


KRISTENSEN, B. Subject Composition by Roles. Em: *Anais Object-Oriented Information Systems*, Australia, 1997.


SILVA, V.; LUCENA, C.; ALENCAR, P.; COWAN, C. A Model-Based Transformational Approach for Implementing Multi-Agent Systems, Technical Report CS2004-12, School of Computer Science, University of Waterloo, Canada, 2004d.


Este apêndice apresenta duas gramáticas parcialmente descritas no Capítulo 5.3. A gramática MAS-ML ilustrada na Figura 119 define a estrutura das entidades, propriedades e relacionamentos descritos na entrada da fase de transformação das entidades e a estrutura dos relacionamentos e classes gerados na saída. A gramática descrita na Figura 120, que é um subconjunto da gramática MAS-ML, define a estrutura dos relacionamentos e classes descritos na entrada da fase de transformação dos relacionamentos e a estrutura do conjunto de classes de saída.
define relationshipDescription
'RELATIONSHIP [relationship][NL] [NL]
end define

define relationship
[ownership]
| [inhabit]
| [play]
| [control]
| [dependency]
| [association]
| [aggregation]
| [specialization]
end define

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% RELATIONSHIPS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%
% ownership
define ownership
[organizationMainID] '1 'OWNS [multiplicity][agentRoleID]
| [organizationID] '1 'OWNS [multiplicity][objectRoleID]
end define

define organizationMainID
[organizationID]
end define

%%%%%%%%%%%%%%%%%%%%
% inhabit
define inhabit
[agentID][multiplicity] 'INHABITS '1 [environmentID]
| [organizationMainID][multiplicity] 'INHABITS '1 [environmentID]
| [entityClassID][multiplicity] 'INHABITS '1 [environmentID]
end define

%%%%%%%%%%%%%%%%
% play
define play
[agentID] '1 'PLAYS [multiplicity][agentRoleID] 'IN
[organizationID]
| [organizationID] '1 'PLAYS [multiplicity][agentRoleID] 'IN
[organizationID]
| [entityClassID] '1 'PLAYS [multiplicity][objectRoleID] 'IN
[organizationID]
end define

%%%%%%%%%%%%%%%%%%%%%%%%
% specialization
define specialization
[agentID] 'SPECIALIZES [agentID]
| [organizationID] 'SPECIALIZES [organizationID]
define role

[environmentID] 'SPECIALIZES [environmentID]
| [agentRoleID] 'SPECIALIZES [agentRoleID]
| [objectRoleID] 'SPECIALIZES [objectRoleID]
| [entityClassID] 'SPECIALIZES [entityClassID]
end define

% control
define control
[agentRoleID][multiplicity] 'CONTROLS
[multiplicity][agentRoleID][labelRel]
end define

% dependency
define dependency
[agentRoleID][multiplicity][role] 'DEPENDS 'ON
[role][multiplicity][agentRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'DEPENDS 'ON
[role][multiplicity][objectRoleID][labelRel]
| [objectRoleID][multiplicity][role] 'DEPENDS 'ON
[role][multiplicity][objectRoleID][labelRel]
| [entityClassID][multiplicity][role] 'DEPENDS 'ON
[role][multiplicity][objectRoleID][labelRel]
end define

% association
define association
[objectRoleID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][objectRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][objectRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][objectRoleID][labelRel]
| [environmentID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][objectRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][entityClassID][labelRel]
| [organizationID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][entityClassID][labelRel]
| [entityClassID][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][entityClassID][labelRel]
end define

% aggregation
define aggregation
[objectRoleID][multiplicity][role] 'AGGREGATES
[role][multiplicity][objectRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'AGGREGATES
[role][multiplicity][objectRoleID][labelRel]
| [agentRoleID][multiplicity][role] 'AGGREGATES
[role][multiplicity][objectRoleID][labelRel]
| [entityClassID][multiplicity][role] 'AGGREGATES
[role][multiplicity][entityClassID][labelRel]
end define

define role
[id]
| [empty]
end define
define multiplicity
  [multi]
  [empty]
end define

define labelRel
  [id]
  [empty]
end define

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENTITIES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% AGENT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

define agent
  'AGENT [NL] '(' [agentID] [IN][NL][goal*][NL][belief*][NL][plan*][NL][actionPrePostCondition *][NL][EX] ')
  [[entityClass*]] %agent se transforma em class
end define

define agentID
  [id]
end define

define goal
  [leafGoal]
  [compositeGoal]
end define

define leafGoal
  'GOAL '( [attrBody] [goalPlans+] ')
  |'objectGoal' = 'new 'LeafGoal '( [stringlit] ', [stringlit] '
  ', [stringlit] ');[NL]
  'this.goals.add(objectGoal);[NL]
  [goalPlans+]
  |'objectSubGoal' = 'new 'LeafGoal '( [stringlit] ',
  [stringlit] ', [stringlit] ');[NL]
  'objectCompositeGoal.setSubGoal(objectSubGoal);[NL]
  [goalPlans+]
end define

define compositeGoal
  'GOAL '( [attrBody] [goalPlans*] 'SUBGOAL '(' [goal+] ')')
  |'objectCompositeGoal' = 'new 'CompositeGoal '( [stringlit] '
  ', [stringlit] ', [stringlit] ');[NL]
  'this.goals.add(objectCompositeGoal);[NL]
  [goalPlans+]
  [goal*]
  'objectCompositeGoal.setSubGoal(objectSubCompositeGoal);[NL]
  [goalPlans+] [goal*]
end define

define goalPlans
'RELATED 'TO 'PLAN [stringlit]
|'objectGoal.setPlan( [stringlit] ');[NL]
|'objectCompositeGoal.setPlan( [stringlit] ');[NL]
|'objectSubGoal.setPlan( [stringlit] ');[NL]
|'objectSubCompositeGoal.setPlan( [stringlit] ');[NL]
end define

