文章编号:1001-2486(2007)05-0104-07

HLA数据回放及其可通用性问题研究*

张新宇,陈 彬,尹全军,黄柯棣
(国防科技大学 机电工程与自动化学院,湖南 长沙 410073)

摘 要:为了给HLA分布仿真系统提供运行数据的回放能力并解决各种回放策略都必须面对的数据完备、时序准确、运行高效及回放方法的可通用性等问题,对HLA联邦中的数据回放问题进行了研究,分析了HLA仿真中的回放需求以及HLA对回放实现技术的影响,指出回放方法的可通用性实质上是联邦独立性的概念,研究了将回放数据对象化处理的方法,针对成员式数据回放策略、基于对象序列化技术和成员自动生成技术提出了一种可通用的HLA数据回放实现方案,最后介绍了通用HLA数据回放工具的应用情况。

关键词:高层体系结构;数据回放;数据对象;通用工具
中图分类号:TP391.9 **文献标识码**:A

A Disquisition to General Purpose HLA Data Playback

ZHANG Xin-yu, CHEN Bin, YIN Quan-jun, HUANG Ke-di
(College of Mechatronics Engineering and Automation, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: For supplying the capability of replaying the HLA federation execution data and solving the issues demanded by playback that data must be integrated as a whole, time order must be exactly, run efficiency should be highly and the solution should be general-purpose, this paper explores the issues of data replaying in HLA federation, analyzes the requirements and the impacts of HLA on playback, points out that the essentially general purpose is a concept of federation independent, and excogitates a data processing method to deal with the replayed data by means of object. Relying on the strategy of a concentrated playback federate, the playback solution was putted forward, which is based on the technology of data's object serializing and federates' automatic-generating. The solution realizes the goal to make a general purpose data playback. In the end, this paper introduces the application of the general-purpose playback tool.

Key words: HLA(High Level Architecture); data playback; data object; general tool

仿真系统及试验是建立在模型和数据基础上的,仿真运行数据的回放是应用仿真系统的关键步骤,是分析数据、评估结果、测试系统和仿真重演的重要支持手段。作为分布交互仿真领域的标准,HLA的应用范围在逐步扩大,为实现互操作、可扩展、可重用目标及减少网络数据,HLA采用了新的体系结构和技术措施。这些新特性在带来好处的同时,也为可通用HLA联邦回放的实现增加了难度,需要对HLA数据回放技术进行更进一步的研究。国内外十分重视HLA数据回放问题的研究[1-2,5,7],典型成果有美国MÄK Technologies公司的Data Logger和Virtual Technology Corporation公司的hlaResults®2.0,以及文献[5,7]的实现办法;它们各自在一定程度上解决了HLA数据的回放问题,其共同特点是将联邦中交互的各种数据看成互不相关的二进制块,为每个数据块添加标识信息以确保其可解析性,然后在定制的句柄管理及数据解包/打包规则支持下工作。这种做法未考虑HLA以面向对象(OO,Object-Oriented)方式组织数据的特点,实现逻辑复杂,对标识信息的重复添加和操作导致了冗余数据大量产生并影响到运行效率。

针对上述实现方法的不足,本文首先对HLA数据回放问题的含义和需求进行了定义,引入对象序列化机制[6]来处理联邦执行数据与回放数据,基于回放数据对象与成员自动生成技术[4],提出了新的通用HLA数据回放实现方法,并介绍了它的实际应用情况。

* 收稿日期:2007-04-30
基金项目:国家自然科学基金资助项目(60374065)
作者简介:张新宇(1978—),男,博士生。

1 HLA中的数据回放问题分析

1.1 HLA数据回放问题的定义、实现策略和实现原则

与传统回放重视仿真结果的重现而不重视仿真的重复运行不同,本文所面对的回放需求是"关于在回放联邦中回放执行机构如何'重建'仿真对象,并按时序一致要求向回放联邦提供原始联邦运行交互数据,以准确'复现'相关仿真对象间数据交互过程的问题",这也是本文关于HLA数据回放问题的定义。其中,回放联邦是指容纳回放执行机构工作的仿真联邦;原始联邦是指为执行回放任务而进行数据采集的仿真联邦;回放执行机构是执行回放任务的载体,与回放相关的仿真对象由其产生、数据由其发出;仿真对象是回放任务的受体,包括被回放的成员、对象/交互等实体和对象实例注册、属性更新及交互发送等操作;时序一致要求指的是回放数据的发送时序要与原始联邦中交互数据的产生时序相一致;"重建"和"复现"都是克隆操作,分别作用于原始联邦的仿真对象和数据交互过程。

