High Level Architecture and Agent Technology based Astronautics Simulation Platform and Cluster Computing Environment’s Construction

Zhen SHEN
School of Science, National University of Defense Technology
Changsha, Hunan, 410073, China

And

Jing YAO
School of Science, National University of Defense Technology
Changsha, Hunan, 410073, China

And

Dong-yun YI
School of Science, National University of Defense Technology
Changsha, Hunan, 410073, China

ABSTRACT

The astronautics oriented computer simulation has two remarkable characteristics: one is the high dynamics of the simulation scenario; the other is the huge quantity of computing. For developing an astronautics simulation platform and cluster computing environment (abbr. ASPCCE), five strategies are presented in this paper to meet the above two requirements in complex scenario simulating computing (such as constellation optimization, resource scheduling and space-based early warning et al): 1. To increase ASPCCE’s numerical calculation ability and controllability, distributed clusters are employed as the computing environment; 2. To enhance ASPCCE’s dynamic simulation adaptation, the development is based on the high level architecture and the agent technology; 3. To meet the complex imaging model and the scenario demonstration’s requirements, a new 3D rendering engine is designed; 4. To achieve ASPCCE’s expansibility, the combination of the plug-in framework and the scripting language are adopted; 5. To accelerate the development of ASPCCE, many existing open source software are integrated following the open source strategy, and the outcome will be an open source software as well.

Keywords: Astronautics Simulation, Agent Technology, Cluster Computing, High Level Architecture, Parallel Virtual Machine.

1. INTRODUCTION

The researcher in astronautics simulation field should know the famous Satellite Tool Kit (STK), but as a general simulation platform, STK has the following disadvantages:
1. STK is not a complete simulation framework, while it is mainly a program majored in satellite related computing and demonstrating; It just provide the utility to interact with other simulation framework, but the integration needs more programming works.
2. STK is not designed for high performance cluster system, and has no extra performance advantage in huge quantity computing tasks;
3. STK is expansive and is not free, although it has many function about astronautics field, there are problem it can not do easily, such as: anti-missile and mid-course maneuvering, small satellite formation, special sensor imaging and volume covering, et al [1, 2].

Of course, there are more VO models and physical models which STK provides for us to learn from, and we are doing this. The original intention of this paper is to build an open source STK based on use of open source software, and become a set of general use astronautics simulation platform with cluster calculation environment [3-6]. This paper set up a simulation environment, with its main function to make up above-mentioned three disadvantages.

2. THE STRUCTURE

The horizontal structure of the simulation platform is divided into the following three layers: the interactive layer, the logic supporting layer, the data storage layer.

The interactive layer is mainly responsible for interaction between users and systems. In the interactive layer, the system will use double-agent mechanism, the logical support layer and interface interaction layer loosely coupled links. The interactive layer invokes the logic supporting layer functions through the
logical proxy, the logic supporting layer feedback through the interface proxy. The logical layer is mainly responsible for the main business logic calculations. On one hand it receives commands coming from interaction layer interface information, on the other hand it read or store information to the data storage layer through the data agent when needed. The data storage layer is mainly responsible for data storage capabilities. The data storage layer manage a variety of data including: data files, parameter configuration, components, scripts, scene script, simulation state, simulation results and so on.

From the vertical the simulation platform can be divided into three kinds of running status: simulation design state, simulation run state, simulation evaluation state.

Simulation design first appeared as a running state is primarily aimed at the development of component configuration and updates, scene script development and update, components and scripts management. This work will be the basis of simulation, should be completed first time and most of its major components and typical scenarios should be designed in the development phase first. However, as the system needs change after the completion of this part, this is the most vulnerable part, and should change after the completion of secondary development according to the needs.

Simulation run state mainly refers to the entire period from the start of the simulation until the end of it. In the simulation run state, the system mainly do the following tasks: parsing scene script, building simulation federal, starting passive simulation components, managing initiatives component, driving simulation forward, accepting and executing real-time control command, recording simulation state, saving the simulation results and so on. Simulation evaluation status is mainly refers to the simulation state about the analysis and evaluation about simulation result data after the end of the simulation. Mainly including: various forms of simulation demonstration, statistical analysis, report generation, etc.

