Crowd Federate Architecture and API Design

Frederic(Rick) D. McKenzie
Qingwen Xu
Quynh-Anh H. Nguyen
Mikel D. Petty
Virginia Modeling, Analysis and Simulation Center
Old Dominion University
Norfolk, VA. 23529
757.683.5590, 757.683.6367
fmckenzi@ece.odu.edu, mpetty@odu.edu

Keywords:
Crowd behavior, API design, game technology

ABSTRACT: *Our United States military is increasingly engaged in urban combat or peace-keeping missions. As a result, soldiers are also increasingly engaged with the civilian non-combatant inhabitants of various nations. Yet, current military simulation models have little or no representation of these effects which can lead to suboptimal training or experimentation results. More realistic and sophisticated crowd models are desired to address this growing need.*

It is unlikely that one crowd model will meet all our military's crowd requirements since models are needed with a variety of behaviors depending upon the type of mission, the size of encounters, and the user application. A federate that may be used to provide such variety of civilian behaviors in a crowd context would need to be flexible, configurable, and extensible. In this paper, we report about one such instantiation and the framework which supports it. The framework is a layered architecture that is composed of a physical layer in which movements and other actions of the crowd are manifested; and also a cognitive layer in which the motivations of these activities are generated and propagated. Connecting these two layers is an API layer that provides mapping and communication services for the stimuli, activities, and accompanying parameters that are crowd behavior centric.

Included in the paper are details about the API and the design process used to achieve it. Also included is a description of the integration of game technology used to provide the physical layer portion for this particular instantiation of the crowd federate.

1. INTRODUCTION

Crowds of non-combatants play a large and increasingly recognized role in modern military operations, and often create substantial difficulties for the combatant forces involved.

> "In Somalia, U. S. Marines often faced hostile crowds of rock-throwing women and children. In Bosnia, U. S. Army soldiers had to disperse angry mobs of Serb hard-liners near the town of Banja Luka. More recently, Danish, French, and Italian forces attempted to control riots between ethnic Albanians and Serbs in Mitrovice, Albania." [1]

> "All military operations, large or small, have a crowd control/crowd confusion factor. … [C]rowds are one of the worst situations you can encounter. There is mass confusion; loss of control and communication with subordinates; potential for shooting innocent civilians, or being shot at by hostiles in the crowd; potential for an incident at the tactical level to influence operations and policy at the strategic level." [2]

In spite of the military challenges and risks imposed by crowds, models of crowds are essentially absent from current production military simulations. This omission has been understandable in the context of legacy simulations that were historically focused on large-scale engagements between heavy mechanized forces in primarily non-urban settings. However, in the last decade

the threat has changed and future engagements are expected to often involve lighter forces in urban settings. In simulations of such scenarios the absence of crowds and of non-combatants in general is a more serious departure from realism. The absence of models of crowds in military simulation, and the need to include them, has been widely recognized.

> "Military forces are increasingly called upon to support operations other than war in which they come into contact with civilian populations. In some cases, the interaction takes place with crowds of civilians. Unfortunately, the computer generated forces that support virtual training systems do not yet support the simulation of crowds of civilians." [3]
>
> "Representations are needed for … (neutrals or civilians) to represent operations other than war and the interactions among these forces." [4].
>
> "[T]he ability to represent the behavior of crowds is currently lacking in military modeling and simulation …" [5]
>
> "With the Army's growing emphasis on low-intensity conflicts and operations other than war, the need to consider the civilians that live in the environment in which our forces will operate has become increasingly important. … [C]ivilian populations can have a profound affect in a crowded battle space. … There is, however, little representation of the civilians in today's military simulations." [6]

VMASC is engaged in a two-phase research project aimed at developing a crowd modeling capability for military simulation. The first phase, now complete, consisted of three parts: a requirements analysis to identify military simulation crowd modeling requirements, a literature survey to examine psychological research relevant to crowd modeling, and a design study to explore design issues in the implementation of a crowd simulation [7].

In the second phase, now well underway, we are developing a crowd simulation, implemented as a distributed simulation federate, that will be interoperable with existing military simulations and will have a credible psychological basis for the crowd behavior it generates [8]. The second phase of the project has seven interrelated tasks. They are:

1. *Crowd federate implementation*; design and development of a simulation that generates and controls crowd members, is interoperable with existing military simulations via HLA, and has a reconfigurable architecture to allow later replacement of its component models. This architecture is the subject of this paper.
2. *Cognitive model development*; acquisition of psychological information describing the behavior of crowds via both literature review and direct psychological research [9] [10], the development of a computational model of crowd member behavior based on the psychological information, and the integration of that model into the crowd federate.
3. *Requirements analysis continuation*; continuation of the process of identifying requirements for crowd modeling in military simulation.
4. *Historical survey*; study and analysis of historical incidents where crowds had a significant effect on the course or outcome of military engagements.
5. *Reference scenarios*; development of documented, historically accurate scenarios in a military simulation of historical events involving crowds, for testing and validation of the crowd federate [11]. This includes the development of geospecific terrain [12].
6. *Experiments*; conduct of two experiments planned to test the crowd federate, the first to examine the level of crowd behavior fidelity needed, and the second to test the architectural reconfigurability of the crowd federate.
7. *PMFserv evaluation*; independent evaluation of a psychological model based on performance moderator functions.

The relationships between the three tasks of Phase 1 and the seven tasks of Phase 2 are summarized in Figure 1.

2. DESIGN PROCESS AND GOALS

The design process started with an investigative software development process to initially create an exploratory prototype aimed at elaborating requirements of the Crowd Behavior application programming interface (API). The goals of developing this prototype were twofold. Firstly, we mitigate risk by investigating areas of the design that contain either much complexity or involve reuse of software which may be difficult to integrate. Specifically, this involves the reuse of game AI solutions from the entertainment industry and integrating them with traditional military distributed simulation solutions. Those risk areas involve software integration and terrain correlation. Secondly, growing familiarity with the capabilities of the commercial-off-the-shelf/government-off-the-shelf (COTS/GOTS) software and the theoretical necessities of crowd behaviors and their effect on military outcomes, allow iterative refinement of the requirements of the Crowd Behavior API.

