한국경영과학회/대한산업공학회 2003 춘계공동학술대회
2003년 5월 16일-17일 한동대학교(포항)

분산 생산 시스템을 위한 에이전트 기반의 협업 시뮬레이션 체계

Agent-based Collaborative Simulation Architecture for Distributed Manufacturing Systems

차영필*, 정무영*
*포항공과대학교 산업공학과/제품생산기술연구소

Abstract

Maintaining agility and responsiveness in designing and manufacturing activities are the key issues for manufacturing companies to cope with global competition. Distributed design and control systems are regarded as an efficient solution for agility and responsiveness. However, distributed nature of a manufacturing system complicates production activities such as design, simulation, scheduling, and execution control. Especially, existing simulation systems have limited external integration capabilities, which make it difficult to implement complex control mechanisms for the distributed manufacturing systems. Moreover, integration and coupling of heterogeneous components and models are commonly required for the simulation of complex distributed systems.

In this paper, a collaborative and adaptive simulation architecture is proposed as an open framework for simulation and analysis of the distributed manufacturing enterprises. By incorporating agents with their distributed characteristics of autonomy, intelligence, and goal-driven behavior, the proposed agent-based simulation architecture can be easily adapted to support the agile and distributed manufacturing systems. The architecture supports the coordination and cooperation relations, and provides a communication middleware among the participants in simulation.

Keyword

Distributed Manufacturing System, Agent Technology, Distributed Simulation

1. 서론

오늘날의 무한경쟁 체제로 인해, 제조 산업은 고품질의 제품을 적은 비용으로 빠르고 효율적으로 생산해야 할 상황에 직면하였다. 이러한 환경 하에서 제조 산업의 성공은 고객의 요구에 신속하고 효과적으로 대응하여, 제품을 빠른 시간 내에 설계, 제조, 시험, 공급할 수 있는 능력에 크게 좌우된다. 즉, 소비자의 요구 변화에 따라 제품 생산을 위한 shop이 재구성되기 쉬워야 하며, 확장이 용이해야 한다. 이러한 생산 시스템의 변화는 shop floor control system (SFCS)의 변화로 나타난다. 과거의 SFCS는 주로 hierarchical architecture를 기반으로 했으나, 현재는 heterarchical architecture 또는 distributed architecture등으로 변화해 가고 있다. 특히 distributed architecture는 이전의 전통적 architecture의 최하위 레벨의 controller인 equipment controller 만으로 module을 구성하여 모듈간의 의사 협상(negotiation) 과정을 통해 모든 작업을 수행하여 shop의 유연성(flexibility)을 확보한다. Shop 내에서의 module의 분산 뿐 아니라 제품의 설계부터 공급에 이르기까지의 전체 제조 공정을 구성하고 있는 shop들 역시 지역적으로 분산되어 서로 연계하고 있는 네트워크 구조를 이루고 있다.

Manufacturing 분야의 distributed system으로의 변화는 simulation을 제조 시스템에 이용하는 분야에도 영향을 미치고 있다. 과거에는 simulation을 하나의 시스템으로 간주하여 그 하나의 시스템 내에 생산 시스템 전체를 반영하여 simulation 하였으나, 오늘날에는 simulation system 역시 분산된 구조를 취하는 방향으로 변화하고 있다. 이와 관련하여 distributed manufacturing system을 위한 효과적인 distributed simulation architecture가

필요하다. 본 연구에서는 mobile agent를 이용한 collaborative simulation architecture를 제시하고자 한다. 이를 위하여 (i) 먼저 manufacturing system에서의 simulation 이용에 대하여 언급하고, (ii) distributed simulation과 distributed manufacturing과의 관계를 살펴보고, (iii) mobile agent를 이용한 simulation architecture를 제시하고, (iv) 마지막으로 구현 방향에 대하여 살펴본다.

2. Manufacturing System과 Simulation

Simulation은 다양한 산업분야에서 널리 사용되고 있는 도구이다. Simulation을 이용하여 모델링하고 분석함으로써 얻을 수 있는 중요한 이점은 다음과 같다 [1].

- 사실적인 model을 만들 수 있다. simulation을 통해 industry system과 그 내부에 존재하는 interaction을 실질적으로 나타낼 수 있다.
- Alternative design이나 option이 실제 실험 없이 수행되고 평가될 수 있다.
- 실제 시스템에서 사용되는 performance measure와 동일한 measure를 simulation으로부터 도출할 수 있다.
- 존재하지 않는 시스템도 가상적으로 modeling할 수 있다.
- 시각적인 형태를 통해 쉽게 modeling 하고 validation할 수 있다.

