has-msg-receiver (or agent-class-path agent-instance-path
    object-class-path object-instance-path
    sub-organization-class-path sub-organization-instance-path
    environment-class-path environment-instance-path)))
  (and (some has-msg-sender (or main-organization-class-path main-organization-instance-path
    environment-class-path environment-instance-path))
  (all has-msg-receiver (or agent-class-path agent-instance-path
    object-class-path object-instance-path
    sub-organization-class-path sub-organization-instance-path
    environment-class-path environment-instance-path)))
  (and (some has-msg-sender (or object-class-path object-instance-path)
  (all has-msg-receiver (or object-class-path object-instance-path))))
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
)
(and (some has-msg-sender (or main-organization-class-path main-organization-instance-path))
  (all has-msg-receiver (or object-class-path object-instance-path))) )

(implies (and (some has-head agent) (some is-msg-receiver role-cancel-msg))
  (and agent-instance-path (some is-msg-sender role-cancel-msg)))

(implies (and (some has-head agent-class) (some is-msg-receiver role-cancel-msg))
  (and agent-class-path (some is-msg-sender role-cancel-msg)))

(implies (and (some has-head sub-organization) (some is-msg-receiver role-cancel-msg))
  (and sub-organization-instance-path (some is-msg-sender role-cancel-msg)))

(implies (and (some has-head sub-organization-class) (some is-msg-receiver role-cancel-msg))
  (and sub-organization-class-path (some is-msg-sender role-cancel-msg)))

;; agents and sub-organizations can send role-activate-msgs to other agents or sub-organizations
;; in order to activate an inactive role.
(implies role-activate-msg
  (or (and (some has-msg-sender (or agent-class-path agent-instance-path))
    (all has-msg-receiver (or agent-class-path agent-instance-path
    object-class-path object-instance-path)))
    (and (some has-msg-sender (or sub-organization-class-path sub-organization-instance-path))
    (all has-msg-receiver (or sub-organization-class-path organization-instance-path)))))

;; if the sender and the receiver of a role-activate-msg is an agent-path then their heads must be
;; the same. This means that the same agent instance will activate an agent-role that was inactive.
(implies (and (some has-head agent) (some is-msg-receiver role-activate-msg))
  (and agent-instance-path (some is-msg-sender role-activate-msg)))
(implies (and (some has-head agent-class) (some is-msg-receiver role-activate-msg))
  (and agent-class-path (some is-msg-sender role-activate-msg)))
(implies (and (some has-head sub-organization) (some is-msg-receiver role-activate-msg))
  (and sub-organization-instance-path (some is-msg-sender role-activate-msg)))
(implies (and (some has-head sub-organization-class) (some is-msg-receiver role-activate-msg))
  (and sub-organization-class-path (some is-msg-sender role-activate-msg)))

;; agents and sub-organizations can send role-deactivate-msgs to other agents or sub-organizations
;; in order to deactivate an active role.
(implies role-deactivate-msg
  (or (and (some has-msg-sender (or agent-class-path agent-instance-path))
    (all has-msg-receiver (or agent-class-path agent-instance-path
    object-class-path object-instance-path)))
    (and (some has-msg-sender (or sub-organization-class-path sub-organization-instance-path))
    (all has-msg-receiver (or sub-organization-class-path sub-organization-instance-path)))))

;; if the sender and the receiver of a role-deactivate-msg is an agent-path then their heads must be
;; the same. This means that the same agent instance will deactivate an agent-role that was active.
(implies (and (some has-head agent) (some is-msg-receiver role-deactivate-msg))
  (and agent-instance-path (some is-msg-sender role-deactivate-msg)))
(implies (and (some has-head agent-class) (some is-msg-receiver role-deactivate-msg))
  (and agent-class-path (some is-msg-sender role-deactivate-msg)))
(implies (and (some has-head sub-organization) (some is-msg-receiver role-deactivate-msg))
  (and sub-organization-instance-path (some is-msg-sender role-deactivate-msg)))
(implies (and (some has-head sub-organization-class) (some is-msg-receiver role-deactivate-msg))
  (and sub-organization-class-path (some is-msg-sender role-deactivate-msg)))

;; agents and sub-organizations can send role-change-msgs to other agents or sub-organizations
;; in order to:
;; cancel an agent-role and commit to another agent-role,
;; cancel an agent-role and activate a deactive role,
;; deactivate an agent-role and activate a deactive role,
// deactivate an agent-role and commit to a new agent-role.
(implies role-change-msg
 (or (and (some has-msg-sender (or agent-class-path agent-instance-path))
          (all has-msg-receiver (or agent-class-path agent-instance-path
                                   object-class-path object-instance-path)))
     (and (some has-msg-sender (or sub-organization-class-path sub-organization-instance-path))
          (all has-msg-receiver (or sub-organization-class-path sub-organization-instance-path))))
)

// if the sender and the receiver of a role-change-msg is an agent-path then their heads must be the same. This means that the same agent instance will activate an agent-role that was inactive.
(implies (and (some has-head agent) (some is-msg-receiver role-change-msg))
         (and agent-instance-path (some is-msg-sender role-change-msg)))
(implies (and (some has-head agent-class) (some is-msg-receiver role-change-msg))
         (and agent-class-path (some is-msg-sender role-change-msg)))
(implies (and (some has-head sub-organization) (some is-msg-receiver role-change-msg))
         (and sub-organization-instance-path (some is-msg-sender role-change-msg)))
(implies (and (some has-head sub-organization-class) (some is-msg-receiver role-change-msg))
         (and sub-organization-class-path (some is-msg-sender role-change-msg)))

// internal properties
//

// agent state' semantics
//

// an agent must have at least one goal
(implies agent-class (some has-goal goal))

// an agent has at least one plan
(implies agent-class (some has-plan plan))

// an agent has at least one action
(implies agent-class (some has-action action))

// an agent has beliefs
(implies agent-class (some has-belief belief))

// an agent play at least one agent-role in an organization
(implies agent-class
 (some is-end1 (and play (some has-end2 (and agent-role-class (some is-end2 ownership))))))

// an agent inhabits an environment
(implies agent-class
 (some is-end1 (and inhabit (all has-end2 environment-class)))))

// organization state' semantics
//

// an organization has at least one goal
(implies organization-class (some has-goal goal))

// an organization has at least one plan
(implies organization-class (some has-plan plan))

// an organization has at least one action
(implies organization-class (some has-action action))

// an organization has axioms
(implies organization-class (some has-axiom axiom))

// an organization has beliefs
(implies organization-class (some has-belief belief))

// an organization inhabits an environment
(implies organization-class (some is-end1 (and inhabit (all has-end2 environment-class))))

// an organization owns at least one role
(implies organization-class (some is-end1 (and ownership (some has-end2 role-class)))))

// a sub-organization play at least one agent-role in an organization
(implies sub-organization-class
 (some is-end1 (and play (some has-end2 (and agent-role-class (some is-end2 ownership))))))
(some is-end1 (and play (all has-end2 (and agent-role-class (some is-end2 ownership))))))

;; main-organizations have msgs
(implies main-organization-class (some has-msg msg))

;;;;;;;;;;;
;;;;;;;;;;; agent-role state' semantics
;;;;;;;;;;;
;; an agent-role must have at least one goal
(implies agent-role-class (some has-goal goal))

;; an agent-role has beliefs
(implies agent-role-class (some has-belief belief))

;; an agent-role has protocols
(implies agent-role-class (some has-protocol protocol))

;; an agent-role is owned by an organization
;; we don't say which are the ownership end-1 since it was done during the organization-class
;; semantics definition
(implies agent-role-class (some is-end2 ownership))

;; an agent-role is played by someone
(implies agent-role-class (some is-end2 play))

;;;;;;;;;;;
;;;;;;;;;;;; object state' semantics
;;;;;;;;;;;
;; an object has attributes
(implies object-class (some has-attribute attribute))

;; an object has methods
(implies object-class (some has-method method))

;; an object plays object-roles
(implies object-class (some is-end1 (and play (all has-end2 object-role-class)))))

;;;;;;;;;;;
;;;;;;;;;;;; object-role state' semantics
;;;;;;;;;;;
;; an object-role has attributes
(implies object-role-class (some has-attribute attribute))

;; an object-role has methods
(implies object-role-class (some has-method method))

;; an object-role is played by an object
(implies object-role-class (some is-end2 play))

;; an object-role is owned by an organization
(implies object-role-class (some is-end2 (and ownership (all has-end1 organization-class))))

;;;;;;;;;;;
;;;;;;;;;;;; environment state' semantics
;;;;;;;;;;;
;; a passive-environment has attributes
(implies passive-environment-class (some has-attribute attribute))

;; a passive-environment has methods
(implies passive-environment-class (some has-method method))

;; an active-environment has goals
(implies active-environment-class (some has-goal goal))

;; an active-environment has plans
(implies active-environment-class (some has-plan plan))

;; an active-environmentagent has actions
(implies active-environment-class (some has-action action))

;; an active-environmentagent has beliefs
(implies active-environment-class (some has-belief belief))

;; an active-environmentagent has msgs
;; this was introduced, since it isn't mentioned in the MAS-ML metamodel
(implies active-environment-class (some has-msg msg))
Apêndice D