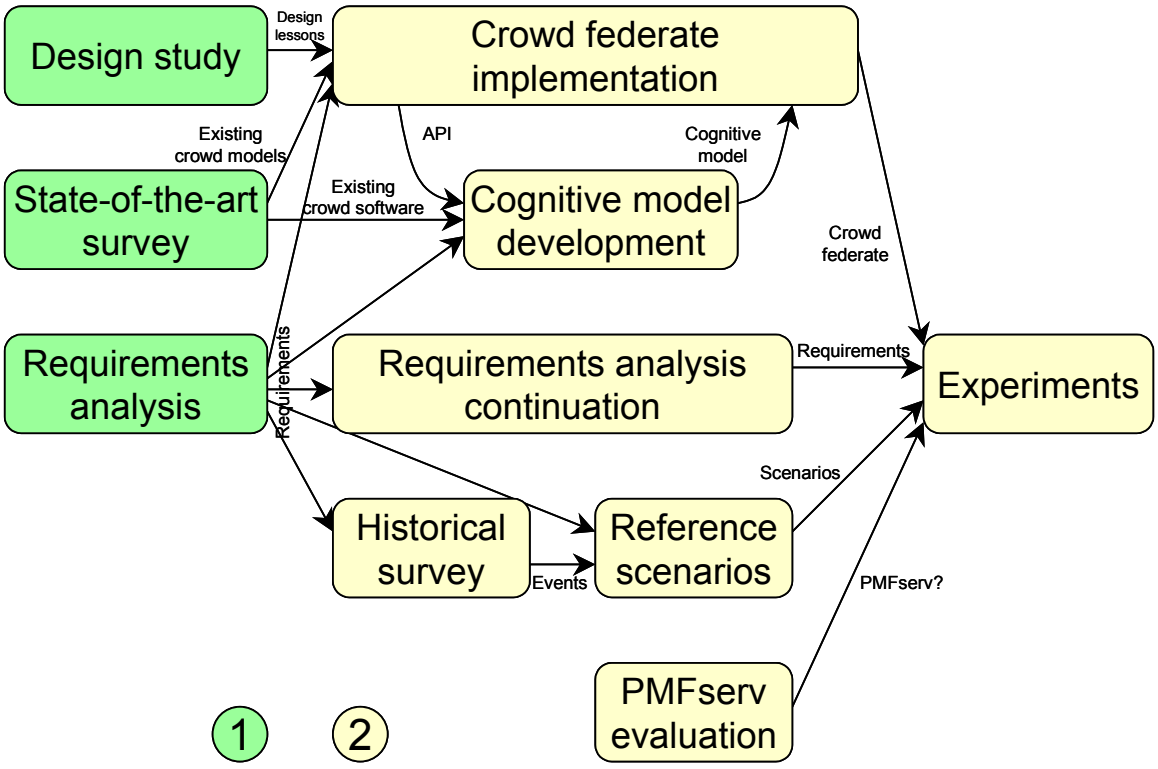

Figure 1. Crowd Modeling project tasks.

The exploratory prototype and refined requirements were then used to implement a functional prototype designed to support the Crowd Behavior API. This prototype can be expanded as additional capabilities are needed.

This process is in effect the beginnings of a spiral development effort tailored to the goals and objectives of this project. Subsequent cycles in the spiral would be carried out in follow-on crowd modeling efforts related to this project.

Upon conclusion of the design study the following design and implementation goals were achieved:

1. A prototype was implemented that demonstrated the integration of game AI software within a military-oriented distributed simulation architecture, such as HLA. This prototype resulted in a federation that combined crowd modeling capabilities from entertainment industry software with semi-automated forces modeling capability of JSAF.
2. An initial draft of the Crowd Behavior API was achieved to a level of detail specification that can be implemented as part of a follow-on phase to this project.
3. Crowd models with differing fidelity were created in order to determine the effects of the different fidelity crowds on the outcome of a military mission.
4. The team also developed and correlated terrain between the game technology and the military simulation. This process was evaluated and documented.

3. CROWD FEDERATE ARCHITECTURE

3.1 Architecture Background

Much research has been conducted on human behavior models that focus on individual autonomous human agents. Fewer models have addressed group or crowd behavior. Crowd behavior models in the literature range from purely physical representations to purely cognitive representations and combinations of both have been successful to varying degrees in the domain to which they were intended. Motivations range from improving graphical rendering performance to providing realistic behavior representations with emotions and other drivers. Crowd models include particle systems, flocking systems, or behavioral systems -- the difference being an increasing level of interaction among participants in the crowd and with the environment in general. Along with increased interaction comes increased attention to

modeling the social and emotional interactions among crowd members.

A recurring theme throughout the literature is the necessity to model behavior either at the individual level and allow crowd behavior to emerge as a result of many goal-oriented individuals within the same location or at a crowd level where the behavior of the crowd may be completely controlled by a crowd entity. Again, there is a range of efforts between these two extremes. Recent findings [13] indicate that there is empirical evidence that crowds are not wholly individuals nor wholly a single entity but rather a congregation of individuals and small groups from which crowd behavior emerges out of a process of assembling, temporarily interacting, and then dispersing. This suggests there is a hierarchy of behavior needed at the individual and group level and an awareness and affect of the evolution of the crowd over time. Such complexity could only arise from behavioral models that include aspects of cognition. Nevertheless, a variety of models do exist. Where realism and human decision-making based on simulation stimuli are important, models that focus on physical visual representation of crowds are important. Where motivations and consequences take precedence over visual realism, robust cognitive models are necessary. A framework that allows the integration of diverse models at the cognitive level with differing physical models would serve to provide useful crowd models to fit a variety of needs. In recognition of this variety, the Crowd Behavior API must be flexible enough to allow such mappings of differing philosophy.

We are using COTS tools such as AI.implant, Maya, and simulation engines from the gaming and entertainment industry that provide much of the physical and visual realistic representation. AI.implant is an autonomous character tool that calculates and updates the position and orientation of each entity, chooses the correct set of animation (locomotion) cycles, and enables the correct simulation logic. Maya is a graphical design and animation tool that provides the workspace for AI.implant and produces the animation clips that a game engine would use to visualize the simulation environment.

Military alternatives at the physical layer of crowd representation include DISAF [14]. Recently, DISAF began including a flocking-based representation of crowd behavior allowing the incorporation of standing around behavior and flocking to waypoints. In addition, crowd individuals have traits such as curiosity, fear, motivation, and hostility. The mental states achieved in individuals stimulate crowd behavior resulting in stay or flee.

More robust models of crowd cognitive behavior are being developed [15] [16] [17] [18] [19]. These models include crowd behaviors such as dispersing, gathering, swirling, clustering, flocking, safe wandering, following, goal changing, attraction to a location, repulsion from a location, group splitting, and space adaptability. Crowd evolution is also tracked so that one can determine whether the crowd is advancing, gathering, retreating, or dispersing. These models include environmental knowledge, group beliefs, intentions or goals, and also desires. Silverman builds a cognitive model based on Markov chains and a BDI (Belief, Desires, Intensions) model [15].

