Distributed Artificial Intelligence and Intelligent Agents

Design Agent Platform

2012

Mahboobeh Abdal Mahmood Abadi & Mehran Nasseri

12/3/2012

1. **Model system via GAIA AOSE methodology**
   1. **Analysis**
      1. **Role Model**

|  |  |
| --- | --- |
| **Role Schema** | Bidder |
| **Description** | Evaluating auctioneer’s offer and accepting or ignoring offered price for artifacts in the Auction model |
| **Protocols and Activities** | RegisterDF, Dutch auction, Deliver |
| **Permission**  **Reads**  **Generates** |  |
| Offers from auctioneer |
| Evaluation of the auctioneer’s offer (accept/ reject) |
| **Responsibilities**  **Liveness** |  |
| Bidder= RegisterDF.(Dutch auction.(Accept.Deliver| reject))\* |
| **Safety** | true |

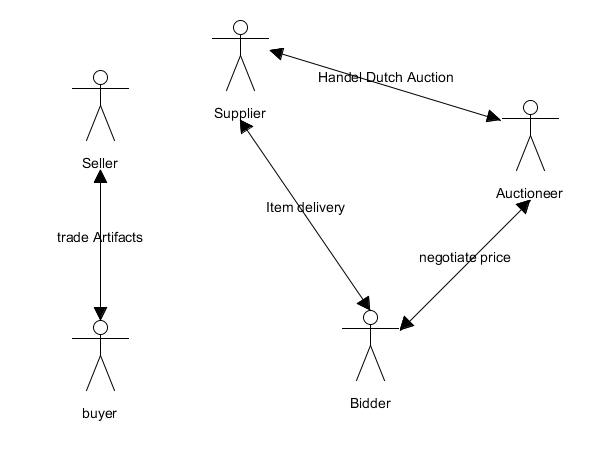
|  |  |
| --- | --- |
| **Role Schema** | Seller |
| **Description** | Offering and selling artifacts to the profilers according to their interest. |
| **Protocols and Activities** | RegisterDF, RequestItem, FindArtifact, OfferPrice, Deliver |
| **Permission**  **Reads**  **Generates** |  |
| Request from profiler |
| Detailed information about each of the artifacts |
| **Responsibilities**  **Liveness** |  |
| Seller= RegisterDF.(SellArtifact)\*  SellArtifact= RequestItem.FindArtifact.OfferPrice.(Accepte. Deliver | rejecte). |
| **Safety** | True |

|  |  |
| --- | --- |
| **Role Schema** | Auctioneer |
| **Description** | Mediating between suppliers and bidders in the Auction model |
| **Protocols and Activities** | QueryDF, Dutch auction, inform, Takedown, moveback |
| **Permission**  **Reads**  **Generates** |  |
| Offer presented by supplier (highest and reserve price) |
| Propose to the supplier the highest price offered by bidders. |
| **Responsibilities** |  |
| **Liveness** | Auctioneer= QueryDF. Dutch Auction. Report  Report = moveback. inform.Takedown |
| **Safety** | Price to bid >= reserve price  Number of bid>=1 |

|  |  |
| --- | --- |
| **Role Schema** | Supplier |
| **Description** | Proposing artifacts in auction model  Attempts to find the market price for a good |
| **Protocols and Activities** | Register mobility ontology, QueryPlatform, cloning, inform, Deliver |
| **Permission**  **Reads**  **Generates** |  |
| Best price from each container |
| Winner AID |
| **Responsibilities**  **Liveness** |  |
| Supplier= Register mobility ontology.( QueryPlatform.cloning. inform)\*.Deliver |
| **Safety** | Artifacts stock>0 |

|  |  |
| --- | --- |
| **Role Schema** | Buyer |
| **Description** | Search for Curators to buy desired artifacts at the lowest available price. |
| **Protocols and Activities** | QueryDF, RequestItem, OfferPrice, Deliver |
| **Permission**  **Reads**  **Generates** |  |
| Offers from curators |
| Evaluation of the artifacts (accept/ reject) |
| **Responsibilities**  **Liveness** |  |
| Buyer= QueryDF.(Shopping)\*  Shopping= RequestItem. OfferPrice.(accept.Deliver | reject) |
| **Safety** | Description= interest |

* + 1. **Interaction Model**

****

Negotiate price

|  |  |  |
| --- | --- | --- |
| Dutch auction | |  |
| Auctioneer | Bidder | * Offer presented by supplier (include highest and reserve price) |
| the auctioneer sends artifact detail and price to the bidders to bid. If none of the bidders satisfied with price Auctioneer, reduce the price and inform bidders about the price reduction. The price reduction will be repeated until reaching to the reserve price or one of the bidders accepts the proposed price. | | * Price and detail of artifact * Refuse the offered price   OR  Accept the Offered price |

Handel Dutch auction

|  |  |  |
| --- | --- | --- |
| inform | |  |
| Auctioneer | supplier | * End of the Auction |
| When the auction done, auctioneer informs supplier about the result of auction. (To what price the items are bided and who offered the highest price.) | | * Offered price for each item   OR   * Refuse(No bid) |

Item delivery

|  |  |  |
| --- | --- | --- |
| Deliver | |  |
| Supplier | Bidder | * Payment, AID |
| Supplier evaluates the price offered by each bidder for a particular item and delivers the item to one that offered the highest price. | | * Sending item   AND/OR   * Inform runner ups |

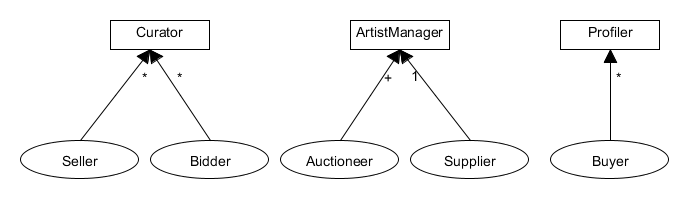
Trade Artifact

|  |  |  |
| --- | --- | --- |
| RequestItem | |  |
| Buyer | Seller | * Interest criteria |
| Buyer sends a query to retrieve price and detail of available artifacts according to predefined interest | | * Interest area * Propose artifact(s)   OR   * Reject (nothing available) |

|  |  |  |
| --- | --- | --- |
| OfferPrice | |  |
| Seller | Buyer | * RequestItem received * Item availability |
| Seller sends the price of available artifact to the buyer. | | * Price and detail of item(s) * Accept offer   OR   * Reject offer |

|  |  |  |
| --- | --- | --- |
| Deliver | |  |
| Seller | buyer | * OfferPrice accepted |
| Seller send the ordered item to the buyer agent | | * Sending item |

