A Reusable Behavior Modeling Method Based on Atom Action and Atom Condition

Yingqian Bao
College of Information System and Management, National University of Defense Technology
137 Yanwachi, Changsha
China
baoyingqian15@nudt.edu.cn

Qingjun Qu
College of Information System and Management, National University of Defense Technology
137 Yanwachi, Changsha
China
qjqu2012@163.com

Yiping Yao
College of Information System and Management, National University of Defense Technology
137 Yanwachi, Changsha
China
ypyao@nudt.edu.cn

Jiawei Fei
College of Information System and Management, National University of Defense Technology
137 Yanwachi, Changsha
China
feijiawei11@nudt.edu.cn

Miao Zhang
College of Information System and Management, National University of Defense Technology
137 Yanwachi, Changsha
China
zhangmiao15@nudt.edu.cn

ABSTRACT

Computer Generated Forces (CGF) behavior modeling technology is the core of the CGF modeling technology, it can make the CGF entities show a certain degree of human behavior characteristics, with varying degrees of autonomy, reasoning ability and adaptability. However, the current CGF behavior modeling methods are difficult to reflect the reusability of the behavior model, result in the workload of behavior modeling become very heavy. To solve this problem, this paper proposes a reusable behavior modeling method based on atom action and atom condition: by means of combining atom actions into the State, connecting the States by atom condition groups to form Doctrine, and designing XML (Extensible Markup Language) template file for State and Doctrine, we can realize the modeling, representation and storage of CGF behavior. Research shows that this method has good reusability, composability and extendibility, which can reduce the workload of behavior modeling.

CCS CONCEPTS

• **Computing methodologies → Modeling methodologies;**

SIMUTOOLS '17, September 11–13, 2017, Hong Kong, China

ACM ISBN 978-1-4503-6388-4/17/09…$15.00
https://doi.org/10.1145/3173519.3173539

KEYWORDS

CGF, behavior modeling, reusability, atom action, atom condition

1 INTRODUCTION

CGF refers to the simulation entity generated by computers in the simulation system, mainly include weapons and personnel. It can respond to the events and states in the virtual battlefield environment according to simulating the combat behavior of human [1]. CGF modeling is composed of environment modeling, physic modeling and behavior modeling. Among these modeling technologies, CGF behavior modeling technology is the core of CGF modeling technology, for this technology is able to make the simulation entity to show a certain degree of human behavior, with different degrees of autonomy, reasoning ability and adaptability. Ultimately the simulation entity is able to determine its own action function all by itself, which is also the difference between CGF and other forces in a simulation system.

The current popular CGF behavior modeling methods are based primarily on the following key technologies, such as FSM (Finite State Machine), rule set, agent and multi-agent, cybernetics etc. Although these behavior modeling methods have been applied in a large number of simulation applications, they all have their own limitations, especially in terms of reusability. Aiming at this problem, this paper proposes a reusable behavior modeling method based on atom action and atom condition.

The rest of this paper is organized as follows: in section 2, the current popular CGF behavior modeling methods is introduced. The behavior modeling method proposed by us is based on component-based modeling technology, so the section 3 introduced the component–based model's family framework. The paper introduces some basic concepts, they are atom action, atom

condition, State, Doctrine, and XML template files of State and Doctrine refer to the behavior modeling method in detail in section 4. The Section 5 gives an example to show how to implement the behavior modeling of a doctrine belongs to a bomber named "Bombing Enemy Radar Base". In section 6, the paper makes a conclusion and looks forward to our future work.

2 RELATED WORKS

As a hot spot and difficulty of CGF modeling technology, CGF behavior modeling technology owns a lot of research findings.

2.1 Behavior Modeling Method Based on FSM

According to tracking the current state and state transition condition of CGF entity (Entity for short), the behavior modeling method based on FSM [2] can determine what to do in next step. Currently, ModSAF, CCTTSAF and some other mature CGF systems have adopted this method. However, this method is difficult to meet the increasing demand of model fidelity. On the one hand, too few states cannot effectively describe the complex behavior. On the other hand, too much state will lead to a sharp increase in modeling workload. Therefore, FSM-based behavior modeling method is difficult to reflect the reusability of behavior models.