根据回放执行机构的特点,HLA数据回放有两种实现策略:(1)成员式,回放执行机构为专门的回放成员,回放数据通过运行时间支撑系统(RTI, runtime infrastructure)传送给相关仿真应用成员,时间管理与同步控制也由RTI负责;(2)嵌入式,回放执行机构是仿真应用成员的内嵌模块,可单独实现本节点全部数据的回放,需开发时间管理与同步控制机制以支持多节点协同回放。比较而言,成员式实现策略有利于可通用性的实现,本文选择其作为通用HLA数据回放的实现策略。

通用HLA数据回放的实现需兼顾仿真系统扩展、仿真标准未来发展等需求,遵循以下几条原则:

(1)准确——回放数据的内容和处理时序都应与原始联邦相关数据保持一致,无添加、删减或更改;

(2)完备——应能回放各层次的仿真对象:联邦、成员和仿真对象实例;还应包括全部回放控制功能;

(3)高效——应尽量减少回放运行对计算资源的占用,具备尽可能高的运行性能;

(4)可扩展——在不影响其初始设计的情况下,回放产品的数据结构能够适应仿真应用系统的改变;

(5)可通用——回放产品的全部或部分组件能被两个或多个功能领域的任务所使用。广义"通用"指能够适用于各种RTI、仿真应用联邦及HLA系统使用的各种计算机平台和网络;狭义"通用"专指适用于各种仿真联邦,即联邦独立性(federation-independent)。由于缺乏标准支持,广义"通用"难以实现,所以本文中"可通用"特指狭义"通用"。

1.2 HLA对数据回放的影响

HLA对数据回放实现技术的影响,比较重要的有:

数据交互机制:HLA采用组播作为数据通信的基础,在高层提供了声明管理、对象管理和数据分发管理(DDM,Data Distribute Management)等服务来实现数据交互;联邦中数据交互须由成员通过公布订购、发送接收等服务来完成;HLA未规定DDM的实现标准,不同RTI实现方法可能不同。因此,应考虑以成员方式实现回放并考察RTI使用DDM服务。

用户自定义FOM:联邦对象模型(FOM,Federation Object Model)是联邦实现互操作的基础,FOM中的对象及其数据结构由用户定义,不同联邦的定义存在差异,因此联邦独立的回放执行机构必须能够解析不同FOM的数据结构。

对象属性的部分更新:与DIS不同,HLA规定在联邦运行过程中对象类实例状态的改变可以通过只更新部分属性来实现,以减少网络数据流量。但这给回放功能的实现带来了困难,不完整的仿真对象信息将难以保证"对象重建"与"联邦重建"结果的准确性。因此要求采集数据时保存相关现场数据,以确定对象属性的更新信息。另外,当一次对象更新由几个不同传输模式或排序策略的属性组成时,将会被RTI分解成几次反射,但HLA却未提供能让接收方识别该现象的方法,因此原始联邦中的一次对象更新就可能在回放联邦中作为多次更新被发送,给回放对象的时序组织带来困难。

时间管理机制:仿真成员工作于不同时间模式时可产生TSO(Time-Stamp-Ordered)消息和RO

(Receive-Ordered)消息,二者处理机制差别甚大,TSO 消息带有时戳和发送成员信息,对回放成功有重要影响。时间信息的缺失将难以保证数据的回放时序与其产生时一致,因此每个仿真成员(包括采集和回放)都应选择合适的时间管理策略。

对象实例的属性所有权转移:HLA 规定对象实例的属性所有权可以在成员间转移,但当属性所有权发生改变时,RTI 并不通知无关成员,因此数据采集成员难以获得属性所有权转移的完整信息。而回放的完备性要求包括特定成员数据的回放,因此最好避免属性所有权在不同成员间转移的情况,以方便回放通用性的实现和保证回放的准确性。

