摘　要

随着经济全球化及 Internet 技术的发展，使工业设计、制造业不断呈现出全球化、网络化的发展趋势。不同地域企业之间的合作也越发明显。因此，异地异构系统之间的数据交换成为了数据交换领域研究的热点。

STEP(ISO10303)标准是国际上用于产品表达的数据交换标准。在发布的几十年里，发展异常迅速。已覆盖到机械行业的各个领域。其中，AP214 是面向汽车设计全过程的应用协议，涉及到机械设计和制造、工艺规划、产品信息等各个方面。但是 STEP 中性文件对于异地系统之间的数据传输与交换支持不够。难以实现异地合作者之间的数据交换和信息共享。可扩展标记语言 XML 凭借其网络适用性成为了网络信息传递的载体。本文主要基于 STEP 标准的 AP214，对 STEP 数据和 XML 数据的交换问题进行了研究，开发了基于 STEP/XML 的数据交换系统，包括前处理器和后处理器两部分，完成 STEP AP214 文件向 XML 文件及其逆向的转换，达到数据交换的目的。

首先，针对 STEP 标准的描述语言 EXPRESS 和 XML 之间的映射关系进行了研究。在 ST-Developer 工具环境下，通过 ROSE 库函数完成 STEP AP214 文件中几何配置信息的提取，找寻信息匹配的 EXPRESS 定义。根据 EXPRESS 与 XML 之间的映射关系和实例表达，完成 STEP AP214 向 XML 文件的转换。由于在此之前编辑了 STEP AP214 文件的 XML Schema 结构，因此生成的 XML 文件具有有效性。转换后的 XML 文件可作为数据源在网络上发布。

通过对 XML DOM 解析器的研究设计，可以解析网络发布的 XML 文件，在系统内存中形成以 AP214 文件中实体、属性为结点的 XML 文件结点树。通过调用 DOM 接口函数，完成 DOM 结点树的结点信息提取。利用 EXPRESS Complier 将 AP214 的 AIM(应用解释模型)中对应的实体转换为 C++类，通过 ROSE 函数创建类对象和 DOM 树中结点信息的赋值过程，生成实体实例。完成 XML 文件向 STEP AP214 文件的转换。供支持 STEP 格式的系统使用。

最后，通过一具体实例证明 STEP/XML 数据交换系统可以实现异地异构系统在基于 Internet 上的产品数据交换。

关键词： 数据交换；XML；STEP；EXPRESS

ABSTRACT

As the development of global economy and Internet technology, there is an international trendence on the fields of inderstries design and manufactury. Different districts's enterprices cooperations become more and more obvious. So data exchange being the emphases among heterogeneous systems in different districts.

STEP (ISO10303) standard is the criterion of international data exchange used for product expression. In these decade years of its release, STEP expanded speedily and had covered with various mechanical scopes. AP214 is the application protocol oriented the whole process of auto design. It contains mechanical design, manufacture, technics layout, product information and other aspects. But there are not enough supports for STEP neutral files to transfer and exchange data among systems of different districts, then, it is also difficult to implement data exchange and share. Due to its Web flexibility, XML being the carrier of Web information transformation. This issue mainly studied the data exchange problems between STEP and XML based on STEP AP214, develop a STEP/XML system of data exchange, and include pre-processor and post-processor, to achieve the aim of data exchange from STEP AP214 to XML and their reverse.

Firstly, the mapping relationship between XML and EXPRESS is studied which define STEP standard. Under the condition of ST-Developer, distill the information of geometry and configure in STEP AP214 file used by ROSELIB functions. Then, match EXPRESS definition corresponded, complete the conversion from STEP AP214 to XML according to the mapping relationship and instance representation. Just because I have compiled the XML Schema structure of STEP AP214, the XML file we have got is valid and could release on Web being data fountain.

Throngh design the XML parse of DOM, we can parse the XML file which release on Web. Generate the node tree within system that the nodes are entities

and attributes of AP214. Sequencely, make use of functions of DOM nodes, and then distill the node information of DOM node tree. We must transfer the corresponding entities of AIM (Application Interprise Model) in STEP AP214 to C++ classes, produce entities instances just through ROSE fuctions build classes objects and endow value from DOM nodes information. At last, accomplish the conversion from XML file to STEP AP214 file, and then provide systems which support STEP style for operation.

Finally, prove system of STEP/XML data exchange can realized the product data exchange based on Internet among heterogeneous systems in different districts through a concrete instance.

Keywords: data exchange; XML; STEP; EXPRESS

哈尔滨工程大学
学位论文原创性声明

本人郑重声明：本论文的所有工作，是在导师的指导下，由作者本人独立完成的。有关观点、方法、数据和文献的引用已在文中指出，并与参考文献相对应。除文中已注明引用的内容外，本论文不包含任何其他个人或集体已经公开发表的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者（签字）：付秀娟

日期：2009 年 3 月 12 日

哈尔滨工程大学
学位论文授权使用声明

本人完全了解学校保护知识产权的有关规定，即研究生在校攻读学位期间论文工作的知识产权属于哈尔滨工程大学。哈尔滨工程大学有权保留并向国家有关部门或机构送交论文的复印件。本人允许哈尔滨工程大学将论文的部分或全部内容编入有关数据库进行检索，可采用影印、缩印或扫描等复制手段保存和汇编本学位论文，可以公布论文的全部内容。同时本人保证毕业后结合学位论文研究课题再撰写的论文一律注明作者第一署名单位为哈尔滨工程大学。涉密学位论文待解密后适用本声明。

本论文（☑在授予学位后即可　☐在授予学位 12 个月后　☐解密后）由哈尔滨工程大学送交有关部门进行保存、汇编等。

作者（签字）：付秀娟　　导师（签字）：郑光华

日期：2009 年 3 月 12 日　　2009 年 3 月 12 日

第1章 绪论

随着科学技术的不断发展，CAD(计算机辅助设计)软件在现今的工业设计、制造业领域有着举足轻重的地位。由于工业集成化趋势逐渐明显，各CAD软件之间的沟通与共享也越来越多。因此，当今主流的CAD软件正朝着集成化、智能化、网络化的方向发展[1]。STEP(产品数据的表达和交换)标准是关于数字化产品信息表达与交换的国际标准，它提供了一种不依赖具体系统，能够描述产品全生命周期的中性机制，在产品的全生命周期做到信息共享。XML(可扩展标记语言)是一种描述数据和数据结构的语言，具有自描述性及可扩展性。可以在任何两个遵守XML模式的应用系统间进行数据交换和信息传递，各系统在收到数据后又可以根据需要自行处理[2]。XML的这些优点，使其非常适合于网络异构系统间的数据交换。因此，将STEP标准与XML语言相结合的数据交换方法(即STEP/XML数据交换方法)的研究与应用是主流CAD软件间数据交换的趋势之一。