QV2 para o método aplicado a MAS-ML

Neste apêndice encontram-se as consultas QV2, definidas durante a aplicação do método para a linguagem MAS-ML.

```prolog
;;;;;;;;
;;; interdependências entre diagramas de classe e diagramas
;;; de organizações e de papéis
;;;;;;;
(RETRIEVE (?CLMD) (?CLMD CLASS-MODEL))

(RETRIEVE (?OBJ-CLASS ?ORG-DIAGRAM ?CLMD)
 (AND (?OBJ-CLASS OBJECT-CLASS) (?ORG-DIAGRAM ORGANIZATION-MODEL)
  (?ORG-DIAGRAM ?OBJ-CLASS HAS-CLASS) (?CLMD CLASS-MODEL)
  (NOT (?CLMD ?OBJ-CLASS HAS-CLASS))))

(RETRIEVE (?OBJ-CLASS ?ROLE-DIAGRAM ?CLMD)
 (AND (?OBJ-CLASS OBJECT-CLASS) (?ROLE-DIAGRAM ROLE-MODEL)
  (?ROLE-DIAGRAM ?OBJ-CLASS HAS-CLASS) (?CLMD CLASS-MODEL)
  (NOT (?CLMD ?OBJ-CLASS HAS-CLASS))))

;;;;;;;;
;;; interdependências entre diagramas de organizações e diagramas
;;; de classes e de papéis
;;;;;;;
(RETRIEVE (?ORGMD) (?ORGMD ORGANIZATION-MODEL))

(RETRIEVE (?AG-CLASS ?CLASS-DIAGRAM ?ORGMD)
 (AND (?AG-CLASS AGENT-CLASS) (?CLASS-DIAGRAM CLASS-MODEL)
  (?ORGMD ORGANIZATION-MODEL) (?CLASS-DIAGRAM ?AG-CLASS HAS-CLASS)
  (NOT (?ORG-MD ?AG-CLASS HAS-CLASS))))

(RETRIEVE (?ORG-CLASS ?CLASS-DIAGRAM ?ORGMD)
 (AND (?ORG-CLASS ORGANIZATION-CLASS) (?CLASS-DIAGRAM CLASS-MODEL)
  (?ORGMD ORGANIZATION-MODEL) (?CLASS-DIAGRAM ?ORG-CLASS HAS-CLASS)
  (NOT (?ORGMD ?ORG-CLASS HAS-CLASS))))

(RETRIEVE (?RCLASS ?ROLE-DIAGRAM ?ORGMD)
 (AND (?RCLASS ROLE-CLASS) (?ROLE-DIAGRAM ROLE-MODEL)
  (?ROLE-DIAGRAM ?RCLASS HAS-CLASS) (?ORGMD ORGANIZATION-MODEL)
  (NOT (?ORGMD ?RCLASS HAS-CLASS))))

;;;;;;;;
;;; interdependências entre diagramas estáticos e diagramas de
;;; sequências
;;;;;;;
;;; classless
(RETRIEVE (?IPATH ?SEQMD ?HEAD)
 (AND (?IPATH INSTANCE-PATH) (?SEQMD SEQUENCE-MODEL) (?SEQMD ?IPATH HAS-PATH)
  (?IPATH ?HEAD HAS-HEAD) (?HEAD NIL IS-INSTANCEOF)))

;;; caso especial de classless onde quem recebe a msg de criação
;;; de agente está associado a um papel de agente que é instância
;;; de uma classe que não está modelada em diagramas estáticos.
(RETRIEVE (?SEQMD ?AGENT-IPATH ?CREATE ?ROLE)
 (AND (?SEQMD SEQUENCE-MODEL) (?AGENT-IPATH AGENT-INSTANCE-PATH)
  (?SEQMD ?AGENT-IPATH HAS-PATH)
  (?CREATE CREATE-MSG) (?SEQMD 7CREATE HAS-MSG)
  (?AGENT-PATH 7CREATE IS-MSG-RECEIVER) (?AGENT-PATH 7ROLE-PATH HAS-TAIL)
  (?ROLE AGENT-ROLE) (?ROLE-PATH 7ROLE HAS-HEAD)
  (?ROLE NIL IS-INSTANCEOF)))
```
Apêndice E
QD1 e QD2 para o método aplicado a MAS-ML

Neste apêndice encontram-se as consultas QD1 e QD2, sobre sugestões de boas práticas de modelagem, definidas durante a aplicação do método para a linguagem MAS-ML.

;;;;;;;;;;;;;;;;;;;;;;;;; QD1
;;; Guideline rules

;;; Q1 helping the designer to build plans which have the set of allowed actions already assigned
(retrieve (?plan ?1st-act)
  (and (?plan plan)
    (?1st-act action)
    (?plan ?1st-act has-action)
    (?1st-act NIL has-precond)
  )
)

;;; return a plan and the candidates to be its last actions
(retrieve (?plan ?1ast-act)
  (and (?plan plan)
    (?1ast-act action)
    (?plan ?1ast-act has-action)
    (?1ast-act ?cond has-poscond)
    (?plan ?cond has-goal)
  )
)

;;; return a plan with 3 or more actions and the possible ordering between them. It shows to designer possible structures to the plan
(retrieve (?plan ?act1 ?act2 ?act3)
  (and (?plan plan)
    (?act1 action) (?act2 action) (?act3 action)
    (?plan ?act1 has-action)
    (?plan ?act2 has-action)
    (?plan ?act3 has-action)
    (?act1 ?cond1 has-poscond)
    (?act1 ?cond1 has-precond)
    (?act2 ?cond2 has-poscond)
    (?act2 ?cond2 has-precond)
    (?act3 ?cond2 has-precond)
  )
)

;;; Q2 helping the designer to assign suited roles to the agent-classes
(retrieve (?agent ?role)
  (and (?agent agent-class)
    (?role agent-role-class)
    (?agent ?goal has-goal)
    (?role ?goal has-goal)
  )
)

;;; Q3 helping the designer to relate agent-role-classes according to the protocols they contain. This query can also be posted in the execution time, considering given agent-role-class and protocol in order to find agent-role classes to relate.
(retrieve (?role-1 ?role-2 ?prtcl)
  (and (?role-1 agent-role-class)
    (?role-2 agent-role-class)
    (?prtcl protocol)
    (?role-1 ?prtcl has-protocol)
    (?role-2 ?prtcl has-protocol)
  )
)

;;;;;;;;;;;;;;;;;;;;;;;;; QD2
;;;
consultas referentes à sugestões de boas práticas de modelagem usando MAS-ML e considerando diagramas de organização e papel

(retrieve (?agrl ?orgmd)
  (and (?agrl agent-role-class)
    (?orgmd role-model)
    (?orgmd organization-model)
    (?orgmd ?agrl has-class)
    (not (?rolemd ?agrl has-class))))

(retrieve (?rolemd) (?rolemd role-model))

;;; consulta referente a boas práticas de modelagem, envolvendo diagrama de papéis e de sequência.

(retrieve (?prtcl-not-modeled)
  (and (?rel (or association dependency aggregation control))
    (?rel has-end1)
    (?rel has-end2)
    (?role-class1 agent-role-class)
    (?role-class2 agent-role-class)
    (?rel ?role-class1 has-end1)
    (?rel ?role-class2 has-end2)
    (?prtcl-not-modeled protocol)
    (?role-class1 ?prtcl-not-modeled has-protocol)
    (?role-class2 ?prtcl-not-modeled has-protocol)
    (?struct sequence)
    (?prtcl-not-modeled ?struct has-structure)
    (?seqmd sequence-model)
    (not (?seqmd ?struct has-structure))
  )
)
Apêndice F

Resultados do estudo de caso sobre mercados virtuais usando o método aplicado a MAS-ML

Código da base de conhecimento usada na fase F1

Neste apêndice encontra-se o código da base de conhecimento usada no estudo de caso sobre mercados virtuais, usando a aplicação do método para a linguagem MAS-ML. Esta base refere-se à execução da fase F1 do Observed-MAS e possui erros de modelagem, cuja introdução foi deliberada.

```prolog
(in-abox phase1-ABox MASML-TBox)
;; class-model
(instance class-diagram class-model)
;; organization-model
(instance org1-diagram organization-model)
;; role-model
(instance role-diagram role-model)
;; sequence-model
(instance seq-model sequence-model)
(instance seq-model-struct sequence)
(instance seq-model-struct-2 sequence)
;; virtual-market is an environment-class
(instance virtual-market passive-environment-class)
(instance supermarket passive-environment-class)
;; general-store, imported-bookstore, 2nd-hand-bookstore are organization-classes
(instance general-store main-organization-class)
(instance imported-bookstore sub-organization-class)
(instance 2nd-hand-bookstore sub-organization-class)
;; user-agent and store-agent are agent-classes
(instance user-agent agent-class)
(instance store-agent agent-class)
(instance super-user agent-class)
;; item and book are object-classes
(instance item object-class)
(instance book object-class)
(instance magazine object-class)
;; has-plan (GeneralQueries-Q33-OK)
(instance plan-1 plan)
(related user-agent plan-1 has-plan)
(related plan-1 seq-model-struct-2 has-structure)
;; has-action (GeneralQueries-Q34-OK)
(instance action-1 action)
(related user-agent action-1 has-action)
;; ERROR has-plan (GeneralQueries-Q33-OK)
(instance plan-2 plan)
;; ERROR has-plan (GeneralQueries-Q33-OK)
(instance plan-3 plan)
(related book plan-1 has-plan)
;; ERROR has-action (GeneralQueries-Q34-OK)
(instance action-2 action)
;; ERROR has-action (GeneralQueries-Q34-OK)
(instance action-3 action)
(related book action-1 has-action)
;; roles
(instance seller agent-role-class)
(instance buyer agent-role-class)
(instance offer object-role-class)
(instance desire object-role-class)
(instance mediator agent-role-class)
;; agents
```
(instance mary agent)
(related mary user-agent is-instanceOf)
(related mary-path agent-instance-path)
(related mary-path mary has-head)
(related user-agent-path agent-class-path)
(related john agent)
(related john user-agent is-instanceOf)
(related john-path agent-instance-path)
(related vendor-americanas agent)
(related vendor-americanas store-agent is-instanceOf)
(related vendor-americanas vendor-americanas-path has-path)
(related store-agent store-agent-path has-path)

;; defining a messages for an agent-role and inserting it into a protocol (GeneralQueries-Q31-OK)
(related agent-msg-1 agent-msg)
(related buyer agent-msg-1 has-msg)
(related agent-msg-2 agent-msg)
(related buyer agent-msg-2 has-msg)
(related agent-msg-3 agent-msg)
(related buyer agent-msg-3 has-msg)
(related protocol-1 protocol)
(related buy protocol-1 has-protocol)
(related protocol-1 agent-msg-1 has-msg)
(related protocol-1 agent-msg-2 has-msg)
(related protocol-1 agent-msg-3 has-msg)

;; protocol has-structure (GeneralQueries-Q37-OK)
(related protocol-1 seq-model-struct has-structure)
(related seq-model-struct has-first (GeneralQueries-Q39-OK))
(related seq-model-struct agent-msg-1 has-first)
(related seq-model-struct has-last (GeneralQueries-Q39-OK))
(related seq-model-struct agent-msg-1 has-last)
(related seq-model-struct has-element
(related seq-model-struct agent-msg-2 has-element)
(related agent-msg-3 has-first (GeneralQueries-Q39-OK))
(related agent-msg-1 agent-msg-2 is-before)
(related agent-msg-2 agent-msg-3 is-before)

;; ERROR sequence has-first (GeneralQueries-Q39)
;; ERROR sequence has-last (GeneralQueries-Q39)

;; organization
(related americanas-dotcom main-organization)
(related americanas-dotcom main-organization-instance-path)
(related americanas-dotcom general-store is-instanceOf)
(related americanas-dotcom americanas-dotcom-path has-path)
(related general-store general-store-path has-path)
(related intl-bookstore-dotcom sub-organization)
(related intl-bookstore-dotcom-path sub-organization-instance-path)
(related imported-bookstore-path sub-organization-class-path)

;; environment
(related webshopping p-environment)
(related webshopping-path p-environment-instance-path)
(related virtual-market-path passive-environment-class-path)
(related webshopping virtual-market is-instanceOf)
(related webshopping webshopping-path has-path)

;; objects
(related sommerville2001 object)
(related sommerville2001-path object-instance-path)
(related book-path object-class-path)
(related sommerville2001 book is-instanceOf)
(related sommerville2001 sommerville2001-path has-path)
(related book book-path has-path)