1.3 以数据块处理方式实现 HLA 回放的不足

我们首先基于数据块处理方式实现了通用 HLA 数据回放工具,并发现其若干不足。该工具由三个互相联系的子模块组成,分别是配置模块、动态公布模块、解包模块。如图 1 所示。

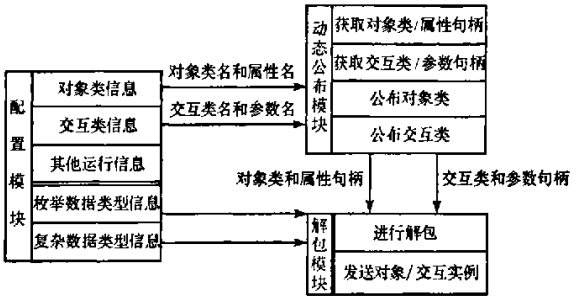

图 1 配置、公布、解包模块的关系

Fig.1 Relationship among config, publish and parser modules

配置模块的主要功能是读取 FOM 中包含的对象类和交互类信息及其数据结构,形成配置文件供动态公布模块和解包模块使用。这样做是因为在仿真过程中,计算机无法识别数据结构的定义、不能根据定义生成相应的仿真对象来发送数据,所以需要将数据结构以字符串形式储存起来,供动态公布和解包模块通过比较数据类型名称来确定数据结构并动态分配内存,进行数据辨认与解析。

动态公布的过程分为三个步骤,首先要读配置文件得到相关仿真对象的名称;然后调用 RTI 服务获取相关仿真对象的运行时句柄;再利用运行时句柄,调用 RTI 服务进行公布和发送回放数据操作。

解包模块是数据块处理技术的核心部分,也是该回放实现方法的特点和不足之处。为使计算机能够识别数据块,每块数据都由报头和主体组成,报头格式统一、大小固定包含处理数据块所需的元数据,主体格式各异、大小不定包含仿真数据的内容。解包操作对每个数据块的处理步骤依次是:在配置文件生成时将仿真对象的数据结构分解为原子(不可再分)数据项并保证其名称的唯一性;在回放预处理过程中读配置文件中的类型信息,为每个原子数据项动态分配内存,读出数据块的内容、筛选并依时序排列数据块,生成专用于回放的数据文件;在回放进行过程中根据配置文件中的类型信息,为每个数据块动态分配内存以将其从回放文件中导出,并通过 RTI 服务将其发送给回放联邦。

实现解包模块的主要难点在于原子数据项命名规则的繁琐和回放预处理过程的复杂。考虑到 FOM 用户自定义行为的多样性,需采用如下命名规则以保证原子数据项名称的唯一性:如果仿真对象的某属性数据类型是简单类型,则直接用属性名作为数据项名称,如果是复杂类型,则用属性名加上复杂类型的数据域名,如果数据域类型还是复杂类型,则继续添加域名,直到每个数据项都是不可再分的;如果上述过程中某数据项是多粒度的情况,还需为其添加索引号,以唯一标识该数据项。回放预处理过程的复杂性主要是由内存的动态分配机制和数据块重组过程的繁琐造成的:(1)由于各编译器有不同的复杂结构数据对齐机制,所以在动态分配内存的时候需要根据编译器的具体设置、考虑由数据对齐而导致的数据内容间有空隙的情况,以保证解包操作的正确性;以 VC6.0 为例,比如依次包含两个 double、一个 float 型数据域的复杂数据类型 Position,一般理解所占空间为 $8+8+4=20$ 个字节,但由于数据对齐机制的作用,它实际上占用的空间是 24 个字节;如果不处理好这个问题,就会造成数据错位,得不到正确结果;考虑到数据对齐机制是要求在内存中的地址和其数据内容的长度要成整数倍的关系,我们设计了相应的数据对齐解析算法,利用该算法可以将 FOM 中所有复杂类型所含数据域占用空间的大小和位置都计算出来,在解包复杂类型数据时,每解析出一个原子数据项的内容,就根据其所占空间,向后推移,跳过数据内容的间隔,保证解析数据的正确性;(2)由于无法直接识别,因此数据块重组时需要遍历、解