define attrBody
[varType] ': [varName] ' = [varValue]
end define

define varType
[stringlit]
end define

define varName
[stringlit]
end define

define varValue
[stringlit]
|'null
end define

define belief
'BELIEF '({ [attrBody] ')
|'this.beliefs.add(new 'Belief '( [stringlit]',
[stringlit] ', [stringlit]',[NL]
|'this.beliefs.add(new 'Belief '( [stringlit]',
[stringlit]', 'null '));[NL]
end define

define plan
'PLAN '(' [planName] [planActions*] [planGoal]')
[entityClass*]
|'objectPlan '= 'new [planName] '();[NL]
|'objectPlan.setAgent(this);[NL]
|'this.plans.add(objectPlan);[NL]
[planActions*]
[planGoal]
|'objectPlan '= 'new [planName] '();[NL]
|'objectPlan.setOrganization(this);[NL]
|'this.plans.add(objectPlan);[NL]
[planActions*]
[planGoal]
end define

define planActions
'COMPOSED 'OF 'ACTION [stringlit]
|'actionAux ' = 'null;[NL]
|'enumActions '= 'this.actions.elements();[NL]
|'while(enumActions.hasMoreElements())[NL]
'|'(NL)[IN]
|'actionAux '= '(Action)enumActions.nextElement();[NL]
|'if '(actionAux.getClass().getName().equals([stringlit])'))[NL]
|'(NL)[IN]
|'objectPlan.setAction(actionAux);[NL][EX]
'|')(NL)[EX]
end define
define planGoal
    'RELATED TO GOAL [varName]
    |'goalAux' = 'null';[NL]
    'enumGoals' = 'this.goals.elements();[NL]
    'while(enumGoals.hasMoreElements())[NL]
    'if 'goalAux.getName().equals([stringlit])
    ')')[NL]
    'if 'goalAux.checkSubGoalsName([stringlit])
    'goalAux.getSubGoals();[NL]
    'Vector 'vGoals = 'vGoals.elements();[NL]
    'while 'enumvGoals.hasMoreElements
    'if 'theGoal.getName().equals([stringlit])
    'objectPlan.setGoal(theGoal);[NL][EX]
end define

define goalName
    [stringlit]
end define

define planName
    [id]
end define

define actionPrePostCondition
    [action][NL][preCondition*][postCondition*]
    [[entityClass*]
end define

define action
    'ACTION [actionAssignment]
    |'objectAction' = 'new [actionAssignment] '(); [NL]
    'this.actions.add(objectAction);[NL]
end define

define actionAssignment
    [id]
end define

define preCondition
define postCondition
'POSTCONDITION '(' [attrBody] ')
|'objectCondition ' = 'new 'Condition '(' [stringlit] ',
[stringlit] ', [stringlit]'); [NL]
'objectAction.setPostCondition '(objectCondition);[NL]
end define

$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%$
% ORGANIZATION

define organizationMain
[organization]
end define

define organization
'ORGANIZATION [NL]
[organizationID][IN][NL][goal*][NL][belief*][NL][plan*][NL][actionPrePostCondition*][NL][axiom*][NL][EX] ')
| [entityClass*]
end define

define organizationID
[id]
end define

define axiom
'AXIOM '(' [attrBody] ')
|'this.axioms.add(new 'Axiom '(' [stringlit]', [stringlit]',
[stringlit]' ));[NL]
|'this.axioms.add(new 'Axiom '(' [stringlit]', [stringlit]',
'null ');[NL]
end define

$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%$
% AGENT ROLE

define agentRole
'AGENTROLE [NL]
'([agentRoleID][IN][NL][roleGoal+][NL][belief*][NL][duty*][NL][right*][NL][protocolMessage*] [NL][EX]')
|[entityClass*] %agentRole se transforma em class
end define

define agentRoleID
[id]
end define

define roleGoal
[roleLeafGoal]
| [roleCompositeGoal]
end define

define roleLeafGoal
'ROLEGOAL '(' [attrBody] ')'
end define

define roleCompositeGoal
'ROLEGOAL '(' [attrBody] 'SUBGOAL '(' [roleGoal+] ')')
'objectCompositeGoal ' = 'new 'CompositeGoal '('
'this.goals.add(objectCompositeGoal);[NL]
[roleGoal*]
'objectSubCompositeGoal ' = 'new 'CompositeGoal '('
'objectCompositeGoal.setSubGoal(objectSubCompositeGoal);[NL]
[roleGoal*]
end define

define duty
'DUTY [stringlit]
'objectCompositeGoal ' = 'new 'CompositeGoal '('
'this.duties.add(new 'Duty '('
[stringlit] '));[NL]
end define

define right
'RIGHT [stringlit]
'this.rights.add(new 'Right '('
[stringlit] '));[NL]
end define

define protocolMessage
[protocol][message+]
[entityClass*]
end define

define protocol
'PROTOCOL [id]
'objectCompositeGoal ' = 'new [id] '();[NL]
'this.protocols.add(objectProtocol);[NL]
end define

define message
'MESSAGE [IN][NL]'( [labelMessage][NL][IN][opt content][NL][sender][NL][receiver][NL][EX] '')[NL][EX]
'objectMessage ' = 'new 'Message '('
[stringlit] ',
'objectProtocol.setMessage(objectMessage);[NL]
end define

define labelMessage
'LABEL [stringlit]
end define

define content
'CONTENT [stringlit]
end define

define sender
'SENDER [stringlit]
end define
define receiver
'RECEIVER [stringlit]
end define