3. KEY DESIGN LAYER

Parallel and management layer
Parallel virtual machine based distributed cluster system is the foundation of ASPCCE’s parallel computing and management layer, its main functions are:
1. parallel computing task management, such as computer node organization, computing resource distribution configuration, computing task process management, computing task inter-process communication, et al;
2. Simulation management, such as: simulation federation’s configuration and start, runtime control and sudden event simulation, simulation monitoring and data collection, simulation result demonstration, et al.

Simulation engine
We develop the ASPCCE’s simulation engine according to the high level architecture (HLA) specification. To realize dynamic adaptation, the global unique federation object model (FOM) is separated and distributed into every federate, and it will be aggregated by the simulation engine just before the normal HLA simulation starts. Every federate is modeled as agent object pattern with the internal handling threads and autonomous execution willingness based on the specific simulation logic. The simulation engine processes simulation execution according to HLA’s time advance mechanism. Every federate agent must interact with the engine through the communication mechanism that is provided strictly by the engine.

The functional difference between simulation engine and management layer can be summarized as: the simulation engine is the core communication mechanism and the driving engine for the simulation’s autonomous execution; The management layer is responsible for the preparation works before the simulation starts, the collection works after the simulation stops, the human machine interface when simulation is running, and the background parallel computing tasks. The simulation engine is developed based on the open source RTI implementation, CERTI.
Object Management: Register, Discover, Delete, Remove, Update, Reflect, Send, And Receive.

As a kind of parallel computing tasks, the simulation federation’s basic federal management service is implemented based on the parallel task management. The above services constitute a subset of HLA specifications, and can be designed in relatively simple and efficient ways to meet the needs of the astronavigation simulation.

Rendering engine
Fully functional 3d rendering engine is an important part of ASPCCE and the key component to meet the visualization requirement. ASPCCE does not employ the OpenGL engine or other OpenGL based libraries, it uses a simple but functional 3d rendering engine designed by our own. As not using hardware acceleration, the rendering efficiency could be improved. However it achieves the following advantages:
1. It Eliminates the frequent coordinates transformation problems brought by the inconsistency between the coordinate system inside the rendering engine and the coordinate system used in astronautics field;
2. It easily satisfies the need of the special visualization and special sensor’s imaging type by adding new projection model;
3. It Solves the super wide visual field rendering and the super far depth test calculation efficiency problem;
4. It minimizes the operating system dependency, and improves ASPCCE’s portability. The rendering engine is designed based on the BurningVideoDriver interface in the open source library Irrlicht.

The features which the three-dimensional rendering engine in simulation platform achieved include: perspective projection, cylindrical coordinates projection, depth testing, alpha channel color mixing and simple lighting effects and so on. The perspective projection is based on the usual three-dimensional affine transformation and its projection transformation as follows:

![Perspective Projection Diagram]

Cylindrical coordinate transformation is a special projection model in order to achieve a similar two-dimensional map model which STK designed; it directly keeps the correspondence of three-dimensional scene with the two-dimensional map, and is better to achieve the consistency of two-dimensional and three-dimensional scene. The depth testing and color mixing is designed according the usual three-dimensional algorithm. Although the advantages of ray tracing algorithm is a measure and an important symbol of three-dimensional rendering engine, but in space-oriented scientific research field and simulation demonstrate field the role of ray tracing algorithm is not so important as in the game, so this three-dimensional engine lighting effects using a very simple implementation, based solely on the triangle to be rendered with the camera line of sight angle to properly adjust the depth of color only.

Plug-in framework
The structure of the complex software system tends to adopt plug-in integration mode. The plug-ins collaborates together under an extensible plug-in framework to accomplish the system’s functionality. For the sake of portability, ASPCCE is developed based on the portable GUI toolkit wxWidgets. We chose the plug-in framework and integrate development environment CodeBlocks as ASPCCE’s basic GUI frame, and develop ASPCCE’s components as plug-in, including simulation engine, management layer, simulation federate, simulation scenario modeling, 3d rendering, monitoring and controlling, analyzing and demonstrating, et al. We use the squirrel script language as the simulation scenario representation form, and depend on it to guide specific simulation scenario execution.

Simulation system will use plug-in architecture to achieve maximum flexibility. The system is mainly built uses the dynamic components form plug-ins. The so-called dynamic component form plug-in is an independent component outside the framework of the main platform, can dynamically load and call, can be independently configured and updated, with a standard communication interface, support for HLA architecture specification.

