DOI:10.16836/j.cnki.jcuit.2012.03.005
第27卷第3期 成都信息工程学院学报 Vol.27 No.3
2012年6月 JOURNAL OF CHENGDU UNIVERSITY OF INFORMATION TECHNOLOGY Jun. 2012

文章编号:1671-1742(2012)03-0253-06

基于事件通知服务的雷达产品生成系统的数据分发改进

甘 兵[1,2], 朱 毅[2], 王红艳[1], 刘黎平[1]

(1.成都信息工程学院,四川 成都 610225;2.中国气象科学研究院灾害天气国家重点实验室,北京 100081)

摘要:雷达观测数据有回波强度、径向速度、速度谱宽及偏振参量等多种基数据,不同气象算法任务模块需要输入的基数据不尽相同,需要对其进行按需发送。针对事件通道和任务模块间存在大量不必要的通信,根据分布式雷达产品生成系统实时数据传输和松散的数据驱动通信模型的需要,结合CORBA命名服务和通知服务的特性,提出一个基于通知服务的实时数据传输模式。分布式雷达产品生成系统的数据流的基础由标准CORBA通知服务的核心部件构成,通过命名服务绑定算法任务名和对象引用。该模式提供了过滤机制,这种机制通过在事件提供者和消费者任务子模块之间高效的路由事件来支持松散的数据驱动的通信。

关 键 词:计算机应用技术;CORBA通知服务;过滤机制;分布式雷达产品生成系统

中图分类号:TN957.53 文献标志码:A

0 引言

图1是双偏振雷达数据处理软件系统,包括雷达产品生成系统和产品显示系统。

分布式雷达产品生成[1-2]软件是一个由大量雷达气象算法任务组成的软件系统,包括复杂的计算任务,复杂的控制流和数据流,并且实时性要求高,是一个兼有计算密集和IO密集的系统。系统主要以后台复杂气象算法数据处理为核心,辅以简单的用户控制界面。每个算法任务定义为一个CORBA对象,作为独立进程运行,任务间通过COBRA的事件通道传输数据,包括中间数据和最终产品。

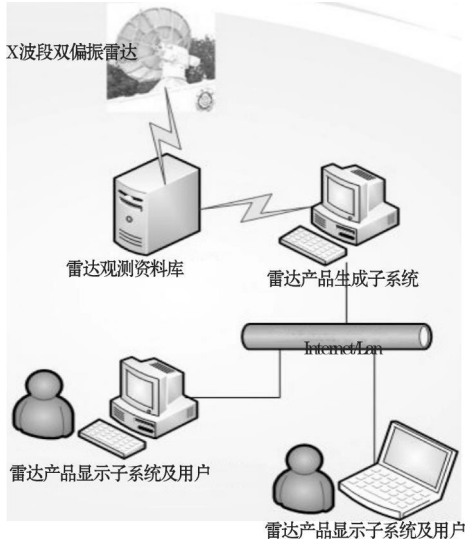

图1 X波段双偏振雷达数据处理系统结构图

事件通道[3]将所有的基本数据发送给所有的消费者任务子模块,每一个消费者任务子模块接收连接到同一个事件通道的提供者发来的所有的基本数据。事件通道将所有的基本数据发送给所有消费者任务子模块是对系统资源的一种浪费,因为很多消费者任务子模块会将大部分接收到的无关的基本数据进行丢弃(通道发送大量与接收任务无关的数据之前被迫存储一些基本数据并占用时间来连接发送)。这样将导致在通道和其接收任务模块之间有大量不必要的数据事件通信。每一种类型的数据通道只能处理一种类型的数据,使用多个事件数据类型需要多个事件通道,事件在独立的事件通道上是相同的(代码必须由消费者任务子模块写不同的处理这些数据事件),CPU必须花费时间判断和丢弃不需要事件,影响消费者任务的执行效率。使用ANY类型的数据在通道中无法进行分析,只有接收者任务接收到数据以后转换成一般类型结构后才可以处理,这对通道内部复杂职能的实现造成了困难。

针对以上的不足,需要对通道进行改进,增加任务模块间通信的智能性,事件通道需要对发送给所有的消费者任务子模块的数据事件做一些过滤操作(对从提供者发给接收任务模块的数据事件进行解释),这样接收任务子模块就不会接收到大量无关数据。

收稿日期:2012-02-18

基金项目:国家自然科学基金资助项目(40975013);青海省三江源自然保护区人工增雨工程软件开发及系统集成—探测资料处理软件开发资助项目(2009-Q-02)

