

基于 GAUDI 框架的原始数据管理系统*

张晓梅¹⁾ 马秋梅 王喆 王大勇 尤郑云 毛泽普 邓子艳 邱进发 刘怀民

李卫东 张学尧 冒亚军 袁野 黄性涛 蒋林立 臧石磊

(中国科学院高能物理研究所 北京 100049)

摘要 主要阐述了原始数据管理系统的基本原理. 原始数据管理系统主要完成在线获得的原始数据字节流格式和适合 GAUDI 框架中瞬态数据缓存(Transient Data Store)存取的格式之间的转换工作. 整个实现过程基于 GAUDI 框架, 采用了面向对象设计模式, 因此整个系统有很好的灵活性和可维护性.

关键词 GAUDI 原始数据 面向对象 BESIII 数据分析

1 引言

北京谱仪(BES)是北京正负电子对撞机(BEPC)上的大型通用谱仪. 它已经成功地在 τ 粲能区运行了14年. 在未来几年里, BES/BEPC将升级改造成为 BESIII/BEPC II. BEPC II 的亮度将比 BEPC I 提高100倍, 最高将达到 $10^{33}\text{cm}^{-2}\text{s}^{-1}$. 设计建造中的北京谱仪 III 主要部件有主漂移室, 电磁量能器, 飞行时间计数器, μ 子计数器及超导磁铁等. 它主要用于 τ 粲能区的物理研究如弱电相互作用研究, 强相互作用和新物理的寻找等^[1].

BESIII 的数据处理流程将分为在线取数、离线数据处理和物理分析3个主要部分^[2]. 其中在线取数部分就是通过电子学读出, 触发和在线计算机系统将正负电子对撞产生的事例记录下来形成原始数据字节流文件, 而离线数据处理部分的主要任务是把在线获得的原始实验数据文件进行分析处理, 为物理分析提供粒子信息, 如动量、能量、空间位置和粒子种类等. 故原始数据是在线取数过程和离线数据处理过程的连接纽带, 在整个数据处理流程中起着很重要的作用.

随着计算机工业的高速发展, 高能物理软件技术已经从原来以 Fortran, C 语言为主导的结构化的程序设计发展到了以 C++, Java 为主导的面向对象的程序设计. GAUDI 框架就是将面向对象设计思想成功运

用于高能物理软件的一个典范. 面向对象设计模式使得 GAUDI 框架比结构化程序语言编写的框架拥有更多的优点: 模块独立性强, 可重用性好, 架构清晰, 层次分明等. BESIII 数据分析框架是以 GAUDI 框架为基础开发的符合 BESIII 实验要求的离线数据处理平台. 原始数据管理系统则是在 GAUDI 框架下开发的一套原始数据处理系统, 设计符合面向对象设计的思想, 该服务提供了方便统一的用户接口, 程序具有重用性好, 可读性等优点.

2 GAUDI 框架

GAUDI 是在 LHCb 实验背景下发展起来的用于开发物理数据处理软件系统的公共平台. 它提供了一套设计清晰, 面向对象思想运用成熟的体系结构, 其他实验可以按照该体系结构设计和实现适合本实验的数据分析框架.

2.1 GAUDI 体系结构^[3]

GAUDI 体系结构中的主要组成部分以及它们之间的关系如图 1 所示: 算法模块或用户模块(Algorithm), 应用管理器(Application Manager), 瞬态数据缓存(Transient data store, TDS), 服务模块(Service). 应用管理器指挥着整个数据处理过程的进行, 例如

2005 - 03 - 29 收稿

* 国家自然科学基金(19991480)资助

1) E-mail: zhangxm@mail.ihep.ac.cn

它知道何时应该启动哪个运算模块或者哪种服务模块。算法模块是数据处理的中心，一套完整有序的算法模块可以实现特定的物理任务。而服务模块则是为算法模块提供各种直接或间接的服务，图 1 可以看到许多重要的服务。它们包括提供算法模块从瞬态数据缓存中存取事例数据的服务(如 Event Data Service)，负责数据文件和瞬态数据缓存之间数据交换的宿存服务(Persistency Service)，信息打印服务(Message Service)，作业配置服务(Job Options Service)等。