2.2 Behavior Modeling Method Based on Rule Set

The behavior modeling method based on rule set [3] has advantages in aspect of expressing knowledge, dealing with abrupt situation, and reducing the workload of modeling. And this method can be used to realize the behavior modeling of Entity of platform level and aggregation level. In the new generation of OneSAF, CCTTSAF and some other CGF systems, this method has been widely applied. But this method has some problems, for example lacking of intelligence and poor reusability.

2.3 Behavior Modeling Method Based on Agent and Muti-Agent

The behavior modeling method based on Agent [4,5] and Multi-Agent [6,7] can make the Entity to make independent decisions, and achieve different levels of reasoning ability, reactivity and autonomy. This method has been widely used in WARSIM2000. However, there is no unified concept and definition, as well as clear theoretical system for Agent. In addition, this method has poor reusability.

2.4 Behavior Modeling Method Based on Cybernetic

The behavior modeling method based on cybernetics [8] regards the behavior of the Entity as a system, and researches its dynamic characteristics from three aspects: input, output and control, which can reflect the interaction between Entity behavior and external environment. However, this method cannot reflect reusability of systems.

In general, the current CGF behavior modeling methods lack a support framework for reusability.

3 COMPONENT-BASED MODEL ARCHITECTURE FRAMEWORK

Component-based modeling technology has been the focus in the domain of modeling and simulation. Modeling framework based on component has good reusability, decoupling, extendibility and composability, which can greatly improve the development efficiency of the modeling. Thus the behavior modeling method proposed in this paper is based on this modeling framework as well. Before introducing the behavior modeling method, we firstly introduce the model's family framework based on component, as shown in Fig. 1.

This modeling framework consists of the following categories of function components:

Platform component: any Entity requires a platform component as a support, so as to give the Entity external feature and behavior attributes, so that the Entity can has physical meaning.

Motion component: used to simulate the motion of an Entity.

Target selection component: used to simulate the process that an Entity selects a variety of targets.

Sensor component: used to simulate the process that an Entity detects another Entity.

Electronic countermeasure component: used to simulate the interference of electromagnetic energy to the signal received by the sensor or the communication equipment.

Communication component: used to simulate the process that an Entity sends messages to another Entity.

Weapon component: used to simulate the management and launch of ammunition loaded on the Entity.

Damage component: used to simulate the calculation of physical damage effect when the Entity is hit by certain weapons.

By ordering the above function components, we can complete the physical modeling of CGF. Next step, on the basis of the model's family framework, we will complete the design of behavior modeling method.

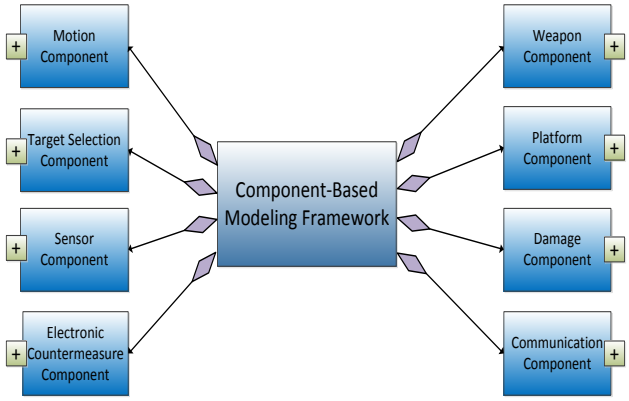

Figure 1: Component-Based Model Architecture Framework

4 Behavior Modeling Method BASED ON ATOM ACTION AND ATOM CONDITION

The behavior of an Entity is a set of actions that are organized by a variety of actions according to certain relations. On the basis of the modeling thought as shown in Fig.2, we assume there are some the most basic actions and conditions. All the behaviors of an Entity can be represented by these basic actions and conditions through the combination and connection. In the light of this idea, we first give the definition of atom action, atom condition, State and Doctrine. And then we can give the XML template file for the State and Doctrine.