1 通知服务、多线程技术和并行计算在雷达产品生成系统的应用

1.1 CORBA 通知服务[4] 技术原理

这部分简单的描述核心部件的结构在标准 CORBA 通知服务。论文中,这些核心部件构成分布式雷达产品生成系统的数据流的基础。

通知服务的体系结构如下:(1)通知发送者和通知接收者;(2)事件传送模式;(3)事件通道;(4)供应者管理器和消费者管理器;(5)服务质量和过滤。

1.1.1 事件通道

在客户和服务器之间加入事件通道,用 CORBA 通知服务实现事件信息的传递。事件通道使事件的提供者和消费者关系变得松散,无需相互了解对方,而只是同事件通道通信,这也适应了现在异构环境的发展。事件对于提供方来说担任消费者的角色,而对消费方来说担当了提供者的角色,这两种角色都是有相关的接口完成。在通信过程中事件通道在提供方创建一个提供方管理器(SupplierAdmin),再由提供方管理器为本地提供方创建一个消费者代理(ProxyConsumer);同样在消费方创建一个消费方管理器(ConsumerAdmin),再由消费方管理器为本地消费方创建一个供应者代理(ProvySupplier)。CORBA 通知服务使事件通道中可以传递结构化事件,此事件可以进行过滤等优化操作[5]。在事件的传递过程中,消费者任务子模块依照一定语法,描述过滤条件,形成过滤对象,可对事件提供者提供的事件和事件消费者接收的事件进行过滤,避免消费者接收自己不希望得到的数据事件。消费者类继承自 org.omg.CosNotifyComm.StructuredPushConsumer 类中接口 push-structured-event(),并实现该接口。当有数据从事件通道上传递过来时,可以通过该接口实现数据的接收和处理。

1.1.2 过滤机制

过滤器[6] 允许消费者任务订阅特定的事件,CORBA 通知服务中有两种过滤机制,一种是普通过滤机制,另一种是映射过滤机制。分布式 RPG 系统主要使用的是普通过滤机制,普通过滤机制决定事件是否前转,映射过滤机制不决定事件能否前转,普通的过滤机制的一个过滤对象包含多个约束,每一个约束有两部分组成:前一部分是事件类型的序列,给出了约束所适应的事件类型,后一部分是一个使用特定语法的布尔表达式字符串,给出了约束的内容。代理对象调用与其关联的过滤器对象合适的 match(匹配)操作决定是否是用户希望得到的事件。具体来说一个代理对象可能会关联多个过滤器对象,这些过滤器对象之间的关系直接影响到事件的转发策略,通知服务规定它们之间的关系是逻辑与的关系。也就是说只要一个匹配为真代理就转发这个事件,只有当所有的匹配为假,代理才会丢弃数据事件,通知服务的管理(admin)接口也可以关联一组过滤器对象,这是因为它和代理接口一样都继承了 CosNotifyFilter::FilterAdmin 接口。

1.1.3 结构化事件

结构化事件[6-7] 提供了良好定义的一个标准数据结构,可以映射到更多的数据类型,事件消息能存储在这个结构事件中。在通知服务中明确定义了处理结构化事件的接口,这样操作结构化事件算法就可以得到优化。结构化事件的格式包括事件头和事件体,事件头又分为固定域和变长域。事件体又分为可过滤部分和剩余部分。CORBA 结构化事件吸收了无类型事件的通用和易用的特点,而且引入了一定的类型约束的机制[8]。数据事件消费者任务可以使用结构化事件给通道代理添加过滤条件,以说明数据事件是希望得到的。

1.2 多线程和 OpenMP 在 TaskProdForecasts 的应用

为了不让负载过重的任务阻塞通道,文中应用多线程技术。接收数据和处理数据分解成两个线程处理,避免阻塞通道的目的(避免了该任务影响整个系统的运行情况),从而提高系统的可响应性。

OpenMp[9] 是一个并行程序的标准在共享内存的环境里。提供一系列语法给编程者,很容易并行化代码,程序员可以使用 OpenMp 提供的一系列语法并行化代码。如在 TREC 算法中使用 OpenMp 后,CPU 的最大使用率达到 99%,充分使用了计算机的运算能力,提高了多核 CPU 的利用率。在双核 CPU 加速比如表 1 所示。

表 1　在双核 CPU 加速比

基数据文件列表	未使用 OpenMp(ms)	使用 OpenMp(ms)	加速比
2	12734	9798	1.3
3	3265	2514	1.3
4	4206	2804	1.5