Musse [18] in particular discuss three ways of controlling crowd behavior: 1) scripts, 2) rules with events and reactions, and 3) real-time external control. An entity hierarchy is provided where the smallest unit is the virtual human agent. Next up in the hierarchy are groups that are composed of agents and then crowds that are composed of groups. “Crowd behavior corresponds to a set of actions applied according to entities’ intentions, beliefs, knowledge and perception” [18]. Different levels of realism are explored to support simple to complex crowd behaviors. Simple crowd behaviors would approximate crowd behavior based on flocking systems. Additionally, crowds may have a predominant emotion (sad, calm, regular, happy, or explosive which is extremely happy). These emotions may affect the posture and walk characteristics of individuals.

Our desire is to benefit from the variety of models by creating an architecture with an API specifically to support crowd behaviors in distributed simulations. This framework would allow the integration of robust cognitive models with models existing at the physical layer.

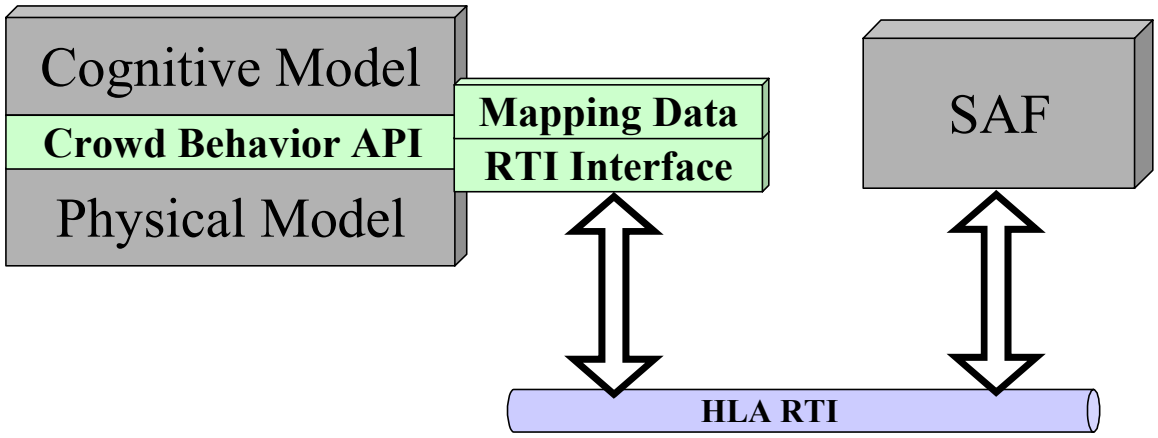

Figure 2: Generic crowd federate architecture

3.2 Architecture Approach

The Crowd Behavior API facilitates integration, communication, and interoperability of crowd cognitive models with crowd physical models. HLA compliance allows another level of interoperability that may be exploited. Physical models that are uniquely military oriented are immediately available with the use of the HLA. This combination of the Crowd Behavior API and HLA compliance allows for the development of a crowd federate architecture that may draw from the military as well as the entertainment industry to utilize the best of both worlds in carrying out realistic simulations of crowds in military scenarios.

The approach used in the design for the crowd federate is to start with a generalized API for an architecture that supports the reuse of existing cognitive models as well as existing physical models. The API is intended to facilitate control of the physical model by the cognitive model as well as event and state feedback from the physical model to the cognitive model. It is thought that the API would also provide the access to the entity and interaction data needed by the RTI Interface. Figure 2 illustrates a generic crowd federate architecture and shows the Crowd Behavior API as a layer between the cognitive model and the physical model. This API facilitates the exchange of information between the two models tempered by the reconfigurable mapping data. Information about the crowd is exchanged across the RTI with other federates such as a SAF simulation.

3.3 Architecture design details

Details of the crowd federate are most apparent in the Crowd Behavior API. In Figure 3, general designs for the cognitive model and the physical model are shown. These designs are intended to represent functionality that may be mapped to any given specific cognitive or physical model. This is important since the Crowd Behavior API will support such functionality and, therefore, specifics of the API contains the framework necessary to convey functionality and information given the necessary set of mappings.

Shown in the design of the general cognitive model are the two main interfaces with the physical model and thus the API. These are the components that include perception and action selection. The API contains mechanisms to provide sensory data to the cognitive model so that this data may be perceived in the required manner. The API must also contain mechanisms to provide command and control to the cognitive layer over the physical layer. The converse is true with regard to the physical layer.

Sensor stimuli must be provided to the cognitive layer and actions provided by the cognitive layer must have a means to be executed. All this must occur via the Crowd Behavior API.

The specifications of the interface have been derived and refined based upon these general cognitive and physical designs derived from the crowd modeling literature, the implemented prototype, and the experiments conducted. Specific design criteria for the interface include the flexibility of the API to handle behavior transference from an individual influencing group behaviors or a group influencing an individual's behavior. Also of interest are individual or group behaviors that are influenced by triggers or persistent phenomena. It is believed but not proven that such capability is needed, as these are unanswered questions in the psych community. Whether or not these capabilities are needed in military simulations are questions we may be able to answer in surveying the military simulations community. Our intent is to provide the flexibility in the API to support the changing psychology of crowd behavior.