;; roles
(related activate-role-3 john-path has-msg-sender)
(related activate-role-3 mary-path has-msg-receiver)
;; role-deactivate-msg (GeneralQueries-Q27-OK)
(instance deactivate-role-1 role-deactivate-msg)
(related deactivate-role-1 john-path has-msg-sender)
(related deactivate-role-1 john-path has-msg-receiver)
;; ERROR role-deactivate-msg (GeneralQueries-Q27-OK)
(instance deactivate-role-2 role-deactivate-msg)
(related deactivate-role-2 john-path has-msg-sender)
(related deactivate-role-2 special-offer-path has-msg-receiver)
;; instance change-role-1 role-change-msg
(related change-role-1 john-path has-msg-sender)
(related change-role-1 special-offer-path has-msg-receiver)
;; object-msg (GeneralQueries-Q20)
(instance getTitle object-msg)
(related getTitle user-agent-path has-msg-sender)
(related getTitle book-path has-msg-receiver)
;; ERROR object-msg (GeneralQueries-Q20)
(instance getAbstract object-msg)
(related getAbstract book-path has-msg-sender)
(related getAbstract book-path has-msg-receiver)
;; agent-msg (GeneralQueries-Q21)
(instance inform agent-msg)
(related inform user-agent-path has-msg-sender)
(related inform store-agent-path has-msg-receiver)
;; ERROR agent-msg (GeneralQueries-Q21-OK)
(instance proposal agent-msg)
(related proposal user-agent-path has-msg-sender)
(related proposal book-path has-msg-receiver)
;; specialization
(instance spec-1 specialization)
(related spec-1 book has-end1)
(related spec-1 item has-end2)
(related spec-2 specialization)
(related spec-2 user-agent has-end1)
(related spec-2 super-user has-end2)
;; aggregation
(instance aggr-1 aggregation)
(related aggr-1 imported-bookstore has-end1)
(related aggr-1 2nd-hand-bookstore has-end2)
;; ownership
(instance own-1 ownership)
(related own-1 general-store has-end1)
(related own-1 buyer has-end2)
(instance own-2 ownership)
(related own-2 imported-bookstore has-end1)
(related own-2 desire has-end2)
;; ERROR (GeneralQueries-Q42-OK)
;; O papel seller está sendo desempenhado pelo agente store-agent mas não está definido em
nenhuma organização
;; (instance own-1 ownership)
;; (related own-1 general-store has-end1)
;; (related own-1 seller has-end2)
(GeneralQueries-Q42-OK)
(instance play-5 play)
Respostas às consultas QV1

Respostas obtidas a partir das consultas QV1 aplicadas à base de conhecimento descrita na seção anterior.
Código da base de conhecimento usada na fase F2

```
(in-abox phase2-ABox MASML-2-TBox)
;; class-model
(instance class-diagram class-model)
;; organization-model
(instance org1-diagram organization-model)
;; sequence-model
(instance seq-model sequence-model)
(instance seq-model-struct sequence)
;; virtual-market is an environment-class
(instance virtual-market passive-environment-class)
;; general-store, imported-bookstore, 2nd-hand-bookstore are organization-classes
(instance general-store main-organization-class)
(instance imported-bookstore main-organization-class)
(instance 2nd-hand-bookstore main-organization-class)
;; user-agent and store-agent are agent-classes
(instance user-agent agent-class)
(instance store-agent agent-class)
(instance super-user-agent agent-class)
;; item and book are object-classes
(instance item object-class)
(instance book object-class)
(instance magazine object-class)
;; role-model
(instance role-diagram role-model)
;; roles
(instance seller agent-role-class)
(instance buyer agent-role-class)
(instance offer object-role-class)
(instance desire object-role-class)
(instance mediator agent-role-class)
(instance market-of-used-goods agent-role-class)
(instance market-of-special-goods agent-role-class)
;; teste protocol
(instance market-of-used-goods negotiate-prtcl has-protocol)
(instance market-of-special-goods negotiate-prtcl has-protocol)
;; association
(instance assoc-2 association)
 stale assoc-2 book has-end1
 stale assoc-2 magazine has-end2
 stale assoc-3 book has-end1
 stale assoc-3 desire has-end2
 stale assoc-5 association
 stale assoc-5 buyer has-end1
 stale assoc-5 desire has-end2
 stale assoc-6 association
 stale assoc-6 seller has-end1
 stale assoc-6 offer has-end2
 stale org1-diagram organization-model
 stale general-store main-organization-class
 stale item object-class
 stale book object-class
 stale spec-1 specialization
 stale spec-1 book has-end1
 stale spec-1 item has-end2
 stale org1-diagram general-store has-class
 stale org1-diagram item has-class
 stale org1-diagram book has-class
 stale org1-diagram spec-1 has-relationship
 stale inhabit-1 inhabit
 stale inhabit-1 user-agent has-end1
 stale inhabit-1 virtual-market has-end2
 stale inhabit-2 inhabit
 stale inhabit-2 store-agent has-end1
 stale inhabit-2 virtual-market has-end2
 stale inhabit-3 inhabit
 stale inhabit-3 general-store has-end1
 stale inhabit-3 virtual-market has-end2
 stale inhabit-4 inhabit
 stale inhabit-4 2nd-hand-bookstore has-end1
 stale inhabit-4 virtual-market has-end2
 stale inhabit-4 inhabit
 stale inhabit-4 imported-bookstore has-end1
 stale inhabit-4 imported-bookstore has-end2
 stale inhabit-5 inhabit
 stale inhabit-5 book has-end1
 stale inhabit-5 virtual-market has-end2
 stale inhabit-6 inhabit
 stale inhabit-6 item has-end1
 stale inhabit-6 virtual-market has-end2
 stale spec-1 specialization
 stale spec-1 book has-end1
 stale spec-1 item has-end2
 stale own-1 ownership
 stale own-1 general-store has-end1
 stale own-1 buyer has-end2
 stale own-2 ownership
 stale own-2 general-store has-end1
 stale own-2 seller has-end2
 stale own-3 ownership
 stale own-3 general-store has-end1
 stale own-3 desire has-end2
 stale own-4 ownership
```
Respostas às consultas QV2

(IN-TBOX MASML-2-TBOX) --> MASML-2-TBOX
(TBOX-COHERENT-P) --> T
(IN-ABOX PHASE2-ABOX MASML-2-TBOX) --> PHASE2-ABOX
(CHECK-ABOX-COHERENCE) --> T
(RETRIEVE (?IPATH INSTANCE-PATH) (AND (?SEQMD SEQUENCE-MODEL) (?SEQMD ?IPATH HAS-PATH) (?IPATH HEAD HAS-HEAD) (?HEAD NIL IS-INSTANCEOF))) ) --> (((?IPATH PAPER01-PATH) (?SEQMD SEQ-1) (?HEAD PAPER-01))))
Apêndice G

Resultados do estudo de caso Expert Committee usando o método aplicado a MAS-ML

Código da base de conhecimento usada na fase F1

Nesta seção encontra-se o código da base de conhecimento usada no estudo de caso sobre o Expert Committee, usando a aplicação do método para a linguagem MAS-ML. Esta base refere-se à execução da fase F1 do Observed-MAS e possui erros de modelagem, cuja introdução foi deliberada.

```
(in-abox phase1-ABox MASML-TBox)
; creating the environment of the EC system
(instance expert-committee passive-environment-class)
(instance expert-committee-path passive-environment-class-path)
(related expert-committee-path expert-committee has-head)

; creating the main-organization of the EC System
(instance organizing-institution main-organization-class)
(instance organizing-institution-path main-organization-class-path)
(related organizing-institution-path organizing-institution has-head)
(related organizing-institution-path expert-committee-path has-tail)

;;;;;;; internal state of the main organization

; creating the main-organization goals
(instance start-event goal)
(related organizing-institution start-event has-goal)

; creating the main-organization axioms
(instance dont-play-CA axiom)
(instance dont-play-CHCM axiom)
(instance dont-play-CR axiom)
(instance dont-play-CCH axiom)
(instance reviewer-not-author axiom)
(instance pcmember-not-author axiom)
(instance dont-play-CMC axiom)

(related organizing-institution dont-play-CA has-axiom)
(related organizing-institution dont-play-CHCM has-axiom)
(related organizing-institution dont-play-CR has-axiom)
(related organizing-institution dont-play-CCH has-axiom)
(related organizing-institution reviewer-not-author has-axiom)
(related organizing-institution pcmember-not-author has-axiom)
(related organizing-institution dont-play-CMC has-axiom)

; creating the main-organization actions
(instance to-create-roles action)
(instance to-create-reviewer action)
(instance to-create-PCmember action)
(instance to-create-PCchair action)
(instance to-create-PCmanager action)

(related organizing-institution to-create-roles has-action)
(related organizing-institution to-create-reviewer has-action)
(related organizing-institution to-create-PCmember has-action)
(related organizing-institution to-create-PCchair has-action)
(related organizing-institution to-create-PCmanager has-action)
(related organizing-institution to-create-agenda has-action)

; creating the main-organization plans
```
(instance start-event-plan-plan)
(related organizing-institution start-event-plan-plan has-plan)
(instance start-event-plan-struct sequence)
(related start-event-plan-plan start-event-plan-struct has-plan)
(related start-event-plan-plan start-event-plan-struct has-structure)
(related start-event-plan-plan to-create-PCmanager has-action)
(related start-event-plan-plan to-create-PCmember has-action)
(related start-event-plan-plan to-create-reviewer has-action)
(related start-event-plan-plan to-create-roles has-action)
(related start-event-plan-plan to-create-PCchair has-action)

(related start-event-plan-struct to-create-PCmanager has-first)
(related start-event-plan-struct to-create-PCchair has-last)

(related start-event-plan-struct to-create-PCmanager has-element)
(related start-event-plan-struct to-create-PCmember has-element)
(related start-event-plan-struct to-create-roles has-element)
(related start-event-plan-struct to-create-PCchair has-element)

(related to-create-PCmanager to-create-PCmember is-before)
(related to-create-PCmember to-create-reviewer is-before)
(related to-create-reviewer to-create-roles is-before)
(related to-create-roles to-create-PCchair is-before)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;end internal state main-org

; creating roles that are owned by the main-organization
(instance reviewer agent-role-class)
(instance reviewer-path agent-role-class-path)
(related reviewer-path reviewer has-head)
(related reviewer-path organizing-institution-path has-tail)

(instance chair agent-role-class)
(instance chair-path agent-role-class-path)
(related chair-path chair has-head)
(related organizing-institution-path chair-path has-tail)

(instance manager agent-role-class)
(instance manager-path agent-role-class-path)
(related manager-path manager has-head)
(related manager-path organizing-institution-path has-tail)

(instance pc-member agent-role-class)
(instance pc-member-path agent-role-class-path)
(related pc-member-path pc-member has-head)
(related pc-member-path organizing-institution-path has-tail)

(instance author agent-role-class)
(instance author-path agent-role-class-path)
(related author-path author has-head)
(related author-path organizing-institution-path has-tail)

; creating the main-organization's ownership relationships
(instance own-1 ownership)
(instance own-2 ownership)
(instance own-3 ownership)
(instance own-4 ownership)
(instance own-5 ownership)

; resources of the system
(instance paper object-class)
(instance presentation object-class)
(instance proposition object-class)
(instance result object-class)
(instance vote object-class)
(instance conflict object-class)
(instance camera-ready object-class)

; creating the relationships between objects
(instance assoc-1-has association)
(instance assoc-2-has association)
(instance assoc-3-has association)
(instance assoc-4-has association)
(instance assoc-conflict association)

(instance dep-1 dependency)

(instance inhabit-1 inhabit)
(instance inhabit-2 inhabit)
(instance inhabit-3 inhabit)
(instance inhabit-4 inhabit)
(instance inhabit-5 inhabit)
(instance inhabit-6 inhabit)
(instance inhabit-7 inhabit)
(instance inhabit-8 inhabit)