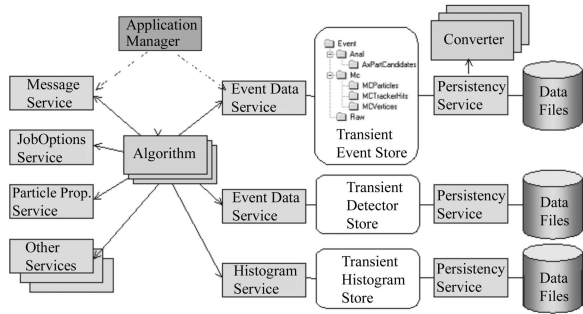


图 1 GAUDI 框架的体系结构

瞬态数据缓存是程序运行期间各功能模块所需事例数据，探测器描述数据和直方图统计数据所存放的内存空间。因此它只存在于框架运行期间。与它相对的是宿存数据文件(Persistent Data File)，宿存数据是在存储介质中存放的数据，不依赖程序运行。瞬态数据缓存的数据组织模式是树型，类似 Unix 中的文件系统。每个节点可以是一个数据对象或一组数据对象的容器。瞬态数据缓存可以规范和简化内存中的数据流动，大大减少功能模块之间数据和功能模块的之间的耦合，使得存储格式的选择更加的灵活和方便。

2.2 GAUDI 框架中的数据存取过程

用户在 GAUDI 框架上存取数据的过程主要是宿存数据文件和瞬态数据缓存之间的数据转换过程。在数据转换过程中，转换服务(Conversion Service)和转换器(Converter)在转换过程中是不可缺少的。不同的数据存储类型对应不同的转换服务，而且不同的数据类型需要不同的转换器。也就是说，特定存储格式的文件读取数据时需要调用专门的转换服务，而该转换服务一般因为需要处理多种数据类型而负责管理一组转换器。图2描述了用户模块读取数据文件的过程。

1) 首先，事例数据服务模块(Event Data Service) 接到请求后，首先在瞬态事例数据缓存(Transient Event Data Store, TDS)中搜索所需要的数据对象。如果数据对象当前存在于瞬态数据缓存，则返回值，程序结束。

2) 如果所需的数据对象不存在于瞬态事例数据缓存中，数据服务模块就会将申请递交给宿存服务(Persistency Service)。而宿存服务则负责根据所处理的存储类型将任务指派给合适的转换服务。一般转换服务都负责管理一系列的转换器，每种转换器负责特定的数据类型。因此转换服务又将任务根据所处理的数据类型交给专门的转换器。