包回放时段中的每块数据,以识别其内容、判断相关性,遍历解包操作浪费并占用计算资源;而且由于对象属性部分更新机制的影响,数据块所含仿真对象信息可能不完整,当判断相关性时,需从选定时刻"前溯"各块数据比较并补齐相关对象的属性数据,过程繁琐且执行效率低,所以回放预处理过程通常比较耗时。此外,当在回放运行过程中执行向前跳进回放功能时,为保证在指定时刻更新对象信息的完整并且不遗漏当前时刻和指定时刻之间的重要仿真事件(如注册/删除对象实例等),也必须将两个时刻间的各块数据全部解包处理,因此该回放方法常常不能及时响应用户的跳进回放等时间控制指令。

不难看出,将回放数据视为互不相关的数据块并以此为基础的 HLA 数据回放方法虽然也能具备准确、可通用等特点,但其不足也同样明显:数据处理效率难以提高,回放控制功能难以灵活响应用户需求。

2 以对象化方式实现可通用 HLA 数据回放的方法

2.1 回放数据的对象化处理和回放数据对象的序列化技术

回放数据的采集与存储可借鉴文献[3,6]中的解决方案。而由 1.3 节可知,回放数据的处理方式对回放实现的质量有关键性的影响。因此,本文依据 OO 方法和 HLA 对象的描述规则,给出关于回放数据的有关特征的统一描述框架,规定回放数据对象是回放数据有关特征的集合{特征 1,特征 2,…,特征 n},集合中的特征描述原本只是互不相关的数据块,但经过对象序列化技术和相关知识处理之后,该集合就成为机器可理解的信息包——回放数据对象,从而使各式各样的回放数据块(更新的属性、发送的交互和仿真事件等)有了统一的描述和处理格式;利用 OO 多态机制,定义抽象类 CPObject 作为所有回放数据对象的根类,其中包含了回放数据对象所共有的操作,操作的具体实现由各回放数据对象负责;CPObject 是回放执行逻辑操作各回放数据对象的接口,保证了回放执行逻辑与具体数据的无关性。

对象序列化是回放数据对象处理技术的重要基础。对象序列化技术实际上就是要求对象具有保存自身的能力,即 OO 中的对象持久能力(Persistence),为此序列化包括了文件访问、缓冲区管理、数据读写、运算符重载、动态对象生成等一系列技术。Windows 平台下 MFC 序列化(Serialize)的主要用途是配合 MFC 的文档/视/框架三元组体系的应用程序结构,将整个文档作为一个对象进行保存和恢复,因此不能直接用于回放数据对象的序列化。回放数据对象的序列化技术是在 MFC 序列化机制的基础上改进实现的:考察 MFC 序列化机制,发现其在记录重复的内存地址的对象时,只在第一次记录完整的数据,后面都只保存一个索引,因为它保存的是内存中某个时刻的一个映像,同一地址指向的对象必然是完全相同的,这是 MFC 为节约存储空间而进行的优化;而仿真中每次数据的产生(属性更新,交互发送和仿真事件)都必须作为新的数据对象予以保存和处理,比如:即使每次属性更新都是更新的同一块内存,但因为它代表了仿真对象的不同状态,所以每次更新值都必须被记录;因此必须改造 MFC 序列化的优化机制,经过对 MFC 的 CArchive 类进行分析发现,它使用 map 结构来实现不重复保存同一内存位置数据的功能,所以每次保存完数据之后需对 map 执行清除操作,以保证下次保存的是数据内容而不是地址索引。由于在回放数据对象的 Serialize 函数里可以同时实现存储和读取过程,所以回放数据对象的存储与读取可以共用接口和数据文件。回放数据对象序列化之后按时序组织在一起,其格式如图 2 所示。

回放数据对象统一了各种回放数据的描述形式,可以简化回放执行逻辑的实现。序列化使回放数据对象成为机器可理解的信息包,存储和读取过程共用接口与数据,无需人工控制内存分配并遍历解包数据块,可提高回放数据的处理效率和回放控制功能的灵活响应能力。