Dynamic component is divided into two categories: functional components and simulation components. Algorithm models, and other functional components to provide a passive call functions; the simulation component have initiative and intelligence as a simulation federation member. All dynamic components consist of the system's "hardware" components. And the system's "software" part is composed of the scenario script. Scenario script is written in script languages about the description of specific scenarios simulated planning system. In the scene inside the script, detailed description about the components and its configuration need to participate in the simulation, component communication mechanisms, processes and sequence of events the scene, scene simulation presentation attributes and so on are specified.
High Level Architecture-based simulation is the industrial standards about simulation application; it is the best choice to adopt the dynamic component form simulation federation members targeted at space-based simulation system for real-time simulation of the dynamic characteristics. Dynamic component refers to loosely coupled with the simulation platform for the dynamic component, usually resides in the script or dynamic link library. In the system’s running status, the simulation platform will be required to notify the relevant loads or dynamic components and call their services or data processing functions. Dynamic component-based modular separation systems, the platforms itself retaining only the basic support functions, the update and expansion of the module is just the update and expansion of the system functions, this can be with a very great flexibility.

![Fig 3 Plug-in Framework](image)

4. SYSTEM EXECUTION

Scripts used to control the process and guide the implementation of information processing is a major feature of the simulation system. The script is a plain text stored procedures, in general, a computer script is a combination of computing, in which the logic can be achieved with certain branches. Script development, it is rather the general procedure is relatively close to natural language, may not be interpreted by the compiler, which will do some help to quickly develop or some light-weight control.

There are many scripting language now, the general implementation of a scripting language interpreted only with a specific interpreter related, so long as the computer has the appropriate language interpreter it can be cross-platform. In choice of script language specification, there are two options: One is to build a custom syntax for a simple scripting language to achieve the basic functions; one is using the existing standard scripting language. For the first approach, requires a certain amount of work, the resulting language skills are limited, but it does not require additional development kits support, can maintain platform-independent features. The latter is necessary to use some existing scripting language development kit, which kit may not be platform-independent, and will bring the system constraints.

There are two ways about the development of the script: one is text-based style of development, such a design development of the workload is very small, as long as there is a text editor on the list; another way is graphical development, the performance capabilities of this approach is strong, and in particular is very intuitive.

There are basically two ways about script execution, one is interpreting and the other is compiling. The interpretation implementation has a built-in script interpreter according to the contents of the script line by line analysis. The benefit is that the process can be carried out in the implementation of flexible control. The scripting languages which express the logic within the system are only for advanced users to show high-level control. For the average user, the system will provide an intuitive user-friendly graphical modeling and design capabilities. In the script system development and design, system configuration maintenance, etc., we will adopt a common modeling environment UMLPad-based visual development.

For the simulation scenario’s execution, we can use a static built-in mechanism for the development described in the system, but also a dynamic scripting mechanisms can be used to descript it outside. Scene script mechanism more flexible and scalable, able to better adapt to the changing needs of simulation programs, to better support analysis and decision-making. The system uses process-based script to describe the structure of data flow simulation scenarios, using script to control the flow of data processing logic flow. Workflow script describe the process information in scripting language form, the main program read into the workflow script, analyze and understand the formation process and in accordance with the implementation of data processing.

In the scripting language we establish the following basic language mechanisms: data representation mechanisms, the
simulation component synchronous service invocation mechanism, the simulation component of the mechanism of asynchronous service calls, data flow and control flow mechanisms.

![Diagram](image)

5. CONCLUSIONS

We take the full integrated existing achievement's way as a foundation, elaborated the construction plan about general simulation platform face the astronavigation simulation domain. We use our knowledge and mastery about existing aerospace-related open-source software, and integrate the appropriate library at the appropriate entry point, but it’s not a simple integration, it’s the selective re-creation on the existing software’s ideas (for example: the owner designed 3D engine, the high-level architecture and cluster system’s integration, etc.). The structured and functional simulation platform is able to fill in the field of aerospace simulation of open-source general-purpose simulation platform gaps. We believe that the completion of the platform can play an important role for scientific research in the field of aerospace simulation.

Some astronauts field simulation tests showed that ASPCCE can fulfill complex astronauts simulation and computing works. We hope it could play an important role in the decision support and the system optimization in the astronauts field. Meanwhile, as open source software, we hope it is valuable in the scientific research field for the astronauts.

[2] Li, Ke-Xin, Cong, Ming-Yu, Zhang, Wei, Infrared sequence image generation of point target in deep space, Optics and Precision Engineering, Vol 17, No 12, 2009, pp. 3062-3068.