3) 接着转换器从宿存数据文件中读取数据后，创建所要求的数据对象，赋值后将该对象返回给负责它的转换服务。

4) 转换服务再将该数据对象注册到瞬态事例数据缓存，最终数据服务模块将数据对象的指针返回给先前提出要求的算法模块。

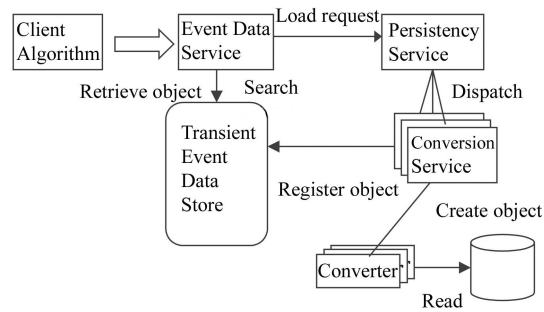


图 2 GAUDI 框架下从存储介质读取数据信息的过程

与读取过程相反的存储过程如下：

1) 在事例处理结束时数据服务模块遍历瞬态事例数据缓存，收集在用户运行配置文件(Job Options)中所要求存储的数据对象。

2) 将收集到的这些数据对象的指针传给宿存服务，由宿存服务再提交给负责该存储类型的转换服务，转换服务再将这些数据对象指针分配给相应的转换器。

3) 转换器则按一定的存储格式将取得的相应数据对象进行相应的格式转换，最后存入存储介质中。

3 BESIII 原始数据流动过程

在 BESIII 数据处理软件系统中，与原始数据有关的软件包主要包括原始数据管理系统(Raw Data Management System)，蒙特卡罗，事例重建，离线刻度，以及在线数据获取。如图 3 所示，原始数据的流动主要包括以下两个过程：

1) 蒙特卡罗软件包产生的 TDS 原始数据对象通过原始数据管理系统封装产生原始数据字节流文件。

2) 重建软件包和刻度软件包通过原始数据管理

系统从实验数据字节流文件中将原始数据读出,并以数据对象的形式存在瞬态事例数据缓存中进行读取.

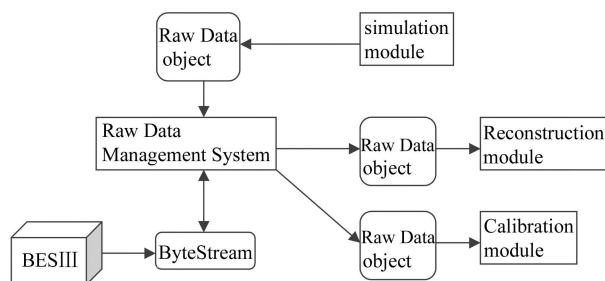


图 3 BESIII 原始数据流动示意图

从上面的过程,可以清楚地看出原始数据管理系统在原始数据流动中重要作用和主要任务.

4 原始数据管理系统的设计和实现

原始数据管理系统^[4]包括原始数据字节流格式的管理,瞬态数据库中原始数据组织模式的管理,以及原始数据转换服务.它主要实现两个功能:字节流文件的数据格式到瞬态事例数据缓存中的数据格式的转换以及瞬态事例数据缓存中的数据格式到字节流文件的数据格式的转换.

4.1 原始数据字节流存放格式

字节流文件的原始数据是按事例顺序安排的.图 4 是一个完整的事例,一般由事例头和事例数据组成.事例头记录了事例大小,run 号,事例号,事例类型,探测器名,以及记录各个子探测器数据位置的偏移地址表.事例数据部分由各个子探测器的数据头和数据部分组成.每个子探测器的数据头分配两个字,第一个字的前半部分和后半部分分别记录子探测器标识和数据大小,第二个字用来标识该子探测器数据的状态.

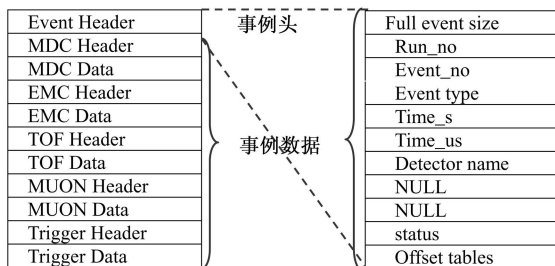


图 4 BESIII 字节流数据格式示意图

4.2 瞬态数据缓存中原始数据组织模式

瞬态数据缓存的数据组织模式是树型的.每个

节点可以是数据对象,也可以是一些小数据对象的容器(ObjectVector<ContainedObject>).无论是数据对象还是数据对象的容器,它们都是被唯一标识的.用户可以使用类似查找目录或文件的逻辑地址来存取数据,例如,如果要取出图 5 中 Event 数据对象,则可以用“/Event”来查找.