2.2 回放数据对象和回放成员的自动生成

回放数据对象需根据 FOM 信息生成,因此回放数据对象与仿真应用紧密相关,这种相关性与联邦独立性相矛盾,是影响 HLA 数据回放方法通用性实现的主要障碍。

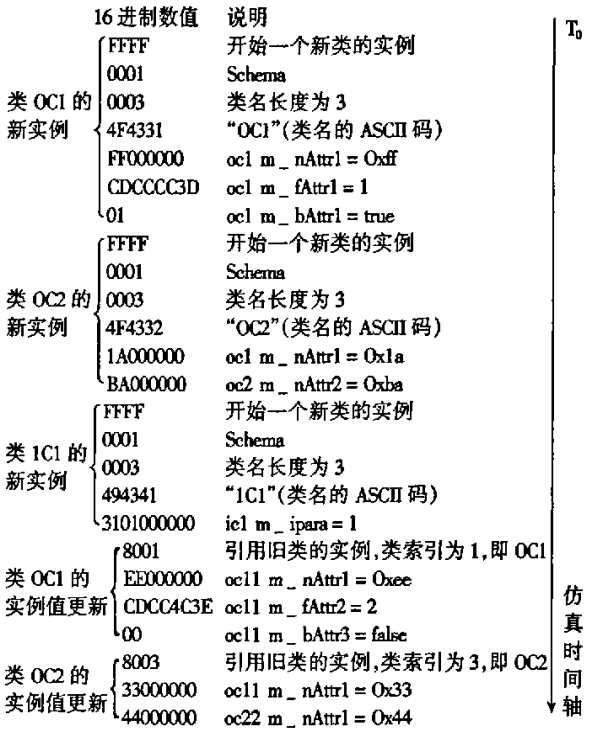

图2 改进的对象序列化存储机制的数据格式

Fig.2 The storage format of the improved object serializing

问题关键在于如何能让计算机处理这种相关性,参考传统的产生代码、编译链接、生成可执行程序,如果这个过程能够自动完成,不需人工操纵,那么就可以克服上述相关性对HLA回放通用性实现的不利影响。这也是本文采用的解决办法:利用FOM在联邦互操作中的基础地位,根据回放数据的特点,开发了可通用的HLA数据回放工具控制台,它可以驱动成员代码生成器根据解析FOM得到的仿真对象信息,自动生成回放成员源代码并驱动编译器和链接器将其生成为可执行文件——回放成员,回放成员作为代理(Proxy)成员加入回放联邦并在回放控制台的指令控制下完成数据回放工作。如图3所示,从解析FOM→生成源代码→生成回放成员→回放成员工作,整个过程都在联邦回放工具控制台的管理驱动下自动完成,通过该方法——从模型到执行代码的自动生成,可实现数据回放的通用性和扩展性要求。

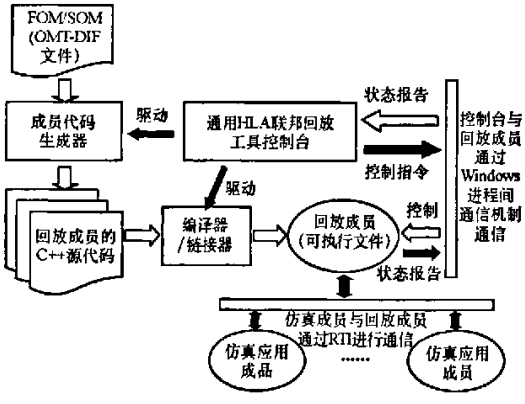

图3 通用HLA联邦回放方法实现原理

Fig.3 The principle of the general HLA federation playback

回放成员中包含与应用无关的回放执行逻辑和与应用相关的回放数据对象,二者通过自动生成技术被粘合到一起并协同工作。回放成员与回放联邦中的其它成员通过RTI进行通信,实现回放功能;控制台并非回放联邦成员,是独立于回放成员的执行进程,通过进程间通信(Windows IPC)机制实现控制功能。

2.3 回放成员时间控制和重演功能的实现

回放成员的时间控制机制用于实现回放成员时间推进、变步长、选时、向前跳进和变速回放等功能。时间控制机制的实现与回放数据对象的时戳紧密相关。回放成员的时间步长最好能与原始联邦中相关