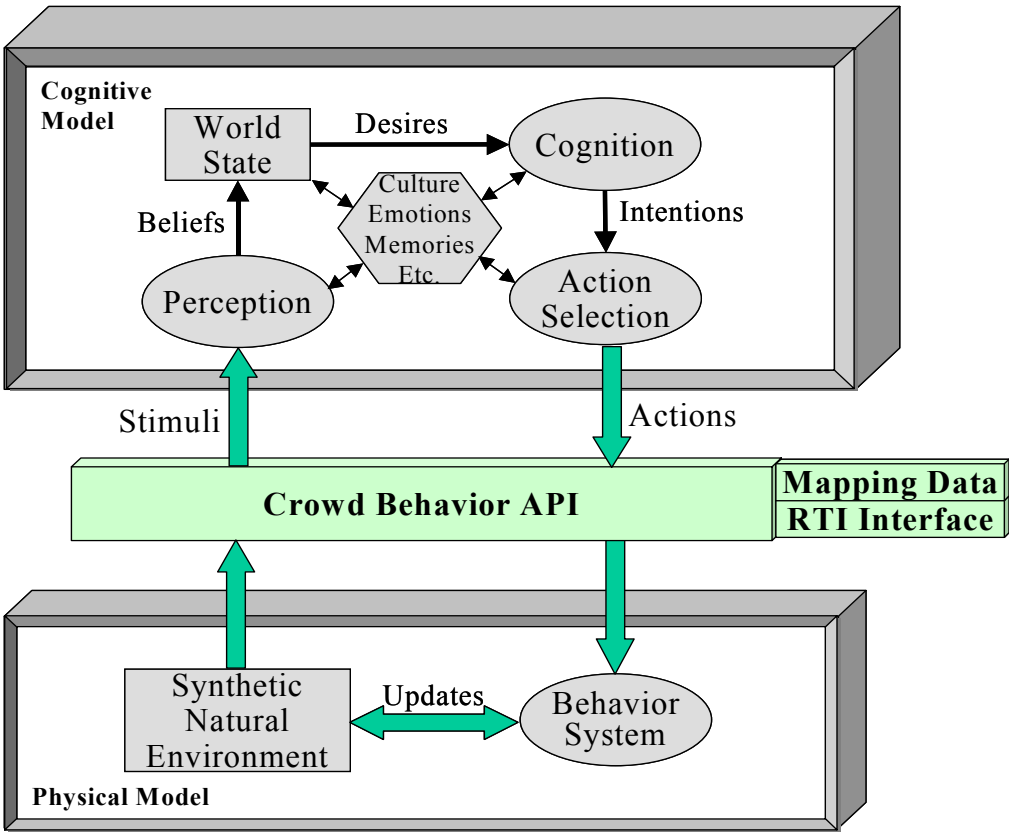

Figure 3: General cognitive and physical model designs

Attributes Category:
External states => position, orientation, speed …
Internal states => alive, anger, hungry …

Actions Category:
Observation: see, hear, feel …
Movement: stay, walk, run, flock with, follow path …
Interaction: throw rock, shoot, talk …
State change (callback): set attribute value

Figure 4. Example attributes and actions categories.

3.3.1 Human characters

Characters and groups of characters are defined by a set of attributes and set of actions to be carried out by these groups and the characters in the groups. These distinctions allow individual characters to carry out individualized actions while sharing actions designated to be group activities with members of its group. Figure 4 shows representative attributes and actions.

3.3.2 General command and control

The ability to perceive and affect the characters and their environment is provided through the execution of a predetermined set of crowd behavior actions that may be mapped to similar actions in the cognitive and physical models. It is believed that such a selection may be possible since this API is focused on the behaviors of

crowds and there are reasonable sets of activities that are realistic for crowds to perform. Further, mappings of semantically exact actions, although desirable, may not be a necessary condition in that semantic differences should be small enough to be insignificant in the randomness of what it is to be human and unconstrained within the context of civilian life. This is of course based on the axiom that civilians are significantly more non-deterministic than military units, and therefore there is a wider range of believable behavior for crowds and the individuals within them.

3.3.3 Character HLA interface

Mappings for exchange of information with other federates will depend upon the ability of the Federation Object Model (FOM) to support the necessary information. Initially, the Real-time Platform Reference Federation Object Model (RPR FOM) will be used to support interoperability with selected semi-automated forces (SAF) systems. It is believed that there will be a set of attributes and interactions that will not be supported by the RPR FOM nor any of the SAF systems. Such a set would remain part of the simulation object model (SOM) of the crowd federate unless and until the cognitive model portion of the crowd federate were to be executed on a separate node from the physical model portion of the crowd federate to in fact create a distributed crowd federate. In such a case, the RPR FOM would need to be augmented to support transfer of the extraneous set of data. The HLA interface therefore is a mapping dependent upon the FOM used.

4. USER REQUIREMENTS ANALYSIS

VMASC consulted with M&S users in the joint community (and others) regarding their current and anticipated needs for crowd modeling in joint simulations. Figure 6 shows the results of an effort to group requirements suggesting similar type behavior in crowds.

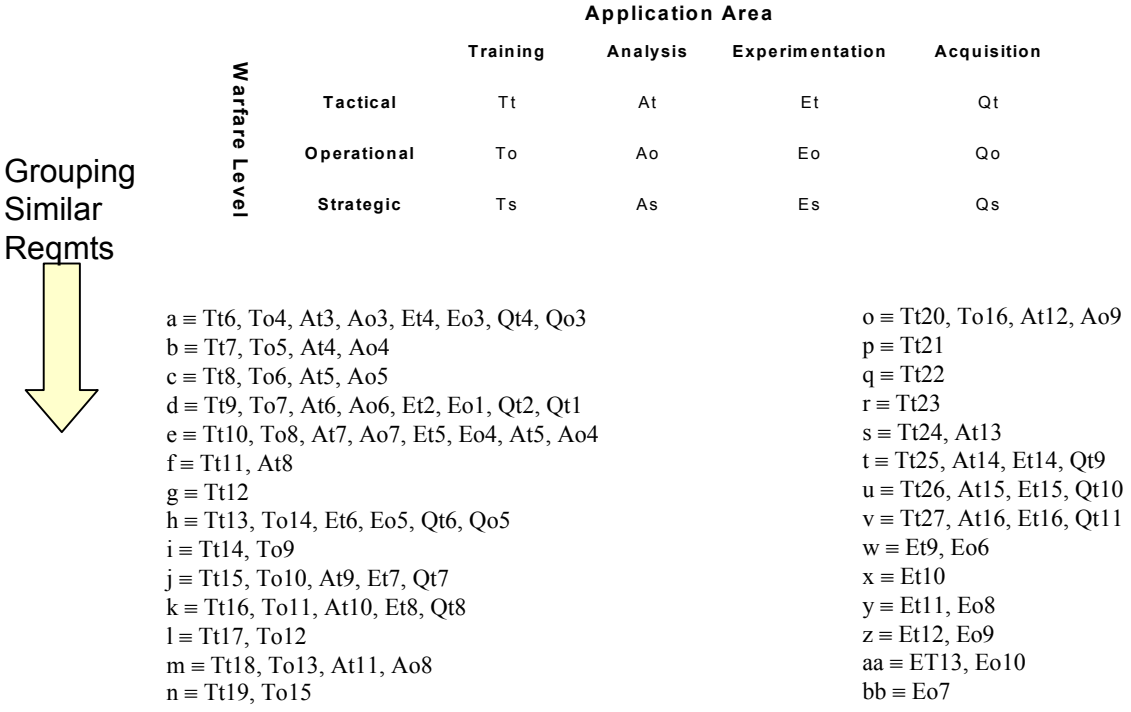

Figure 6: Grouping of user requirements

These groupings were further categorized as to whether the associated behaviors were generic or specific and whether they were pertaining to the action of an individual or groups of individuals. The results of these groupings are shown in Figure 7.