; agents of the system
; creating the researcher goals
(instance receive-camera-ready goal)
(instance submit-paper goal)
(instance distribute-paper goal)
(instance receive-final-results goal)
(instance review-paper goal)
(instance solve-conflict goal)
(instance accept-paper goal)
(instance present-final-results goal)
(instance vote-to-solve goal)
(instance make-review goal)
(instance send-camera-ready goal)
(instance engage-reviewer goal)
(instance provide-reviewer goal)

(instance related researcher-agent receive-camera-ready has-goal)
(instance related researcher-agent submit-paper has-goal)
(instance related researcher-agent distribute-paper has-goal)
(instance related researcher-agent receive-final-results has-goal)
(instance related researcher-agent review-paper has-goal)
(instance related researcher-agent accept-paper has-goal)
(instance related researcher-agent solve-conflict has-goal)
(instance related researcher-agent present-final-results has-goal)
(instance related researcher-agent vote-to-solve has-goal)
(instance related researcher-agent make-review has-goal)
(instance related researcher-agent send-camera-ready has-goal)
(instance related researcher-agent engage-reviewer has-goal)
(instance related researcher-agent provide-reviewer has-goal)

; creating the researcher actions
(instance to-make-additional-relation action)
(instance to-make-presentation action)
(instance to-select-new-reviewer action)
(instance to-register-vote action)
(instance to-search-for-conflict action)
(instance to-vote action)
(instance to-register-result action)
(instance to-make-final-result action)
(instance to-review action)
(instance to-prepare-paper-pack action)
(instance to-check-less-than-3-reviewer action)
(instance to-make-camera-ready action)
(instance to-make-paper action)
(instance to-register-paper-and-presentation action)
(instance to-select-reviewer action)
(instance to-make-review-proposition action)
(instance to-refresh-reviewer action)
(instance to-register-camera-ready action)
(instance to-assess-review-proposition action)

(instance related researcher-agent to-make-additional-relation has-action)
(instance related researcher-agent to-make-presentation has-action)
(instance related researcher-agent to-select-new-reviewer has-action)
(instance related researcher-agent to-register-vote has-action)
(instance related researcher-agent to-search-for-conflict has-action)
(instance related researcher-agent to-vote has-action)
(instance related researcher-agent to-register-result has-action)
(instance related researcher-agent to-make-final-result has-action)
(instance related researcher-agent to-review has-action)
(instance related researcher-agent to-prepare-paper-pack has-action)
(instance related researcher-agent to-check-less-than-3-reviewer has-action)
(instance related researcher-agent to-make-camera-ready has-action)
(instance related researcher-agent to-make-paper has-action)
(instance related researcher-agent to-register-paper-and-presentation has-action)
(instance related researcher-agent to-select-reviewer has-action)
(instance related researcher-agent to-make-review-proposition has-action)
(instance related researcher-agent to-refresh-reviewer has-action)
(instance related researcher-agent to-register-camera-ready has-action)
(instance related researcher-agent to-assess-review-proposition has-action)

; creating the researcher plans
(instance receive-camera-ready-plan plan)
(instance send-camera-ready-plan plan)
(instance submit-paper-plan plan)
(instance receive-paper-plan plan)
(instance assign-paper-plan plan)
(instance accept-paper-plan plan)
(instance provide-reviewer-plan plan)
(instance engage-reviewer-plan plan)
(instance make-review-plan plan)
(instance review-plan plan)
(instance solve-conflict-plan plan)
(instance vote-conflict-plan plan)
(instance show-final-result-plan plan)

; assigning the researcher's plans to it
(related researcher-agent receive-camera-ready-plan has-plan)
(related researcher-agent send-camera-ready-plan has-plan)
(related researcher-agent submit-paper-plan has-plan)
(related researcher-agent assign-paper-plan has-plan)
(related researcher-agent receive-paper-plan has-plan)
(related researcher-agent accept-paper-plan has-plan)
(related researcher-agent provide-reviewer-plan has-plan)
(related researcher-agent engage-reviewer-plan has-plan)
(related researcher-agent make-review-plan has-plan)
(related researcher-agent review-plan has-plan)
(related researcher-agent solve-conflict-plan has-plan)
(related researcher-agent vote-conflict-plan has-plan)
(related researcher-agent show-final-result-plan has-plan)

;; turning the researcher into a citizen
(instance inhabit-9 inhabit)
(related researcher inhabit-9 is-end)
(related expert-committee inhabit-9 is-end)

; structuring the researcher plans
(instance accept-paper-plan-struct sequence)
(related accept-paper-plan-struct has-structure)
(instance submit-paper-plan-struct sequence)
(related submit-paper-plan-struct has-structure)
(instance review-plan-struct sequence)
(related review-plan-struct has-structure)
(instance receive-paper-plan-struct sequence)
(related receive-paper-plan-struct has-structure)
(instance make-review-plan-struct sequence)
(related make-review-plan-struct has-structure)
(instance engage-reviewer-plan-struct sequence)
(related engage-reviewer-plan-struct has-structure)
(instance provide-reviewer-plan-struct sequence)
(related provide-reviewer-plan-struct has-structure)
(instance receive-camera-ready-plan-struct sequence)
(related receive-camera-ready-plan-struct has-structure)
(instance send-camera-ready-plan-struct sequence)
(related send-camera-ready-plan-struct has-structure)
(instance show-final-result-plan-struct sequence)
(related show-final-result-plan-struct has-structure)
(instance solve-conflict-plan-struct sequence)
(related solve-conflict-plan-struct has-structure)
(instance vote-conflict-plan-struct sequence)
(related vote-conflict-plan-struct has-structure)

; creating the researcher beliefs (no beliefs assigned)

; creating the play relationships the researcher plays
(instance play-1 play)
(instance play-2 play)
(instance play-3 play)
(instance play-4 play)
(instance play-5 play)

; creating the internal structure of the agent-roles

; agent-role beliefs and goals (agent-role goals are the same of the researcher agent goals)
(instance review-proposal-belief belief)
(instance package-list-belief belief)
(instance paper-belief belief)
(instance paper-list-belief belief)
(instance result-belief belief)
(instance review-belief belief)
(instance assessment-belief belief)
(related reviewer review-proposal-belief has-belief)
(related reviewer package-list-belief has-belief)
(related reviewer paper-list-belief has-belief)
(related reviewer result-belief has-belief)
(related reviewer review-belief has-belief)
(related reviewer assessment-belief has-belief)
(related reviewer accept-paper has-goal)
(related reviewer review-paper has-goal)

(instance presentation-belief belief)
(instance final-result-belief belief)
(instance final-result-list-belief belief)
(instance camera-ready-belief belief)
(instance reviewers-list-belief belief)
(instance presentation-list-belief belief)
(instance deadline-reviewer-belief belief)
(instance result-list-belief belief)
(instance deadline-paper-belief belief)
(instance deadline-camera-ready-belief belief)
(instance review-acceptance-belief belief)
(instance voting-list-belief belief)
(instance camera-ready-list-belief belief)
(instance additional-relation-belief belief)

(related author presentation-belief has-belief)
(related author paper-belief has-belief)
(related author review-belief has-belief)
(related author final-result-belief has-belief)
(related author camera-ready-belief has-belief)

(related author submit-paper has-goal)
(related author receive-final-results has-goal)
(related author send-camera-ready has-goal)

(related chair chair-reviewers-list-belief has-belief)
(related chair paper-list-belief has-belief)
(related chair presentation-list-belief has-belief)
(related chair deadline-reviewer-belief has-belief)
(related chair review-proposal-list-belief has-belief)
(related chair review-acceptance-belief has-belief)
(related chair package-list-belief has-belief)
(related chair result-list-belief has-belief)
(related chair voting-list-belief has-belief)
(related chair camera-ready-list-belief has-belief)
(related chair final-result-list-belief has-belief)
(related chair additional-relation-belief has-belief)

(related chair deadline-paper-belief has-belief)
(related chair deadline-camera-ready-belief has-belief)

(related chair receive-paper has-goal)
(related chair distribute-paper has-goal)
(related chair engage-reviewer has-goal)
(related chair make-review has-goal)
(related chair solve-conflict has-goal)
(related chair present-final-results has-goal)
(related chair receive-camera-ready has-goal)

(instance reviewers-list-belief belief)

(related manager reviewers-list-belief has-belief)

(related manager provide-reviewer has-goal)

(instance voting-belief belief)
(instance conflict-belief belief)

(related pc-member voting-belief has-belief)
(related pc-member conflict-belief has-belief)

(related pc-member solve-conflict has-goal)

agent-role duties
(instance duty-make-additional-relation duty)
(instance duty-make-presentation duty)
(instance duty-select-new-reviewer duty)
(instance duty-register-vote duty)
(instance duty-search-for-conflict duty)
(instance duty-vote duty)
(instance duty-register-result duty)
(instance duty-make-final-result duty)
(instance duty-review duty)
(instance duty-prepare-paper-pack duty)
(instance duty-check-less-then-3-reviewer duty)
(instance duty-make-camera-ready duty)
(related chair submit-msg has-msg)
(related chair send-vote-msg has-msg)
(related chair send-msg has-msg)
(related chair inform-msg has-msg)
(related chair inform-deadline-msg has-msg)

(related manager send-request-msg has-msg)
(related manager send-relation-msg has-msg)

(related pc-member send-conflict-msg has-msg)
(related pc-member send-vote-msg has-msg)

(related reviewer send-proposal-msg has-msg)
(related reviewer send-pack-msg has-msg)
(related reviewer send-result-msg has-msg)
(related reviewer inform-msg has-msg)

(related author submit-msg has-msg)
(related author send-final-result-msg has-msg)
(related author inform-deadline-msg has-msg)
(related author send-msg has-msg)

; structuring the agent-roles protocols
(instance submit-struct sequence)
(related submit-paper-and-presentation-prtcl submit-struct has-structure)
(related submit-paper-and-presentation-prtcl submit-msg has-msg)
(related submit-struct submit-msg has-element)
(related submit-msg author-path has-msg-sender)
(related submit-msg chair-path has-msg-receiver)
(related submit-struct submit-msg has-first)
(related submit-struct submit-msg has-last)

(instance send-proposal-struct sequence)
(related send-review-proposal-prtcl send-proposal-struct has-structure)
(related send-review-proposal-prtcl send-proposal-msg has-msg)
(related send-proposal-struct send-proposal-msg has-element)
(related send-proposal-msg chair-path has-msg-sender)
(related send-proposal-msg reviewer-path has-msg-receiver)
(related send-proposal-struct send-proposal-msg has-first)
(related send-proposal-struct send-proposal-msg has-last)

(instance inform-acceptance-struct sequence)
(related inform-review-acceptance-prtcl inform-acceptance-struct has-structure)
(related inform-review-acceptance-prtcl inform-acceptance-msg has-msg)
(related inform-acceptance-struct inform-msg has-element)
(related inform-msg reviewer-path has-msg-sender)
(related inform-msg chair-path has-msg-receiver)
(related inform-acceptance-struct inform-msg has-first)
(related inform-acceptance-struct inform-msg has-last)

(instance send-result-struct sequence)
(related send-result-prtcl send-result-struct has-structure)
(related send-result-prtcl send-result-msg has-msg)
(related send-result-struct send-result-msg has-element)
(related send-result-msg reviewer-path has-msg-sender)
(related send-result-struct send-result-msg has-first)
(related send-result-struct send-result-msg has-last)