1.1 课题研究的目的和意义

正是由于CAD软件已经逐渐朝着集成化、智能化和网络化的方向发展，不同的CAD系统之间的交流与共享就成为了不可或缺的环节。如图1.1所示，各CAD软件是由不同公司开发完成的，其建模和造型思想都会有所不同。再加之网络协同设计的出现，也给异构CAD系统之间的数据交流提出了新的挑战。因此，基于STEP/XML的数据交换技术的研究已被提上日程。主要表现在两个方面：

1、为了实现异地异构CAD系统之间的数据交换

在经济全球化发展的时代，现代工业的集成化越来越明显，特别是汽车、飞机等制造工业。产品的整个设计制造过程已经不再局限于单一的地点，完全可以在全球范围内，由不同领域的专家共同合作完成。另外，计算机集成制造系统CIMS(Computer Integrated Manufacturing System)的理念和技术得到了广泛的应用。特别是各企业之间以及企业内部的不同部门之间的并行设计，

计，围绕供应链的协同开发等等[3]。这就需要不同的企业之间、不同的 CAD 系统之间实现互操作。在地域间及应用系统间的数据交换，多采用单独开发特定数据交换程序来实现，有点对点的交换和星式交换方式。然而，过多的单点连接会增加系统的复杂性和不稳定性，也不存在应用的广泛性。

STEP AP214 文件可以以中性文件（Part21）的形式，通过标准数据访问接口(SDAI)来完成不同 CAD 系统之间的数据交换，像 Pro/E、UG、CATIA 等 CAD 软件都支持 STEP 接口。彼此之间都可以通过 STEP 中性文件形式达到动态数据共享。但考虑到地域的广泛性，不同工作地点的用户若想实现数据共享，必须要借助 Internet 技术帮助。然而 STEP 中性文件形式不能与 Web 很好的结合支持网络的传输功能。应用 STEP 中性文件进行网络传输会大大降低系统的效率。因此，XML 语言的网络适用性可以在此发挥巨大的作用。

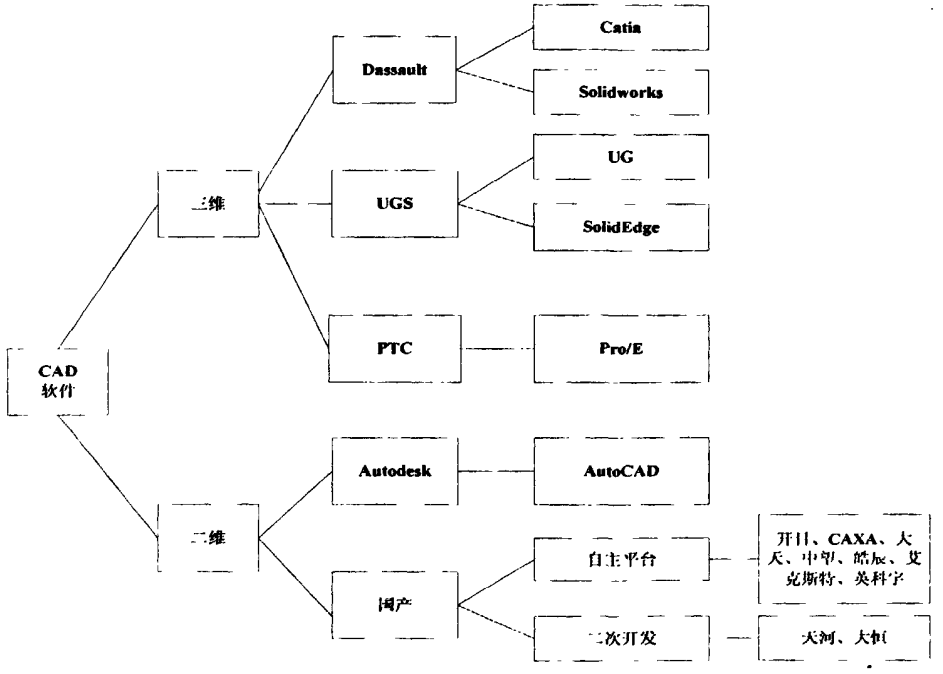

图 1.1 各 CAD 软件所属公司分布图

2 、为了满足网络化信息共享的要求

传统的产品信息共享技术是在传统的信息技术基础上实现企业内部产品的信息集成过程，它有效地支持企业的产品并行设计，提高产品开发速度，缩短产品开发、生产、管理和销售周期，降低产品成本。例如产品数据管理 PDM、工程数据管理 EDM、产品信息管理 PIM、技术文档管理 TDM 等[4]。

然而随着市场竞争日益激烈，信息共享不仅局限在企业内部，多企业之间的合作也屡见不鲜。随着 Internet 技术的广泛应用，网络化的信息共享已成为多企业之间合作的基础。它旨在建立共享、集成、并行、协作的产品开发模式。Web 技术是实现网络化信息共享环境的主要技术支撑。各企业之间的合作人员若想获取合作方有用的数据资源，必须通过网络信息共享来实现。对于 CAD 产品开发人员来说，其中关键的问题是，如何在 Internet 上获得不同的 CAD 系统所输出的 STEP 中性文件数据信息，这就需要选择一种合适的信息载体来代替 STEP 中性文件完成网络上的传输任务。提供强大 Web 服务的 XML 语言成为了首选。

XML 已成为标识 Internet 文档结构和内容的标准语言，它可以通过简单对象访问协议(SOAP)在 Web 上轻松的传输，解决 Internet 环境下数据的传输问题，保证系统的效率，因此用 XML 语言作为数据交换及信息共享的中间载体无疑是正确地选择。XML 支持 Internet 上的网页格式，并且可以被统一资源定位地址 URL(Uniform Resource Locator)所引用。这样就可以轻松地把 XML 文件信息发布到 Internet 上。用户就可以通过 XLink 和 XPointer 实现与网络上 XML 文档的连接[5]，实现网络化的信息共享。