특히 manufacturing system에서 simulation을 많이 사용하고 있는데, 그 주요한 이유는 다음과 같다 [2].

- 자동화된 대규모 생산 시스템은 일반적으로 analytic한 방법에 의해 modeling하고 분석하기 어렵다.
- Computing 기술의 발달로 인해 simulation 비용이 감소하였다.
- Simulation 소프트웨어의 발달로 인해 model 개발 시간과 분석 시간이 감소하였다.
- Animation의 사용을 통해 engineering manager가 쉽게 model과 결과를 이해할 수 있다.

Simulation이 manufacturing system의 modeling 과 분석에 중요하게 응용되고 있지만, 과거의 manufacturing simulation은 몇 가지 한계를 가지고 있다. 일반적으로 기존 manufacturing system에서 사용되는 방법론들은 functional decomposition을 기반으로 하기 때문에, factory scheduling, process control, 그리고 simulation이 분리된 information system으로 간주되어져 왔다. 결과적으로 simulation system은 실제 manufacturing system과의 integration이 제한되어져 왔고, 복잡한 실제의 control mechanism을 반영하기 어려웠다 [3]. 실제 system의 복잡한 control, interaction, cooperation behavior를 dispatching rule이나 script 등으로 단순화 하여 simulation 상에 반영하였다. 뿐만 아니라, 기존의 simulation tool은 decision이나 control modeling에 부적합에 부적합하기 때문에, simulation 중에 decision process의 결과를 포함시킬 수 없었다 [4]. 즉, Simulation과 decision을 위한 control modeling이 분리된 채로 수행되었다.

3. Distributed Simulation

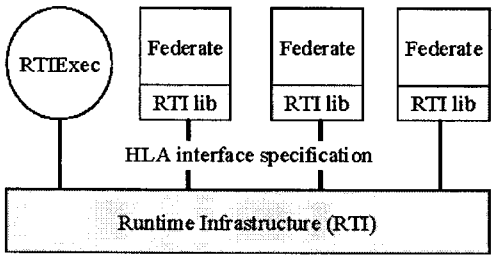
Distributed system이 computer science, information management, manufacturing 등 다양한 분야에 도입되고 응용됨에 따라, simulation 역시 영향을 받아 distributed simulation에 대한 연구가 활발히 이루어지기 시작하였다. Distributed simulation은 분산된 다수의 개체나 모듈을 각각 표현하는 simulation module이나 simulator들이 서로 interaction을 통하여 전체 시스템을 simulation하는 것이다. 따라서 분산된 실제 시스템을 더욱 현실적으로 표현할 수 있고, 실제 시스템과의 integration 역시 쉬워진다. 뿐만 아니라 분산된 각각의 simulation module의 많은 부분을 독립적으로 구현할 수 있기 때문에 복잡한 시스템을 효과적으로 반영할 수 있고, simulation module의 특화 및 재사용이 용이해진다.

이러한 distributed simulation에서 module들 간의 interaction과 interoperability를 위해서는 communication architecture에 대한 표준이 필요하다. 1990년대에 들어서서 distributed simulation에 대한 본격적인인 연구가 진행되었다. 특히 미국에서는 국방부를 중심으로 1991년경 DIS(Distributed Interactive Simulation)[5] 와 1995년경 HLA(High Level Architecture)[6] 표준에 대한 연구를 진행하였다.

DIS에서는 다양한 종류의 simulation application간에 교환할 수 있는 data message를 정의하고 있다. DIS에서 특정 simulation application은 여러 가지 다른 simulation이 실행되는 네트워크에 simulation 노드 접속을 통하여 분산 실행에 참여할 수 있고, 분산 실행에 참여하는 각 simulation application은 PDU(Protocol Data Unit)로 정의된 data를 주고받으며 자율적으로 simulation을 진행하면서 상호작용을 수행한다.