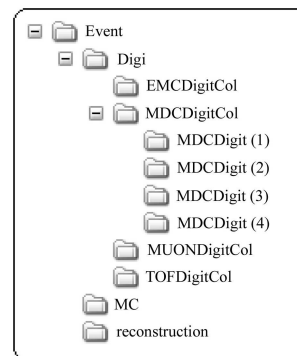


图 5 原始数据在 TDS 中的组织模式

图 5 给出了原始数据在瞬态数据缓存中的组织模式,其中 Event, Digi 都是从 DataObject 类派生的可标识的数据对象,Event 数据对象包含有事例号,run 号等属性,而 Digi 数据对象包含有关于原始数据的属性,如数据是否来自蒙特卡罗产生的.MDCDigitCol 是装有 MDCDigit 对象的容器,它定义为 ObjectVector<MDCDigit>.MDCDigit 类从 Contained Object 继承而来,每个 MDCDigit 对象含有一个 MDC 击中信息包括单元地址信息和 TDC/ADC 值.其他探测器也是采用相同的方法存储击中信息.根据以上的结构模式,用户模块可以很方便地利用 Gaudi 框架的数据服务取到所需的原始数据.

4.3 原始数据转换过程设计

为了实现上述两个数据格式的双向转换,原始数据转换过程的设计共使用了 7 种服务,原始数据经历了 4 种不同存放模式.这 7 种服务包括:输入服务,输出服务,解包服务,封装服务,瞬态数据服务,离线地址标识转换到在线地址标识服务以及在线地址标识转换到离线地址标识服务.4 种存放模式包括:字节流文件模式,单个事例字节流模式,临时事例缓存模式,瞬态数据库数据对象模式.

图 6 显示了双向转换过程中数据流动以及在各阶段各个服务所起到的作用.字节流文件数据到瞬态数据缓存数据转换过程可以分成下面 4 个子过程:

1) 事例输入过程 原始数据一般以字节流形式存储于文件中,由输入服务将数据以事例为单位顺序

从文件读到内存中. 这时内存的字节流数据是一个完整的事例数据;

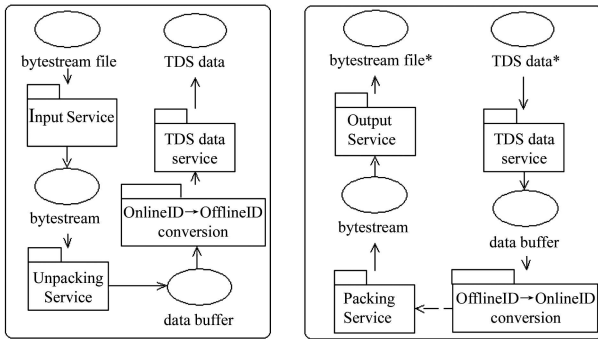


图 6 原始数据服务中双向数据流动过程

2) 事例解包过程 该字节流数据通过解包服务可以很方便地获得各部分的数据, 如事例头, 子探测器数据部分等, 解包服务继续把这些解开的数据依次填入临时事例缓存中, 方便下一次随机读取;

3) 地址标识转换过程 为了使原始数据可以存储到瞬态数据库的瞬态数据对象中, 临时事例缓存中的原始数据还需要做些必要的转换工作, 转换工作主要包括两个方面: 通过地址标识转换服务将探测器单元地址从在线地址标识转换成离线地址标识, 以及分离被压缩在一个字中的探测器单元地址和 TDC/ADC 信息;

4) 存入瞬态数据库过程 最后通过瞬态数据缓存的数据服务将临时事例缓存中的数据依次赋值给相应的瞬态数据缓存数据对象。

同样地, 瞬态数据缓存数据转换成字节流文件有类似的过程. 原始数据首先是以瞬态数据对象的形式存在于瞬态数据缓存中, 通过瞬态数据缓存数据服务读到临时事例缓存中; 接着经过一些格式转换如地址标识的转换, 以及按规定格式压缩数据等; 再由封装服务封装成单个事例字节流格式; 最后再由输出服务按事例顺序存到指定的字节流文件中去。