因此，将 XML 这种网络信息的载体语言应用于数据交换中，可以充分发挥 XML 语言的优势，已成为数据交换领域的发展趋势之一。

1.2 课题的研究现状

1.2.1 XML 用于数据交换的研究现状

数据在不同的信息实体之间交互的过程即为数据交换。数据交换由于能够加快企业间业务流程、提高业务的效率、降低人力、物力、成本等诸多优点，越来越受到人们的关注。

XML 作为跨平台和异构系统之间数据交换的中间载体，极大的促进了数据交换的应用与发展。XML 应用于数据交换领域首先出现在电子商务行业。现在已发展的比较成熟。目前，国际上有很多机构都成功地建立了基于 XML 的电子商务架构标准。如 OBI（Open Buying on the Internet），BizTalk[6]。OBI 采用了 EDI（Electronic Data Exchange）的消息模式，它的架构描述都采用

XML 的消息模式来表达。BizTalk 是通过一个中央 Web 网站，提供 XML 存储管理和 Schema 验证等功能，来实现各个子系统的信息共享和信息验证。

DvadiCehnug，SD.Le 等人提出了一种分布式和可升级的 XML 文档处理架构并应用于具体的数据交换中[7]，该结构采用 XML、XSLT、HTTP 和 Javasevrelts 等技术的互通应用，具有很好的通用性、灵活性和可扩展性。

Poylly.M.S. Poon 提出采用 XML 模式来定义 Web 通信的消息属性[8]，不同的应用系统间基于此统一的定义来产生 XML 文档实例，应用于各自操作系统中实现数据的交换过程。其具有不局限于单一系统的网络信息共享的功能。

Kenneth.Chin,WeiLu 在研究 XML 结构的基础上，提出了一种基于 XSLT 的平行的 XML 映射实现方法。重点研究了 XML 向关系数据库的数据交换。利用 PXP[9](一种利用 DOM 解析器的树形结构来平行映射内存中结点信息的方法)结构来完成 XML 的映射，达到 XML 向其他形式数据交换的目的。

在国内，基于 XML 的数据交换技术也被许多学者研究和探讨。

中国科学院针对银行业务中解决各分支银行向总行实时报送数据业务的现状，提出基于 XML 的 Socket 方式实时数据交换，Socket 是套接字技术，用来实现 Client/Server 方式的网络程序设计，以实现不同地域、不同系统之间地实时数据交换和信息共享[10]。

北京航空航天大学针对大型系统中实现跨平台高效地、便捷地、可扩展地、无歧义的数据交换，采用 XML 作为数据交换的基本技术。利用 XML 的可扩展性和网络适用性，并结合大型网络体系结构提出了一个多级交换的模型，并设计和实现了数据交换的内部结构[11,12]。

徐享忠、王精业等人在分析了 XML 作为应用程序间的数据交换格式的优良特性之后，将其应用到分布式仿真系统的信息集成中[13]，研究了采用基于 XML 的数据交换格式时系统的体系结构和系统的主要组成部分。

谢莉莉、林春梅等人讨论了用 XML 实现数据交换的方法，以及基于 XML 数据交换平台的应用前景，着重研究了基于 XML 数据交换平台原型系统的设计和实现[14]。

现在，基于 XML 的数据交换平台已被应用于许多的领域。随着 Internet 技术的不断发展，XML 凭借它的可扩展性和网络适用性正逐渐被越来越多的

领域所采用，已成为数据交换语言的首选。

1.2.2 基于 STEP/XML 数据交换的研究现状

当前，把 XML 语言和 STEP 标准相结合，形成了 Internet 环境下新一代信息集成的方法，国外有许多科研单位如 ISOTC184/SC4 工作组、Product Data Integration Technologies 公司、Object Management Group 公司等都在从事这方面的研究工作。

ISO10303(即 STEP 标准)意识到自身难以满足工业制造领域的发展要求，于 1999 年新开发了 Part 28 标准，称为"EXPRESS 模式和数据的 XML 表达"，实现方法是：用 XML 表示 EXPRESS 定义的数据。为基于 STEP/XML 的数据交换提供了理论基础。

美国 STEP Tools 公司开发了一种支持 STEP 数据共享的 XML 编程技术。集成在应用工具 ST-Developer 中，其中有很多功能模块，可支持多个 STEP 标准中的应用协议。并支持应用协议的一致性检测等。旨在方便用户基于 Internet 技术实现 STEP 数据地处理。目前他们正致力于 AP238 的开发完善及 CAD 与 CAE 的集成技术研究。

德国 Fraunfofer 研究所提出分布式开发环境下应用 shared 3D viewer 协同设计产品模型的方法[15]。shared 3D viewer 主要是 STEP 数据和 XML 语言在分布式环境下的可视化研究。但是它不能形成协同造型和编辑修改。

Lowa 大学的 Internet 实验室主要进行 Internet 环境下的协同设计研究，提出了基于 Web 的协同设计整体结构，称为 CyberView[16]。用 STEP 标准来表达产品模型，用 VRML 来实现协同设计的浏览。主要是实现 STEP 和 VRML 之间的转换。但其只能形成协同浏览，无法实现协同造型。

国内对基于 STEP 标准和 XML 的数据交换研究也有很多进展。

浙江大学的简琤峰博士采用 XML 作为 Internet 环境下产品信息知识表达语言[17]，充分利用 XML 语言在网络上的优势，建立符合 STEP Part21 文件的 XML 文档定义类型。重点研究了网络化环境下 STEP 数据的 Web 描述方法。

华中科技大学在研究 STEP 和 XML 进行产品信息共享的方法时，提出了通过对应模式间接转换的方法[18]，对不支持 STEP 中性文件的应用程序想要共享 STEP 数据的情况，则通过将 STEP 的语法和具体编程语言 Java 语言联

系起来，把 XML 作为中间转换实体来实现信息的共享。他主要是利用 Java 的 JAXB 来设计一个转换器，将 XML 模式生成 XML 文件。但是他并没有系统地建立与具体 STEP 应用协议相对应的 XML 模型。

东南大学的仇晓黎博士主要针对不同企业异构系统之间的数据交换方式不被其他系统识别的情况，采用 STEP 标准与 XML 语言相集成的方式，他指出 STEP 标准可以统一数据格式，便于信息交换，但 STEP 的信息不能在网络上发布，提出借助 XML 技术来实现 STEP 的对外发布。利用 STEP Tools 公司的 ST-Repository 作为中间转化工具，并针对 STEP 标准中的 AP203 协议完成了 STEP 数据的 Web 描述[19,20]。