HLA는 분산된 네트워크 환경에서 simulation

의 상호운용성(interoperability)을 보장하고, simulation 소프트웨어의 재사용성을 촉진시키기 위한 architecture이다. HLA는 simulation에 대한 standard structure, design rule 그리고 interface에 대하여 규정하고 있다. HLA에서는 [그림 1]에서와 같이 각 simulation 개체(federate)가 RTI (Runtime Infrastructure)라는 공통 module을 통하여 상호 message를 주고받으며 RTI하에서 모든 참여 simulation model이 연합하여 전체 시스템을 simulation할 수 있도록 하고 있다. DIS는 architecture에 data를 포함시켜 protocol의 확장이 제한적이지만 HLA는 이를 분리하여 application의 요구에 따라 data structure를 변화시킬 수 있다. 또한 DIS는 full broadcast distribution approach를 사용하고 있지만 HLA는 RTI를 통해 simulator간에 선택적으로 data를 전송할 수 있다.



[그림 1] HLA의 구조

4. Mobile-agent-based Collaborative Simulation Architecture

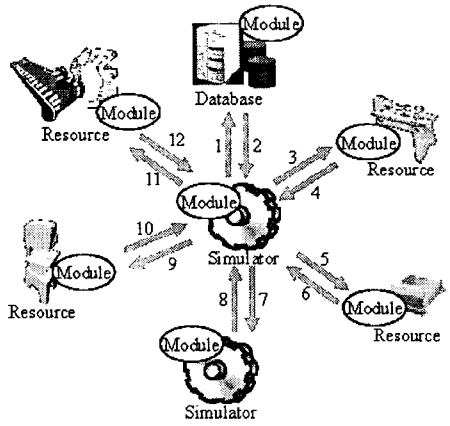
4.1. Mobile Agent의 이용

DIS의 경우 PDU에서 미리 정의하고 있는 제한된 data만을 이용하여 다른 simulation model의 internal behavior를 monitoring하거나, behavior에 영향을 미치는 attribute를 control할 수 있다. 따라서 manufacturing system과 같은 분야에서 분산된 resource들을 표현하고 그들간의 interaction을 통한 manufacturing activity를 simulation하기에는 한계가 있다. 또 full broadcast distribution approach를 이용하기 때문에 module의 수가 많을 경우 communication load의 증가도 문제가 된다. HLA의 경우 RTI 구조를 통해 다양한 data를 표현하고 교환할 수 있지만, RTI라는 공통 structure를 거쳐야 한다는 점에서 distributed system에서의 module간의 자율성 확보에 제한을 받게 된다. 기존의 distributed simulation architecture가 가지는 이러한 단점들을 극복하기 위한 방안으로써 본 연구에서는 mobile agent를 이용하는 collaborative simulation architecture를 제안한다.

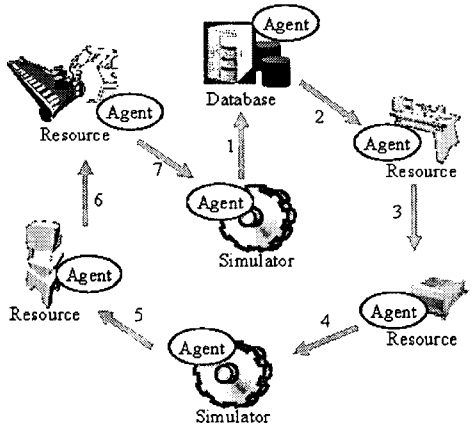
Software agent는 각자의 목적을 달성하기 위하여 자율적으로 행동하고 환경에 반응할 수 있는 computer program이다 [7]. Agent는 자율적으로 생성/작동/소멸하면서 다른 agent와의 협력을 통해 문제를 해결하고 목표를 달성해 나간다. Manufacturing system에서의 agent는 system에 포함된 자원에 연결되는 controller와 네트워크 상에 존재하는 software이다. Agent는 기존의 software component와는 다르게 autonomous, goal-oriented, flexible, self-starting, temporal continuity, communicative, adaptive, 그리고 mobile한 특성을 가질 수 있다. Software agent의 이러한 특징은 distributed system에서 요구되는 특징과 잘 부합할 뿐 아니라, computing power와 네트워킹 기술의 발전은 이의 구현을 가능하게 하고 있다.

특히 mobile agent는 실행 가능한 code를 가지고 네트워크 상에 존재하는 processor들 사이를 이동하는 software 개체로써, 그 개체가 이동을 제어하고 실행 상태를 유지한다. Manufacturing resource와 simulation model을 나타내고 제어할 수 있는 mobile agent를 distributed simulation에 도입함으로써 기존의 DIS나 HLA가 지니는 한계를 극복할 수 있다.