成员的运行时最小步长保持一致。在回放成员的时间控制机制中,以下几个仿真时间是非常重要的:原始联邦运行的起始时刻 T_0,结束时刻 T_e;回放联邦运行的起始时刻 T_r;回放成员的当前时刻 T_{grant},回放成员将要推进到的时刻 T_{next},回放成员的时间推进步长 T_{step},回放成员的前瞻量 $T_{lookahead}$,回放成员要跳进到的时刻 T_{plunge};回放数据对象的时戳值 T_{tso}。

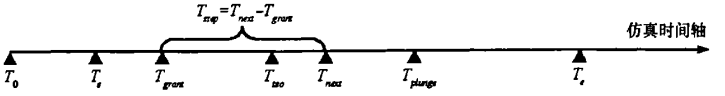

图 4 回放时间关系轴

Fig.4 The axis of playback time scheduling

如图 4 所示,回放数据对象沿回放时间关系轴顺序排列,回放成员时间控制机制按如下方式实现:

(1)回放成员推进时间——当 $T_{grant} = T_0$时,回放成员首先将 T_0 时刻的数据对象发送出去,而不是立即请求推进时间;T_0 时刻的数据对象主要是原始联邦中的发现对象实例事件,在回放时采取对应操作——向回放联邦注册相应对象实例;当 $T_{grant} > T_0$ 时,为保证回放成员不向 *RTI* 发送过时数据,回放成员的时间推进需考虑与回放数据对象时戳相协调以确定是否推进时间:当 $T_{tso} < T_{next} + T_{lookahead}$时,回放成员不推进时间而是直接将回放数据对象发送出去,当 $T_{tso} \geqslant T_{next} + T_{lookahead}$时,回放成员首先请求推进时间到 T_{next},然后再发送回放数据对象;变步长回放也服从本规则;

(2)选时回放——当 $T_0 < T_r < T_e$,在回放数据准备时,在 T_0 至 T_r 时段中只保留发现对象实例事件和删除对象实例事件;其余规则与 1)中相同,只是需要将规则 1)中的 T_0 替换为 T_r;

(3)向前跳进回放——在回放成员运行过程中收到向前跳进指令时,回放成员会快速浏览 T_{grant} 至 T_{plunge}时段中的数据对象,只保留发现/删除对象实例事件并将这些数据对象发送之后,直接请求推进到 T_{plunge}时刻;回放运行前,所有数据对象已由序列化机制导入内存,因此处理效率能满足应用要求;

(4)变速回放——变速回放用于调节回放成员运行的快慢,通过仿真时间与墙上时间的比率来加快或减缓回放的运行过程;

还实现了如暂停与恢复回放等控制机制较为简单的功能。回放时间控制机制实现逻辑的简化得益于回放数据以对象方式处理和数据对象按时序排列的存储机制。

此外,由于 RTI 不支持仿真时间的逆向推进,因此不能将按时序排列的回放数据对象逆向发送出去。但为满足仿真重演对回退功能的需求,本回放实现方法提供了折中方案:先用回放数据对象驱动演示系统(二维和三维)完成整个回放过程并将演示系统的屏幕输出录像,然后使用媒体播放器就可对仿真重演进行 DVD 式操作,包括回退、快进、暂停等,而不需再受 RTI 时间控制机制的约束。解决方案的基础是分清回放与重演的区别:回放强调的是数据按时序关系正确发出,受 RTI 时间控制机制约束;重演,更强调的是演示系统的表现效果。在设计良好的仿真联邦中,演示系统通常作为独立模块存在(如二维、三维成员),仅作为数据接收方工作,因此这种区别回放与重演的实现办法通常是有效的。

2.4 对象属性部分更新问题的解决

由 1.3 节可知,数据块处理方式在解决属性部分更新机制引发的回放问题时,效果并不理想。而回放数据对象化处理方式以对象为单位处理数据的特点,使得在对象状态更新时,不仅可以存储本次更新的属性值,并且可方便地添加属性更新标识,而其它未更新属性值仍保持原值和未更新标识;因此在任意时刻都能得到一个对象实例的完整信息及其属性更新信息;在正常回放时根据更新标识更新对应属性,在选时或跳进回放时只需忽略更新标识并将对象实例的相关属性更新即可,回放实现逻辑简单并且运行效率高。对对象序列化存储机制而言,状态累积存储方式不会额外占用存储空间,因为数据对象的大小由 FOM 中定义的数据结构确定,与运行过程无关。