(instance send-pack-struct sequence)
(related send-pack-to-reviewer-prtcl send-pack-struct has-structure)
(related send-pack-to-reviewer-prtcl send-pack-msg has-msg)
(related send-pack-struct send-pack-msg has-element)
(related send-pack-msg chair-path has-msg-sender)
(related send-pack-struct send-pack-msg has-first)
(related send-pack-struct send-pack-msg has-last)

(instance send-final-result-struct sequence)
(related send-final-result-and-review-prtcl send-final-result-struct has-structure)
(related send-final-result-and-review-prtcl send-final-result-msg has-msg)
(related send-final-result-struct send-final-result-msg has-element)
(related send-final-result-msg chair-path has-msg-sender)
(related send-final-result-struct send-final-result-msg has-first)
(related send-final-result-struct send-final-result-msg has-last)

(instance send-camera-ready-struct sequence)
(related send-camera-ready-prtcl send-camera-ready-struct has-structure)
(related send-camera-ready-prtcl inform-deadline-msg has-msg)
(related send-camera-ready-prtcl send-camera-ready-msg has-msg)
(related send-camera-ready-struct inform-deadline-msg has-element)
(related send-camera-ready-struct send-camera-ready-msg has-element)
(related inform-deadline-msg chair-path has-msg-sender)
(related inform-deadline-msg author-path has-msg-receiver)
(related send-msg author-path has-msg-sender)
(related send-msg chair-path has-msg-receiver)
(related send-camera-ready-struct inform-deadline-msg has-first)
(related send-camera-ready-struct send-msg has-last)
(related inform-deadline-msg send-msg is-before)

(instance engage-reviewer-struct sequence)
(related engage-more-reviewer-ptrcl engage-reviewer-struct has-structure)
(related engage-more-reviewer-ptrcl send-request-msg has-msg)
(related engage-reviewer-struct send-request-msg has-element)
(related engage-reviewer-struct send-relation-msg has-element)
(related send-request-msg manager-path has-msg-sender)
(related send-relation-msg manager-path has-msg-sender)
(related engage-reviewer-struct send-relation-msg has-last)
(related send-request-msg send-relation-msg is-before)

(instance solve-conflict-struct sequence)
(related solve-conflict-with-pc-ptrcl solve-conflict-struct has-structure)
(related solve-conflict-with-pc-ptrcl send-vote-msg has-msg)
(related solve-conflict-struct send-conflict-msg has-element)
(related solve-conflict-struct send-vote-msg has-element)
(related send-conflict-msg pc-member-path has-msg-receiver)
(related send-vote-msg pc-member-path has-msg-receiver)
(related solve-conflict-struct send-conflict-msg has-first)
(related solve-conflict-struct send-vote-msg has-last)

; creating a class-model
(instance ec-class-model class-model)

; populating the ec-class-model

; INTRODUCING INTRA-MODEL ERRORS
// ERROR 1 : AN AGENT-ROLE-CLASS MODELED IN A CLASS DIAGRAM
(related ec-class-model chair has-class)
(related ec-class-model expert-committee has-class)
(related ec-class-model organizing-institution has-class)
(related ec-class-model paper has-class)
(related ec-class-model presentation has-class)
(related ec-class-model camera-ready has-class)
(related ec-class-model conflict has-class)
(related ec-class-model result has-class)
(related ec-class-model proposition has-class)
(related ec-class-model vote has-class)
(related ec-class-model assoc-1-has has-relationship)
(related ec-class-model assoc-2-has has-relationship)
(related ec-class-model assoc-3-has has-relationship)
(related ec-class-model assoc-4-has has-relationship)
(related ec-class-model assoc-conflict has-relationship)

; related assoc-1-has dep-1 has-relationship
(related dep-1 result has-end1)
(related dep-1 conflict has-end2)
(related assoc-1-has paper has-end1)
(related assoc-1-has camera-ready has-end2)

(related assoc-2-has paper has-end1)
(related assoc-2-has proposition has-end2)
(related assoc-3-has paper has-end1)
(related assoc-3-has presentation has-end2)
(related assoc-4-has paper has-end1)
(related assoc-4-has result has-end2)
(related assoc-conflict vote has-end1)
(related assoc-conflict conflict has-end2)

(related dep-1 result has-end1)
(related dep-1 conflict has-end2)
(related inhabit-1 paper has-end1)
(related inhabit-1 expert-committee has-end2)
(related inhabit-2 camera-ready has-end1)
(related inhabit-2 expert-committee has-end2)
creating an organization-model

; populating the the-organizing-model
(related the-organizing-model reviewer has-class)
(related the-organizing-model chair has-class)
(related the-organizing-model manager has-class)
(related the-organizing-model pc-member has-class)
(related the-organizing-model author has-class)
(related the-organizing-model organizing-institution has-class)
(related the-organizing-model researcher-agent has-class)

; creating a role-model

; populating the the-role-model

; ERROR 2: AN AGENT-CLASS MODELED IN A ROLE-MODEL
(related the-role-model researcher-agent has-class)
(instance assoc-5 association)
(instance assoc-6 association)
(instance ctrl-1 control)
(instance ctrl-2 control)

(related the-role-model assoc-5 has-relationship)
(related the-role-model assoc-6 has-relationship)
(related the-role-model ctrl-1 has-relationship)
(related the-role-model ctrl-2 has-relationship)

(related assoc-5 chair has-end1)
(related assoc-5 manager has-end2)

(related assoc-6 chair has-end1)
(related assoc-6 author has-end2)

(related ctrl-1 chair has-end1)
(related ctrl-1 pc-member has-end2)

(related ctrl-2 chair has-end1)
(related ctrl-2 reviewer has-end2)

; creating sequence-models
(instance seq-1 sequence-model)
(related seq-1 submit-struct has-structure)
(related seq-1 author-path has-path)
(related seq-1 chair-path has-path)

(instance seq-2 sequence-model)
(related seq-2 send-proposal-struct has-structure)
(related seq-2 chair-path has-path)
(related seq-2 reviewer-path has-path)

(instance seq-3 sequence-model)
(related seq-3 inform-acceptance-struct has-structure)
(related seq-3 reviewer-path has-path)
(related seq-3 chair-path has-path)

(instance seq-4 sequence-model)
(related seq-4 send-result-struct has-structure)
(related seq-4 reviewer-path has-path)
(related seq-4 chair-path has-path)

(instance seq-5 sequence-model)
(related seq-5 send-pack-struct has-structure)
(related seq-5 chair-path has-path)
(related seq-5 reviewer-path has-path)

(instance seq-6 sequence-model)
(related seq-6 send-final-result-struct has-structure)
(related seq-6 chair-path has-path)
(related seq-6 author-path has-path)

(instance seq-7 sequence-model)
(related seq-7 send-camera-ready-struct has-structure)
(related seq-7 chair-path has-path)
(related seq-7 author-path has-path)

(instance seq-8 sequence-model)
(related seq-8 engage-reviewer-struct has-structure)
(related seq-8 chair-path has-path)
(related seq-8 manager-path has-path)

(instance seq-9 sequence-model)
(related seq-9 solve-conflict-struct has-structure)
(related seq-9 chair-path has-path)
(related seq-9 pc-member-path has-path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(check-abox-coherence)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;
Respostas às consultas QV1

Respostas obtidas a partir das consultas QV1 aplicadas à base de conhecimento descrita na seção anterior.
Código da base de conhecimento usado na fase F2

Nesta seção encontra-se o código da base de conhecimento usada no estudo de caso sobre o Expert Committee, usando a aplicação do método para a linguagem MAS-ML. Esta base refere-se à execução da fase F2 do Observed-MAS e possui erros de modelagem, cuja introdução foi deliberada.

```
in-abox phase2-ABox MASML-2-TBox
;
creating the environment of the EC system
(instance expert-committee passive-environment-class)
(instance expert-committee-path passive-environment-class-path)
(related expert-committee-path expert-committee has-head)
;
creating the main-organization of the EC System
(instance organizing-institution main-organization-class)
(instance organizing-institution-path main-organization-class-path)
(related organizing-institution-path organizing-institution has-head)
(related organizing-institution-path expert-committee-path has-tail)

; ; ; ; ; ; ; ; internal state of the main organization
;
creating the main-organization goals
(instance start-event goal)
(related organizing-institution start-event has-goal)
;
creating the main-organization axioms
(instance dont-play-CA axiom)
(instance dont-play-CHCM axiom)
(instance dont-play-CR axiom)
(instance dont-play-CCH axiom)
(instance reviewer-not-author axiom)
(instance pcmember-not-author axiom)
(instance dont-play-CMC axiom)
(related organizing-institution dont-play-CA has-axiom)
(related organizing-institution dont-play-CHCM has-axiom)
(related organizing-institution dont-play-CR has-axiom)
(related organizing-institution dont-play-CCH has-axiom)
(related organizing-institution reviewer-not-author has-axiom)
(related organizing-institution pcmember-not-author has-axiom)
(related organizing-institution dont-play-CMC has-axiom)
;
creating the main-organization actions
(instance to-create-roles action)
(instance to-create-reviewer action)
(instance to-create-PCmember action)
(instance to-create-PCchair action)
(instance to-create-PCmanager action)
(instance to-create-agenda action)
(related organizing-institution to-create-roles has-action)
(related organizing-institution to-create-reviewer has-action)
(related organizing-institution to-create-PCmember has-action)
(related organizing-institution to-create-PCchair has-action)
(related organizing-institution to-create-PCmanager has-action)
(related organizing-institution to-create-agenda has-action)
;
creating the main-organization plans
(instance start-event-plan action)
(related organizing-institution start-event-plan has-plan)
(related organizing-institution start-event-plan has-plan)
(related start-event-plan struct sequence)
(related start-event-plan start-event-plan has-structure)
(related start-event-plan to-create-PCmanager has-action)
(related start-event-plan to-create-PCmember has-action)
(related start-event-plan to-create-agenda has-action)
(related start-event-plan to-create-roles has-action)
(related start-event-plan to-create-PCchair has-action)
(related start-event-plan-struct to-create-PCmanager has-first)
(related start-event-plan-struct to-create-PCchair has-last)
```
(related start-event-plan-struct to-create-PCmanager has-element)
(related start-event-plan-struct to-create-PCmember has-element)
(related start-event-plan-struct to-create-reviewer has-element)
(related start-event-plan-struct to-create-roles has-element)
(related start-event-plan-struct to-create-PCchair has-element)

(related to-create-PCmanager to-create-PCmember is-before)
(related to-create-PCmember to-create-reviewer is-before)
(related to-create-reviewer to-create-roles is-before)
(related to-create-roles to-create-PCchair is-before)

;;;;;;;;;;;;;;;;;;;;;;;;;end internal state main-org

; creating roles that are owned by the main-organization
(instance reviewer agent-role-class)
(instance reviewer-path agent-role-class-path)
(related reviewer-path reviewer has-head)
(related reviewer-path organizing-institution-path has-tail)

(instance chair agent-role-class)
(instance chair-path agent-role-class-path)
(related chair-path chair has-head)
(related chair-path organizing-institution-path chair-path has-tail)

(instance manager agent-role-class)
(instance manager-path agent-role-class-path)
(related manager-path manager has-head)
(related manager-path organizing-institution-path manager-path has-tail)