기존의 distributed simulation에서 사용되던 data communication을 mobile agent의 transferring으로 대체함으로써 simulation module과 resource가 많아지는 경우 발생하는 communication load를 줄일 수 있다. [그림 2]는 기존의 distributed simulation과 mobile agent를 이용한 distributed simulation에서 발생하는 communication의 방식을 비교한 것이다. Manufacturing system 내에 resource, simulation model, DB들이 분산되어 있는 환경에서 특정 simulation module 또는 simulator가 다른 resource module 또는 simulation module을 monitoring/control 하여 shop 전체의 simulation status를 update할 경우 발생하는 data transaction process를 나타낸다. 기존의 distributed simulation의 경우 simulation에 참여하는 다른 모든 module들과 data를 주고받는 과정을 거쳐야 하지만, mobile agent를 이용할 경우 agent가 직접 네트워크를 통해 다른 module의 site로 이동하여 그 모듈과 직접 communication을 하게 되므로 module들을 한번씩 거치기만 하면 된다. 결과적으로 message 기반으로 data를 주고받는 경우에 비하여 네트워크 상에 발생하는 traffic을 줄일 수 있고, 특히 그 효과는 distributed simulation system의 규모가 커 질수록 더욱 증가하게 된다.



(a) Traditional distributed simulation



(b) Mobile-agent-based distributed simulation

[그림 2] Distributed simulation에서의 communication 방식 비교

4.2. Agent Type과 Interaction

Mobile-agent-based Collaborative Simulation Architecture에서는 기본적으로 manufacturing resource, simulation model 등의 객체에 의해 각각을 표현하는 agent들이 나누어지지만 기능에 따라 몇 가지 type의 agent로 분류될 수 있다.

- Resource agent

Resource의 behavior를 나타내는 agent로 일종의 가상적인 resource controller이다. Resource의 behavior를 표현하는 resource model을 가지고 있으며, 실제의 resource와 integration을 통하여 shop을 control 할 수도 있다.

- Simulation agent

Shop의 sub-function을 simulation할 수 있는 model을 포함하고 있는 agent이다. Resource agent를 monitoring 또는 control하여 실제 simulation을 수행하고 결과를 낸다. 한 shop에는 한개 이상의 simulation agent가 존재한다.

- Interaction M-agent

Simulation agent가 resource agent와 실제 data를 교환하기 위하여 생성해 내는 mobile agent이다. Simulation agent의 monitoring 또는 control과 관련된 일부 function을 상속받아서 생성되어 네트워크 상을 이동하여 실제로 resource agent와 communication한다.

- Operation M-agent

Simulation manager가 user interface를 통하여 simulation을 operating할 수 있도록 user client에게 interface를 제공하는 mobile agent이다. Simulation agent가 자신의 일부 function을 상속시켜 생성해낸다.

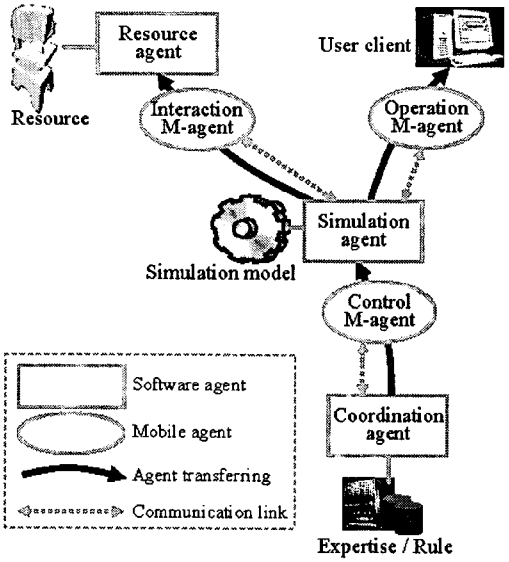
- Coordination agent

Manufacturing system에 존재하는 다수의 simulation agent로부터 받은 simulation 결과로부터 system의 performance를 평가할 뿐 아니라 performance를 증가시키기 위한 rule이나 expertise를 표현하는 model을 가지고 있다.

- Control M-agent

Coordination agent가 simulation agent와 실제 data를 교환하기 위하여 생성해 내는 mobile agent이다. Simulation agent로부터 simulation 결과를 받고 simulation agent의 attribute를 수정함으로써 simulation의 behavior를 조정할 수 있다.