* 1. **Design**
     1. **Agent Model**



* + 1. **Service Model**
       1. Service model for ArtistManager

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | Dutch auction | Deliver | inform | Prepare for auction |
| **Inputs** | Max price and reverse price | Payment, AID | End of auction | Container |
| **Outputs** | Winner AID and bided price or no bid made | Sending item  AND/OR  Inform runner ups | Offered price for each item | Cloned agents |
| **Pre-condition** | (Number of bidders>=2)  ^ ( item≠nil) | (Money received= wining price)  ^ (AID≠nil) | Auction finished | container≠ nil |
| **Post-condition** | - | Stock =stock-item | - | Cloned agents moved |

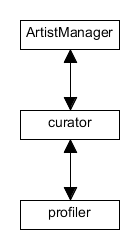
* + - 1. Service model for curator

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | Trade artifact | Artifact delivery | Dutch Auction |
| **Inputs** | Request item | Payment, AID | Item to bid |
| **Outputs** | (Matched item^price) ˅ N/A | Sending item | Accept or reject offer |
| **Pre-condition** | request≠nil | (AID≠nil) ^ (payment=item price) | Item≠nill |
| **Post-condition** | - | Stock=stock-item | - |

* + - 1. Service model for profiler

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | RequestItem | Evaluate item | Order |
| **Inputs** | Search criteria, curator AID | Item, price | Item, price |
| **Outputs** | List of artifacts | Accept ˅ reject | Payment |
| **Pre-condition** | curator≠nil | Item≠nil | Agreement |
| **Post-condition** | - | - | Payment=price |

* + 1. Acquaintance Model



1. **Model interactions among agents in UML**
   1. **Level 1 representation for overall System**

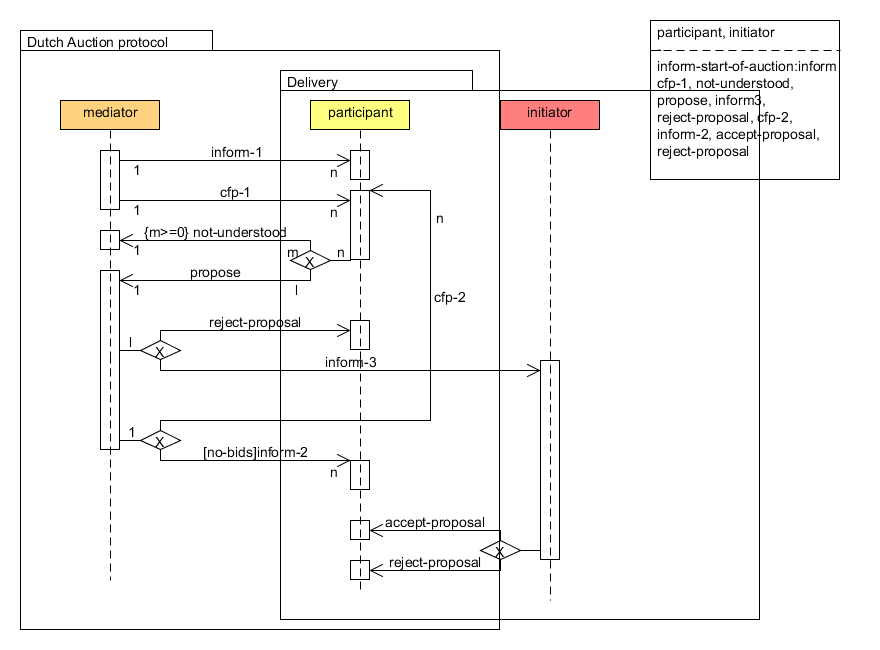


Figure1: Level 1 overall protocol

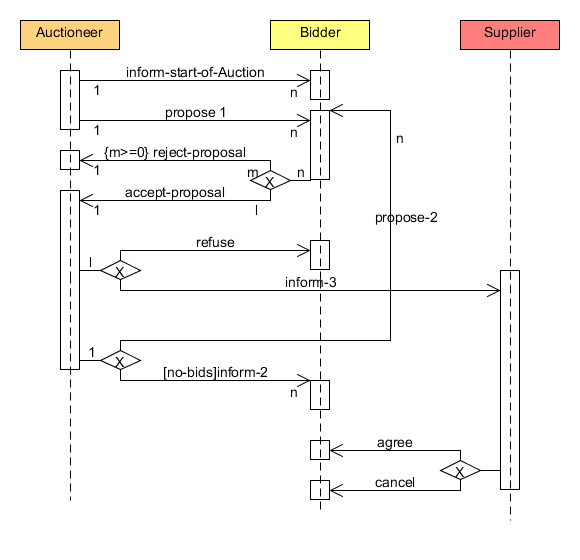


Figure 2: Applying the template in fig 1 to a particular scenario involving Auctioneer, bidder and supplier