表1 使用OpenMp 和未使用OpenMp 的比较性能注意事项: 拥有正确且可执行的OpenMp 程序之后, 应该考虑其整体性能。可以利用一些常规技术和 Sun 平台专有技术改善 OpenMp 应用程序的效率和可伸缩性。

1.3　采用过滤机制对系统数据分发进行改进

消费者任务子模块给通道上的代理添加过滤条件, 消费者任务子模块把希望得到的数据以过滤条件的形式包装在过滤对象中, 然后把过滤对象和代理对象关联起来。增加了任务子模块间通信的智能性。使用过滤器订阅特定基本数据事件代码片段示例:

```
void ProdGenerator::RegisterRequiredData(CosNotifyFilter::ConstraintExpSeq &NeedBaseData)
{
NeedBaseData.length(1);
NeedBaseData[0].constraint_expr = CORBA::string_dup("VolDataType=='BREF'");
OR
NeedBaseData.length(1);
NeedBaseData[0].event_types.length(1);
NeedBaseData[0].event_types[0].domain_name = CORBA::string_dup("VolumeScan");
NeedBaseData[0].event_types[0].type_name = CORBA::string_dup("BREF");
NeedBaseData[0].constraint_expr = CORBA::string_dup("");}
```

实验证明, 在通知服务 4 种数据传输模式中, Push-Push 的效率最高, 可靠性最好, 系统也选用该模式。

数据接收代码示例如下, 任务子模块首先需要从希望得到的结构化数据中提取事件数据。

```
void ProdGenerator::push_structured_event ( const ::CosNotification::StructuredEvent & strutPPIData )
ACE_THROW_SPEC ((
::CORBA::SystemException,
::CosNotifyComm::Disconnected))
 ……
CommonPPIData *pPPIData;
if(!(strutPPIData.remainder_of_body>>= pPPIData)){
Send_log("Get invalid data");
return;};
 ……
```

采用了过滤机制的分布式雷达产品生成系统的优点是:

(1)按需发送数据事件

雷达观测数据有回波强度、径向速度、速度谱宽及偏振参量等多种基数据, 不同产品、算法子任务需要输入的基数据不尽相同, 按需发送数据后, 产品算法接收任务子模块不会接收到大量无关的数据。通道和其接收任务模块之间不存在大量不必要的数据事件通信。

(2)良好的灵活性和可扩展性

使用 CORBA 通知服务可以减少通道的数目。系统的数据流是动态变化的。用户可以动态增加或者删除任务, 从而极大的提高了系统的灵活性和扩展性, 实现"即插即用"。

(3)不同任务子模块良好的交互性

消费者任务子模块可以给通道代理添加过滤条件, 使用过滤器结合订阅来精确的接收希望得到的基本数据

事件。忽略判断丢弃等等细节。

(4)良好的可靠性和容错性

借助 CORBA 的成熟的标准,任务模块间可以可靠的数据事件传送,忽略重传等等细节。数据事件实时传输的测试实验中表明:所有数据均正常传送,未出现数据丢失的现象。系统在处理数据量大的双线偏振雷达数据时,仍能实时、稳定运行,具备与雷达扫描同步运转的能力。

系统也具有检测错误以及可以在任务子模块发生和已经发生的错误中恢复的能力。某个任务模块出错对其他的任务模块不会造成影响,系统照样可以正常工作,不会因为一个模块的错误或停止而造成整个产品生成系统的崩溃。当异常发生时,系统将会检测到哪个任务子模块发生异常,并将此异常任务退出,系统仍能正常工作。

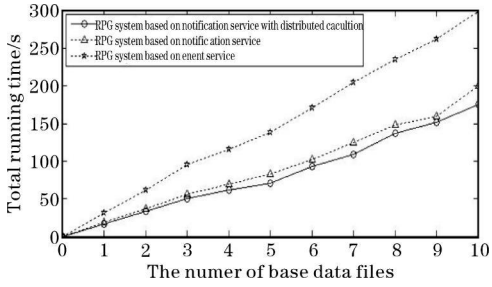

图 2 总的运行时间对比

(5)改善整个系统的数据处理效率

图 2 显示基于事件和通知服务的雷达产品生成系统每个体扫总的运行时间的对比。

2 采用了过滤机制的分布式雷达产品生成系统的架构

整个雷达产品生成系统分成 3 层:基数据读取层、产品生成层和和产品发布层。系统的各个任务子模块定义统一的 IDL 接口,雷达产品生成系统监控界面通过接口进行任务控制,获取和设置算法参数等。算法参数任务 IDL 接口定义如下:

```
interface Task: CosNotifyComm::StructuredPushConsumer{
void GetTaskAlgParams(out ParamSeq params);//获取任务算法参数
void SetTaskAlgParams(in ParamSeq params);//设置任务算法参数
oneway void ExecCmd(in Command cmd);
//修改任务运行状态
void CheckStatus();
//系统监控界面定时查询任务运行状态
void get-status(out TaskStatus st);
//获取任务运行状态
};
```