These categorized groupings were then examined for API specific requirements ignoring effects that are more appropriate for consideration in the physical and cognitive models. The result was a small set of generic use cases that expressed the tasks that the crowd behavior API would perform. Those use cases are shown in Figure 8.

- Generic Individual behavior
  - Stay at point (p), Move w/ neighbors (p), Drive vehicles (h), Employ crude or improvised weapons (h), etc.
- Specific Individual behaviors
  - Satisfy curiosity [goal] (q), Confront antagonist [behavior] (q), Assist combatants (e.g. medical aid) (f), etc.
- Generic group behavior
  - Flocking (s)
    * Move w/ neighbors (p)
  - Looting (g)
  - Etc.
- Specific group behavior
  - Comply w/or violate cordons and curfews [behavior](y), Mass action to overwhelm checkpoint [goal](g)
  - Attack barrier [behavior] (h)
  - Mass action to force combatant withdrawal (g) [goal]
    * Take hostile action against combatants [behavior] (e.g. throw rocks [action]) (f)
  - Provide human shield cover for combatants [goal] (g)
  - Etc.
- Other
  - Usability, Characteristics, Scenarios, Operational Needs, Considerations

Figure 7: Categorized Groupings

- Generic Set of Use Cases For API
  - Provide Action
    * The action may be simple or complex with a parameter that is the name of an action
  - Provide Group Action
    * Group entity may be implemented as a list of individuals or an actual group object.
  - Receive Sensor Information
  - Group Goal Change
  - Role Player Controlled Group
  - Receive State Information

Figure 8: Generic Crowd Behavior API use cases

5. API CONCEPTS AND MODELS

A *simulation entity* represents an individual or a group of individual that simulated by a set of computational models. The *entity type* of a simulation entity defines a set of *actions* and *stimulus* that the simulation entity may have. Actions and Stimulus are in the form of *requests* that will be sent back and forth between the computational modes.

A physical model is a computational model that carries out low-level realistic locomotion actions of a simulation entity. The results of a locomotion action will update the position and orientation of the simulation entity. The locomotion actions are selected and controlled by a cognitive model of the simulation entity. A physical model also provides sensing information (stimulus) to its cognitive model for processing.

A cognitive model performs cognitive tasks and produces actions than can be compared to data from human performance. It is intended to be an explanation of how some aspect of cognition is accomplished by a set of primitive computational processes. The primary computational processes include perception, knowledge representation, decision, planning, learning, action selection etc. Generally speaking, a cognitive model

receives stimulus that provided by it corresponding physical model and produces actions that will be carried out by the physical model.

Crowd behavior API provides a flexible request exchanging mechanism between physical models and cognitive modes. Typically, we can implement a separate program called API executive for forwarding requests between physical models and cognitive models. The API executive contains all the simulation entity information (entity type, identity and owners—simulation models, etc) required to forwarding requests. Alternatively, The API can be realized as software components integrated directly into physical models and cognitive models.

5.1 Crowd Behavior API Classes

Figure 9 shows a high level logical view of crowd behavior API classes. A brief description of each of these classes is provided below.

The **Crowd_API** class is the main interaction point between the API and other computational models. It implements two general API interfaces. The control interface (ICTL_API) provides initialization and simulation run time control functionalities. The model interface (IMOD_API) provides model registration, type and entity creation and request forwarding functionalities.

A Crowd_API object contains a type system in the simulation. The type system will be created during API initialization. Particularly, all simulation entity types are defined in the type system. Any simulation entity will have a corresponding entity type, which specifies the actions and stimulus in the simulation entity.

The **API_Type** class defines the entity, action and stimuli types in the simulation world. It provides the common information for each type, such as name, category etc. An entity type keeps track of all the simulation entities in the simulation that have this type. Typically, it will contain a list of action types, a list of stimuli types, and a list of API_Obj objects. Action and stimuli types give the format information of the "real" action and stimuli in the simulation.

The types in the system is specified in a CBOM (Crowd Behavior Object Model) file and created by a TypeFactory object.

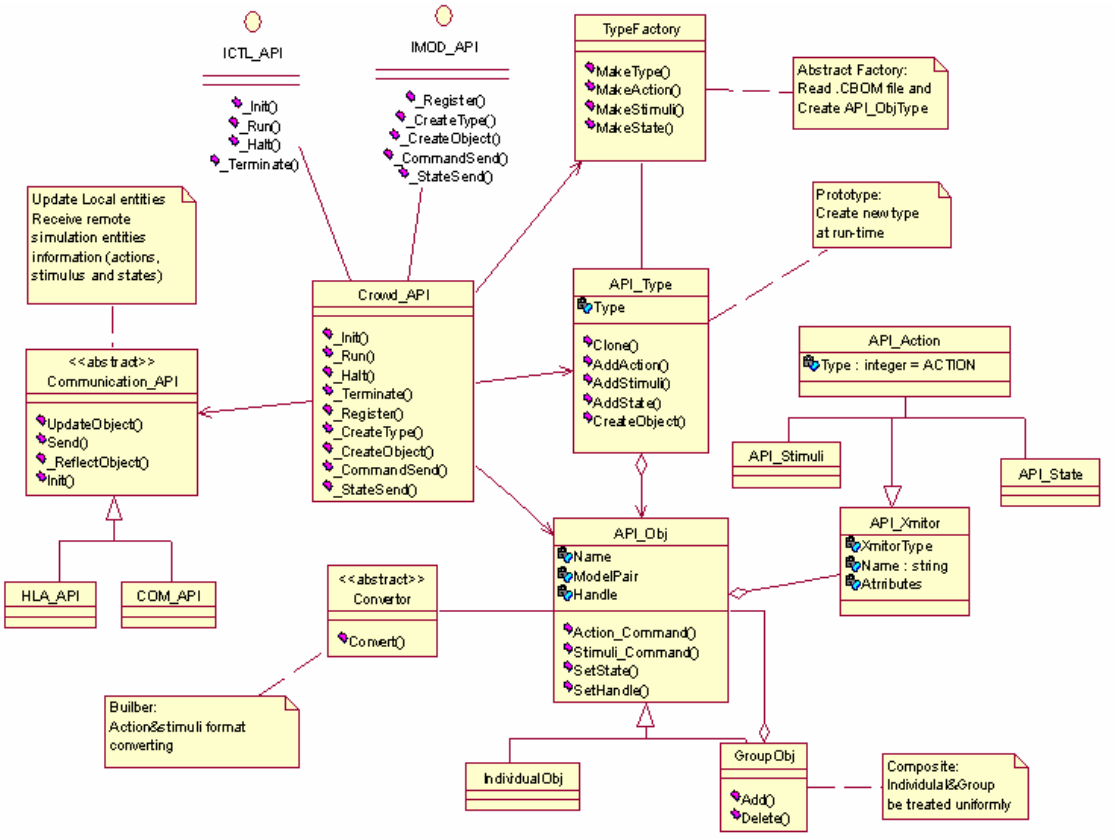

Figure 9: Logic view of Crowd Behavior API

A TypeFactory object reads the CBOM file and creates an API_Type object for each type defined in the file. The TypeFactory class realizes the Abstract Factory pattern [20] that has the following advantages:

1). Supports multiple type standards.
2). Isolates type creation from type specification.
3). Enforces dependencies between actions, stimulus in a type. However, it
4) doesn't support run-time type creation.