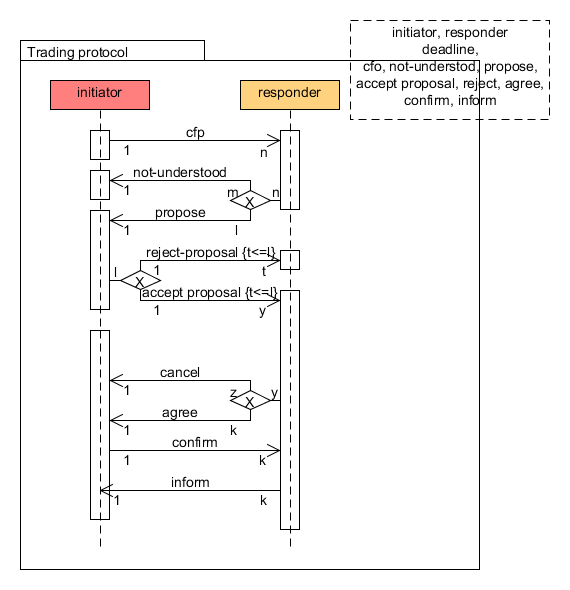


Figure3: Level 1 overall protocol

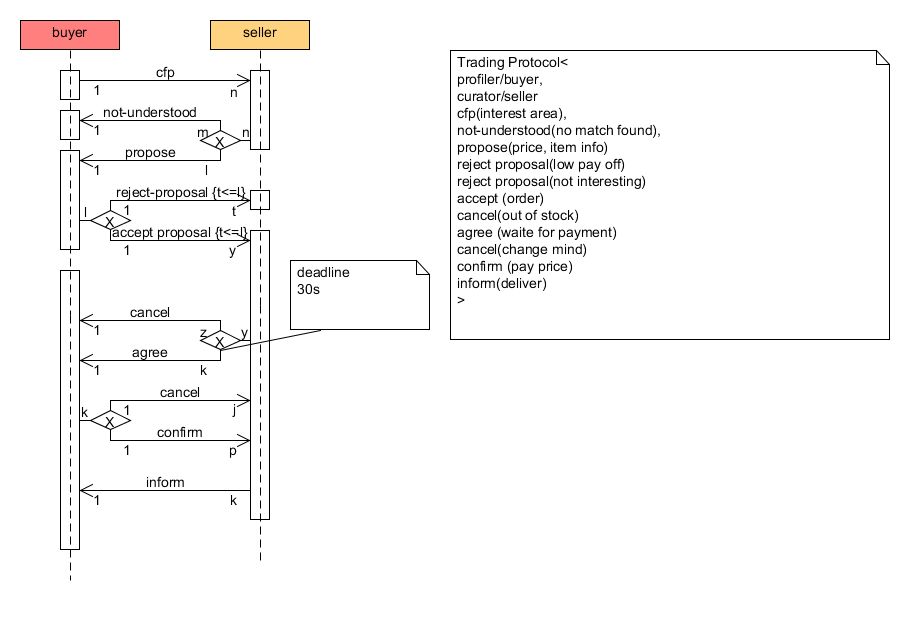
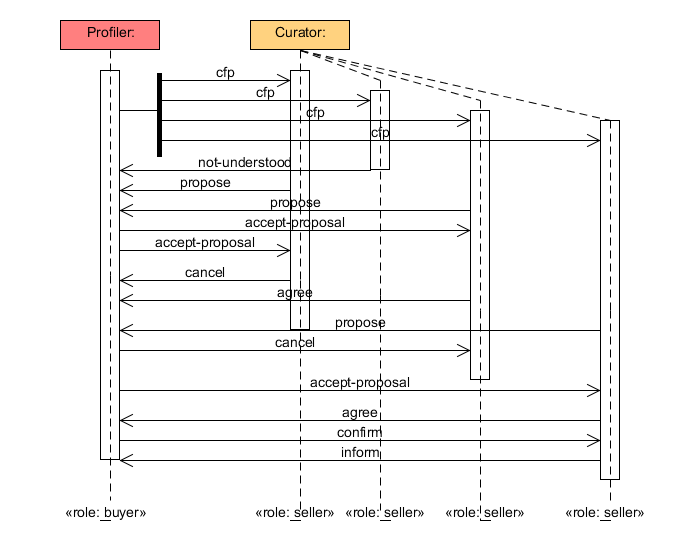
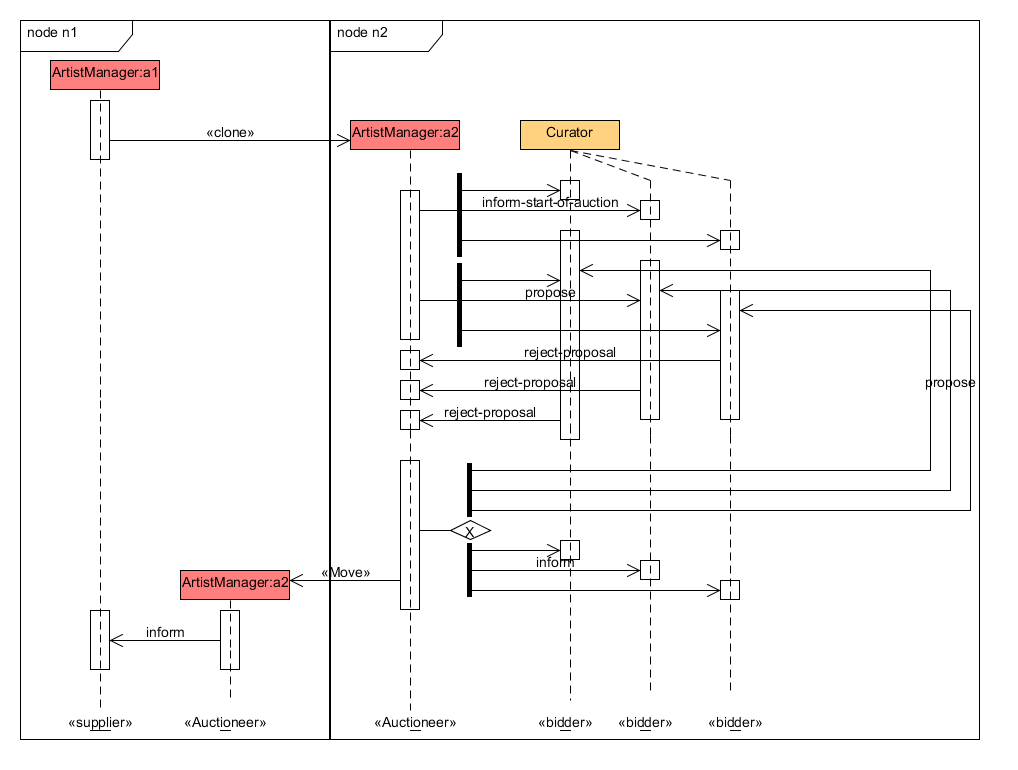
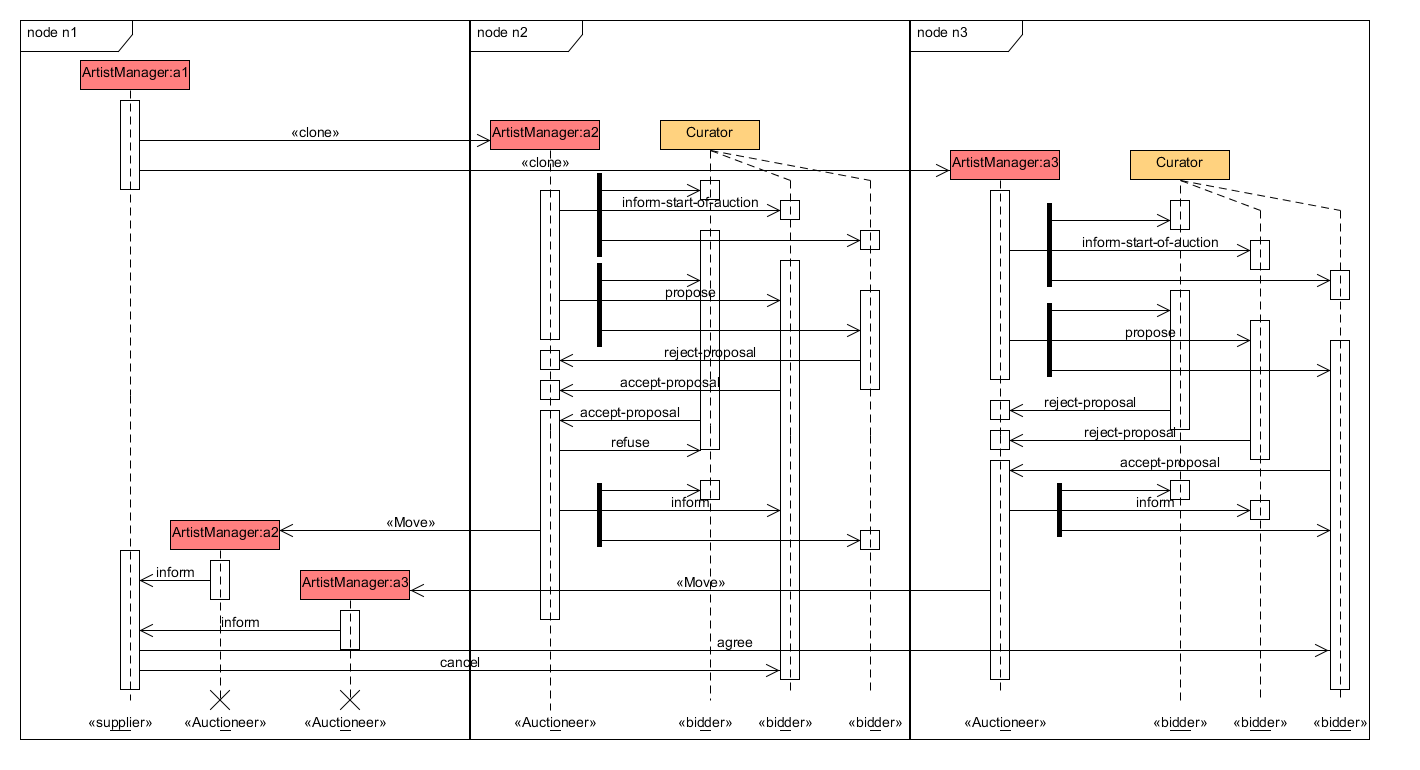