[그림 3]은 앞에서 언급한 6 종류의 agent 간의 interaction을 나타낸다. Simulation agent와 resource agent, simulation agent와 coordination agent, 그리고 simulation agent와 user client 간의 communication은 네트워크를 통해 직접적으로 이루어지는 것이 아니라 각각 mobile agent인 interaction M-agent, Control M-agent, Operation M-agent를 이용하기 때문에 네트워크를 통하지 않고 직접 연결을 통해서 이루어진다. 결과적으로 standard communication protocol을 이용하지 않아도 되기 때문에 다른 module의 internal behavior에 대한 monitoring과 control을 구현하기 쉬워진다. 뿐만 아니라 simulation agent와 interaction M-agent, simulation agent와 operation M-agent, 그리고 Coordination agent와 Control M-agent간의 communication link는 네트워크에서 protocol을 통해 이루어지지만, 다른 외부 module이나 agent에 의해 이해될 필요 없는 communication이기 때문에

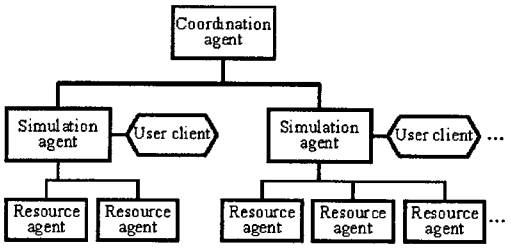


[그림 3] Agent interaction diagram

임의의 protocol을 이용해 구현 가능하다는 장점을 가진다.

4.3. Global Structure

[그림 4]는 다수의 분산된 simulation agent가 resource agent와 coordination agent와의 협업을 통해서 system 전체를 simulation하기위한 structure를 나타낸다. 앞에서 언급한 바와 같이

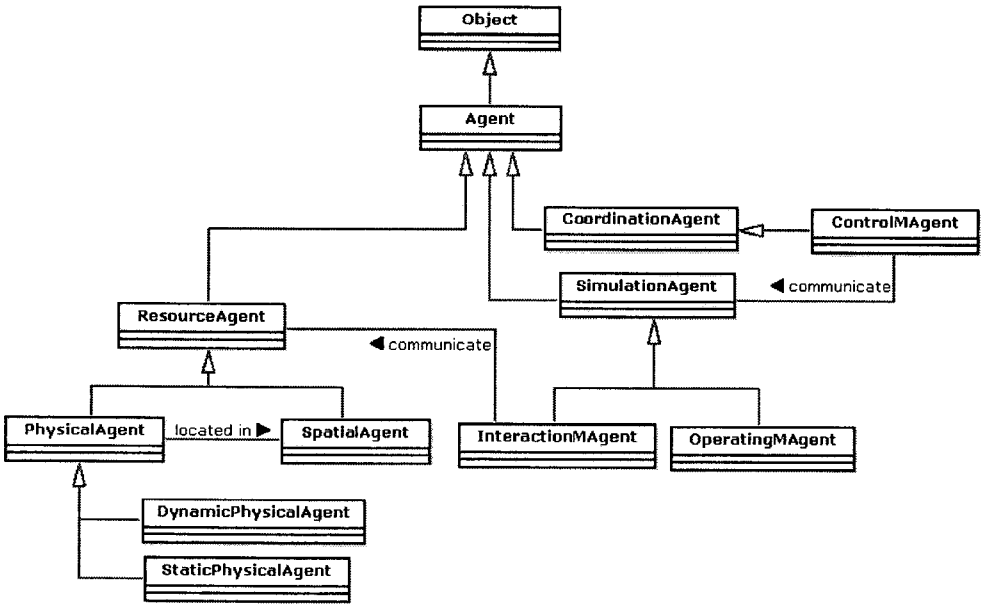


[그림 4] Global simulation structure

[그림 4]에 나타난 연결은 네트워크를 통해 직접적으로 이루어지는 것이 아니라 mobile agent를 통해 간접적으로 이루어진다. Simulation agent는 일반적으로 다수의 resource agent를 이용하여 simulation을 수행한다. FMS(Flexible Manufacturing System)를 예를 들면 하나의 simulation agent는 전형적으로 하나의 machine cell을 나타낸다. Simulation agent는 system 전체의 performance 조절이 필요할 경우에만 coordination agent에 의해 영향을 받으면서 각자 autonomous하게 simulation을 진행한다. 각자의 독립된 simulation operator는 user client를 통해 해당 부분의 simulation model을 monitoring 또는 control한다.