大连理工大学研究了基于 XML 的系统设计中的异构产品信息集成[21]。主要在分析、综合了 STEP 技术和 XML 技术的基础上，指出 XML 在网络化产品信息集成中的作用，研究了 XML 的 Web 数据模型，同时，结合新一代网络技术 Web Services 构建了一个基于 XML 的协同设计网络产品信息集成系统框架。

上海交通大学在分析了 STEP Part21 中性文件作为数据交换手段的不足之后，讨论了 EXPRESS 语言到 XML 语言的映射机制，并结合 XML 相关技术，提出了一个制造业企业之间数据交换系统的简单模型[22]。

1.3 本课题的主要研究内容

为了实现异地异构 CAD 系统之间的数据交换及信息共享，以及适应当前网络化信息共享的要求，主要是针对 STEP AP214 文件的网络化共享，本课题以 ISO10303 中 Part 28 为理论基础，对 STEP AP214 文件和 XML 文件之间的数据交换进行了研究，并开发了从 STEP AP214 文件向 XML 文件的转换器及其逆向转换器，重点实现两个应用处理器。具体工作分为以下几个方面：

（1）STEP 标准的描述语言 EXPRESS 与可扩展标记语言 XML 之间的映射研究。要实现 STEP AP214 文件和 XML 文件的交换过程，首先要找到 EXPRESS 和 XML 之间的映射关系。其中，以 ISO10303 Part28 为理论基础，包括模式映射、数据类型映射、约束规则映射等等。

（2）STEP AP214 文件中实体及属性信息的提取。了解 STEP Part 21 文件的语法及语义，找到 AP214 文件的 EXPRESS Schema 结构，利用 STEP Tools

公司提供的 ROSE 类库和 EXPRESS 编译器，可将 EXPRESS Schema 编译成 C++类。再将其与 VC++ 6.0 很好的集成，通过 ROSE 库函数的调用，以及应用程序的编译执行，最终完成所需实体属性信息的提取。

（3）XML 文件解析器的设计开发以及结点信息的提取。利用 DOM 这种 XML 解析器中的相应函数接口调用，来实现 XML 文件的解析过程，并在系统内存中生成 XML 的文件结点树，通过定位结点方法可对 XML 文件中的结点遍历并提取所需信息。

（4）前、后处理器的开发。找到 EXPRESS 语言和 XML 语言之间的映射关系之后，建立符合 STEP AP214 物理文件的 XML Schema 结构，确保转换生成 XML 文件的有效性。通过对 STEP 数据的分析找寻匹配的 EXPRESS 表达，再利用映射关系和应用程序的编译，及其实体属性的赋值过程，完成 STEP AP214 文件向 XML 文件正向和反向转换地实现，即前处理器和后处理器的开发过程。

第 2 章 STEP/XML 数据交换系统

本文主要是通过结合STEP标准和XML技术的独特优势来实现数据交换系统的主要功能，分别利用 STEP 标准的强大信息描述能力和 XML 语言优越的网络传输能力，以达到异地异构系统之间数据交换的目的。

2.1 STEP 标准

2.1.1 STEP 标准的基本原理及构成

STEP（Standard for the Exchange of Product Model Data）产品数据的表达和交换是国际标准化组织(ISO)制定的系列标准(ISO10303)，是一个关于产品数据的计算机可理解的表达与交换的国际标准[23]。目的是提供一种不依赖于具体系统的中性机制，用来建立包括产品整个生命周期的，完整的，语义一致的产品数据模型。从而满足产品生命周期内各阶段对产品信息的不同需求，以及保证对产品信息理解的一致性。

STEP 标准支持完整的产品数据模型，除了支持几何信息外，还支持全面的非几何数据。如:性能、公差规范、材料性质和表面加工规范等。其原理是采用一种用 EXPRESS 建模语言描述的中性文件机制，表达产品全生命周期中的信息定义和数据交换的外部描述。从而把产品的数据表达从数据交换的实现方法中分离出来。这种中性文件形式是一种无二义性表达格式，并且此格式可由计算机解释。

STEP 标准的体系结构可以分为三层：应用层，逻辑层，物理层。这三层组织结构在形式上类似于数据库的外模式、概念模式、内模式三级模式结构。整个 STEP 规范是一个内容庞大的标准，由如下六部分组成：描述方法、实现方法、集成资源、应用协议、一致性测试、抽象测试。各个部分之间的关系和范围如图 2.1[24]所示。其中，每一类又包括若干个部分(Part)。已制定的有关设计制造方面的 STEP 应用协议(Application Protocol，简称 AP)有 38 个(AP201-AP238)。其中，与机械制造方面有关的 STEP 应用协议主要有：

AP203(几何定义)、AP214(面向汽车设计全过程的定义)，AP219(用于公差定义)，AP224(用于特征描述)，AP238(用于数控加工)等等。本文主要针对AP214 协议。

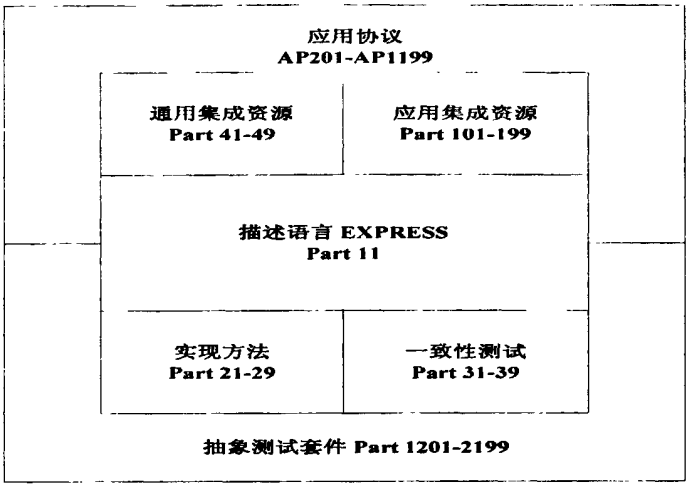

图 2.1 STEP 标准体系结构图

2.1.2 EXPRESS 语言和中性文件结构

STEP 拥有自己的信息描述语言—EXPRESS 语言，在 STEP 标准中的第11 部分。这种语言是目前唯一广泛采用的可以描述三维实体复杂性的语言，既能描述任意产品信息的结构，也能描述其间的约束关系。因此，利用EXPRESS 语言可以将 STEP 向任意应用领域扩展。