An entity type should be able to extend itself to form a new type at run time. The new type may be only slightly different from the original. However, it is impossible to define new type classes for every type extension in the CBOM file. To support a type extension capability, the API_Type class realizes the Prototype pattern [20] which supports run-time API_Type creation and reduces the number of classes/subclasses in the system.

The **API_Obj** class is an abstract concept class that represents the simulation entity in the API. It provides general information of the simulation entity for the API executive. An API_Obj object contains information such as name, type, and handle of a simulation entity. It also contains the information about the computational models that simulates it. A simulation entity could be an individual or a group of individual. To support this hierarchical structure of the simulation entity, API_Obj class realizes the Composite pattern [20] which presents part-whole hierarchies of the entities and provides a uniform interface to individual and group classes

The **API_Xmitor** class is an abstract class that represents the simulation entity requests (stimulus, actions etc). A concrete API_Xmitor class defines a request that can be sent from one model to another model in the simulation. It provides detail information of the exchanging request between physical models and cognitive models. The action/stimuli request is encapsulated in **API_Xmitor** class to support parameterized request objects and allow the physical model/cognitive model be structured around high-level operations.

If the simulation allows different models to use their own request format, request converting has to take place before the request is sent or after it is received. The **Converter** class is a utility class that provides request format converting between different computational models and the API executive. It realizes the Builder pattern which provides an independent algorithm for converting actions or stimulus and supports different representations for actions or stimuli.

The **Communication_API** class is an abstract communication class that implements communication mechanisms. Currently, it is a placeholder that indicates that the API component will have cross-machine (or cross-process) communication capabilities.

5.2 Physical and cognitive model classes

Figure 10 and 11 illustrate logical views of a physical model and a cognitive model, respectively. To be integrated with the API component, computational models need to provide an interface (port) that receives the requests and control commands from the API executive. The received control commands will be executed and the received requests will be sent to a particular physical simulation entity (the receiver) for processing. The interfaces for our prototype computational models are the ICM interface for the cognitive model and the IPM interface for the physical model shown in the figures.