4.4 原始数据转换过程的实现

4.4.1 封装和解包服务^[5]中类设计和实现

为了获得较好的扩展性和重用性, 采用面向对象思想^[6]进行设计和实现, 可以做到: (1) 事例以及事例各部分数据的处理模块化. (2) 可以适用于多种数据存取介质, 如文件, 内存. (3) 对于探测器数目有很好的伸缩性, 即可以方便地增加或减少子探测器的数目. (4) 可以根据需要灵活选择不同的内存分配方式。

图 7 所示是用 UML 建模语言^[7]表达的设计类图,

设计了 Fragment, Header, Storage 和 Allocator 4 个主要基类来完成整个封装和解包过程. 其中 Fragment 和 Header 类用来描述原始数据, Fragment 类定义一个基本数据处理单元, 它知道该单元数据的存放格式, 并可以独立将该单元数据按预定格式读出和写入规定的存储空间. 它有两个继承类: FullEventFragment 和 SubdetectorFragment 分别负责整个事例数据和子探测器数据的处理. 相应地, Header 基类也有 FullEventHeader 类和 SubdetectorHeader 两个继承类。

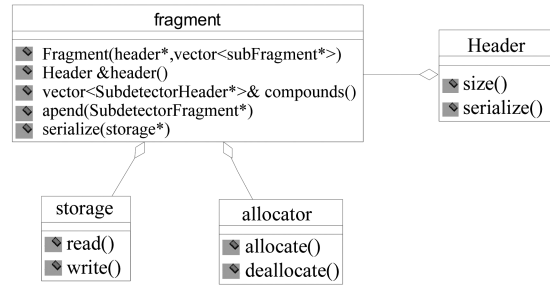


图 7 封装和解包服务中的基类关系图

Storage 和 Allocator 基类为了支持多种存储技术和内存分配技术而设计的. Storage 类实现在存储介质上的读写, 它有 3 个继承类 FileStorage, GzFileStorage 和 RawStorage, 分别提供对文件, 压缩文件和指定数据缓冲区的读写. Allocator 类负责运用不同的内存分配技术. 本软件包提供了简单内存分配和快速内存分配两种内存分配技术, 其中简单内存分配是保证线程安全的, 快速内存分配不能保证线程安全, 但是它可以达到快速分配内存空间的效果. 快速内存分配主要实现方法是: 首先分配一些等大的内存空间, 利用堆栈来管理, 一旦对象请求分配内存时, 就直接从堆栈中弹出一块, 不用的内存块不立刻释放, 而是压入堆栈等待重新再利用. 只有堆栈中内存块供应不足时, 才重新要求系统分配内存空间。

4.4.2 基于 Gaudi 框架的数据转换过程的类设计和实现

在 Gaudi 框架下实现瞬时数据存储和宿存数据存储之间的转换, 需要下列 3 种类的合作: IConversionSvc, IConverter, IOpaqueAdress. 在运行过程中, IConversionSvc 对象负责指挥整个数据转换过程的顺利进行, 它管理着一系列 IConverter 对象以便对不同的数据类型进行格式处理. IConverter 对象则负责具体的数据格式转换工作, 例如从 IOpaqueAdress 对象中获取数据信息来创建并填充瞬态数据缓存中相应的数据对象. IOpaqueAdress 也是完成数据转换服务中

不可缺少的对象. 它可以在宿存服务, 转换服务, 数据服务之间传递信息, 它的SvcType函数帮助宿存服务确定所要的转换服务, cId函数帮助转换服务确定当前应该选择的转换器, 还可以帮助转换器获得足够信息来创建所需的瞬态数据对象. 因此在原始数据过程中实现了原始数据转换服务, 4个转换器(主漂移室数据转换器, 量能器数据转换器, 飞行计数器数据转换器, μ 计数器数据转换器), 以及原始数据地址服务, 它们之间的协作如图8所示.