(instance pc-member agent-role-class)
(instance pc-member-path agent-role-class-path)
(related pc-member-path pc-member has-head)
(related pc-member-path organizing-institution-path pc-member-path has-tail)

(instance author agent-role-class)
(instance author-path agent-role-class-path)
(related author-path author has-head)
(related author-path organizing-institution-path author-path has-tail)

; creating the main-organization's ownership relationships
(instance own-1 ownership)
(instance own-2 ownership)
(instance own-3 ownership)
(instance own-4 ownership)
(instance own-5 ownership)

; resources of the system
(instance paper object-class)
(instance presentation object-class)
(instance proposition object-class)
(instance result object-class)
(instance vote object-class)
(instance conflict object-class)
(instance camera-ready object-class)

; creating the relationships between objects
(instance assoc-1-has association)
(instance assoc-2-has association)
(instance assoc-3-has association)
(instance assoc-4-has association)
(instance assoc-conflict association)

(instance dep-1 dependency)

(instance inhabit-1 inhabit)
(instance inhabit-2 inhabit)
(instance inhabit-3 inhabit)
(instance inhabit-4 inhabit)
(instance inhabit-5 inhabit)
(instance inhabit-6 inhabit)
(instance inhabit-7 inhabit)
(instance inhabit-8 inhabit)

; agents of the system
(instance researcher-agent agent-class)
(instance researcher-agent-path agent-class-path)
(related researcher-agent-path researcher-agent has-head)
(related researcher-agent-path reviewer-path has-tail)
(related researcher-agent-path chair-path has-tail)
(related researcher-agent-path manager-path has-tail)
(related researcher-agent-path pc-member-path has-tail)
(related researcher-agent-path author-path has-tail)

; creating the researcher goals
(instance receive-camera-ready goal)
(instance submit-paper goal)
(instance distribute-paper goal)
(instance receive-final-results goal)
(instance review-paper goal)
(instance solve-conflict goal)
(instance accept-paper goal)
(instance present-final-results goal)
(instance vote-to-solve goal)
(instance make-review goal)
(instance send-camera-ready goal)
(instance engage-reviewer goal)
(instance provide-reviewer goal)

(related researcher-agent receive-camera-ready has-goal)
(related researcher-agent submit-paper has-goal)
(related researcher-agent distribute-paper has-goal)
(related researcher-agent receive-final-results has-goal)
(related researcher-agent review-paper has-goal)
(related researcher-agent receive-paper has-goal)
(related researcher-agent solve-conflict has-goal)
(related researcher-agent accept-paper has-goal)
(related researcher-agent present-final-results has-goal)
(related researcher-agent vote-to-solve has-goal)
(related researcher-agent make-review has-goal)
(related researcher-agent send-camera-ready has-goal)
(related researcher-agent engage-reviewer has-goal)
(related researcher-agent provide-reviewer has-goal)

; creating the researcher actions
(instance to-make-additional-relation action)
(instance to-make-presentation action)
(instance to-select-new-reviewer action)
(instance to-register-vote action)
(instance to-search-for-conflict action)
(instance to-vote action)
(instance to-register-result action)
(instance to-make-final-result action)
(instance to-review action)
(instance to-prepare-paper-pack action)
(instance to-check-less-than-3-reviewer action)
(instance to-make-camera-ready action)
(instance to-make-paper action)
(instance to-register-paper-and-presentation action)
(instance to-select-reviewer action)
(instance to-make-review-proposition action)
(instance to-register-camera-ready action)
(instance to-assess-review-proposition action)

(related researcher-agent to-make-additional-relation has-action)
(related researcher-agent to-select-new-reviewer has-action)
(related researcher-agent to-register-vote has-action)
(related researcher-agent to-search-for-conflict has-action)
(related researcher-agent to-vote has-action)
(related researcher-agent to-register-result has-action)
(related researcher-agent to-make-final-result has-action)
(related researcher-agent to-review has-action)
(related researcher-agent to-prepare-paper-pack has-action)
(related researcher-agent to-check-less-than-3-reviewer has-action)
(related researcher-agent to-make-camera-ready has-action)
(related researcher-agent to-register-paper-and-presentation has-action)
(related researcher-agent to-select-reviewer has-action)
(related researcher-agent to-make-review-proposition has-action)
(related researcher-agent to-make-paper has-action)
(related researcher-agent to-refresh-reviewer has-action)
(related researcher-agent to-register-camera-ready has-action)
(related researcher-agent to-assess-review-proposition has-action)

; creating the researcher plans
(instance receive-camera-ready-plan plan)
(instance send-camera-ready-plan plan)
(instance submit-paper-plan plan)
(instance receive-paper-plan plan)
(instance assign-paper-plan plan)
(instance accept-paper-plan plan)
(instance provide-reviewer-plan plan)
(instance engage-reviewer-plan plan)
(instance make-review-plan plan)
(instance review-plan plan)
(instance solve-conflict-plan plan)
(instance vote-conflict-plan plan)
(instance show-final-result-plan plan)

; assigning the researcher's plans to it
(related researcher-agent receive-camera-ready-plan has-plan)
(related researcher-agent send-camera-ready-plan has-plan)
(related researcher-agent submit-paper-plan has-plan)
(related researcher-agent assign-paper-plan has-plan)
(related researcher-agent receive-paper-plan has-plan)
(related researcher-agent accept-paper-plan has-plan)
(related researcher-agent provide-reviewer-plan has-plan)
(related researcher-agent engage-reviewer-plan has-plan)
(related researcher-agent make-review-plan has-plan)
(related researcher-agent review-plan has-plan)
(related researcher-agent solve-conflict-plan has-plan)
(related researcher-agent vote-conflict-plan has-plan)
(related researcher-agent show-final-result-plan has-plan)

;; turning the researcher into a citizen
(instance inhabit-9 inhabit)
(related researcher inhabit-9 is-end1)
(related expert-committee inhabit-9 is-end2)

;; structuring the researcher plans
(instance accept-paper-plan-struct sequence)
(related accept-paper-plan accept-paper-plan-struct has-structure)

(instance submit-paper-plan-struct sequence)
(related submit-paper-plan submit-paper-plan-struct has-structure)

(instance review-plan-struct sequence)
(related review-plan review-plan-struct has-structure)

(instance make-review-plan-struct sequence)
(related make-review-plan make-review-plan-struct has-structure)

(instance receive-paper-plan-struct sequence)
(related receive-paper-plan receive-paper-plan-struct has-structure)

(instance engage-reviewer-plan-struct sequence)
(related engage-reviewer-plan engage-reviewer-plan-struct has-structure)

(instance provide-reviewer-plan-struct sequence)
(related provide-reviewer-plan provide-reviewer-plan-struct has-structure)

(instance receive-camera-ready-plan-struct sequence)
(related receive-camera-ready-plan receive-camera-ready-plan-struct has-structure)

(instance send-camera-ready-plan-struct sequence)
(related send-camera-ready-plan send-camera-ready-plan-struct has-structure)

(instance show-final-result-plan-struct sequence)
(related show-final-result-plan show-final-result-plan-struct has-structure)

(instance solve-conflict-plan-struct sequence)
(related solve-conflict-plan solve-conflict-plan-struct has-structure)

(instance vote-conflict-plan-struct sequence)
(related vote-conflict-plan vote-conflict-plan-struct has-structure)

;; creating the researcher beliefs (no beliefs assigned)

;; creating the play relationships the researcher plays
(instance play-1 play)
(instance play-2 play)
(instance play-3 play)
(instance play-4 play)
(instance play-5 play)

;; creating the internal structure of the agent-roles

;; agent-role beliefs and goals (agent-role goals are the same of the researcher agent goals)

(instance review-proposal-belief belief)
(instance package-list-belief belief)
(instance paper-belief belief)
(instance result-belief belief)
(instance review-belief belief)
(instance assessment-belief belief)

(related reviewer review-proposal-belief has-belief)
(related reviewer package-list-belief has-belief)
(related reviewer paper-list-belief has-belief)
(related reviewer result-belief has-belief)
(related reviewer review-belief has-belief)
(related reviewer assessment-belief has-belief)

(related reviewer accept-paper has-goal)
(related reviewer review-paper has-goal)

(instance presentation-belief belief)
(instance final-result-belief belief)
(instance final-result-list-belief belief)
\begin{verbatim}
(related engage-reviewer-struct send-relation-msg has-last)
(related send-request-msg send-relation-msg is-before)

(instance solve-conflict-struct sequence)
(related solve-conflict-with-pc-prtcl solve-conflict-struct has-structure)
(related solve-conflict-with-pc-prtcl send-conflict-msg has-msg)
(related solve-conflict-with-pc-prtcl send-vote-msg has-msg)
(related solve-conflict-struct send-vote-msg has-element)
(related send-conflict-msg chair-path has-msg-sender)
(related send-conflict-msg pc-member-path has-msg-receiver)
(related solve-conflict-struct send-vote-msg has-first)
(related solve-conflict-struct send-vote-msg has-last)

; creating a class-model
(instance ec-class-model class-model)

; populating the ec-class-model
(related ec-class-model expert-committee has-class)
(related ec-class-model organizing-institution has-class)
(related ec-class-model paper has-class)
(related ec-class-model presentation has-class)
(related ec-class-model conflict has-class)
(related ec-class-model result has-class)
(related ec-class-model proposition has-class)
(related ec-class-model vote has-class)
(related ec-class-model assoc-1-has has-relationship)
(related ec-class-model assoc-2-has has-relationship)
(related ec-class-model assoc-3-has has-relationship)
(related ec-class-model assoc-4-has has-relationship)
(related ec-class-model assoc-conflict has-relationship)
(related ec-class-model dep-1 has-relationship)
(related ec-class-model inhabit-1 has-relationship)
(related ec-class-model inhabit-2 has-relationship)
(related ec-class-model inhabit-3 has-relationship)
(related ec-class-model inhabit-4 has-relationship)
(related ec-class-model inhabit-5 has-relationship)
(related ec-class-model inhabit-6 has-relationship)
(related ec-class-model inhabit-7 has-relationship)
(related ec-class-model inhabit-8 has-relationship)
(related assoc-1-has paper has-end1)
(related assoc-1-has camera-ready has-end2)
(related assoc-2-has paper has-end1)
(related assoc-2-has proposition has-end2)
(related assoc-3-has paper has-end1)
(related assoc-3-has presentation has-end2)
(related assoc-4-has paper has-end1)
(related assoc-4-has result has-end2)
(related assoc-conflict vote has-end1)
(related assoc-conflict conflict has-end2)
(related dep-1 result has-end1)
(related dep-1 conflict has-end2)
(related inhabit-1 paper has-end1)
(related inhabit-1 expert-committee has-end2)
(related inhabit-2 camera-ready has-end1)
(related inhabit-2 expert-committee has-end2)
(related inhabit-3 presentation has-end1)
(related inhabit-3 expert-committee has-end2)
(related inhabit-4 result has-end1)
(related inhabit-4 expert-committee has-end2)
(related inhabit-5 proposition has-end1)
(related inhabit-5 expert-committee has-end2)
(related inhabit-6 vote has-end1)
(related inhabit-6 expert-committee has-end2)
(related inhabit-7 conflict has-end1)
(related inhabit-7 expert-committee has-end2)
(related inhabit-8 organizing-institution has-end1)
(related inhabit-8 expert-committee has-end2)
\end{verbatim}
; creating an organization-model
(instance the-organizing-model organization-model)