4.4. General Agent Classes

[그림 5]는 본 논문에서 다루고 있는 agent를 class diagram을 이용하여 관계를 나타낸 것이다. 종속된 하위 class에 해당하는 agent는 상위 class



[그림 5] Class diagram of agent

에 해당하는 agent로부터 상속받아 instance를 생성한다. Resource agent는 Ramat and Preux [8]의 agent class를 수용하여 수정하였다. Resource agent는 크게 physical agent와 spacial agent로 나누어지는데, physical agent는 manufacturing system상에 존재하는 resource, part, human등의 entity를 나타내고 spacial agent는 resource가 차지하는 공간을 나타낸다. Physical agent는 다시 part, AGV 등의 dynamic physical agent와 machine, conveyor 등의 static physical agent로 나뉜다.

5. 구현 방안

전형적으로 사용되는 네트워크인 TCP/IP 상에서 heterogeneous한 여러 platform에 분산되어있는 simulator 간의 정보 교환을 위해서는 효과적인 communication infrastructure의 구현이 필요하다. CORBA™(Common Object Request Broker Architecture), DCOM, Message exchange server, Java™ 등 다양한 기술을 응용 가능하다. 특히 platform independent한 특징과 web centric 한 특성으로 현재 각광받고 있는 Java™를 이용하는 것이 효과적일 것이다. 더욱이 Java™ 기반의 mobile agent platform도 현재 다수 개발되고 있는 실정이다.

한 processor site 내에서의 mobile agent와 software agent간의 직접적인 communication에는 4.2절에서 언급한 것과 같이 임의의 protocol을 이용 가능하다. 다양한 operation과 attribute를 효과적으로 서로 호출하고 사용하기 위해서는 공통의 ontology를 구축하고 ontology에 부합하는 message를 교환하는 것이 효과적일 것이다.

Simulation operator를 위한 user client의 구현은 Java™로 구현한 mobile agent를 가정 할 경우 Java™ applet과 Java™ servlet을 이용하여 web browser 기반으로 구현하는 것이 적합할 것이다.

6. 결론

본 연구에서는 distributed manufacturing system을 위한 mobile agent 기반의 distributed simulation architecture를 제시하였다. 본 연구에서 제안한 architecture에서는 mobile agent를 도입함으로써 기존의 distributed simulation architecture의 문제점이라 할 수 있는 제한된 control 과 monitoring 지원, 그리고 communication load 문제를 해결할 수 있는 방향을 제시하였다. 그리고 coordination agent를 비롯한 functional module을 이용하여 simulation performance control을 simulation 내에 포함시킬 수 있는 방향을 제시하였다.

본 연구에서 제시한 mobile-agent-based distributed simulation architecture를 실제 구현하고 시스템에 대한 성능평가를 하는 것은 추후 계속적으로 연구되어야 할 사항이다.

감사의 글

본 논문은 한국과학재단의 목적기초연구과제 (2001-1-31500-005-1) 와 교육인적자원부의 BK21과제 지원으로 수행되었습니다. 이에 감사드립니다.

참고문헌

[1] P. A. Fishwick, "Computer simulation : growth through extension," *Transactions of the Society for Computer Simulation International*, Vol. 14, pp. 13 - 23, 1997.

[2] A. M. Law, and W.D. Kelton, Simulation Modelling and Analysis, second ed., McGraw-Hill, New York, 1991.

[3] A. D. Baker, H. D. Parunak, and K. Erol, "Agents and The Internet: Infrastructure for Mass Customization," *IEEE Internet Computing*, Vol. 3, Issue 5, pp. 62-69, 1999.

[4] G. Habchi, and C. Berchet, "A model for manufacturing systems simulation with a control dimension," *Simulation modeling practice and theory*, Vol. 11, pp. 21-44, 2003.

[5] IEEE Standard for Distributed Interactive Simulation-Application Protocols, The Institute of Electrical and Electronics Engineers, 1998.

[6] IEEE Standard for Modeling and Simulation High Level Architecture(HLA) - Framework and Rules, The Institute of Electrical and Electronics Engineers, 2000.

[7] N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*. Vol.1, Issue 5, pp 7 - 38, 1998.

[8] E. Ramat, and P. Preux, "Virtual Laboratory Environment(VLE): a software environment oriented agent and object for modeling and simulation of complex systems," *Simulation modeling practice and theory*, Vol .11, pp. 45-55, 2003.