EXPRESS 数据模型主要通过模式(Schema)来表达，构成模式的基本元素有：字符集、注释、符号、保留字、标识符和文字。其具体描述的信息模型是由一个或多个模式组成。而 EXPRESS 语言本身并没有规定从 EXPRESS 数据模型生成实体实例的机制，STEP 标准中的 Part21 定义了一种中性文件格式以及由 EXPRESS 描述映射到中性文件的映射规则。这种文件是基于ASCII 字符的，被称为 Part21 physical file，简称为 Part21 物理文件，也作中性文件。用它来实现不同的应用或系统之间的数据交换，Part21 文件可被看作是 EXPRESS 语言所描述的某个数据模型中实体的实例。

STEP 中性文件，即 ISO10303-21 部分，是由基本字符串组成的连续自流符，由具有一定语义 Token 组成不同的数据类型和格式，例如分隔符、关键

字、保留字和简单数据类型等。STEP 中性文件是顺序文件，它用维尔斯句法 WSN（With Syntax Notation）定义具体的结构[25]，WSN 具有无二义性的特点。表示所有可能的概念模式，而不考虑任何固定的概念模式。STEP 中性文件主要是由头部段(Header Section)和数据段(Data Section)两部分构成。

头部段包含了整个交换文件的信息，在文件中必须而且只出现一次。STEP 头部段提供了三个标准实体：

文件描述(FILE_DISCRIPTION)：文件描述实体的属性包括文件内容的说明和后置处理的实施级别。

文件名(FILE_NAME)：文件名实体的属性包括文件名、建立文件的日期、作者姓名、单位、预处理器版本和文件审核人等。

文件模式(FILE_SCHEMA)：文件模式实体描述了数据段实体所引用的应用协议。

STEP 头部段的文件如下所示：

```
ISO-10303-21;
HEADER;                                          /*头部段开始*/
FILE_DESCRIPTION(('STEP AP214'),'1');            /*文件描述*/
FILE_NAME('ZHOU'),                               /*文件名*/
'2008-08-25T 10:20:31',                          /*日期和时间*/
('fxj'),                                         /*作者*/
('CIMS'),                                        /*作者单位*/
'PRO/ENGINEER BY PARAMETRIC TECHNOLOGY
 CORPORATION, 2007450',                          /*创建模型系统*/
FILE_SCHEMA(('CONFIG_CONTROL_DESIGN'))           /*所用应用协议名*/
ENDSEC;                                          /*头部段结尾*/
```

数据段主要包括转换文件产品的数据信息，是 STEP 文件的核心部分。部分数据段实例如下：

```
#53=CYLINDRICAL_SURFACE('',#52,2.5E1);
#54=ORIENTED_EDGE('',*,*,#42,.F.);
#56=ORIENTED_EDGE('',*,*,#55,.T.);
```

下面以 #53=CYLINDRICAL_SURFACE('',#52,2.5E1)为例，说明各个组

成部分的含义。其中，#53 表示实体标识符(Instance identifier)，简称 ID。CYLINDRICAL_SURFACE 为实体关键字，括号内的为实体的属性，‘ ’表示实体属性内的空字符串，#52 为#53 实体属性的下级实体标识符，则#52 所标识的实体即为 CYLINDRICAL_SURFACE 实体的下级实体。2.E1 为具体的属性值，2.E1 用十进制表示为数字 20。

2.1.3 应用协议 AP214

STEP标准的所有应用协议(简称AP)都是依照应用活动模型(AAM)、应用参考模型(ARM)和应用解释模型(AIM)三级模型体系建立的。分别采用IDEF1X /EXPRE-SS、 IDEF0、EXPRESS进行描述。AP214(Core data for automotive mechanical design process)，全称为汽车机械设计过程核心数据。于1992年由德国STEP中心(ProSTEP公司)负责组织开发，几乎世界所有汽车制造厂家都参与了开发工作，目前已被ISO组织接受为STEP标准[26]。AP214是基于特征的，面向汽车设计全过程的应用协议，包括机械设计和制造、工艺规划、产品管理等信息，可支持产品全生命周期的信息需求，主要由应用范围和信息需求等部分组成。

AP214以汽车设计制造为目标，定义了与汽车产品开发过程相关的核心数据，覆盖了整个产品全生命周期的数据模型，所涉及的机械设计与制造过程包括：整体设计、零件和装配件的设计、零件列表与材料清单、文档管理、NC编程、生产规划、制造过程、动力学和机械仿真、质量控制等。由于汽车开发过程涉及到几乎所有的机械设计，规划，加工，管理等过程，代表了机械行业的最高水平，所以AP214同样适用于汽车外的其他机械制造领域。目前流行的三维CAD软件如UG，Pro/E，Solidwork等都支持AP214协议，因此，AP214的应用非常广泛。

AP214是一个庞大的应用协议，在具体的应用过程中，参加数据交换的系统或应用程序并不需要使用完整的AP214协议。因此，开发人员将AP214协议划分为面向具体应用的若干个功能单元(Unit Of Function)简称UOF。每个功能单元又是由不同的功能模型组成的。这些功能单元分布在不同的一致性类(Conformance Classes,简称CC)中。AP214协议中共有20个一致性类[27],这些一致性类应用于不同的方面。如表2.1所示。

表2.1 AP214一致性类列表

一致性类	主要应用
CC1	3D组件设计形状表达
CC2	3D组件组装设计表达
CC3	框架表面形状表示的组成绘图
CC4	框架表面实体形状表示的组装绘图
CC5	格式数据
CC6	除形状表达外的产品数据管理
CC7	3D形状表达的产品数据管理
CC8	除形状表达外的结构控制设计
CC9	3D形状表达的结构控制设计
CC10	外形表达及绘图数据的结构控制设计
CC11	部件设计过程
CC12	带有特征和公差数据组成的设计过程
CC13	装配设计过程的有效控制
CC14	基于特征的设计
CC15	带有特征定位的基于特征的设计
CC16	对于3D外形表达的组成和装配的运动学仿真
CC17	测量数据
CC18	3D外形表达和运动学数据的组成和装配的结构控制过程设计
CC19	3D外形表达的组成和装配的结构控制过程设计（包括特征和运动学数据）
CC20	数据存储及检索系统

上述20个一致性类又是由不同的功能单元UOF组成。AP214中共有34个功能单元，主要涉及产品表达的外形尺寸、几何表达、尺寸公差和产品结构等方面。表2.2是AP214中UOF的逻辑分组。