Figure 4: Applying the template in fig 3 to a particular scenario involving buyer and seller

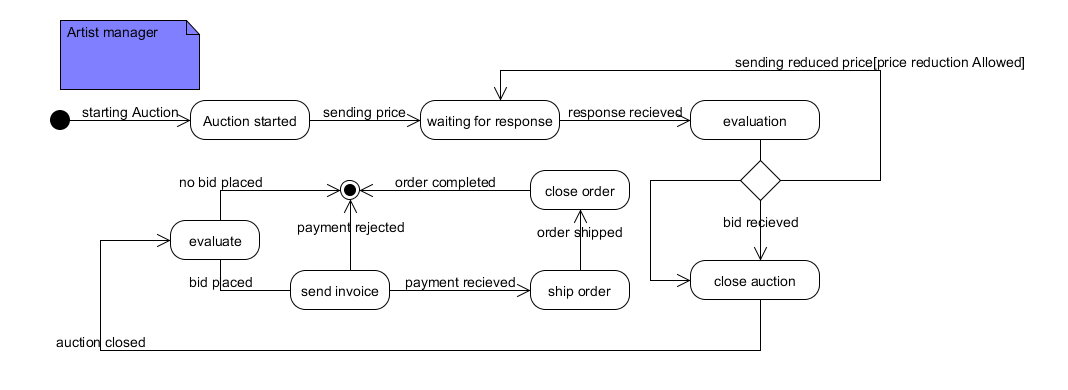
* 1. **Level 2 representation of Agent interactions using Sequence Diagrams**