通过命名服务获取任务对象引用。任务的管理和任务间交互不需要知道任务的具体位置,只根据名字就可以获取对象引用,实现了网络的透明。算法任务名和对象引用的绑定通过命名服务。

```
template< class T> void BindTaskName(const CosNaming::NamingContext-var &naming-context,
T *taskptr){
CosNaming::Name name (1);
name.length (1);
name[0].id = CORBA::string-dup ("DRPG");
try
naming-context-> bind-new-context(name);
catch (const CosNaming::NamingContext::AlreadyBound &)
goto ok;
catch (CORBA::Exception &e) {
cerr<<"CORBA exception raised!"<<e<< endl;
return;
```

```
}
ok:
name.length(2);
name[0].id = CORBA::string_dup("DRPG");
name[1].id=
CORBA::string_dup(taskptr—> get_task_name());
//binding tasks' name by naming service
naming_context—> rebind (name, taskptr—>_this ());
}//get_task_name(); used to get tasks' name
```

其采用了过滤机制的基于 CORBA 通知服务的雷达产品生成系统架构如图 3 所示。

TaskProductVCS, TaskTempCappi, TaskProdCldId, TaskProdForcast, TaskWeatherMod 任务子模块以及发布产品任务模块 TaskProdDistrib 也是要和通道 TaskStatusChance 进行通信。

基于通知服务的雷达产品生成系统主要工作步骤如下:

(1)TaskDetectData 任务遍历指定的各个文件路径的基数据文件,将文件名存储到待处理文件列表中,接着读取雷达基数据文件进行质量控制及统一格式化处理。然后将处理后的结果推到 CommonPPIDataChannel 数据通道上。

(2)注册为 CommonPPIDataChannel 通道上的消费者任务将会被回调 push_structured_event()如果这个基本数据事件满足消费者任务设置的过滤条件(代理对象调用与其关联的过滤器对象合适的 match(匹配)操作决定是否是用户希望得到的事件);从该通道上获取一个结构化类型的事件,然后经过一系列的气象算法,生成气象产品或者中间产品。最后将处理的结果(产品)推到 ProductDataChannel 通道中。或者(中间产品)推到通道 TempPpiDataChannel 中。

(3)注册为 ProductDataChannel 通道的消费者任务 TaskProdDistrit 将收到所有的产品数据(该任务不使用过滤器),然后分发给有请求该产品的连接的客户端(PUP)。预报用户使用 PUP 对气象产品分析,进行研究和预报等工作。