表2.2 AP214 UOF逻辑分组

信　息	简称	信　息	简称
表面条件	C	绘　图	D
外部参考	E	形状特征	FF
几何表达	G	运动学	K
测量数据	MD	特征表达	PR
表达数据	P	产品结构	S
公　差	T		

每个逻辑分组又是由不同的功能模型组成。例如“G”代表几何表达，组成它的功能模型有G1、G2、G3、G4、G5、G6、G7、G8。他们分别代表了不同的涵义。如表2.3所示。

表2.3 AP214“G”功能单元中的功能模型

功能模型	表达涵义
G1	wireframe_model_2d
G2	wireframe_model_3d
G3	connected_surface_model
G4	faceted_b_rep_model
G5	b_rep_model
G6	compound_model
G7	csg_model
G8	geometrically_bound_surface_model

AP214覆盖的范围很广，目前，还没有商用的CAX系统对AP214协议完全支持。如Pro/E仅限于对CC1和CC2的支持，UG软件所集成的AP214主要是对产品结构表达UOF中的S1、S2, 几何表达UOF中的G2、G3、G4、G5、G8，表达数据UOF中的P1，它们主要集中在CC1与CC2中[28]。因此，对AP214一致性类的应用研究还在继续。由于AP214是针对汽车行业指定的国际标准，它几乎覆盖了机械行业的许多领域，对于机械产品的信息描述非常全面，因此本文选用STEP AP214协议。

2.2 XML技术

XML(eXtensible Markup Languange)是一种描述数据和数据结构的语言，可以保存在任何可以存储文本的文档中。XML具有许多灵活性、开放性的特点，使其在Internet和不同企业信息交流环境下颇具价值。它让使用不同系统和不同编程语言的人们能够相互交流和分享数据。其基础在于Web服务器采用XML在系统之间交换数据。交换的数据通常用XML标记，能使协议规范一致，比如在简单对象访问协议(Simple Object Access Protocol简称SOAP)平台上[29]。XML具备访问数据库和将数据传输到网络客户端的新功能。可以更快地生成应用程序，使应用程序更易于维护，并且可以在结构数据上利用其内部技术轻易地提供多种视图。XML的功能非常强大，同时对于人类或计算机程序来说，都容易阅读和编写，现在越来越多的设备都提供对XML的支持，应用非常广泛。

2.2.1 XML的技术规范和语法说明

XML是一种能够在各个领域进行设计的标记语言。XML的语法很简单，包括标记、元素、属性、注释、序码、声明等。它严谨、有效地定义了XML文件的结构。元素及标记是构成XML文件的主体，并能够创建可以使用程序或样式表处理的结构。

(1)标记可分为非空标记和空标记两种。每个标记都必须包括开始标记和结束标记，标记的开始标记和结束标记之间的部分称为“标记的内容”。非空标记的开始标记以“<”作为开始，用“>”作为结束。空标记以“<”开始，用“/>”结束。XML标记有三类意义：结构、语义和样式。结构用来表达XML文件的元素树。语义将元素单元与描述的实际事物联系起来。样式指定显示元素的方式。

(2)元素对命名的信息加以标识，并使用标记构建标识元素的名称、开始和结束。元素还可以包含属性名称和值，提供相关内容的其他信息并指出这些信息的逻辑结构。元素以树形分层结构排列，可以嵌套在其他元素中，但不可交叉嵌套。元素也分为非空元素和空元素两种类型。

(3)属性是指标记的属性，可以为标记添加附加信息，属性是一个名值组合，即属性必须由名字和值组成，属性必须在非空标记的开始标记或空标

记中声明。用“=”为属性指定一个值。属性值必须使用单引号或双引号。例如：“name”与‘name’描述的是相同的属性值。如果要使用左尖括号“<” 、右尖括号“>” 、连接符号“&” 、单引号“‘ ” 、或双引号“ ”时，必须使用字符引用或实体引用。如表2.4所示。

表2.4 XML实体引用表

实体引用	特殊字符	意义
&lt	<	小于号
&gt	>	大于号
&amp	&	和或连接符
&apos	‘’	单引号
&quot	“”	双引号

（4）注释是以“<!-”开头，以“->”结束，可以是一行，也可以是多行，但不允许出现嵌套，同时，字符串“-”、“<”、“>”不能出现在注释中。

（5）序码(Prolog)：XML文件的开始部分是XML文件的序码，在序码中通常出现XML声明和文件类型、模式声明等内容。

（6）声明是以“<?xml”开始，以“?>”结束，出现在XML文件的第一行，其中指明XML声明和文件类型声明等内容。

```
<? xml version = “1.0” standalone =“no”?>
  <! DOCTYPE People SYSTEM “out.dtd”>
    <people>
      <name>wanghao<name/>
      <sex>male<sex/>
      <birthday>1983.04.26<birthday/>
    <people/>
```

上面是一段简单的XML文件，由此段文件可以看出XML文件组成的各部分之间的关系。XML也规定了严格的语法，所有的XML文件都应符合XML的语法规则，这样的XML文件称为格式良好(Well-formed)的文件，同时还需要满足如下的要求：

（1）XML文件有且只有一个根元素；

（2）每个元素都必须有开始标记和结束标记；

（3）开始与结束标记的大小写应一致；

（4）元素彼此不允许交叉嵌套；

（5）属性值要用引号引起。

XML基础标准按其作用可以分为三类，第一类是包括XML文件标准和相关文件支撑标准的XML核心标准，其中关键的是XML语言标准，当前最新版本是2004年2月发布的XML1.1。第二类是XML处理标准，主要用于开发人员对XML文件进行处理，以实现XML所表达信息地生成、查询、传输和表达等。第三类指在XML基础标准上，面向特定领域，如数学、化学、法律等制订的“词汇表”，以便于在不同的组织和计算机应用程序间交换信息。由此可见，XML正在被越来越多的领域所应用。

2.2.2 XML核心技术

在我们编写XML文件时，由于应用程序的需求多种多样，所以有时需要对所编写的XML文件结构加以改变，当文件的结构改变时，就需要修改代码并添加新的元素。然而，当应用程序执行时出现了错误，如果没有显示的文件资料，就无法可靠地捕获XML文件中的错误，只能依靠运行代码步步寻找。这就给操作带来了不便。因此，要描述相同的内容，应采用相同的结构定义，这就需要提前定义XML的文档定义类型DTD和XML Schema。