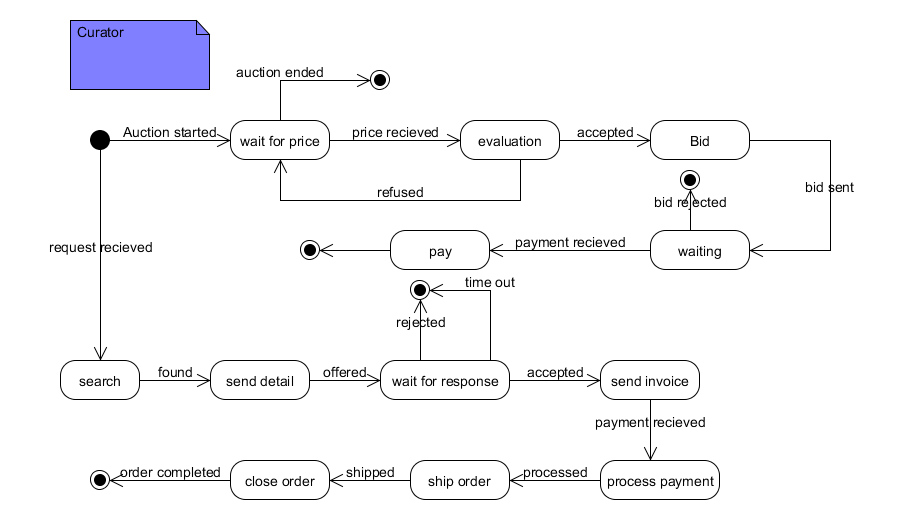
****

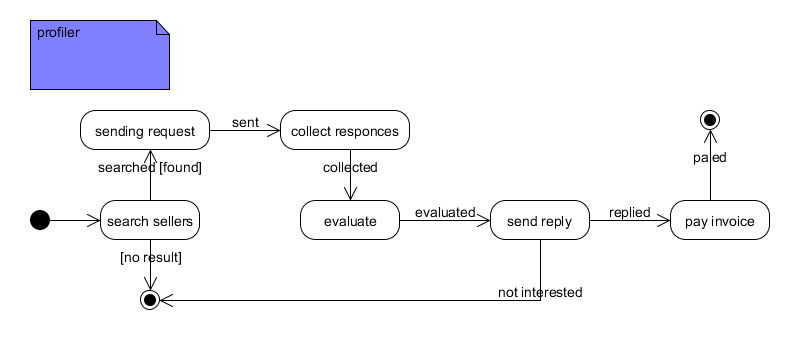
****

****

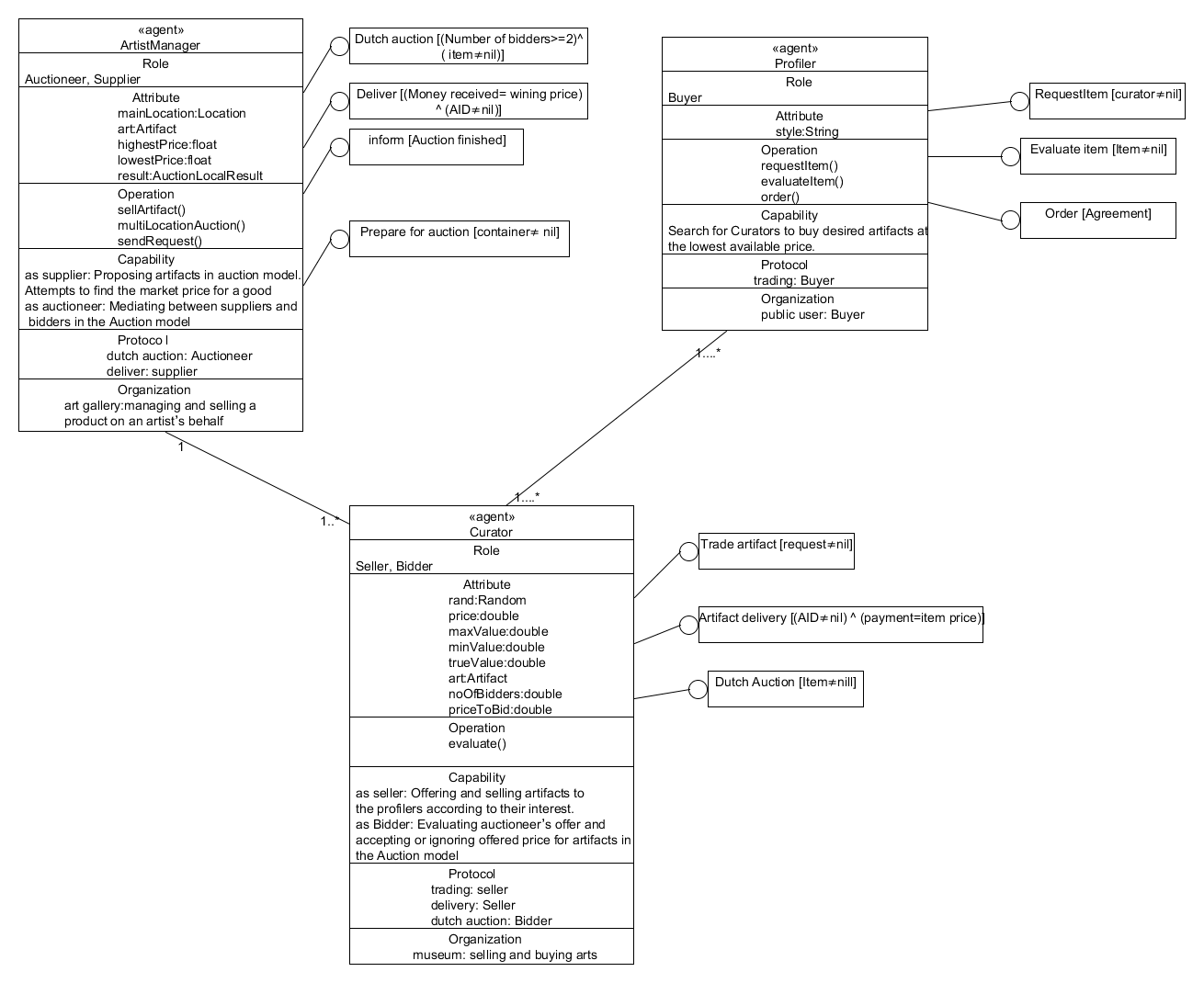
* 1. **Level 3 representation of Agent behaviors using State-chart diagrams**



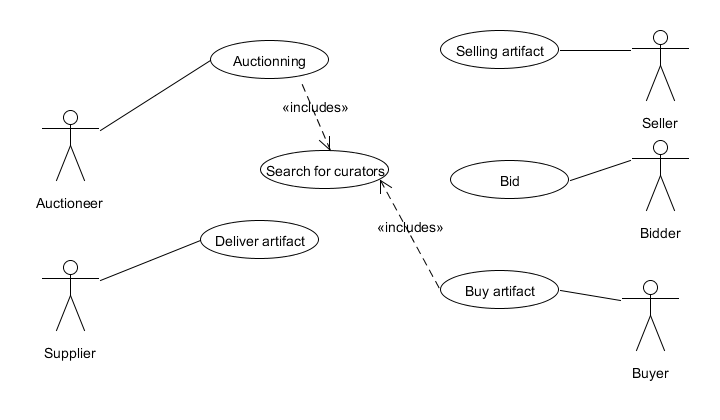




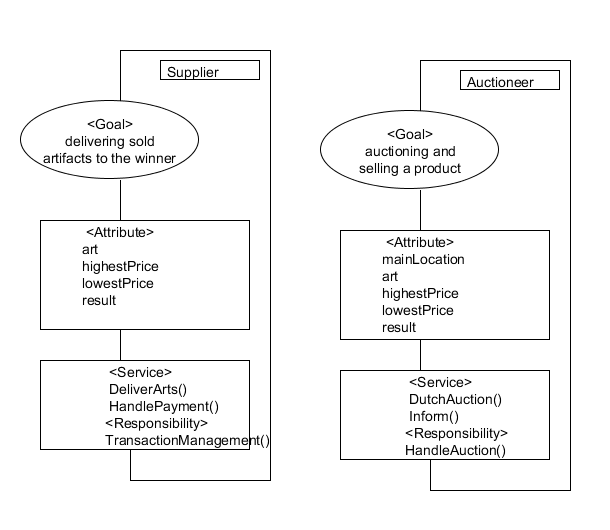
1. **UML Class diagrams to design behavior of agents.**

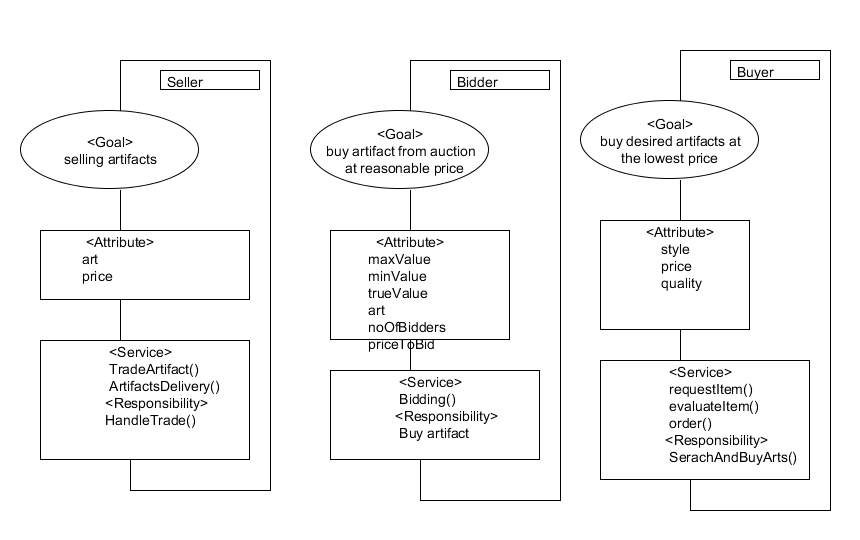
****

1. **Model your system using Role based modeling approach**
   1. **Capture use case**