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OBJECT ROLE

define objectRole
'OBJECTROLE [NL]'
[objectRoleID]'([NL][IN][attributeDescription*][NL][methodDescription*][NL][EX]')
end define

define objectRoleID
[id]
end define

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CLASS

define entityClass
[opt importDef][static][visibility][concret] 'class
[entityClassID][inheritance*][NL]'([NL][IN][attributeDescription*]
[methodDescription*][EX]')[NL][NL]
[entityClass*] % uma classe pode virar mais de uma classe
[actionPrePostCondition*] % action se transforma em class
[plan*] % plano se transforma em class
[protocolMessage*] % protocolo se transforma em class
end define

define importDef
'import 'java.util.*;[NL]
end define

define inheritance
'extends [entityClassID]
end define

define entityClassID
[id]
end define

define static
'static
| [empty]
end define

define concret
'abstract
| [empty]
end define

define visibility
'public
| 'private
| 'protected
end define

define attributeDescription
define coments
'/** [NL]
  ** @associates <([id])>[NL]
  ** @link [id][NL]
  [opt comentsLabel]
  [opt comentsClientRole]
  [opt comentsClientCardinality]
  [opt comentsSupplierRole]
  [opt comentsSupplierCardinality]
  */[NL]
end define

define comentsLabel
  '** @label [id][NL]
end define

define comentsClientRole
  '** @clientRole [id][NL]
end define

define comentsClientCardinality
  '** @clientCardinality [multi][NL]
end define

define comentsSupplierRole
  '** @supplierRole [id][NL]
end define

define comentsSupplierCardinality
  '** @supplierCardinality [multi][NL]
end define

define methodDescription
  [static][visibility][concret][attribType][attribName] '=
  [attribValue];[NL]
end define

define coments
'/** [NL]
  * @associates <([id])>[NL]
  * @link [id][NL]
  [opt comentsLabel]
  [opt comentsClientRole]
  [opt comentsClientCardinality]
  [opt comentsSupplierRole]
  [opt comentsSupplierCardinality]
  */[NL]
end define

define comentsLabel
  * @label [id][NL]
end define

define comentsClientRole
  * @clientRole [id][NL]
end define

define comentsClientCardinality
  * @clientCardinality [multi][NL]
end define

define comentsSupplierRole
  * @supplierRole [id][NL]
end define

define comentsSupplierCardinality
  * @supplierCardinality [multi][NL]
end define

define methodDescription
  [static][visibility][concret][outParameter][assignmen]'( [list inParameter] ') [method*][NL]
end define

define method
  [NL]'{ [NL][IN][methodBody*][EX][NL] '}
end define

define methodBody
  [not 'import][not visibility][not 'ENVIRONMENT][not 'ORGANIZATION][not 'AGENTROLE][not 'AGENT]
  [not 'OBJECTROLE][token]
  [goal+][belief*][actionPrePostCondition+][plan+][axiom*]
  [goal+][belief*][actionPrePostCondition+][plan+]
  [roleGoal+][belief*][duty*][right*][protocolMessage*]
end define

define assignment
  [id]
end define
define outParameter
  [id]
  | 'void
  | [empty]
end define

define inParameter
  [attribType][attribName]
  | [empty]
end define

define attribType
  [id]
end define

define attribName
  [id]
end define

define attribValue
  [id]
  | [stringlit]
  | 'new 'Vector()
end define

define environment
  [proactiveEnvironment]
  | [reactiveEnvironment]
end define

define environmentID
  [id]
end define

define proactiveEnvironment
  'ENVIRONMENT [NL]'([NL][IN][agent] [NL][EX])
  | [agent*]  %environment se tranforma em agent para depois se transformar em classe
end define

define reactiveEnvironment
  'ENVIRONMENT [NL]'([NL][IN][entityClass][NL][EX])
  | [entityClass*]  %environment se transforma em classe
end define

Figura 119 – A gramática MAS-ML usada na fase de transformação das entidades.

% Grammar

compounds
  != // /* */ /** <{ }> /*#
  @associates @link @label @clientRole @clientCardinality
  @supplierRole @supplierCardinality
end compounds

tokens
  multi "\d+\.\d+" | "\d+\.*" | "\d+"
% PROGRAM

define program
[UMLprogram*]
end define

define UMLprogram
[relationshipDescription*][entityClass*]
end define

define relationshipDescription
'RELATIONSHIP [relationship] [NL] [NL]
end define

define relationship
[ownership]
| [inhabit]
| [play]
| [control]
| [dependency]
| [association]
| [aggregation]
| [specialization]
end define

% RELATIONSHIPS

% ownership
define ownership
 [id] [multiplicity] 'OWNS [multiplicity] [id]
end define

% inhabit
define inhabit
 [id] [multiplicity] 'INHABITS '1 [id]
end define

% play
define play
 [id] [multiplicity] 'PLAYS [multiplicity] [id] 'IN [id]
end define

% specialization
define specialization
 [id] 'SPECIALIZES [id]
end define

% control
define control
 [id] [multiplicity] 'CONTROLS [multiplicity][id][labelRel]
end define

%%%%%%%%%%%%%%%%%%%%%%
% dependency
define dependency
  [id][multiplicity][role] 'DEPENDS 'ON
[role][multiplicity][id][labelRel]
end define

%%%%%%%%%%%%%%%%%%%%%%
% association
define association
  [id][multiplicity][role] 'ASSOCIATED 'WITH
[role][multiplicity][id][labelRel]
end define

%%%%%%%%%%%%%%%%%%%%%%
% aggregation
define aggregation
  [id][multiplicity][role] 'AGGREGATES
[role][multiplicity][id][labelRel]
end define

define role
  [id]
  | [empty]
end define

define multiplicity
  [multi]
  | [empty]
end define

define labelRel
  [id]
  | [empty]
end define

%%%%%%%%%%%%%%%%%%%%%%
% ENTITIES

%%%%%%%%%%%%%%%%%%%%%%
% CLASS
define entityClass
  [opt importDef][static][visibility][concret] 'class
[entityClassID][inheritance*][NL]'[[NL][IN]
[comentsAttrib*][methodDescription*][EX']] [NL][NL]
end define

define comentsAttrib
  [generalComents*][attributeDescription*]
end define

define importDef
  'import java.util.*;[NL]
end define

define static
  'static
define visibility
    'public
    | 'private
    | 'protected
end define

define concret
    'abstract
    | [empty]
end define

define entityClassID
    [id]
end define

define inheritance
    'extends [entityClassID]
end define

define attributeDescription
    [static][visibility][concret][attribType][attribName] '=
    [attribValue]' İl [NL]
end define