1、文档定义类型 DTD

DTD是Document Type Definition(文档类型定义)的缩写。XML文件是一种描述标记的语言，它可以由DTD来定义结构(如文件中的元素、属性等)。DTD指定了文件的一系列规则，确保文件的一致性和有效性[30]。

DTD源于SGML(标准通用标记语言)，它的主要优点就是简单易用。由于其开发的时间比较早，现在许多的系统都对其提供了很好的支持。DTD一般由元素声明、属性声明和实体声明等构成，但并不是每一个DTD文件都完全包括这三方面。下面就是一段XML DTD的表达。

```
<? xml version="1.0" encoding="GB2312" standalone="yes" ?>
<! DOCTYPE 工程[
<! ELEMENT 工程 （项目，负责人）>
<! ELEMENT 项目 （任务，负责部门）>
<! ATTLIST 项目 projict_id ID #REQUIRED>
```

```
<! ELEMENT 任务（#PCDATA）>
<! ELEMENT 负责部门（#PCDATA）>
<! ATTLIST 负责部门 person_p_id IDREF #REQUIRED>
<! ELEMENT 负责人 （姓名，负责项目）>
<! ATTLIST 负责人 person_id ID #IMPLIED>
<! ELEMENT 姓名（#PCDATA）>
<! ATTLIST 负责项目 project_project_id IDREF #REQUIRED>
]>
```

DTD在校验XML文件的有效性方面非常有用，它的作用主要是规定在XML文件中所应出现的内容及文件的格式。但它也存在许多缺陷，例如，采用了非XML的语法规则、不支持多种多样的数据类型、继承关系和扩展性较差等，这些缺陷使DTD的应用受到了很大限制。

2、XML Schema

Schema基本思想就是为XML文件制定一种模式。便于XML文件地输出和保证XML文件的结构良好性。经过多年地改进与完善，2001年5月，XML Schema 规范成为了W3C的正式推荐标准。XML Schema是由一套预先规定的XML元素和属性创建的，这些元素和属性定义了文件的结构和内容模式[30]。XML Schema有两种重要的Schema模型：Microsoft XML Schema和W3C XML Schema。两种模式定义的XML文件结构都是相同的。前者Schema文件和其他XML文件的样式非常相似，后缀名为.xml，它由一组元素构成，其根元素是Schema，用于表明该XML文件是Schema文件。相应地，根元素Schema的结束标记一般在文件的末尾。下面是一段简单Microsoft XML Schema 的示例：

```
<? xml version="1.0" ? >
< Schema name="schema.xml" xmlns="urn:schemas">
  < ElementType name="姓名" content="textonly" model="closed" />
  < ElementType name="性别" content="textonly" model="closed" />
  < ElementType name="出生日期" content="textonle" model="closed" />
< ElementType name="员工"content="mixed" model="closed" order="seq" >
    < element type="姓名" />
    < element type="性别" />
    < element type="出生日期" />
  < ElementType/>
```

< Schema/>

W3C Schema的文件元素及属性等全部内容声明均以xsd:开头，文档后缀名为.xsd，文件必须以xsd:schema为根元素。本文主要是采用W3C XML Schema作为约束XML文件的标准。下面是一段简单W3C XML Schema的示例：

```
<? xml version="1.0"? >
<xsd:schema xmlns:xsd="http：//www.w3.org/2008/XMLSchema">
  <xsd：element name="员工">
    <xsd：complexType>
     <xsd：sequence>
        <xsd：element name="姓名"type="xsd：string"/>
        <xsd：element name="性别"type="xsd：string"/>
        <xsd：element name="出生日期"type="xsd：string"/>
      <xsd：sequence/>
    <xsd：complexType/>
  <xsd：element/>
<xsd：schema/>
```

XML只说明数据的结构而并不关心数据是如何具体描述的、数据是否正确，XML文件的强制性结构需求是通过DTD来实现的。DTD可以描述XML文件的结构，但它的灵活性不够，而XML Schema能够有效地解决上述问题，可以说XML Schema有其独特的优势。它不仅从数据结构和数据类型两方面为XML文件提供了强大的约束能力，而且对元素、属性和数据类型定义的继承关系提供了较强的语义验证能力[31]。最重要的是XML Schema本身就是一个XML文件，可以自由地对它进行处理。本文选用XML Schema来定义XML文件结构主要考虑其以下几方面优势：

（1）一致性：XML Schema直接利用XML的基本语法规则来定义文件结构，使XML实现了完美的统一，为XML地进一步发展奠定基础。

（2）扩展性：XML Schema对DTD进行扩充，引入数据类型和命名空间，使其具备更强的可扩展性。而且在一个Schema文件里可以引用其他Schema文件或在相同的文档中参考多种Schema。

（3）互换性：XML Schema可验证XML文件的合法性，并通过特定的映射机制，将不同的XML Schema进行转换，实现高层次的数据交换。

（4）规范性：XML Schema提供完整的机制以约束XML标记的使用，利用元素的内容和属性来定义整体结构。

（5）数据类型多样性：XML Schema支持丰富的数据类型，如时间类型(time)、URI应用类型、日期型(date)等等。

3、XLink

XLink是XML用来指向不同资源的链接方法。它扩展了HTML链接的概念，可以直接应用于Internet, 指向一个具体的URI(统一资源标识符)[32]，是专门为XML设计的链接语言。可以分为简单链接和扩展链接。链接反映了不同资源对象之间的关系，这些资源可以是本地的，也可以是远程的。而连接结构可以采用压缩、二进制或加密的形式。XLink下也包含有链接元素和链接属性。链接属性必须要在链接元素中声明。下面是XLink中几种常用的链接属性。

其中xlink:role和xlink:title属性用来描述远程资源，即链接所指向的文档或其他资源。title包含描绘资源的普通文本，描述网页的功能。role包含一个URI，指向更完整描述该元素的文档。Xlink:show属性定义了激活链接时如何显示具体内容，它的下面包括replace、new、embed、other、none五个值，分别用来表示替换资源、新建资源、插入资源、寻找其他资源和无联接操作。xlink:actuate属性定义了激活链接的方法，此属性包含四个属性值：onRequest、onLoad、other、none。分别表示：只有用户发出请求时，才可访问目标资源、文档加载到浏览器时即执行链接、应用程序查找其它标记、以决定何时执行链接、由程序默认行为来决定何时执行链接。