对由不同传输模式或排序策略引发的单个对象状态更新需通过几次反射才完成的问题,由于缺乏有效方法使数据采集确定哪些反射是同一更新的结果,因此只能根据采集情况在回放联邦中将其作为

多次更新发送,但本回放方法将保证这些更新能在一个处理节拍中被全部发送,所以对回放数据的接收方(仿真应用成员)而言,数据还是按原来方式得到的。

3 通用 HLA 数据回放工具的应用情况

应用该方法开发的通用 HLA 数据回放工具 DPT,已经成功应用于军队重点项目某电子战仿真试验室的通用仿真环境建设中,可针对对象实例、仿真成员和联邦三个层次提供快速、完备的数据回放服务,被广泛应用于联邦开发过程中仿真应用成员的单独调试、多方成员的联合测试、仿真联邦的集成测试、仿真过程重演和仿真结果评估等任务中。

通过选择性的回放相关回放数据对象,通用 HLA 数据回放工具的具体应用包括:

(1)成员测试和联邦集成测试:在测试联邦中,DPT 被用于替换某些缺席成员(已通过测试的成员或暂时不参加测试的成员),缺席成员的数据通过 DPT 被选择性地回放给测试联邦。被测成员可以测试出它们能否对发现/删除对象实例、反射属性值和接/发交互等数据互操作活动做出正确反应,帮助各成员在集成测试之前发现并纠正问题,从而使联邦集成过程的效率更高;

(2)仿真重演和结果评估:由于该系统规模大、仿真实体多、运算逻辑复杂,仿真联邦每完成一次运行需数小时至几十小时不等,并且电子战仿真的表现形式比较抽象,因此回放支持的快速重演在该系统中发挥了重要作用——DPT 与演示系统相结合最快可将每次联邦运行在数分钟内完成重演,其直观、快速的事后回顾能力支持仿真结果的快速评估和问题定位,为校正需求的快速响应和演示汇报工作的顺利完成提供了有力的支持。

4 结 论

通过对 HLA 仿真数据处理机制的比较研究,我们提出了回放数据对象的概念,并基于对象数据处理方式实现了可通用的 HLA 数据回放解决方案,对象数据处理方式使得对回放数据的理解与处理能与仿真对象建模时的概念描述相一致,回放数据对象的时序性使回放实现逻辑更加简洁,实践证明该解决方案使回放数据处理效率和回放运行性能得到显著提高,满足了 HLA 数据回放对准确、完备、高效、可扩展和可通用性的要求。建立在对象描述基础上的该回放方法对标准的依赖程度低,可方便地升级至标准的高级版本或扩展到其它标准。

参 考 文 献:

[1] Head R V, et al. Developing an HLA Federation-independent Playback Application[EB/OL]. http://siso.sc.ist.ucf.edu/siw/00Spring/view-paper.htm/00S-SIW-146.doc, 2000.

[2] Fisher D L, et al. Distributed Simulation Synchronization Replay Tool (DSSRT) [EB/OL]. ht paper.htm/ 04S-SIW-033.doc, 2004.

[3] Zhang X Y, et al. The Research and Implementation of a General HLA Data Collection and Playback Application[C]//Proceedings of Asia Simulation Conference/the 6th International Conference on System Simulation and Scientific Computing, Oct 24-27, 2005, Beijing, China, 2005: 625-629.

[4] Zhang X Y, et al. Research on the Automatic Generating Technique of Federates Based on HLA Object Model and C++[C]//Proceedings of Asian Simulation Conference, October 30-November 1, 2006, Tokyo, Japan: 49-50.

[5] 郑扬飞. 分布交互仿真测试评估工具系统研究[D]. 哈尔滨:哈尔滨工业大学, 2003: 33-66.

[6] 张柯, 张新宇,等. 基于 HLA 的分布仿真系统数据采集解决方案[J]. 系统仿真学报, 2004,12:2725-2728.

[7] 陈宏,等. 基于 HLA 的仿真系统的记录与回放[J]. 系统仿真学报, 2006:629-632.