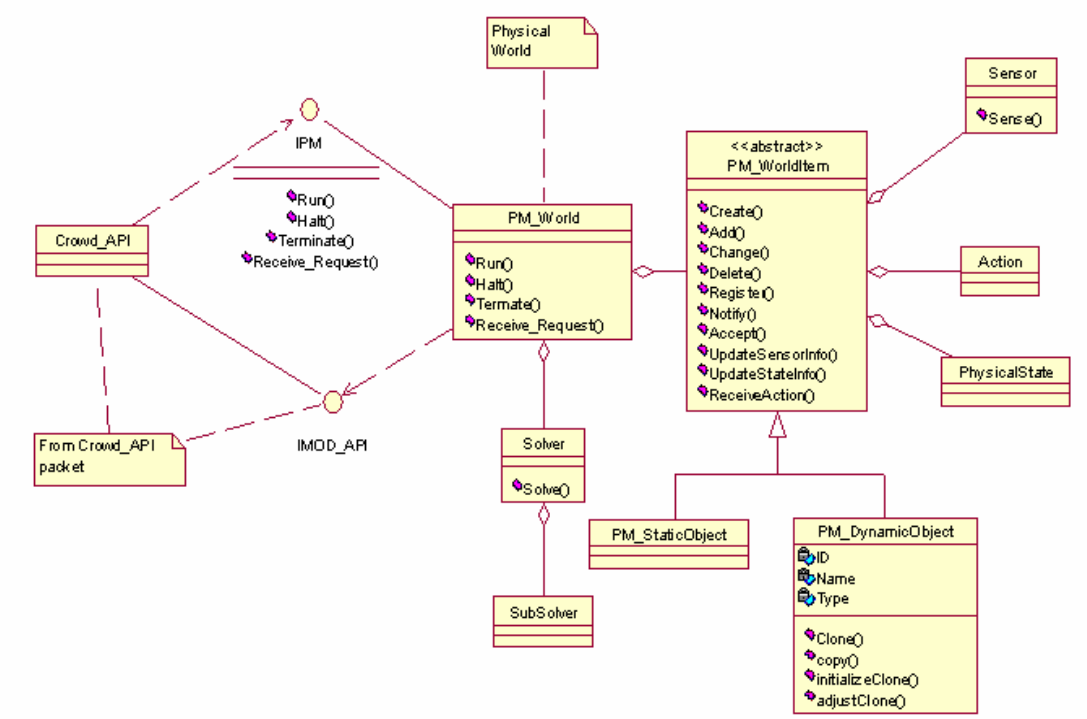

Figure 10: Logic view of a physical model

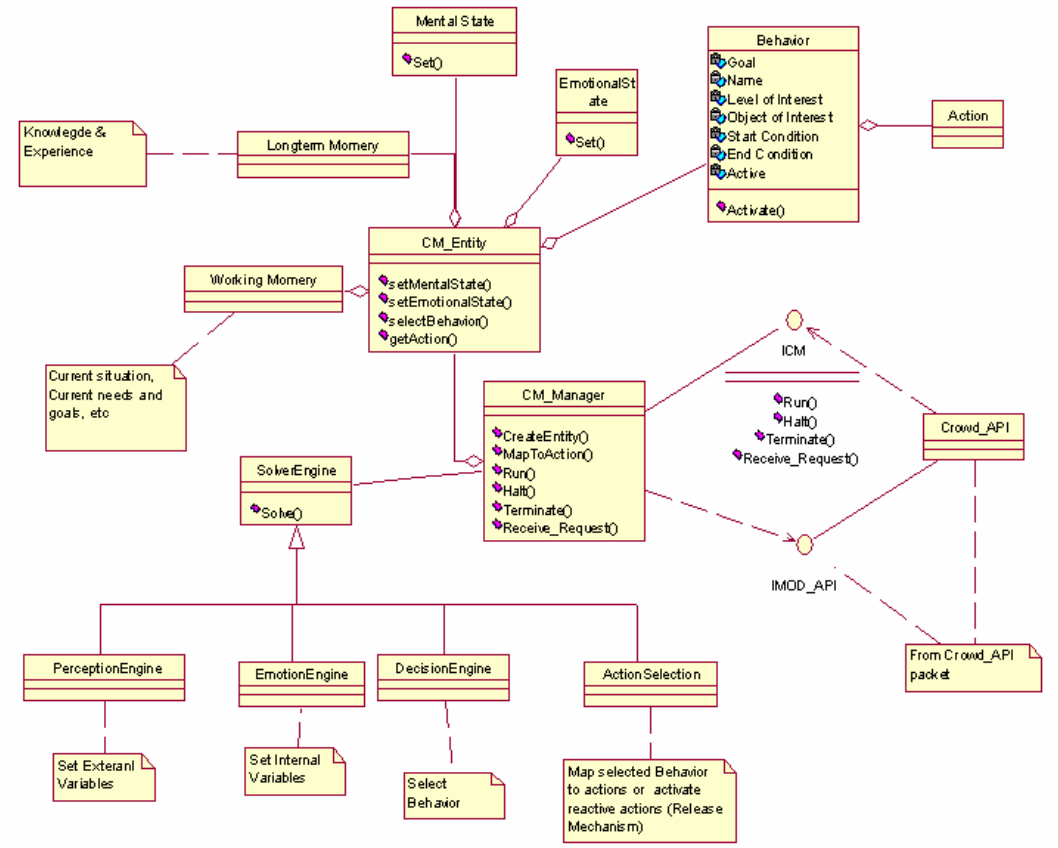

Figure 11: Logic view of a cognitive model

5.3 API Use Cases

Figure 12 shows the core use cases, shown as ovals, for the crowd behavior API. These use cases map to the general use cases from figure 8 with the addition of other use cases for control and initialization. The Provide Action and Provide Group action map to Provide Action Receive Action use cases in Figure 12. Similarly, Group Goal Change and Receive State Information from Figure 8 map to State Forwarding. Receive Sensor Information maps to Receive Stimuli and Provide Stimuli. Finally, Role-Player Controlled Group results in a GUI that interfaces in the same manner as the cognitive model and is excluded from the figure to reduce its complexity.

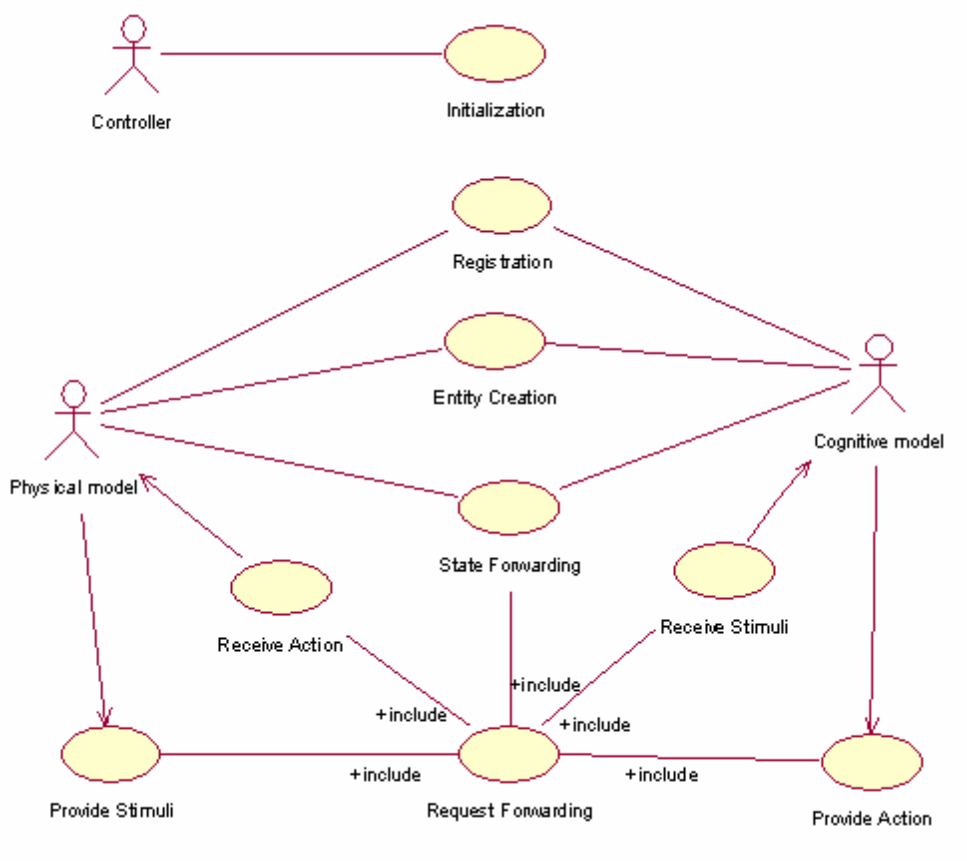

Figure 12: Crowd Behavior use cases

6. CONCLUSIONS

It is unlikely that one crowd model will meet all our military's crowd requirements since models are needed with a variety of behaviors depending upon the type of mission, the size of encounters, and the user application. A federate that may be used to provide such variety of civilian behaviors in a crowd context would need to be flexible, configurable, and extensible.

We discussed and illustrated a design that attempts to provide such capabilities. The framework is a layered architecture that is composed of a physical layer in which movements and other actions of the crowd are manifested; and also a cognitive layer in which the motivations of these activities are generated and propagated. The crowd behavior API connecting these two layers provides mapping and communication services for the stimuli, activities, and accompanying parameters that are crowd behavior centric.

Currently, the Crowd Behavior API has been implemented in a working prototype that recreates a vignette from the Blackhawk Down incident in Mogadishu, Somalia involving civilian, militia, and US military entities. The extent to which the federate is flexible and configurable is a task that has only recently been started and will be reported upon in future publications.

7. REFERENCES

[1] J. M. Kenny and W. L. Gilpin, "Insertion of Crowd Behavior Models into the INIWIC Course", *Proceedings of the 2002 Interservice/Industry Training, Simulation, and Education Conference*, Orlando FL, December 2-5 2002, pp. 1064-1074.

[2] M. Ferguson, *Questionnaire response*, May 8 2003.

[3] D. A. Reece, "Crowd Modeling in DISAF", *Proceedings of the Eleventh Conference on Computer-Generated Forces and Behavior Representation*, Orlando FL, May 7-9 2002, pp. 87-95.