define generalComents
    ' /**
    [NL]
    [coments*]
    ' */[NL]
end define

define coments
    ' @associates <[id]>[NL]
    | '@associates [token][NL]
end define

define comentsLabel
    ' @label [labelRel][NL]
end define

define comentsClientRole
    ' @clientRole [role][NL]
end define

define comentsClientCardinality
    ' @clientCardinality [multiplicity][NL]
end define

define comentsSupplierRole
    ' @supplierRole [role][NL]
end define

define comentsSupplierCardinality
  ' * @supplierCardinality [multiplicity][NL]
end define

define comentsDepAttrib
  '/*# [id] [id] '; '*/[NL]
  |[attributeDescription]
end define

define varType
  [stringlit]
  |[id]
end define

define varName
  [stringlit]
  |[id]
end define

define varValue
  [stringlit]
  |[id]
  |'null
end define

define methodDescription
  [static][visibility][concret][outParameter][assignment]'(
  [list inParameter] ') [method*][NL]
end define

define outParameter
  [id]
  |'void
  |[empty]
end define

define assignment
  [id]
end define

define inParameter
  [attribType][attribName]
  |[empty]
end define

define attribType
  [id]
end define

define attribName
  [id]
end define

define attribValue
  [id]
  |[stringlit]
  |'new 'Vector()
define method
[ NL ] '{
[ NL ][ IN ] [ partialAttribDescription* ][ methodBody* ][ EX ][ NL ] '}
 | ';
end define

define partialAttribDescription
[ attribType ][ attribName ] ' = [ attribValue ] ';[ NL ]
end define

define methodBody
[ token ]';[ NL ]
 | [ NL ] '{ [ NL ][ IN ] [ token ]
 | [ NL ][ EX ] ' } [ NL ]
 | [ not ' import ] [ not visibility ] [ token ]
end define

Figura 120 – O subconjunto da gramática MAS-ML usada na fase de transformação dos relacionamentos.
Apêndice II
A Descrição Textual do Exemplo do Virtual Marketplace

Os diagramas estruturais de MAS-ML que modelam o exemplo *Virtual Marketplace* foram descritos usando a gramática MAS-ML (Apêndice I). A descrição textual do exemplo está ilustrada na Figura 121.

RELATIONSHIP General_Store 1..* INHABITS 1 Virtual_Marketplace
RELATIONSHIP User_Agent 0..* INHABITS 1 Virtual_Marketplace
RELATIONSHIP Store_Agent 0..* INHABITS 1 Virtual_Marketplace
RELATIONSHIP Second_hand_Bookstore 0..* INHABITS 1 Virtual_Marketplace
RELATIONSHIP Imported_Bookstore 0..* INHABITS 1 Virtual_Marketplace

RELATIONSHIP General_Store 1 OWNS 0..* Buyer
RELATIONSHIP General_Store 1 OWNS 0..* Seller
RELATIONSHIP General_Store 1 OWNS 0..* Market_of_Special_Goods
RELATIONSHIP General_Store 1 OWNS 0..* Market_of_Used_Goods
RELATIONSHIP General_Store 1 OWNS 0..* Desire
RELATIONSHIP General_Store 1 OWNS 0..* Offer

RELATIONSHIP User_Agent 1 PLAYS 0..* Buyer IN General_Store
RELATIONSHIP Store_Agent 1 PLAYS 0..* Seller IN General_Store
RELATIONSHIP Book 1 PLAYS 0..* Desire IN General_Store
RELATIONSHIP Book 1 PLAYS 0..* Offer IN General_Store

RELATIONSHIP Second_hand_Bookstore 1 PLAYS 0..* Market_of_Used_Goods IN General_Store
RELATIONSHIP Imported_Bookstore 1 PLAYS 0..* Market_of_Special_Goods IN General_Store

RELATIONSHIP Second_hand_Bookstore 1 OWNS 0..* Buyer_of_Second_hand_Books
RELATIONSHIP Second_hand_Bookstore 1 OWNS 0..* Seller_of_Second_hand_Books
RELATIONSHIP Second_hand_Bookstore 1 OWNS 0..* Desire_of_Second_hand_Books
RELATIONSHIP Second_hand_Bookstore 1 OWNS 0..* Offer_of_Second_hand_Books

RELATIONSHIP User_Agent 1 PLAYS 0..* Buyer_of_Second_hand_Books IN Second_hand_Bookstore
RELATIONSHIP User_Agent 1 PLAYS 0..* Seller_of_Second_hand_Books IN Second_hand_Bookstore
RELATIONSHIP Second_hand_Book 1 PLAYS 0..* Desire_of_Second_hand_Books IN Second_hand_Bookstore
RELATIONSHIP Second_hand_Book 1 PLAYS 0..* Offer_of_Second_hand_Books IN Second_hand_Bookstore

RELATIONSHIP Imported_Bookstore 1 OWNS 0..* Buyer_of_Imported_Books
RELATIONSHIP Virtual_Marketplace 0..* ASSOCIATED WITH 0..*
   Virtual_Marketplace
RELATIONSHIP Virtual_Marketplace 1 habitat ASSOCIATED WITH member
   0..* Item

ENVIRONMENT
{
import java.util.*;
public class Virtual_Marketplace
{
    public Vector getItems ()
    {
        return this.theItem;
    }
    public void setItems (Item newItem)
    {
        this.theItem.add (newItem);
    }
    public void sellItem (Item newItem)
    {
        newItem.setSold();
    }
    public Vector getItems ()
    {
        return this.theItem;
    }
    public Vector getOtherEnvironments ()
    {
        return this.theVirtual_Marketplace;
    }
    public void setOtherEnvironments (Environment newEnvironment)
    {
        this.theVirtual_Marketplace.add (newEnvironment);
    }
    public boolean checkAgentCanLeave (Agent newAgent)
    {
    }
}
}