; populating the the-organizing-model
(related the-organizing-model reviewer has-class)
(related the-organizing-model chair has-class)
(related the-organizing-model manager has-class)
(related the-organizing-model pc-member has-class)
(related the-organizing-model author has-class)
(related the-organizing-model organizing-institution has-class)

; creating a role-model
(instance the-role-model role-model)

; populating the the-role-model
(related the-role-model reviewer has-class)
(related the-role-model chair has-class)
(related the-role-model manager has-class)
(related the-role-model pc-member has-class)
(related the-role-model author has-class)

; creating an object to the role-model to
; create an interdependency between the role
; and the class-diagram
(related the-role-model paper has-class)

; introducing an object to the role-model to
; create an interdependency between the role
; and the class-diagram
(related ctrl-1 pc-member has-end2)
(related ctrl-2 chair has-end1)
(related ctrl-2 reviewer has-end2)

; creating sequence-models
(instance seq-1 sequence-model)
(related seq-1 submit-struct has-structure)
(related seq-1 author-path has-path)
(related seq-1 chair-path has-path)

(instance seq-2 sequence-model)
(related seq-2 send-proposal-struct has-structure)
(related seq-2 chair-path has-path)
(related seq-2 reviewer-path has-path)

(instance seq-3 sequence-model)
(related seq-3 inform-acceptance-struct has-structure)
(related seq-3 reviewer-path has-path)
(related seq-3 chair-path has-path)

(instance seq-4 sequence-model)
(related seq-4 send-result-struct has-structure)
(related seq-4 reviewer-path has-path)
(related seq-4 chair-path has-path)

(instance seq-5 sequence-model)
(related seq-5 send-pack-struct has-structure)
(related seq-5 chair-path has-path)
(related seq-5 reviewer-path has-path)

(instance seq-6 sequence-model)
(related seq-6 send-final-result-struct has-structure)
(related seq-6 chair-path has-path)
(related seq-6 author-path has-path)

(instance seq-7 sequence-model)
(related seq-7 send-camera-ready-struct has-structure)
(related seq-7 chair-path has-path)
(related seq-7 author-path has-path)

(instance seq-8 sequence-model)
(related seq-8 engage-reviewer-struct has-structure)
(related seq-8 chair-path has-path)
(related seq-8 manager-path has-path)

(instance seq-9 sequence-model)
(related seq-9 solve-conflict-struct has-structure)
(related seq-9 chair-path has-path)
(related seq-9 pc-member-path has-path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; INTRODUCING INTER-MODEL ERRORS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;; Q1 agent-class in a class-model and not in an org-model
(instance user-agent agent-class)
(related ec-class-model user-agent has-class)

;;;;; Q2 org-class in a class-model and not in an org-model
(instance reviewers-board sub-organization-class)
(related ec-class-model reviewers-board has-class)

;;;;; Q3 role-class in a role-model and not in an org-model
(instance co-chair agent-role-class)
(related the-role-model co-chair has-class)

;;;;; Q4 object in an org-model and not in a class-model
;;;;; Q5 object in a role-model and ot in a class-model
(related the-organizing-model eval-form has-class)
(related the-role-model eval-form has-class)

;;;;; Q6 classless (ipath without an original class)
(instance paper01-path object-instance-path)
(instance paper01 object)
(related paper01-path paper-01 has-head)
(related seq-1 paper01-path has-path)

;;;;; Q7 classless special case
(instance LES-PUC-Rio main-organization)
(related LES-PUC-Rio organizing-institution is-instanceOf)
(related LES-PUC-Rio-path main-organization-instance-path)
(related LES-PUC-Rio LES-PUC-Rio-path has-path)
(related LES-PUC-Rio-path LES-PUC-Rio has-head)

(instance lucena agent)
(related lucena researcher-agent is-instanceOf)
Resposta às consultas QV2

Nesta seção encontram-se as respostas às consultas QV2 aplicadas à base de conhecimento descrita na seção anterior.
Apêndice H
Dados sobre a aplicação do método para AUML

A ontologia Ont1 para AUML

{in-knowledge-base AUML-TBox phase1-ABox}
{in-tbox AUML-TBox}
{signature
  :atomic-concepts
  entity
class
  agent-class
  organization-class
  object-class
  environment-class
  agent-role-class
  class-instance
  agent
  object
  features
  goal
  belief
  action
  protocol
  attribute
  method
  condition
  relationship
  specialization
  association
  aggregation
  dependency
  msg
  agent-msg
  object-msg

  :roles
  model
  static-model
  class-model
  dynamic-model
  sequence-model
  activity-model
  collaboration-model
  path
  agent-path
  object-path
  sequence
  seq-element
}
(is-end1 :domain class
  :range relationship
  :inverse has-end1)
(is-end2 :domain class
  :range relationship
  :inverse has-end2)
(has-msg :range msg
  :parent has-element)
(has-msg-end :domain msg
  :inverse is-msg-end)
(is-msg-end :domain msg
  :inverse has-msg-end)
(has-msg-sender :parent has-msg-end)
(has-msg-receiver :parent has-msg-end)
(has-msg-sender :inverse has-msg-sender)
(has-msg-receiver :inverse has-msg-receiver)
(is-instanceOf :domain class-instance
  :range class)
(has-attribute :range attribute)
(has-method :range method)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;; AUML properties
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(plays-role ;:domain agent-class
  :range agent-role-class)
(has-class ;:domain static-model
  :range (or class class-instance path) ;class
  :inverse is-in-static-model)
(is-in-static-model :inverse has-class)
(has-relationship ;:domain static-model
  :range relationship
  :inverse is-relationship-of)
(is-relationship-of ;:inverse has-relationship)
(has-path ;range path)
(has-head ;domain path
  :range (or class-instance class))
(has-tail ;domain path
  :range path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;; ordering properties
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(has-structure ;range sequence)
(has-element ;range seq-element)
(has-first ;domain sequence
  :range seq-element
  :parent has-element
  :inverse is-first)
(is-first ;inverse has-first)
(has-last ;domain sequence
  :range seq-element
  :parent has-element
  :inverse is-last)
(is-last ;inverse has-last)
(is-before ;domain seq-element
  :range seq-element
  :transitive t
  :inverse is-after)
(is-after ;domain seq-element
  :range seq-element
  :transitive t
  :inverse is-before)

;;;; concepts restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;; TAO concepts taxonomy
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(implies class entity)
(implies class-instance entity)
(implies path entity)
(implies environment-class class)
(implies agent-role-class class)
(implies agent-class class)
(implies organization-class class)
(implies object-class class)
(implies specialization relationship)
(implies association relationship)
(implies aggregation relationship)
(implies dependency relationship)
(implies goal features)
(implies belief features)
(implies action features)
(implies plan features)

(implies protocol features)
(implies attribute features)
(implies method features)
(implies condition features)
(implies agent class-instance)
(implies object class-instance)
(implies agent-msg msg)
(implies object-msg msg)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; AUML concepts taxonomy (it reuses some TAO concepts)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies static-model model)
(implies dynamic-model model)
(implies sequence-model dynamic-model)
(implies activity-model dynamic-model)
(implies collaboration-model dynamic-model)
(implies class-model static-model)
(equivalent path (or agent-path object-path))

;;; axiom that guarantee the possibility of the dynamic structure definition
(implies msg seq-element)
(implies action seq-element)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; ontology axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies TAO disjoint restrictions)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint class class-instance features msg relationship)
(disjoint agent-class organization-class object-class environment-class agent-role-class)

(disjoint agent object)
(disjoint agent-msg object-msg)
(disjoint goal belief protocol attribute)
(disjoint specialization association aggregation dependency)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint static-model dynamic-model)
(disjoint sequence-model dynamic-model activity-model collaboration-model)
(disjoint object-path agent-path)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(instances-related axioms)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies agent (all is-instanceOf agent-class))
(implies object (all is-instanceOf object-class))
(implies agent-path
(or agent-role-class (and agent (some is-instanceOf
(and agent-class (some plays-role agent-role-class)))))
(and agent-class (some plays-role agent-role-class)))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(order axioms)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies (and (some is-before "top") (some is-after "bottom")
(some is-first "top")

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies (some has-first msg) (some has-msg msg))
(implies (some has-last msg) (some has-msg msg))
(implies (some has-first action) (some has-action action))
(implies (some has-last action) (some has-action action))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(classify-tbox)
(tbox-classified-p)
(tbox-coherent-p)
(tbox-cyclic-p)

Consultas QV1 para a o método aplicado a AUML
Usa a especificação de Bauer

; Q1 busca relacionamentos specialization mal estruturados
(retrieve (?end1 ?end2 ?spec)
 (and (?spec specialization)
   (?end1 ?spec is-end1)
   (?end2 ?spec is-end2)
   (or (and (?end1 object-class) (not (?end2 object-class)))
    (and (?end1 agent-class) (not (?end2 agent-class)))
    (and (not (?end1 object-class)) (not (?end1 agent-class)))))
)

; Q2 busca relacionamentos aggregation mal estruturados
(retrieve (?end1 ?end2 ?agg)
 (and (?agg aggregation)
   (?end1 ?agg is-end1)
   (?end2 ?agg is-end2)
   (or (and (?end1 object-class) (not (?end2 (or agent-class object-class))))
    (and (?end1 agent-class) (not (?end2 agent-class)))
    (and (?end1 agent-role-class) (not (?end2 organization-class)))
    (and (not (?end1 agent-class)) (not (?end1 object-class))
     (not (?end1 agent-role-class)))))
)

; Q3 busca relacionamentos dependency mal estruturados
(retrieve (?end1 ?end2 ?dep)
 (and (?dep dependency)
   (?end1 ?dep is-end1)
   (?end2 ?dep is-end2)
   (or (and (?end1 object-class) (not (?end2 object-class)))
    (not (?end1 object-class))))
)

; Em AUML organizações incluem agentes desempenhando papéis. Estes agentes interagem
usando o ambiente. Assim, tanto papéis como agentes e ambientes são agregados de
organizações. "Representing Social Structures in UML"
As questões 4, 5 e 6 referem-se a esta propriedade

; Q4
(retrieve (?org-without-roles)
 (and (?org-without-roles organization-class)
  (not (?org-without-roles
    (some is-end2 (and aggregation (some has-end1 agent-role-class)))))))
)

; Q5 não deveria ser guideline?????
(retrieve (?org-without-agents)
 (and (?org-without-agents organization-class)
  (not (?org-without-agents
    (some is-end2 (and aggregation (some has-end1 agent-class)))))))
)

; Q6 não deveria ser guideline?????
(retrieve (?org-without-env)
 (and (?org-without-env organization-class)
  (not (?org-without-env
    (some is-end2 (and aggregation (some has-end1 environment-class)))))))
)

; Q7 busca mensagens do tipo object-msg mal estruturadas
(retrieve (?bad-obj-msg)
 (and (?bad-obj-msg object-msg)
  ($?sender ?bad-obj-msg is-msg-sender)
  ($?receiver ?bad-obj-msg is-msg-receiver)
  (or (not ($?sender (or object agent)))
   ($?receiver agent)))
)