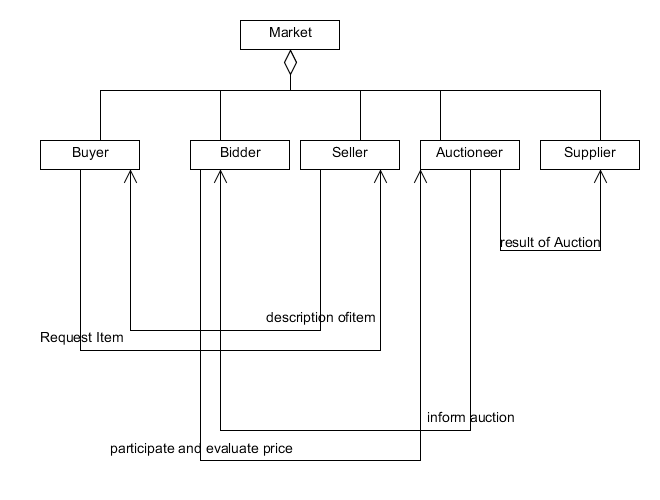
****

* 1. **Identify roles**

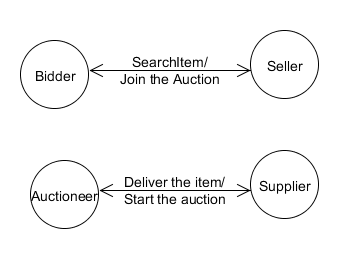
****

****

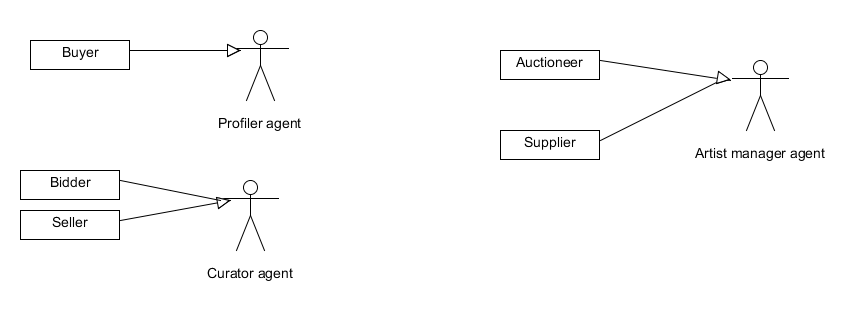
* 1. **Construct role organization**

****

* 1. **Bind roles to agents**
     1. **Role transition**

****

* + 1. **Role Binding**

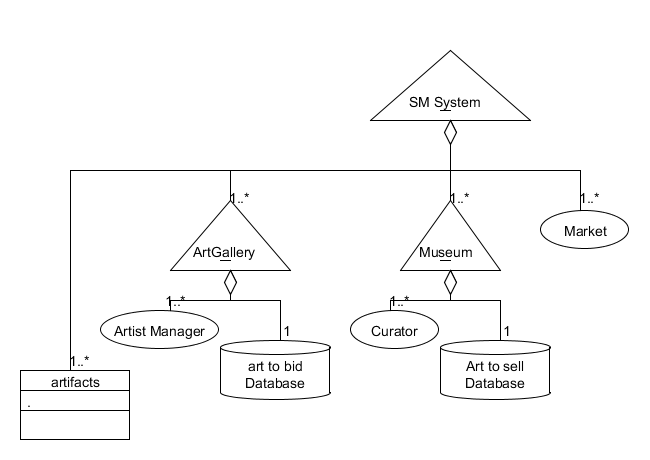
****

* 1. **GAIA vs. ROMAS**

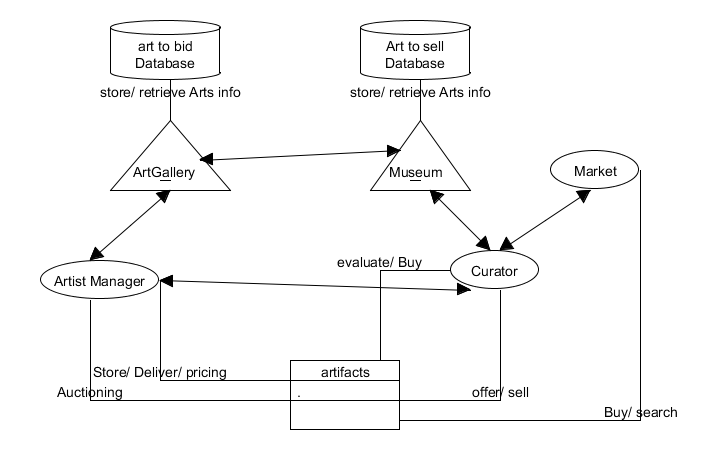
|  |  |
| --- | --- |
| **GAIA** | **ROMAS** |
| * Role is defined implicitly by four attributes:   responsibilities, permissions, activities, and protocols   * An agent cannot change its roles at run-time since:   Fixations of role to agent, the interactions among agents are decided according to the system goal in the phase of design.   * Gaia cannot explicitly model the organizational structure of the agents. It also lacks the ability to model explicitly the social goals, social tasks, or organizational rules within an organization of agents. | * role is an encapsulation of certain attributes and behaviors of the agent it is bound to * Agents bind certain roles; it enables the agents to change their roles dynamically. Roles are not isolate. * Every role communicates and interacts with other roles. * Besides, roles can be specialized or aggregate to other roles. Inheritance and aggregation associations respectively denote specialize/generalize and aggregate/decompose relations among roles. |

1. **Re-model the entire system using MESSAGE UML**
   1. **Level 0 Analysis**
      1. **Organization view**

The Organization View shows ConcreteEntities (Agents, Organizations, Roles, and Resources) in the system and its environment and coarse-grained relationships between them (aggregation, power, and acquaintance relationships). An acquaintance relationship indicates the existence of at least one Interaction involving the entities concerned.