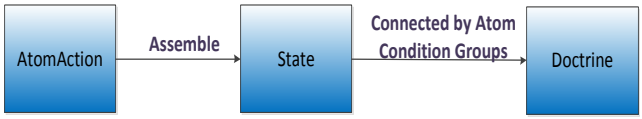

Figure 2: Behavior Modeling Thought

4.1 Definition of Atom Action

Atom action is the minimum execution unit that causes the CGF state to change. And it is the most basic constituent of CGF behavior.

In accordance with the component-based model's family framework, atom action can be divided into six categories, namely, motion action, target selection action, sensor action, electronic countermeasure action, weapon action and communication action. For each atom action, we define it as a quintet:

Action = < ActionName, ActionEnum, ActionType, EntityID, ActionData>

Among them, "ActionName" is the name of atom action; "ActionEnum" is the only enumeration value corresponding to the atom action; "ActionType" is the type of atom action; "EntityID" is a unique identifier of the Entity that invokes the atom action. "ActionData" is a struct which is used to save the parameters needed to perform the atom action. Its code implementation framework is:

int ActionsExecuter::ActionName (long EntityID, ActionData &ActionData).

This function will perform the action of Entity according to "EntityID" and "&ActionData".

4.2 Definition of Atom Condition

The atom condition is the minimum judgment unit that determines whether the State of the CGF will change.

We divide the atom conditions into six categories, namely, entity physic condition, target condition, weapon condition, geographical environment condition, time condition, and fuel level condition. For each atom condition, we define it as a six-tuple:

Condition=< ConditionName, ConditionEnum, ConditionType, EntityID, Operator, RValue>

Among them, "ConditionName" is the name of atom condition; "ConditionType" is the unique enumeration value corresponding to the atom condition; "ConditionType" is the type of the atom condition; "EntityID" is the unique identifier of the Entity that invokes the atom condition; "Operater" is the operator between the left value (Lvalue) and right value (Rvalue) that set by the modeler. Its code implementation framework is:

int ConditionEvaluate::ConditionName (long EntityID, char Operator, double RValue, double &res).

This function will get the Lvalue corresponding to the atom condition, compare the Lvalue with the Rvalue according to the "Operater", and store the result in "&res ".

4.3 Definition of State

The State is a stage in which the Entity performs certain atom actions and waits for some atom conditions. A State consists of one or more different types of atom actions. For each State, we define it as a quadruple:

State=<StateName, StateID, EntityID, ActionGroup>

Among them, "StateName" is the name of the State; "StateID" is the unique identifier of the State; "EntityID" is the unique identifier of the Entity that owns the State; and "ActionGroup" is a set of atom actions owned by the State.

Aiming at the structure of State, we design the XML [9] template file for it, as shown in Fig.3.

```
<State_Set>
    <!-- According to the number of State-->
    <State_Item>
        <State_ID>StateID</State_ID>
        <State_Name>StateName</State_Name>
        <Entity_ID>EntityID</Entity_ID>
        <ActionGroup>
            <ActionCount>Count</ActionCount>
            <!-- According to the value of action count-->
            <AtomAction>
                <Action_Enum>ActionEnum</Action_Enum>
                <Action_Type>ActionType</Action_Type>
                <Parameters>
                    <Count>Count</Count>
                    <!-- According to the number of parameters of action-->
                    <P_Item>
                        <Value>Value</Value>
                    </P_Item>
                </Parameters>
            </AtomAction>
        </ActionGroup>
    </State_Item>
</State_Set>
```

Figure 3: The XML Template File for State

4.3 Definition of Doctrine

Doctrine is an abstract of the combat mode adopted by CGF for a certain combat mission, which is composed of several States and atom condition groups. We define it as a quintet:

Doctrine=< DoctrineName, DoctrineID, EntityID, StateGroups, ConditionGroups>

Among them, "DoctrineName" is the name of Doctrine, "DoctrineID" is the unique identifier of the Doctrine, "EntityID" is the unique identifier of Entity that owns the Doctrine, "StateGroups" is the collection of the state owned by the Doctrine. "ConditionGroups" is a set of condition groups that transfer between States. Aiming at the structure of Doctrine, similarly, we design the XML template file as shown in Fig.4.

```
<Doctrine_Set>
    <!-- According to the real number of Doctrine-->
    <Doctrine_Item>
        <Doctrine_ID>DoctrineID</Doctrine_ID>
        <Doctrine_Name>DoctrineName</Doctrine_Name>
        <Entity_ID>EntityID</Entity_ID>
        <State_Nodes>
            <Node_Count>NodeCount</Node_Count>
            <Initial_State_ID>StateID</Initial_State_ID>
            <!-- According to the value of count-->
            <State_Node>
                <State_ID>StateID</State_ID>
                <!-- According to state transition that may occur-->
                <Transitions>
                <!-- The number of States that the state may transit to-->
                    <Tran_Count>TranCount</Tran_Count>
                    <State_Transition>
                        <Next_StateID>StateID<Next_StateID>
                        <!-- Conditions within a group are "&" relationship-->
                        <!-- Conditions between groups are "||" relationship-->
                        <ConditionGroups>
                            <Group_Count>GroupCount</Group_Count>
                            <!-- According to the number of condition groups-->
                            <ConditionGroup>
                                <Con_Count>ConCount<Con_Count>
                                <!-- According to the number of conditions-->
                                <Condition>
                                    <!--The Lvalue of condition can be get by
Condition_Enum-->
                                    <Condition_Enum>ConEnum</Condition_Enum>
                                    <Condition_Operator>Operator</Condition_Operator>
                                    <!-- The Rvalue of Condition-->
                                    <Condition_RValue>RValue</Condition_RValue>
                                </Condition>
                            </ConditionGroup>
                        </ConditionGroups>
                    </State_Transition>
                </Transitions>
            </State_Node>
        </State_Nodes>
    </Doctrine_Item>
<Doctrine_Set>
```

Figure 4: The XML Template File for Doctrine

5 AN APPLICATION EXAMPLE

In accordance with the behavior modeling method based on atom action and atom condition, in this section, we will model the bomber's "Bombing Enemy Radar Base" in order to clearly demonstrate our behavior modeling method. After that, we will analyze and explain the reusability of the behavior modeling method.

In order to complete the design of "Bombing Enemy Radar Base" Doctrine, first of all, we need to design States composition of the doctrine and the transfer condition groups between the States.

On basis of the definition of doctrine, we design a simple State transition diagram, as shown in Fig.5.

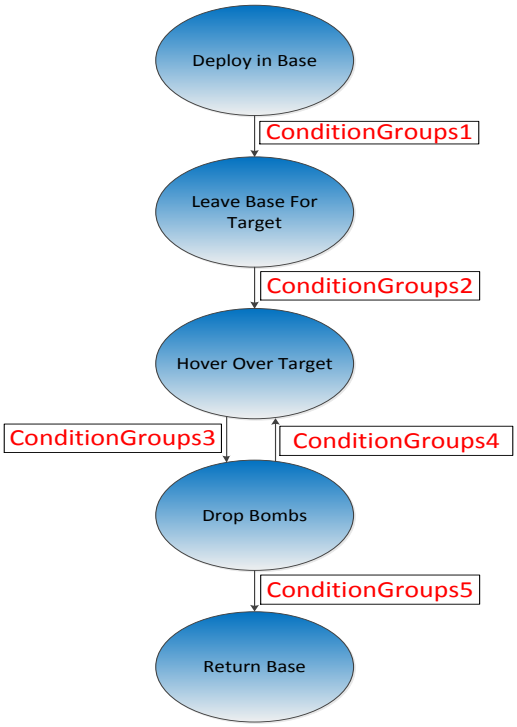

Figure 5: State Transition Diagram for "Bombing Enemy Radar Base" Doctrine

Then, on basis of the definition of atom action, atom condition, and State, we can complete the analysis of States and atom condition groups of this Doctrine, as shown in Fig.6 and Fig.7. The need to pay attention to is that the contents in the double quotation marks mean the parameter of atom action or atom condition should be configured by modeling people.