ORGANIZATION
{
General_Store
    GOAL ( "boolean" : "management_of_sellers" = "true"
        RELATED TO PLAN "Creating_sellers"
    )
    GOAL ( "boolean" : "management_of_orders" = "true"
        RELATED TO PLAN "Update_environment_plan"
    )
    GOAL ( "boolean" : "management_of_profit" = "true"
        RELATED TO PLAN "Evaluating_profit"
    )
    BELIEF ("boolean" : "management" = "false")
    BELIEF ("boolean" : "management_of_sellers" = "false")
    BELIEF ("boolean" : "management_of_orders" = "false")
    BELIEF ("Vector" : "sales" = null)
    BELIEF ("Vector" : "sellers" = null)
    BELIEF ("Vector" : "buyers" = null)
    PLAN ( Creating_sellers
        COMPOSED OF ACTION "Create_seller"
        COMPOSED OF ACTION "Inform_buyer_of_seller"
    )
}
RELATED TO GOAL "management_of_sellers"

PLAN ( Update_environment_plan
    COMPOSED OF ACTION "Update_environment_action"
    RELATED TO GOAL "management_of_orders"
)

PLAN ( Evaluating_profit
    COMPOSED OF ACTION "Evaluate_profit"
    COMPOSED OF ACTION "Register_sale"
    RELATED TO GOAL "management_of_profit"
)

ACTION Create_seller
ACTION Inform_buyer_of_seller
ACTION Update_environment_action
ACTION Evaluate_profit
ACTION Register_order
AXIOM ("boolean" : "Send_information_about_sale" = "true")

AGENTROLE
( Buyer
  ROLEGOAL ("boolean" : "itemBought" = "true"
  )
  DUTY "Search_for_seller"
  RIGHT "Send_answer_proposal_accept"
  RIGHT "Send_answer_proposal_reject"
  PROTOCOL SimpleNegotiation
  MESSAGE
  (LABEL "Request"
   CONTENT "ItemDescription"
   SENDER "Buyer"
   RECEIVER "Seller"
  )
  MESSAGE
  (LABEL "Proposal"
   CONTENT "ItemDescription"
   SENDER "Seller"
   RECEIVER "Buyer"
  )
  MESSAGE
  (LABEL "AnswerProposal"
   CONTENT "RejectItemDescription"
   SENDER "Buyer"
   RECEIVER "Seller"
  )
  MESSAGE
  (LABEL "AnswerProposal"
   CONTENT "AcceptItemDescription"
   SENDER "Buyer"
   RECEIVER "Seller"
  )
  MESSAGE
  (LABEL "Inform"
   CONTENT "Bill"
   SENDER "Seller"
   RECEIVER "Buyer"
  )
  MESSAGE
  (LABEL "Inform"
   CONTENT "Payment"
SENDER "Buyer"
RECEIVER "Seller"

PROTOCOL To_request_to_enter_in_market_of_Special_Goods
MESSAGE
(LABEL "Request"
CONTENT "OrgGoals"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "OrgGoals"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
CONTENT "Roles"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "Roles"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
CONTENT "RoleToPlay"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "PermissionAccept"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Inform"
CONTENT "PermissionReject"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)

PROTOCOL To_enter_in_market_of_Used_Goods
MESSAGE
(LABEL "Request"
CONTENT "OrgGoals"
SENDER "Buyer"
RECEIVER "Market_of_Used_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "OrgGoals"
SENDER "Market_of_Used_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
CONTENT "Roles"
SENDER "Buyer"
RECEIVER "Market_of_Used_Goods"
)

MESSAGE
(LABEL "Inform"
CONTENT "Roles"
SENDER "Market_of_Used_Goods"
RECEIVER "Buyer"
)

MESSAGE
(LABEL "Inform"
CONTENT "RoleToPlay"
SENDER "Buyer"
RECEIVER "Market_of_Used_Goods"
)

AGENT
(User_Agent

GOAL ( "boolean" : "itemBought" = "true"

SUBGOAL
GOAL ( "boolean" : "searchSeller" = "true"
RELATED TO PLAN "Searching_seller"
)

GOAL ( "boolean" : "negotiate" = "true"
RELATED TO PLAN "Buying_item"
RELATED TO PLAN "Complex_Buying_item"
)

GOAL ( "boolean" : "enterOrganization" = "true"
RELATED TO PLAN "Entering_organization"
RELATED TO PLAN "Leaving_environment"
)

)

GOAL ( "boolean" : "itemSold" = "true"

SUBGOAL
GOAL ( "boolean" : "announce" = "false"
RELATED TO PLAN "Announcing_item"
)

GOAL ( "boolean" : "negotiate" = "true"
RELATED TO PLAN "Selling_item"
)

)

BELIEF ("Item" : "itemBought" = "false")
BELIEF ("Item" : "itemSold" = "false")
BELIEF ("Item" : "item" = "null")

PLAN ( Searching_seller
COMPOSED OF ACTION "Search_for_seller"
RELATED TO GOAL "searchSeller"

)

PLAN ( Buying_item
COMPOSED OF ACTION "Create_item"
COMPOSED OF ACTION "Send_request_item"
COMPOSED OF ACTION "Eval_proposal"
COMPOSED OF ACTION "Send_answer_proposal_accept"
COMPOSED OF ACTION "Send_answer_proposal_reject"
ACTION Send_payment
ACTION Search_for_organization
ACTION Check_organization_goals
ACTION Check_organization_roles
ACTION Select_role_to_play
ACTION Register_itself_in_organization
ACTION Search_for_announcement
ACTION Search_for_environment
ACTION Announce
ACTION Create_proposal
ACTION Eval_counter_proposal
ACTION Send_answer_counter_proposal_accept
ACTION Send_answer_counter_proposal_reject
ACTION Calculate_bill
ACTION Inform_organization

AGENT
( Store_Agent
  GOAL ("boolean" : "itemSold" = "true"
              RELATED TO PLAN "Selling_item"
              RELATED TO PLAN "Informing_organization_about_sale"
  )
  BELIEF ("Item" : "itemSold" = "false")