Structural relationships

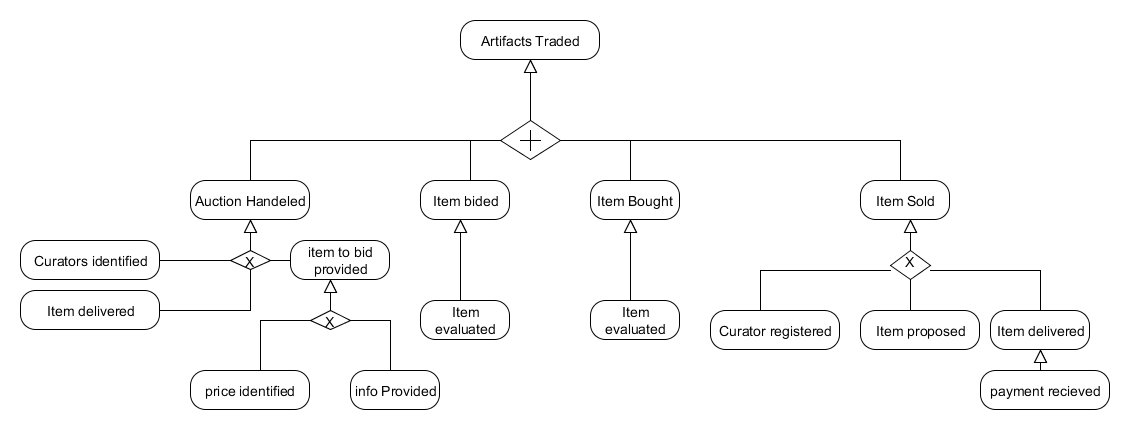


Acquaintance relationships

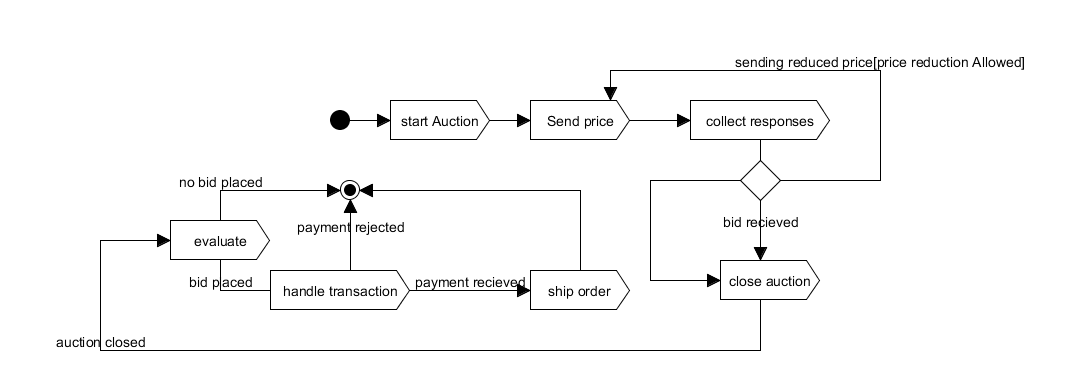
* + 1. **Goal/Task view**

This shows Goals, Tasks, Situations and the dependencies among them. Goals and Tasks both have attributes of type Situation, so that they can be linked by logical dependencies to form graphs that show e.g. decomposition of high-level Goals into sub-goals, and how Tasks can be performed to achieve Goals. Graphs showing temporal dependencies can also be drawn, and we have found UML Activity Diagram notation useful here.

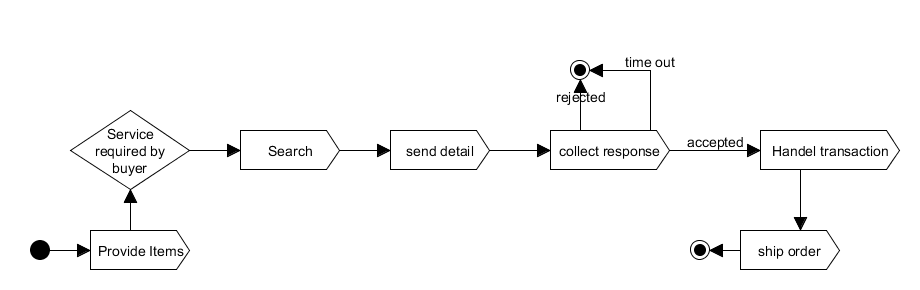
* + - 1. Goal View



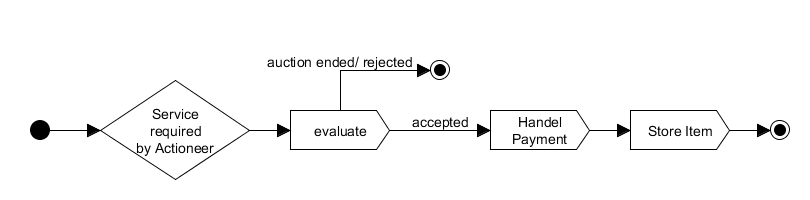
* + - 1. Task View



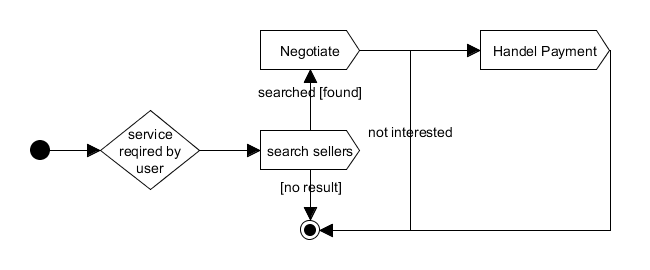
Auction



Selling Item

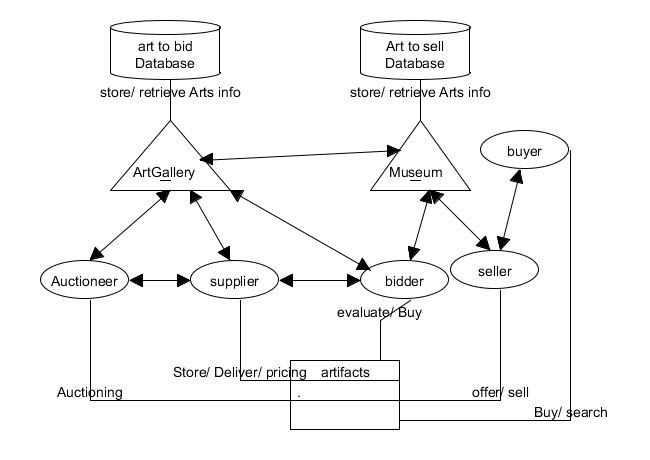


Buying through Auction



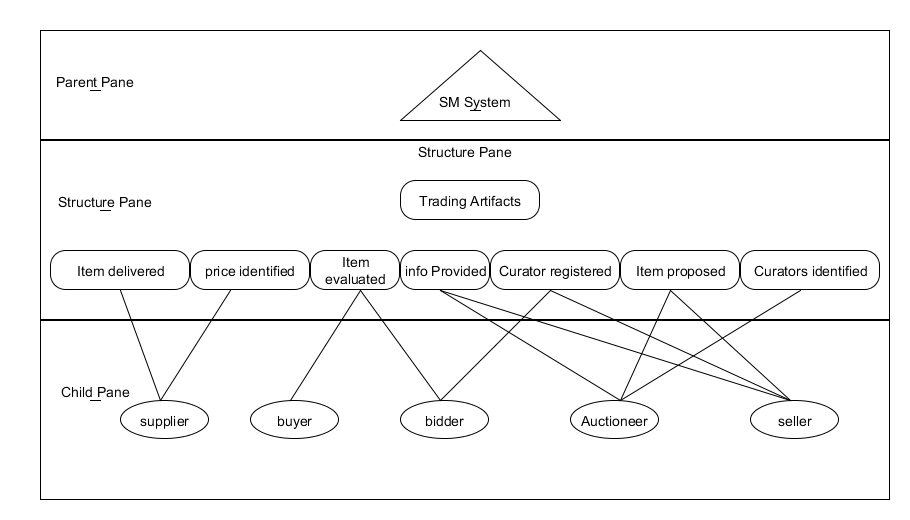
Buying Item

* 1. **Level 1 Analysis**
     1. **Organization view**



Acquaintance relationships

* + 1. **Agent/Role view**



Delegation structure

|  |  |
| --- | --- |
| **Role Schema** | Supplier |
| **Goal** | Item delivered, price identified |
| **Capability** | Some evaluation capability is required to set the max and min price. |
| **Knowledge, Beliefs** | The value and quality of item |
| **Agent requirements** | The role will be played by Artist manager |