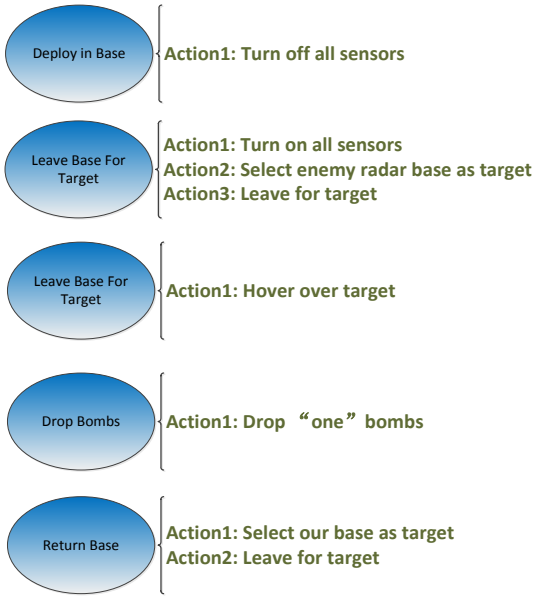

Figure 6: The Analysis of States

Figure 7: The Analysis of Atom Condition Groups

Finally, according to the type value, enumeration value, as shown in Fig.8, and parameter value of atom action and atom condition, as well as the XML template files of State and Doctrine, we can complete the representation and storage of all States and the whole Doctrine, as shown in Fig.9 and Fig.10 (Because of the space constrains, only a screenshot of the XML files of all States and the whole Doctrine is given), so as to complete the behavior modeling of "bombing enemy radar base" Doctrine.

```
enum ActionType
        { ActionTypeManeuvering = 1,
          ActionTypeTargetSelection = 2,
          ActionTypeSensor = 3,
          ActionTypeCM = 4,
          ActionTypeWeapon = 5,
          ActionTypeCommunication = 6,};
enum ActionNames
        { ActionTurnOffAllSensors = 0,
          ActionTurnOnAllSensors = 1,
          ActionSelectEnemyRadarBaseAsTarget = 2,
          ActionSelectOurBaseAsTarget = 3,
          ActionLeaveForTarget = 4,
          ActionHover = 5,
          ActionDrop_Bombs = 6,};
enum ConditionType
        { ConditionEntityState = 1,
          ConditionTarget = 2,
          ConditionWeapon = 3,
          ConditionGeographicalEnvironment = 4,
          ConditionTime = 5,
          ConditionFuelLevel = 6,};
enum ConditionNames
        { ConditionTimeInState = 0,
          ConditionFuelLevel = 1,
          ConditionTargetDistance = 2,
          ConditionDamegePercentOfTarget = 3,
          ConditionBombLoad = 4,};
```