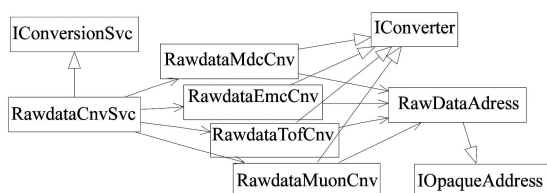


图 8 Gaudi 框架下实现数据转换过程中的类协作图

5 总结

本原始数据管理系统已经基本完成, 并对其做了详细的检查和基本的测试. 测试的内容包括: 数据一

致性, 占用的CPU资源以及运行速度.

在数据一致性测试中所使用的数据来自蒙特卡罗模拟产生的. 测试基本方法: (1) 使用原始数据转换服务软件包对瞬态数据缓存中存储的原始数据进行封装, 生成原始数据字节流文件; (2) 该原始数据字节流文件再通过原始数据转换服务软件包进行解包, 生成瞬态事例数据缓存的原始数据对象; (3) 最后由算法模块读出, 与(1)中的数据对象作比较, 判断是否一致. 最后的测试结果是前后两套数据完全一致, 这证明了该原始数据管理系统的可靠性.

占用的CPU资源以及运行速度的测试我们主要采用比较法, 即算法模块的数据输入分别采用ASCII格式文件直接读入和原始数据文件格式文件经过原始数据管理系统转换后读入, 最后比较它们对当前算法模块运行所造成的影响. 我们选用普通的PC机作为测试机器, 主漂移室快速重建系统作为测试算法模块. 测试结果是两种不同的方法对程序运行时CPU资源的占用和运行速度的影响相差不大, 这表明了原始数据管理系统的可行性, 它在以后的整个离线数据处理过程将不会成为瓶颈.

参考文献(References)

- 1 BES Collaboration. Preliminary Design Report of the BESIII Detector. December, 2003
- 2 MAO Ze-Pu et al(BES Collaboration). HEP & NP, 1992, **16**(9): 769—789(in Chinese)
(毛泽普等(BES 合作组). 高能物理与核物理, 1992, **16**(9): 769—789)
- 3 Cattaneo M et al. Gaudi Developers Guide. December 20, 2001
- 4 Bosman M et al. Event Filter TDR Event Conversion Service. June 28, 2001
- 5 André DOS ANJOS et al. UFRJ/CERN, ATLAS-DQ-EN-0006(EDMS:384821), May 2, 2003
- 6 Bjarne Stroustrup et al. The C++ Programming Language. America: Addison Wesley/Pearson, 1991
- 7 James Rumbaugh et al. The Unified Modeling Language Reference Manual. America: Addison Wesley/Pearson, 2004

Raw Data Management System on GAUDI Framework^{*}

ZHANG Xiao-Mei¹⁾ MA Qiu-Mei WANG Zhe WANG Da-Yong YOU Zheng-Yun MAO Ze-Pu
DENG Zi-Yan QIU Jin-Fa LIU Huai-Min LI Wei-Dong ZHANG Xue-Yao MAO Ya-Jun
YUAN Ye HUANG Xing-Tao JIANG Lin-Li ZANG Shi-Lei
(Institute of High Energy Physics, CAS, Beijing 100049, China)

Abstract The article describes the software package as a raw data management system which is mainly designed to convert persistent data in the form of a packed raw event byte stream into a form suitable for storage in the Transient Data Store of GAUDI. All its design and realizations are based on the GAUDI framework with Object-Oriented Design Patterns in C++, so the raw data management system is feasible and extensible.

Key words GAUDI, raw data, object oriented, BESIII data analysis

Received 29 March 2005

^{*}Supported by NSFC(19991480)

1) E-mail: zhangxm@mail.ihep.ac.cn