Validating Auction Business Processes using Agent-based Simulations

Emilian Pascalau\textsuperscript{1}, Adrian Giurca\textsuperscript{2} & Gerd Wagner\textsuperscript{2}

\textsuperscript{1}Hasso Plattner Institute
University of Potsdam

\textsuperscript{2}Institute of Informatics
Brandenburg University of Technology

2\textsuperscript{nd} INTERNATIONAL CONFERENCE ON BUSINESS PROCESS AND SERVICES COMPUTING

March 23-24, 2009, Leipzig, Germany
Agenda

- Motivation
- Single Item English Auction
  - BPMN model of the Single Item English Auction
- Agent-Object-Relationship
  - The Seller
  - The Bidder
  - The Environment
- Mapping Auction BPMN Model to AOR Model
Motivation

- Business Process implementation is a costly process.
- Therefore simulating business processes before implementation may be a solution to reduce these costs.

A business process management system is "a generic software system that is driven by explicit process representations to coordinate the enactment of business processes." (Weske, 2007)

"an agent is a computer system that is capable of independent action on behalf of its user or owner." (Wooldridge, 2002)

- One solution is to use agent-based simulations
- Small cost, efficiency, easy to be interpreted by business people.
Single Item English Auction

- Only one item is sold at a time
- Bidding is open i.e. all participants bid against each other openly
- Each successive bid must be higher than the old one
- The seller begins the auction by asking for bids at lower price
- Buyers bid against each other, by raising the price, until only one willing buyer remains
BPMN model of the Single Item English Auction

AuctionRequest
createAuction
Send Payment Details
ReceivePayment

AuctionHost
createAuction
AllowToAuction
ReceiveBids
Bid
BidStatus
ReceiveProducts
SendPayment

Bidder
RequestAdmissionToAuction
Bid
AuctionEndNotificationB
PaymentDetails

Seller
AuctionCreationRequestResponse
ReceivePayment

AuctionEnd
AuctionEndNotificationS
SendProducts

AuctionID
auctionStatus
startPrice
sellerID
winnerID
currentHighestPrice
products
cash

AuctionStatus
auctionID
auctionStatus
bidderID
currentHighestPrice
sellerID
bid
winnerID

processBid
DenyAccessToAuction
AuctionEndNotificationS
AuctionEndNotificationB
BidStatus
ReceivePayment

24 March 2009
BPSC 2009
Agent-Object-Relationship

"semantics of business processes may be more adequately captured if the specific business agents associated with the involved events and actions are explicitly represented in the information systems in addition to passive business objects" (Wagner, 2003)

- An entity is either an agent, an event, an action, a claim, a commitment, or an ordinary object.
- AOR models mainly reactive agents having the state represented by a knowledge base and its behavior modeled by means of actions and reaction rules.
- Reaction rules (or Event-Condition-Action Rules) are rules of the form:

  \[ \text{ON } \text{<Event>} \quad \text{IF } \text{<logical-condition>} \quad \text{THEN DO } \text{<actions>} \]
The state of an Agent-based Discrete Event Simulation System

- The state of an Agent-based Discrete Event Simulation System consists of (Wagner, 2004):
  - The simulated time
  - The environment state containing
    - The non-agentive environment (as a collection of objects)
    - The external states of all agents (i.e. their physical state, their geographic position etc)
  - The internal agent states (i.e. perceptions, beliefs, memory, goals)
  - A (possibly empty) queue of future events
The Seller Agent (1)

sellerID: int
startPrice: double
double
endPrice: int
inventory: int
cash: double
winnerID: int
auctionID: int
auctionStatus: int

Objective
Properties

StartAuction
occurrenceTime==1

AuctionRequest
sellerID
startPrice

AuctionCreation
RequestResponse

AuctionEnd
NotificationS

AuctionEnd
NotifcationS

StartAuction
occurrenceTime

SendPayment
products

SendPayment

SendPaymentDetails

SendPaymentDetails

SendProducts

SendProducts
RULE "AR"

ON StartAuction(?ev)

IF Seller(?Seller) AND ?ev.occurrenceTime == 1

THEN

DO AuctionRequest(?Seller.sellerID, ?Seller.startPrice)
The Environment Simulator

AuctionCreation
RequestResponse
auctionID
auctionStatus

AuctionRequest
sellerID
startPrice

ExogenousEvent
occurrenceTime

AuctionEnd
NotificationB
auctionStatus
sellerID

AuctionEnd
NotificationS
auctionStatus
currentHighestPrice
winnerID

Auction
auctionID
sellerID
bidders
currentHighestPrice
auctionTime
auctionStatus
winnerID

currentHighestPrice = ev.bid
winnerID = ev.bidDERID

bidders.add(ev.bidDERID)
bidders.contains(ev.bidDERID) == false

ev.occurrenceTime >= auctionTime
auctionStatus = 0

AdmissionRequest
bidderID

AllowToAuction
auctionID
currentHighestPrice
auctionStatus

Bid
bid
bidderID

BidStatus
currentHighestPrice
The activity pattern is mapped to one reaction rule

Start event is mapped to a no-action event type (StartAuction) with occurrence time 1

AuctionRequest message event is mapped to an AOR message type event. This event encodes also the passed objects during the AuctionRequest activity
The gateway pattern is mapped to two or more reaction rules. 

- **AuctionEndNotificationS** message event is mapped to an AOR message type event. This event encodes the message parameters too (e.g. `currentHighestPrice`).
- The gateway activities (e.g. **Send Products**) are mapped to message events which will trigger the corresponding reaction rules.
A loop is a procedural construct. Looping on the occurrence of a message event is encoded by means of periodic events.

On a periodic basis the loop activity is executed.

The loop exit is encoded by means of the occurrence of a specific event (e.g. BidStatus when currentHighestPrice = bidStatus.currentHighestPrice)
Timers are *exogenous events* i.e. events which are external to the agent process.
Example

- Auction
  - 60 simulation steps

- One Seller
  - Request Auction creation at step 1
  - Start Price: 50

- 4 Bidders
  - Bid: bidder.getBidRate() * Random.uniform(-50, 10) + bidder.getCurrentHighestPrice()
  - They get into auction at different points in time
    - Bidder1: step 4
    - Bidder2: step 4
    - Bidder3: step 10
    - Bidder4: step 20
Running the Example 1
Running the Example 2
Thank you!