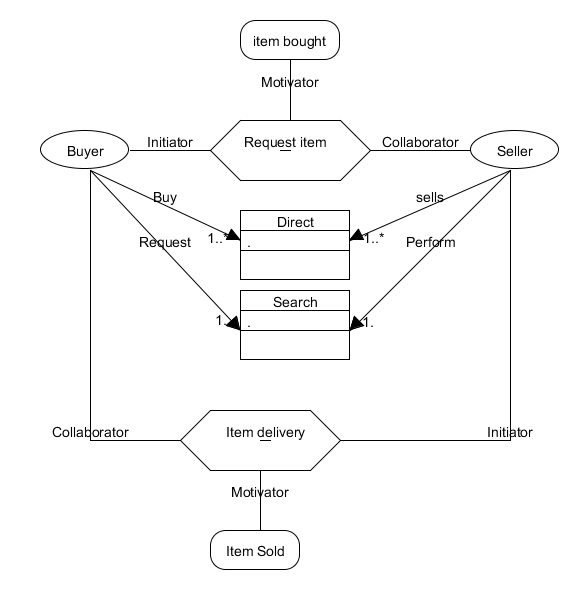
|  |  |
| --- | --- |
| **Role Schema** | Buyer |
| **Goal** | Item Bought, Item Evaluated |
| **Capability** | Search capability is required to find seller, as well as negotiation capability |
| **Knowledge, Beliefs** | A profile of interest area and value of proposed item to be used to evaluate if the proposed item matches the interest area and if a right price is set |
| **Agent requirements** | The role will be played by Profiler |

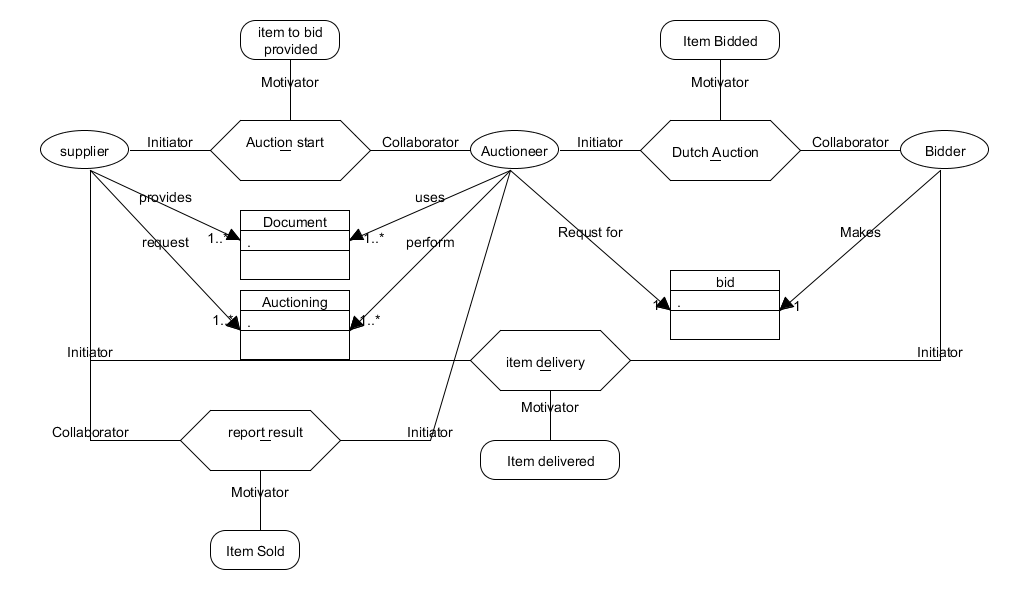
|  |  |
| --- | --- |
| **Role Schema** | Bidder |
| **Goal** | Item evaluated, item bided |
| **Capability** | Some evaluation capability is required to balance the pay off. |
| **Knowledge, Beliefs** | A profile of value of proposed item to be used to evaluate if the Auctioned item worth to bid and at which price |
| **Agent requirements** | The role will be played by Curator |

|  |  |
| --- | --- |
| **Role Schema** | Auctioneer |
| **Goal** | Info provided, item proposed, bidders identified |
| **Capability** | Some evaluation capability is required to reduce the price optimally and balance the pay off. |
| **Knowledge, Beliefs** | A profile of max and min price for each item and bidders location |
| **Agent requirements** | The role will be played by Artist manager |

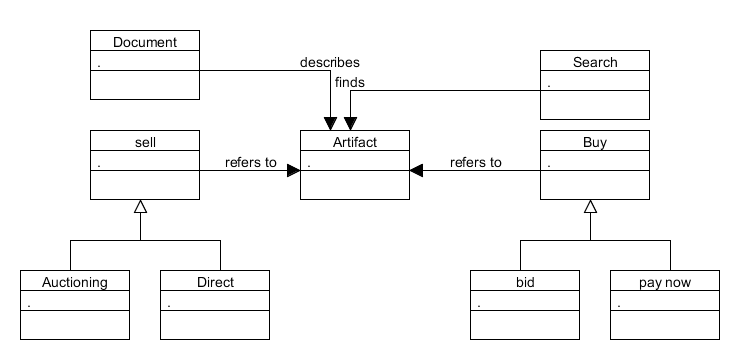
|  |  |
| --- | --- |
| **Role Schema** | Seller |
| **Goal** | Item proposed, info provided, item sold |
| **Capability** | Search capability is required to find and offer suitable item to buyer, as well as negotiation capability. |
| **Knowledge, Beliefs** | A comprehensive documentation of each item to be used to offer the most suitable one to the buyers |
| **Agent requirements** | The role will be played by Curator |

* + 1. **Interaction view**

****

****

* + 1. **Domain view**



1. **Compare MESSAGE UML and GAIA**

|  |  |  |
| --- | --- | --- |
| **GAIA whiteout Role based** | **GAIA with Role Based** | **Message UML** |
| The Gaia methodology has two analysis models and three design models. While the analysis models are based on well-defined concepts, these only represent a subset of the concepts required for agent-oriented analysis and it is not realistic to have such short analysis and design phase. The design models are not clearly explained and the authors envisage OO methods being used for detailed design.  Gaia cannot explicitly model the organizational structure of the agents. It also lacks the ability to model explicitly the social goals, social tasks, or organizational rules within an organization of agents. | They can be applied to existing entities to change their behavior; then, roles can be thought of as solutions common to different problems, so that they can be reused in different situations; further, roles enable a separation of concerns between the ‘business’ logic of the application, which is embedded in the agents, and the ‘coordination’ logic, which is embedded in the roles; finally, roles promote an organizational view of the system, which well suits agent-oriented approaches | MESSAGE is more adaptable to industrial scenarios, because they are evolutions of UML that is a common standard in this kind of environments.  The models that describe the  MAS, opposite to most formalism, can be detailed partially and refined through successive steps (similarly to UML, where class diagrams do not need to be complete from the beginning, but can grow in details during the development).  The MESSAGE approach starts by identifying first the elements required to build a MAS, organizing them in views, and expressing them in meta-models. These metamodels can be applied by integration in a well-proven software development process model, which in the case of MESSAGE is the Unified Software Development Process |