; Q8 busca por agent-msgs mal estruturadas
(retrieve (??sender ?receiver ?ag-msg)
 (and (?ag-msg agent-msg)
  ($?sender ?ag-msg is-msg-sender)
  ($?receiver ?ag-msg is-msg-receiver)
  (not (and ($?sender (and agent (some is-instanceOf
    (and agent-class
     (some plays-role agent-role-class))))
    ($?receiver (and agent (some is-instanceOf
     (and agent-class
      (some plays-role agent-role-class))))))))
)
)
Consultas QD1 para o método aplicado a AUML

;; QD1 - boas práticas usando AUML

;; Q1 AUML não especifica se agentes devem sempre
;; possuir metas, assim vale indicar as agent-classes
;; que não tiveram metas associadas
(retrieve (?agentwithoutgoal)
  (and (?agentwithoutgoal agent-class)
     (?agentwithoutgoal NIL has-goal)
  )
)

;; Q2 AUML não especifica se agentes devem desempenhar papéis
;; logo, vale indicar as agent-classes que não estão associadas
;; a nenhum papel de agente.
(retrieve (?agentwithoutrole)
  (and (?agentwithoutrole agent-class)
     (?agentwithoutrole NIL plays-role)
  )
)

;; Q3 Agentes interagem através de protocolos de interação
;; logo, agent-classes que não possuem protocolos associados
;; precisam ser indicadas para o desenvolvedor
(retrieve (?agentwithoutprotocol)
  (and (?agentwithoutprotocol agent-class)
     (?agentwithoutprotocol NIL has-protocol)
  )
)
Base de conhecimento usada na fase F1

(instance in-abox phase1-ABox AUML-TBox)
(instance instance user-agent agent-class)
(instance instance store-agent agent-class)
(instance instance user-agent-path agent-path)
(instance instance store-agent-path agent-path)
(instance related user-agent user-agent-path has-path)
(instance related store-agent store-agent-path has-path)

(instance instance book object-class)
(instance instance imported-book object-class)
(instance instance used-book object-class)

(instance instance general-store organization-class)
(instance instance imported-bookstore organization-class)
(instance instance 2nd-hand-bookstore organization-class)

(instance instance buyer agent-role-class)
(instance instance seller agent-role-class)
(instance instance imported-book-buyer agent-role-class)
(instance instance imported-book-seller agent-role-class)
(instance instance 2nd-hand-book-buyer agent-role-class)
(instance instance 2nd-hand-book-seller agent-role-class)

(instance instance buy-a-book goal)
(instance instance sell-a-book goal)

(instance instance assoc-1 association)
(instance related assoc-1 buyer has-end1)
(instance related assoc-1 seller has-end2)

(instance instance assoc-2 association)
(instance related assoc-2 imported-book-buyer has-end1)
(instance related assoc-2 imported-book-seller has-end2)

(instance instance assoc-3 association)
(instance related assoc-3 2nd-hand-book-buyer has-end1)
(instance related assoc-3 2nd-hand-book-seller has-end2)

(instance instance agreg-1 aggregation)
(instance related agreg-1 buyer has-end1)
(instance related agreg-1 general-store has-end2)

(instance instance agreg-2 aggregation)
(instance related agreg-2 seller has-end1)
(instance related agreg-2 general-store has-end2)

(instance instance agreg-3 aggregation)
(instance related agreg-3 imported-book-buyer has-end1)
(instance related agreg-3 imported-bookstore has-end2)

(instance instance agreg-4 aggregation)
(instance related agreg-4 imported-book-seller has-end1)
(instance related agreg-4 imported-bookstore has-end2)

(instance instance agreg-5 aggregation)
; (related agreg-5 2nd-hand-book-buyer has-end1)
; (related agreg-5 2nd-hand-bookstore has-end2)

(instance instance agreg-6 aggregation)
; (related agreg-6 2nd-hand-book-seller has-end1)
; (related agreg-6 2nd-hand-bookstore has-end2)

(instance instance class-diag-1 class-model)

; (related class-diag-1 store-agent-path has-entity)
; (related class-diag-1 user-agent-path has-entity)
; (related class-diag-1 general-store has-entity)
; (related class-diag-1 imported-bookstore has-entity)
; (related class-diag-1 2nd-hand-bookstore has-entity)
; (related class-diag-1 buyer has-entity)
; (related class-diag-1 seller has-entity)
; (related class-diag-1 imported-book-buyer has-entity)
; (related class-diag-1 imported-book-seller has-entity)
; (related class-diag-1 2nd-hand-buyer has-entity)
; (related class-diag-1 2nd-hand-seller has-entity)

; (related class-diag-1 assoc-1 has-relationship)
; (related class-diag-1 assoc-2 has-relationship)
; (related class-diag-1 assoc-3 has-relationship)
(related class-diag-1 aggr-1 has-relationship)
(related class-diag-1 aggr-2 has-relationship)
(related class-diag-1 aggr-3 has-relationship)
(related class-diag-1 aggr-4 has-relationship)
(related class-diag-1 aggr-5 has-relationship)
(related class-diag-1 aggr-6 has-relationship)

(related user-agent buyer plays-role)
(related user-agent imported-book-buyer plays-role)
(related user-agent 2nd-hand-book-buyer plays-role)

(related store-agent seller plays-role)
(related store-agent imported-book-seller plays-role)
(related store-agent 2nd-hand-book-seller plays-role)

(instance class-diag-2 class-model)

(instance assoc-4 association)
(related assoc-4 user-agent has-end1)
(related assoc-4 store-agent has-end2)

(instance assoc-5 association)
(related assoc-5 user-agent has-end1)
(related assoc-5 book has-end2)

(instance assoc-6 association)
(related assoc-6 book has-end1)
(related assoc-6 store-agent has-end2)

(instance spec-1 specialization)
(related spec-1 imported-book has-end1)
(related spec-1 book has-end2)

(instance spec-2 specialization)
(related spec-2 used-book has-end1)
(related spec-2 book has-end2)

(instance class-diag-2 user-agent has-entity)
(related class-diag-2 store-agent has-entity)
(related class-diag-2 book has-entity)
(related class-diag-2 imported-book has-entity)
(related class-diag-2 2nd-hand-book has-entity)

(instance class-diag-2 assoc-4 has-relationship)
(related class-diag-2 assoc-5 has-relationship)
(related class-diag-2 assoc-6 has-relationship)

(instance seq-diag-1 sequence-model)

(instance negotiate-prtcl protocol)

(instance request agent-msg)

(instance proposal agent-msg)

(instance reject-proposal agent-msg)

(instance accept-proposal agent-msg)

(instance inform-charge agent-msg)

(instance inform-payment agent-msg)

(instance negotiate-prtcl-struct sequence)

(related negotiate-protocol request has-msg)
(related negotiate-protocol proposal has-msg)
(related negotiate-protocol reject-proposal has-msg)
(related negotiate-protocol accept-proposal has-msg)
(related negotiate-protocol inform-charge has-msg)
(related negotiate-protocol inform-payment has-msg)

(related seq-model-1 request has-first)
(related request proposal is-before)
(related proposal reject-proposal is-before)
(related proposal accept-proposal is-before)
(related accept-proposal inform-charge is-before)
(related inform-charge inform-payment is-before)
(related seq-model-2 inform-payment has-last)

;; introduzindo ERROS

;; Q1

(instance spec-3 specialization)
(related spec-3 user-agent has-end1)
(related spec-3 book has-end2)

;; Q2

(instance aggr-7 aggregation)
(related aggr-7 general-store has-end1)
(related aggr-7 2nd-hand-bookstore has-end2)

;; Q3

(instance dep-1 dependency)
(related dep-1 user-agent has-end1)
(related dep-1 book has-end2)
Respostas às consultas QV1

A ontologia Ont2 para AUML

A ontologia Ont2 para AUML

{in-knowledge-base AUML-phase2-TBox phase2-ABox}
{in-tbox AUML-phase2-TBox}
{signature
atomic-concepts

entity
class
citizen-class
agent-class
organization-class
object-class
environment-class
agent-role-class
class-instance
agent
object
features
goal
belief
action
protocol
attribute
method
condition
relationship
specialization
association
aggregation
dependency
msg
agent-msg
object-msg

model
static-model
class-model
dynamic-model
sequence-model
activity-model
collaboration-model
path
agent-path
object-path
sequence
seq-element

roles

:roles ( has-goal :range goal)
(has-belief :range belief)
(has-action :range action :parent has-element)
(has-precond :domain action :range condition)
(has-poscond :domain action :range condition)
(has-plan :range plan)
(has-protocol :range protocol)
(has-end1 :domain relationship :range class :inverse is-end1)
(has-end2 :domain relationship :range class :inverse is-end2)
is-end1 :domain class :range relationship :inverse has-end1
(is-end2 :domain class :range relationship :inverse has-end2)
(has-msg :range msg :parent has-element)
(has-msg-end :domain msg :inverse is-msg-end)
is-msg-end :inverse has-msg-end
(has-msg-sender :parent has-msg-end)
(has-msg-receiver :parent has-msg-end)
is-msg-sender :inverse has-msg-sender
(is-msg-receiver :inverse has-msg-receiver)
is-instanceOf :domain class-instance :range class
(has-attribute :range attribute)
(has-method :range method)

AUML concepts

AUML properties
;; axiom that guarantee the possibility of the dynamic structure definition
(implies msg seq-element)
(implies action seq-element)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; ontology axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; TAO disjoint restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint class class-instance features msg relationship)
(disjoint agent-class organization-class object-class environment-class agent-role-class)
(disjoint agent object)
(disjoint agent-msg object-msg)
(disjoint goal belief protocol attribute)
(disjoint specialization association aggregation dependency)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; AML disjoint restrictions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(disjoint static-model dynamic-model)
(disjoint sequence-model activity-model collaboration-model)
(disjoint object-path agent-path)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(instances-related axioms)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies agent (all is-instanceOf agent-class))
(implies object (all is-instanceOf object-class))
(implies agent-path
(or agent-role-class (and agent (some is-instanceOf agent-class)))
(and agent-class (some plays-role agent-role-class)))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; order axioms
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies (and (some is-before top) (some is-after bottom)) (some is-first top))
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies (and (some is-after top) (some is-before bottom)) (some is-last top))
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies (some has-first msg) (some has-msg msg))
(implies (some has-last msg) (some has-msg msg))
(implies (some has-first action) (some has-action action))
(implies (some has-last action) (some has-action action))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; specialization
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(or (and (some has-end1 agent-class) (all has-end2 agent-class))
(and (some has-end1 object-class) (all has-end2 object-class)))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; association is commutative
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies association
(or
(and (some has-end1 object-class) (some has-end2 (or object-class)))
(and (some has-end1 agent-class) (some has-end2 object-class))
(and (some has-end1 object-class) (some has-end2 role-class))
(and (some has-end1 role-class) (some has-end2 agent-class))
(and (some has-end1 role-class) (some has-end2 object-class))
(and (some has-end1 role-class) (some has-end2 role-class)))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;; aggregation
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(implies aggregation
(or
(and (some has-end1 object-class) (all has-end2 object-class))
(and (some has-end1 agent-role-class) (all has-end2 agent-role-class))
(and (some has-end1 object-role-class) (all has-end2 object-role-class)))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
end of tbox
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(classify-tbox)
(tbox-classified-p)
(tbox-coherent-p)
(tbox-cyclic-p)