PLAN ( Selling_item
  COMPOSED OF ACTION "Create_proposal"
  COMPOSED OF ACTION "Calculate_bill"
  COMPOSED OF ACTION "Inform_organization"
  RELATED TO GOAL "itemSold"
  )

PLAN ( Informing_organization_about_sale
  COMPOSED OF ACTION "Inform_organization"
  RELATED TO GOAL "itemSold"
  )

ACTION Create_proposal
ACTION Calculate_bill
ACTION Inform_organization

AGENTROLE
( Seller
  ROLEGOAL ("boolean" : "itemSold" = "true"
  )
  DUTY "Inform_organization"
  PROTOCOL SimpleNegotiation
  MESSAGE (LABEL "Request"
           CONTENT "ItemDescription"
           SENDER "Buyer"
           RECEIVER "Seller"
  )
  MESSAGE (LABEL "Proposal"
           CONTENT "ItemDescription"
           SENDER "Seller"
           RECEIVER "Buyer"
  )
  MESSAGE (LABEL "AnswerProposal"
            CONTENT "RejectItemDescription"
SENDER "Buyer"
RECEIVER "Seller"
)
MESSAGE
(LABEL "AnswerProposal"
CONTENT "AcceptItemDescription"
SENDER "Buyer"
RECEIVER "Seller"
)
MESSAGE
(LABEL "Inform"
CONTENT "Bill"
SENDER "Seller"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Inform"
CONTENT "Payment"
SENDER "Buyer"
RECEIVER "Seller"
)

ORGANIZATION
( Second_hand_Bookstore
  GOAL ( "boolean" :  "management_of_entrance" = "true"
       RELATED TO PLAN "Managing_entrance"
     )
  GOAL ( "boolean" :  "management_of_announcement" = "true"
       RELATED TO PLAN "Managing_announcement"
     )
  GOAL ( "boolean" :  "management_of_order" = "true"
       RELATED TO PLAN "Registering_sale"
     )
  BELIEF ( "boolean" :  "management_of_entrance" = "false")
  BELIEF ( "boolean" :  "management_of_announcement" = "false")
  BELIEF ( "boolean" :  "management_of_order" = "false")
  BELIEF ( "Vector" : "sales" = null)
  BELIEF ( "Vector" : "announcements" = null)

PLAN ( Managing_entrance
  COMPOSED OF ACTION "Get_Organization_Goals"
  COMPOSED OF ACTION "Get_Organization_Roles"
  COMPOSED OF ACTION "Register_new_buyer"
  RELATED TO GOAL "management_of_entrance"
)
PLAN ( Managing_announcement
  COMPOSED OF ACTION "Register_announcement"
  COMPOSED OF ACTION "Get_Announcement"
  COMPOSED OF ACTION "Inform_Announcement_and_Seller"
  RELATED TO GOAL "management_of_announcement"
)
PLAN ( Registering_sale
  COMPOSED OF ACTION "Register_sale"
  COMPOSED OF ACTION "Inform_supper_organization"
  RELATED TO GOAL "management_of_order"
)
ACTION Get_Organization_Goals
ACTION Get_Organization_Roles
ACTION Register_new_buyer
ACTION Register_announcement
ACTION Get_Announcement
ACTION Inform_Announcement_and_Seller
ACTION Register_sale
ACTION Inform_supper_organization
AXIOM ("boolean" : "Send_information_about_sale" = "true")

ORGANIZATION
( Imported_Bookstore
  GOAL ( "boolean" :  "management_of_buyers" = "true"
  RELATED TO PLAN "Analyzing_entrance"
  )
  GOAL ( "boolean" :  "management_of_sellers" = "true"
  RELATED TO PLAN "Creating_sellers"
  )
  GOAL ( "boolean" :  "management_of_orders" = "true"
  RELATED TO PLAN "Registering_sale"
  )
  BELIEF ( "boolean" :  "management_of_buyers" = "false"
  )
  BELIEF ( "boolean" :  "management_of_sellers" = "false"
  )
  BELIEF ( "boolean" :  "management_of_orders" = "false"
  )
  BELIEF ( "Vector" :  "sellers" = null
  )
  BELIEF ( "Vector" :  "buyers" = null
  )
  BELIEF ( "Vector" :  "sales" = null
  )
  BELIEF ( "Organization" :  "main_organization" = null

PLAN ( Analyzing_entrance
  COMPOSED OF ACTION "Get_Organization_Goals"
  COMPOSED OF ACTION "Get_Organization_Roles"
  COMPOSED OF ACTION "Analyze_entrance"
  COMPOSED OF ACTION "Register_new_buyer"
  RELATED TO GOAL "management_of_buyers"
  )

PLAN ( Creating_sellers
  COMPOSED OF ACTION "Create_seller"
  COMPOSED OF ACTION "Inform_buyer_of_seller"
  RELATED TO GOAL "management_of_sellers"
  )

PLAN ( Registering_sale
  COMPOSED OF ACTION "Register_sale"
  COMPOSED OF ACTION "Inform_supper_organization"
  COMPOSED OF ACTION "Update_environment_action"
  RELATED TO GOAL "management_of_order"
  )

ACTION Get_Organization_Goals
ACTION Get_Organization_Roles
ACTION Analyze_entrance
ACTION Register_new_buyer
ACTION Create_seller
ACTION Inform_buyer_of_seller
ACTION Register_sale
ACTION Inform_supper_organization
ACTION Update_environment_action
AXIOM ("boolean" : "Send_information_about_sale" = "true")

AGENTROLE
( Market_of_Used_Goods
  ROLEGOAL ("boolean" : "management_of_entrance" = "true"
  )
  )
ROLEGOAL ("boolean": "management_of_announcement" = "true")
ROLEGOAL ("boolean": "management_of_orders" = "true")