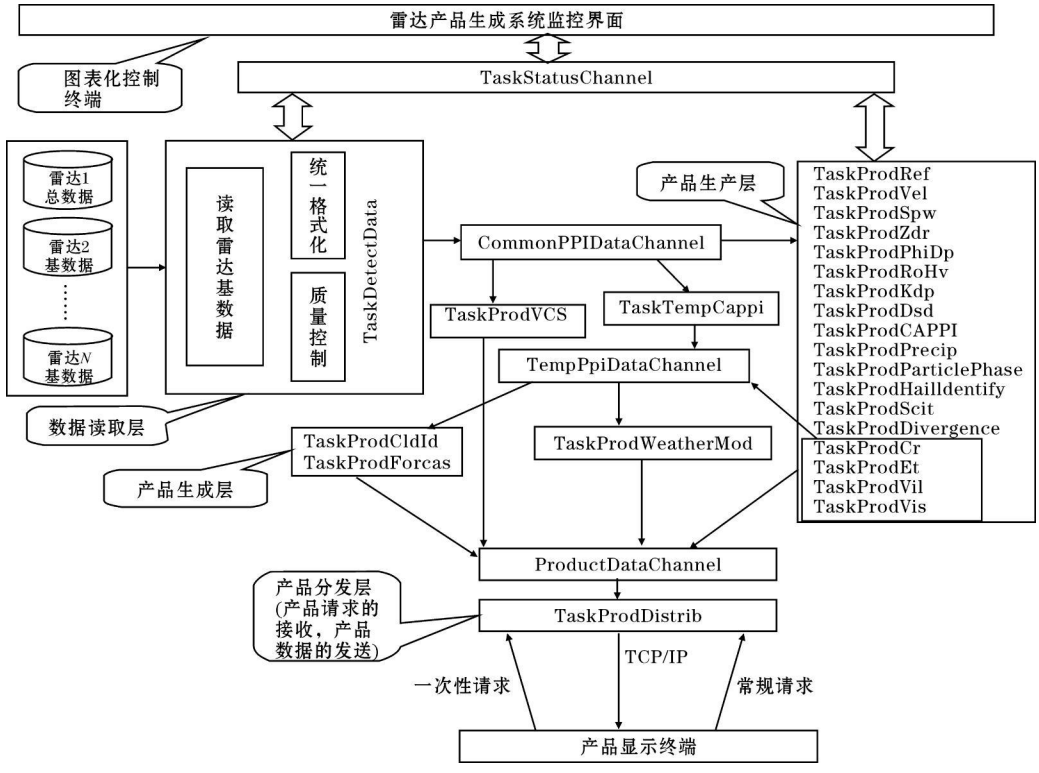

图 3　基于通知服务的分布式雷达产品生成系统架构图

这样就完成了分布式处理大计算量任务的工作。系统充分利用了计算机的处理能力，由通知通道进行统一调度和管理。使用 3 个事件通道保证不同的事件之间不会相互影响，从而保证了通道的质量和系统的效率。通知服务实现保护提供者不受异常的影响。

3 结束语

重点讨论了应用 CORBA 通知服务为任务子模块提供高效、可靠、灵活的数据传送，提高任务子模块间交互性，并可以使该系统获取良好的扩展性。并行计算和多线程技术在 TaskProdForecasts 的应用可以改善通道的阻塞情况并且提高多核 CPU 利用率。CORBA 通知服务在分布式雷达产品生成系统应用方面的实际应用中具有推广的价值。

参考文献:

[1] 杨爱军，王红艳，分 布式雷达产品生成系 统设计[J].山东 农业大学学报(自 然科学版)，2011，42(4)：543—545.

[2] Zabrai A，Zhongqi Jing. A distributed architecture for the WSR-88D (NEXRAD) radar productgenerator (RPG)[C]. Aerospace and Electronics Conference，Proceedings of the IEEE 1997 National，1997，1：302—307.

[3] Miehi，Henning. Steve V' moski. Advanced CORBA Programming with C++[M]. Addison-wesley，1999.

[4] Object Management Group. CORBAservices：Common Object Services Specification[EB/OL]. http://www.omg.org1997-07-04.

[5] 郑先容，陈强. CORBA 构件模型的通告服务集成研究与实现[J].计 算机应用 研究，1996：104—113.

[6] 彭宏，韩仲平. 基于 CORBA 的非耦合异步多点通讯[J].计算机工程与应用，2000，36(7)：47—50，70.

[7] Object Management Group. Notification Service Specification[S]. OMG Document telecom，1999.

[8] Pradeep Gore，Ron Cytron. Designing and Optimizing a Scalable CORBA Notification Service[M]. 2006.

[9] Kang Su Gatlin，Pete Isensee. OpenMP and C++[EB/OL]. http://www.viva64.com/go.php ?url=113.

Improving Radar Products Generating System's Data Distribution Based on Notification Service

GAN Bing[1,2]， ZHU Yi[2]， WANG Hong-yan[1]， LIU Li-ping[1]

(1. State Key Laboratory of Severe Weather，Chinese Academy of Meteorological Sciences，Beijing 100081，China；2. Chengdu University of Information Technology，Chengdu 610225，China)

Abstract：Radar observations contain echo intensity，radial velocity，spectral width and polarization parameters and so on. Different meteorological algorithm task modules don't demand on the same base data. Facing to lots of unnecessary communication between event channel and task modules，on the basic of the requirement of scalable data-driven communication model that decouples suppliers from consumers and real-time data transmission of a distributed radar products generating system，this paper proposes a real-time data transimission model implemented by CORBA Notification Service. the core component of CORBA Notification Service constitute the basis of data flow of distributed radar products generating system. The proposed model provides filter policy designed to support scalable event-driven communication by routing events efficiently between suppliers and consumers at multiple points in a distributed system.

Key words：technology of computer application；CORBA notification service；filter policy； distributed radar product generation system