Figure 8: The Enumeration of Atom Action and Atom Condition and Their Type

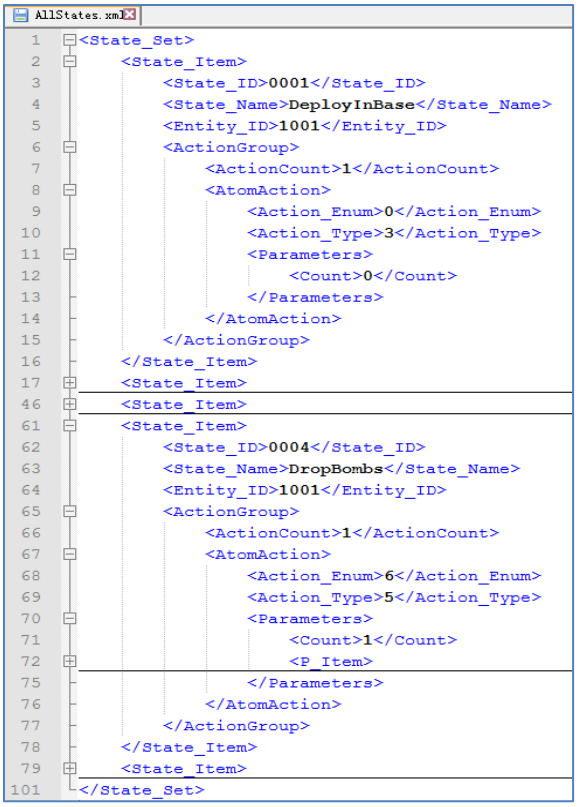

Figure 9: The Screenshot of XML File of All States of "Bombing Enemy Radar Base" Doctrine

From the whole design and realization of the Doctrine, it can be seen that this method can realize the behavior modeling of Entity by constructing various types of atom actions and atom conditions. Compared with other behavior modeling methods, this method has obvious advantages in reusability: (1) for some States that have been created, for example "Deploy in Base", it is available for all aircrafts obviously. By modifying the "Entity_ID" simply, the State can be reused by any other aircrafts CGF entities. (2) For some Doctrines that have been created, for example "Bombing Enemy Base", it is available for all types of bombers. According to modifying "Entity_ID" and some parameters of conditions, this Doctrine can be reused by any bombers. (3) For Doctrines that can just describe some simple combat mission, by means of adding new States and atom condition groups, the complexity and authenticity of these Doctrines can be improved. In addition, this method owns good composability and extendibility.

6 CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

Aiming at the problem of poor reusability of current behavior modeling methods, this paper proposes a reusable behavior modeling method based on atom action and atom condition. First, through the design and combination of atom actions, the States of Entity can be constructed. Then, through the combination of

States and the design of atom condition groups between States, the Doctrines of Entity can be constructed. Finally, according to the XML template files of State and Doctrine, we can complete the representation and storage of all States and the whole Doctrine. Research shows that this method has good reusability, composability and extendibility.

6.2 Future Work

At present, the XML format files of States and Doctrines also need to be configured manually. In the next step, we are looking forward to designing a graphical behavior modeling tool, so as to complete the configuration of parameters of atom actions and atom conditions, and the automatic generation of the XML format files of States and Doctrines.

Figure 10: The Screenshot of XML File of "Bombing Enemy Radar Base" Doctrine

REFERENCES

[1] Yang L, Guo Q. The Development of the Research of Computer Generated Forces [J]. Computer Simulation, 2000.

[2] Liu X L, Huang K D, Zhu X J. The Application of Finite State Machine in the CGF's Behavior Modeling [J]. Acta Simulata Systematica Sinica, 2001.

[3] Hao L S, Xia H B, Tian S C, et al. Research on CGF Entity Behavior Modeling Technology Based on Rule[J]. Fire Control & Command Control, 2015.

[4] Zhang H Y. The Study About The CGF Action Modeling Technology Based On Agent[J]. Computer Simulation, 2003.*control federations.* Journal of Defense Modeling & Simulation, 6(6), 5-16.

[5] Song Y B, Yang Y T. Decision-making behavior model of agent-based CGF[J]. Journal of System Simulation, 2006, 18(5):1319-1300.

[6] Chen J, Liao S Y, Deng F L. On modeling computer generated forces based on multi-agent systems [J]. Systems Engineering & Electronics, 2008, 30(10):1924-1928.

[7] Chen J, Liao S Y, Deng F L. A Collaboration Algorithm for Computer Generated Forces Based on Multi-agent Systems [J]. Computer Simulation, 2010, 27(2):113-117.

[8] Zeng L, Zheng Y, Si-Kun L I, et al. Behavior Model of Computer Generated Forces Based on Cybernetics [J]. Acta Simulata Systematica Sinica, 2005.

[9] Zhen Sun. (2015). Research and Realization of Simulation Scenario Description and Scenario Generation [D]. Beijing Institute of Technology.