PROTOCOL To_enter_in_market_of_Used_Goods
MESSAGE
(LABEL "Request"
 CONTENT "OrgGoals"
 SENDER "Buyer"
 RECEIVER "Market_of_Used_Goods"
)
MESSAGE
(LABEL "Inform"
 CONTENT "OrgGoals"
 SENDER "Market_of_Used_Goods"
 RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
 CONTENT "Roles"
 SENDER "Buyer"
 RECEIVER "Market_of_Used_Goods"
)
MESSAGE
(LABEL "Inform"
 CONTENT "Roles"
 SENDER "Market_of_Used_Goods"
 RECEIVER "Buyer"
)
MESSAGE
(LABEL "Inform"
 CONTENT "RoleToPlay"
 SENDER "Buyer"
 RECEIVER "Market_of_Used_Goods"
)

PROTOCOL Buyer_registration
MESSAGE
(LABEL "Inform"
 CONTENT "Buyer_of_Second_hand_Books"
 SENDER "Buyer_of_Second_hand_Books"
 RECEIVER "Market_of_Used_Goods"
)

PROTOCOL To_announce
MESSAGE
(LABEL "Inform"
 CONTENT "Offer_Seller"
 SENDER "Seller_of_Second_hand_Books"
 RECEIVER "Market_of_Used_Goods"
)

PROTOCOL To_search_announcement
MESSAGE
(LABEL "Request"
 CONTENT "Announcement"
 SENDER "Buyer_of_Second_hand_Books"
 RECEIVER "Market_of_Used_Goods"
)
MESSAGE
(LABEL "Inform"
 CONTENT "Announcement"
 SENDER "Market_of_Used_Goods"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Seller"
SENDER "Market_of_Used_Goods"
RECEIVER "Buyer_of_Second_hand_Books"
)

PROTOCOL To_register_sale
MESSAGE
(LABEL "Inform"
CONTENT "ItemSold_Buyer_Payment"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Market_of_Used_Goods"
)

AGENTROLE
( Market_of_Special_Goods
  ROLEGOAL ("boolean": "management_of_buyers" = "true"
)
  ROLEGOAL ( "boolean": "management_of_sellers" = "true"
)
  ROLEGOAL ( "boolean": "management_of_orders" = "true"
)
DUTY "Analyze_entrance"

PROTOCOL To_request_to_enter_in_market_of_Special_Goods
MESSAGE
(LABEL "Request"
CONTENT "OrgGoals"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "OrgGoals"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
CONTENT "Roles"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "Roles"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE
(LABEL "Request"
CONTENT "RoleToPlay"
SENDER "Buyer"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE
(LABEL "Inform"
CONTENT "PermissionAccept"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
MESSAGE "Inform"
(LABEL "PermissionReject"
CONTENT "PermissionReject"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer"
)
PROTOCOL Buyer_registration
MESSAGE "Inform"
(LABEL "Buyer_of_Imported_Books"
CONTENT "Buyer_of_Imported_Books"
SENDER "Buyer_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)
PROTOCOL To_search_for_seller
MESSAGE "Request"
(LABEL "Seller"
CONTENT "Seller"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer_of_Imported_Books"
)
MESSAGE "Inform"
(LABEL "Seller"
CONTENT "Seller"
SENDER "Market_of_Special_Goods"
RECEIVER "Buyer_of_Imported_Books"
)
PROTOCOL To_register_sale
MESSAGE "Inform"
(LABEL "ItemSold_Buyer_Payment"
CONTENT "ItemSold_Buyer_Payment"
SENDER "Seller_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)
)
AGENTROLE
( Buyer_of_Imported_Books
ROLEGOAL ("boolean" : "itemBought" = "true"
)
DUTY "Register_itself_in_organization"
PROTOCOL Buyer_registration
MESSAGE "Inform"
(LABEL "Buyer_of_Imported_Books"
CONTENT "Buyer_of_Imported_Books"
SENDER "Buyer_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)
PROTOCOL To_search_for_seller
MESSAGE "Request"
(LABEL "Seller"
CONTENT "Seller"
SENDER "Buyer_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)
MESSAGE "Inform"
(LABEL "Seller"
CONTENT "Seller"
SENDER "Market_of_Special_Goods"
AGENTROLE
( Buyer_of_Second_hand_Books
  ROLEGOAL ("boolean" : "itemBought" = "true"
  )
  DUTY "Register_itself_in_organization"
  PROTOCOL Buyer_registration
  MESSAGE
  (LABEL "Inform"
   CONTENT "Buyer_of_Second_hand_Books"
   SENDER "Buyer_of_Second_hand_Books"
   RECEIVER "Market_of_Used_Goods"
  )
  PROTOCOL SimpleNegotiation
  MESSAGE
  (LABEL "Request"
   CONTENT "ItemDescription"
   SENDER "Buyer_of_Second_hand_Books"
   RECEIVER "Seller_of_Second_hand_Books"
  )
(LABEL "Proposal"
  CONTENT "ItemDescription"
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "AnswerProposal"
  CONTENT "RejectItemDescription"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "AnswerProposal"
  CONTENT "AcceptItemDescription"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
  CONTENT "Bill"
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
  CONTENT "Payment"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)

PROTOCOL ComplexNegotiation
MESSAGE
(LABEL "Call_for_proposal"
  CONTENT "Item"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Proposal"
  CONTENT "Item"
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Counter_proposal"
  CONTENT "Item"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Reject_counter_proposal"
  CONTENT ""
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Accept_counter_proposal"
  CONTENT ""
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
  CONTENT "Bill"
  SENDER "Seller_of_Second_hand_Books"
  RECEIVER "Buyer_of_Second_hand_Books"
)

MESSAGE
(LABEL "Inform"
  CONTENT "Payment"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)

MESSAGE
(LABEL "Accept_propose"
  CONTENT "Item"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Seller_of_Second_hand_Books"
)

PROTOCOL To_search_announcement
MESSAGE
(LABEL "Request"
  CONTENT "Announcement"
  SENDER "Buyer_of_Second_hand_Books"
  RECEIVER "Market_of_Used_Goods"
)