[4] R. W. Pew et al, *Modeling Human and Organizational Behavior: Application to Military Simulation*, National Research Council, National Academy Press, Washington D.C., 1998.

[5] Department of the Air Force, *Program Research & Development Announcement NR 03-01-HE*, Online at http://www.eps.gov/spg/USAF/AFMC/AFRLWRS/PRDA-03-01-HE/listing.html.

[6] M. A. Fields and G. Spradlin, "Modeling Civilian Crowds in a Battlefield Simulation", *Proceedings of the Ninth Conference on Computer Generated Forces and Behavioral Representation*, Orlando FL, May 16-18 2000, pp. 451-458.

[7] M. D. Petty, F. D. McKenzie, and R. C. Gaskins, "Requirements, Psychological Models, and Design Issues in Crowd Modeling for Military Simulation", *Proceedings of the Huntsville Simulation Conference 2003*, Huntsville AL, October 29-31 2003.

[8] M. D. Petty, F. D. McKenzie, R. C. Gaskins, and E. W. Weisel, "Developing a Crowd Federate for Military Simulation", *Proceedings of the Spring 2004 Simulation Interoperability Workshop*, Arlington VA, April 18-23 2004, pp. 483-493.

[9] C. M. Boone, R. C. Gaskins, and M. D. Petty, "Observations of Crowd Behavior", *Proceedings of the 2004 Conference on Behavior Representation in Modeling and Simulation*, Arlington VA, May 17-20 2004, pp. 397-398.

[10] R. C. Gaskins, C. M. Boone, T. M. Verna, J. P. Bliss, and M. D. Petty, "Psychological Research for Crowd Modeling", *Proceedings of the 2004 Conference on Behavior Representation in Modeling and Simulation*, Arlington VA, May 17-20 2004, pp. 401-402.

[11] E. W. Weisel and M. D. Petty, "Reference Scenarios for Crowd Federate Validation", *Proceedings of the Fall 2004 Simulation Interoperability Workshop*, Orlando FL, September 19-24 2004.

[12] F. D. McKenzie, H. Garcia, Q. H. Nguyen, J. Seevinck, and M. D. Petty, "Mogadishu Terrain Generation and Correlation for Crowd Modeling", *Proceedings of the Spring 2004 Simulation Interoperability Workshop*, Arlington VA, April 18-23 2004, pp. 944-950.

[13] J. M. Kenny, C. McPhail, D. N Farrer, D. Odenthal, S. Heal, J. Taylor, S. Ijames, and P. Waddington, *Crowd Behavior, Crowd Control, and the Use of Non-Lethal Weapons*, Technical Report, Penn State Applied Research Laboratory, January 1 2001.

[14] D. A. Reece, "Crowd Modeling in DISAF", *Proceedings of the Eleventh Conference on Computer-Generated Forces and Behavior Representation*, Orlando FL, May 7-9 2002, pp. 87-95.

[15] B. G. Silverman, M. Johns, K. O'Brien, R. Weaver, and J. B. Cornwell, "Constructing Virtual Asymmetric Opponents from Data and Models in the Literature: Case of Crowd Rioting", *Proceedings of the Eleventh Conference on Computer-Generated Forces and Behavior Representation*, Orlando FL, May 7-9 2002, pp. 97-106.

[16] J. B. Cornwell, B. G. Silverman, K. O'Brien, and M. Johns, "A Demonstration of the PMF-Extraction Approach: Modeling the Effects of Sound on Crowd Behavior", *Proceedings of the Eleventh Conference on Computer-Generated Forces and Behavior Representation*, Orlando FL, May 7-9 2002, pp. 107-113.

[17] S. R. Musse, C. Babski , T. Capin, and D. Thalmann, "Crowd modelling in collaborative virtual environments", *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, Taipei Taiwan, November 1998, pp. 115-123.

[18] S. R. Musse and D. Thalmann, "Hierarchical model for real time simulation of virtual human crowds", *IEEE Transactions on Visualization and Computer Graphics*, Vol. 7, No. 2, April-June 2001, pp. 152-164.

[19] J. Allbeck, K. Kipper, C. Adams, W. Schuler, E. Zoubanova, N. Badler, M. Palmer, and A. Joshi, "ACUMEN: amplifying control and understanding of multiple entities", *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, Bologna Italy, July 15-29 2002.

[20] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Pub Co. 1995. ISBN 0-201-63361-2.

8. ACKNOWLEDGEMENT

This research described in this paper is sponsored by the Defense Modeling and Simulation Office and managed by the Air Force Research Laboratory. That support is gratefully acknowledged.

9. AUTHOR'S BIOGRAPHIES

Frederic (Rick) D. McKenzie, Ph.D. is an Assistant Professor of Electrical and Computer Engineering at Old Dominion University where he currently serves as Principal Investigator (PI) and Co-PI on projects involving medical modeling and simulation, behavior representation in simulations, and simulation architectures. Prior to joining ODU, he held a senior scientist position at Science Applications International Corporation (SAIC), serving as Principal Investigator for several distributed simulation projects. At SAIC he was a Team Lead on a large distributed simulation system. He has over 10 years of research and development experience in the software and artificial intelligence fields, including object-oriented design and knowledge-based systems. Both his M.S. and Ph.D. work have been in artificial intelligence – focusing on knowledge representation and model-based diagnostic reasoning.

Qingwen Xu is a Ph.D. student in Computer Science at Old Dominion University and a Research Assistant at the Virginia Modeling, Analysis and Simulation Center. He received a M.S. degree in Computer Science from Wake Forest University in 1999 and a B.S. degree in Auditing from Wuhan Unversity in 1991. His dissertation research involves self-organizing wireless sensor networks.

Quynh-Anh (Mimi) H. Nguyen is a Ph.D. student in the Modeling and Simulation (M&S) program at Old Dominion University and a Research Assistant at the Virginia Modeling, Analysis and Simulation Center (VMASC). She received a M.S. degree in M&S from Old Dominion University in 2003 and a B.S. degree in Electrical Engineering from George Mason University.

Mikel D. Petty is Chief Scientist of the Virginia Modeling, Analysis and Simulation Center at Old Dominion University. He received a Ph.D. in Computer Science from the University of Central Florida in 1997. Dr. Petty has worked in modeling and simulation research and development since 1990 in areas that include simulation interoperability, computer generated forces, multi-resolution simulation, and applications of theory to simulation. He has published over 110 research papers and has been awarded over 50 research contracts. He has served on a National Research Council committee on modeling and simulation and is currently an editor of the journals *SIMULATION: Transactions of the Society for Modeling and Simulation International* and *Journal of Defense Modeling and Simulation*.