MESSAGE
(LABEL "Inform"
  CONTENT "Announcement"
  SENDER "Market_of_Used_Goods"
  RECEIVER "Buyer_of_Second_hand_Books"
)

MESSAGE
(LABEL "Inform"
  CONTENT "Seller"
  SENDER "Market_of_Used_Goods"
  RECEIVER "Buyer_of_Second_hand_Books"
)

AGENTROLE
( Seller_of_Imported_Books
  ROLEGOAL ("boolean" : "itemSold" = "true"
  )
  DUTY "Inform_organization"
  PROTOCOL SimpleNegotiation
  MESSAGE
  (LABEL "Request"
    CONTENT "ItemDescription"
    SENDER "Buyer_of_Imported_Books"
    RECEIVER "Seller_of_Imported_Books"
  )

MESSAGE
(LABEL "Proposal"
  CONTENT "ItemDescription"
  SENDER "Seller_of_Imported_Books"
  RECEIVER "Buyer_of_Imported_Books"
)

MESSAGE
(LABEL "AnswerProposal"
  CONTENT "RejectItemDescription"
  SENDER "Buyer_of_Imported_Books"
RECEIVER "Seller_of_Imported_Books"
)
MESSAGE
(LABEL "AnswerProposal"
CONTENT "AcceptItemDescription"
SENDER "Buyer_of_Imported_Books"
RECEIVER "Seller_of_Imported_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Bill"
SENDER "Seller_of_Imported_Books"
RECEIVER "Buyer_of_Imported_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Payment"
SENDER "Buyer_of_Imported_Books"
RECEIVER "Seller_of_Imported_Books"
)
PROTOCOL To_register_sale
MESSAGE
(LABEL "Inform"
CONTENT "ItemSold_Buyer_Payment"
SENDER "Seller_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)
)
AGENTROLE
( Seller_of_Second_hand_Books
  ROLEGOAL ("boolean" : "itemSold" = "true"
  )
  DUTY "Inform_organization"
) PROTOCOL SimpleNegotiation
MESSAGE
(LABEL "Request"
CONTENT "ItemDescription"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Proposal"
CONTENT "ItemDescription"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "AnswerProposal"
CONTENT "RejectItemDescription"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "AnswerProposal"
CONTENT "AcceptItemDescription"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Bill"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
CONTENT "Bill"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Payment"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)

PROTOCOL To_register_sale
MESSAGE
(LABEL "Inform"
CONTENT "ItemSold_Buyer_Payment"
SENDER "Seller_of_Imported_Books"
RECEIVER "Market_of_Special_Goods"
)

PROTOCOL ComplexNegotiation
MESSAGE
(LABEL "Call_for_proposal"
CONTENT "Item"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Proposal"
CONTENT "Item"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Counter_proposal"
CONTENT "Item"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
MESSAGE
(LABEL "Reject_counter_proposal"
CONTENT ""
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Accept_counter_proposal"
CONTENT ""
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Bill"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Buyer_of_Second_hand_Books"
)
MESSAGE
(LABEL "Inform"
CONTENT "Payment"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)
(LABEL "Accept_propose"
CONTENT "Item"
SENDER "Buyer_of_Second_hand_Books"
RECEIVER "Seller_of_Second_hand_Books"
)

PROTOCOL To_announce
MESSAGE
(LABEL "Offer_Seller"
CONTENT "Offer_Seller"
SENDER "Seller_of_Second_hand_Books"
RECEIVER "Market_of_Used_Goods"
)

import java.util.*;
public class Item
{
    protected boolean sold = false;
    public boolean getSold ()
    {
        return this.sold;
    }
    public void setSold ()
    {
        this.sold = true;
    }
}

import java.util.*;
public class Book extends Item
{
    protected String title = null;
    protected String author = null;
    protected String ISBN = null;
    protected int price = null;
    public String getTitle ()
    {
        return this.title;
    }
    public String getAuthor ()
    {
        return this.author;
    }
    public String getISBN ()
    {
        return this.ISBN;
    }
    public int getPrice ()
    {
        return this.price;
    }
    public void setTitle (String title)
    {
        this.title = title;
    }
    public void setAuthor (String author)
    {
        this.author = author;
    }
    public void setISBN (String ISBN)
public void setPrice (int price) {
    this.price = price;
}

import java.util.*;
public class Imported_Book extends Book {
    protected String origin = null;
    public String getOrigin () {
        return this.origin;
    }
    public void setOrigin (String origin) {
        this.origin = origin;
    }
}

import java.util.*;
public class Second_hand_Book extends Book {
    protected String appearance = null;
    public String getAppearance () {
        return this.appearance;
    }
    public void setAppearance (String appearance) {
        this.appearance = appearance;
    }
}

OBJECTROLE
( Desire (
    public int getPrice () {
        return object.getPrice();
    }
    public void setTitle (String title) {
        object.setTitle(title);
    }
    public void setAuthor (String author) {
        object.setAuthor(author);
    }
    public void setISBN (String ISBN) {
        object.setISBN(ISBN);
    }
))

OBJECTROLE
( Offer ( setPrice (int price) {
    object.setPrice(price);
})
public String getTitle ()
{
    return this.title;
}
public String getAuthor ()
{
    return this.author;
}
public String getISBN ()
{
    return this.ISBN;
}

OBJECTROLE
( Desire_of_Imported_Books {
    public void setOrigin (String origin)
    {
        object.setOrigin(origin);
    }
})

OBJECTROLE
( Offer_of_Imported_Books {
    public String getOrigin ()
    {
        return object.getOrigin();
    }
})

OBJECTROLE
( Desire_of_Second_hand_Books {
    public void setAppearance (String appearance)
    {
        object.setAppearance(appearance);
    }
})

OBJECTROLE
( Offer_of_Second_hand_Books {
    public String getAppearance ()
    {
        return object.getAppearance();
    }
})

Figura 121 – Descrição textual do exemplo do Virtual Marketplace.