4、XSLT

eXtensible Stylesheet Language Translation(XSLT)是一种将XML文件转换为其他不同表现形式的语言。最常用的是转换为另一种XML文件。因为，尽管有些XML文件描述同一数据，但结构和元素名称却相差甚远。使用XSLT来转换XML文件，使其结构与XML Schema定义的结构相一致。或者在不支持XML的浏览器上为了在Web上发布信息，将XML文档转换成HTML在浏览器上显示。

2.2.3 XML的主要优势

XML通过定义语义标记，允许不同行业的专业人员开发与自己特定领域有关的标记语言，并可以将数据表示成为一种文本化的、易于阅读和程序理解的格式。这种格式又可以与Web技术很好的结合，充分发挥其网络适用性。并且这种数据表示不依赖于具体的硬件和平台。因此，XML的主要优势体现在技术优势和商业优势两方面：

1、技术优势

（1）XML具有可扩展性：XML是设计标记语言的元语言，可以在XML中定义无限的标记集来扩展其表达性能。

（2）XML结构化集成数据：XML可以以简洁的方式描述复杂数据，一方面简化了复杂数据结构的描述和操作工作量，另一方面也在一定程度上改善了软件的互通性。

（3）XML具有良好的自描述性：这些描述被称作“元数据(mete data)”，这个特点不仅可以方便网络的查询工作，又使得用XML语言书写的文档很容易被理解。

（4）XML便于网络传输：XML是当今互联网上被普遍使用的信息传递载体语言。结构化的XML文件，适合于各种不同的浏览模式。对于非结构化的文件，XML也可以利用XSLT的转换格式转换成特定的形式来支持浏览。并且XML支持现有的各种网络传输协议和网络安全协议，如HTTP，SMTP，SSL等。简单对象访问协议SOAP(Simple Object Access Protel)是XML能够在Web上轻松传输的关键技术。SOAP是基于XML的协议，主要用于在分散的分布式环境中交换信息，而且与XML一样不受特定编译系统和平台的限制，在此基础之上还添加了XML消息信封、消息头、消息体以及用于发送和接受SOAP消息的一组规则[33]。与HTTP网络协议组合之后，SOAP协议就可以随意地在Internet上传送XML内容。因此，近年来，基于XML的网上信息交换和系统集成技术发展非常地快，给XML语言与其他系统的信息集成提供了基础。

2、商业优势

XML使用非专有的格式，不受版权、专利、商业秘密或其他种类的知识产权的限制。XML的功能非常强大，同时对于人类或计算机来说，都容易阅读和编写，因而逐渐成为交换语言的首选。使用XML而不使用专有格式，人们就可以利用任何理解XML的工具来处理数据，还可以为不同的目的使用不

同的工具。XML使用户不必因为数据已经用专有格式编写好了或是接受数据的人只接受专有格式而限制在一个特定的程序上。

总之，XML与软件、硬件和应用程序无关，所以数据可以被更多的用户，更多的设备所应用。XML打破了网络上信息交换中的种种堡垒，使有价值的数据能够在网络之间流通、交换，从而达到信息共享的目的。

2.3 STEP/XML数据交换系统的整体结构

2.3.1 STEP标准与XML相结合

STEP 标准拥有比较成熟的信息描述机制，可面向产品的全生命周期进行信息描述。基于 STEP 的数据表达、数据交换和信息系统的集成机制正成为解决异构系统之间数据交换和信息共享的基本机制。我国也已经把 STEP 标准中的成熟部分，引用为相应的国家标准(GB/T 16656)[34]。XML 语言具有可扩展性及自描述性，又不依赖于单纯的应用系统，兼且适用于在网络上传输数据。因此，将 STEP 的统一机制和强大描述功能与 XML 语言的表达能力和信息传递机制有机地结合起来，就能较好地解决异地异构 CAD 系统网络上产品结构数据(如三维模型数据)的传送和交换问题。

目前，国际上对基于 STEP 标准与 XML 集成机制的网上产品信息交换、系统集成和协同设计的研究非常活跃。为此，国际标准化组织(ISO)也在积极探索研究，以求用 XML 来表述 STEP 数据，使二者优势得以互补。1999 年 5 月，ISO TC184/SC 纽约小组提交“XML Representation Of EXPRESS-Driven Data”，被定为 Part 28，即 STEP Part 28。为 XML 技术与 STEP 标准之间的集成奠定了很好的基础。

2.3.2 STEP/XML 数据交换系统的功能及结构

在 STEP 标准中，用 EXPRESS 语言来描述产品的模型信息，使用应用协议(AP)的形式来规定各个专业领域里的特殊应用。XML 语言中用 XML Schema 来限定 XML 文件的结构。本文所要开发的数据交换系统主要是完成 STEP 数据到 XML 数据及其逆向的转换。则两者之间的关系研究就变的尤为重要。如图 2.2 所示。STEP 文件和 XML 文件分别通过定义自身的 EXPRESS 语言和 XML Schema 之间形成映射关系，来实现两者之间的数据交换的。

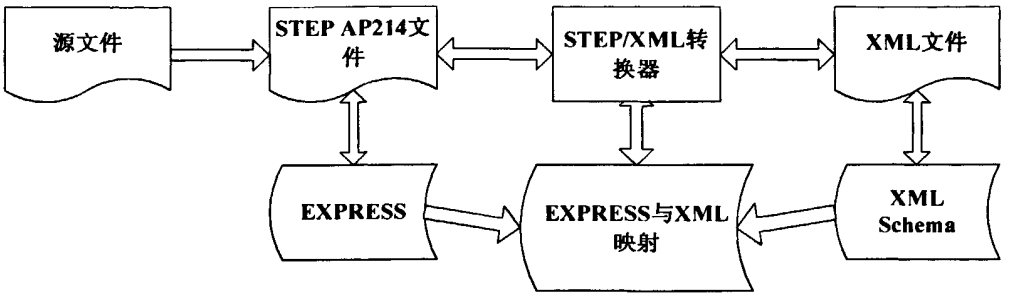

图2.2 STEP数据到XML数据的交换关系结构图

EXPRESS 与 XML 模式之间的映射关系将在第三章详细介绍，在这里不再说明。清楚了二者之间的转换关系，就可以通过具体系统的开发来实现基于 STEP/XML 的数据交换方法。图 2.3 是两异地异构系统采用 STEP/XML 的方法进行数据交换的结构图，主要是通过前处理器和后处理器的开发来实现。

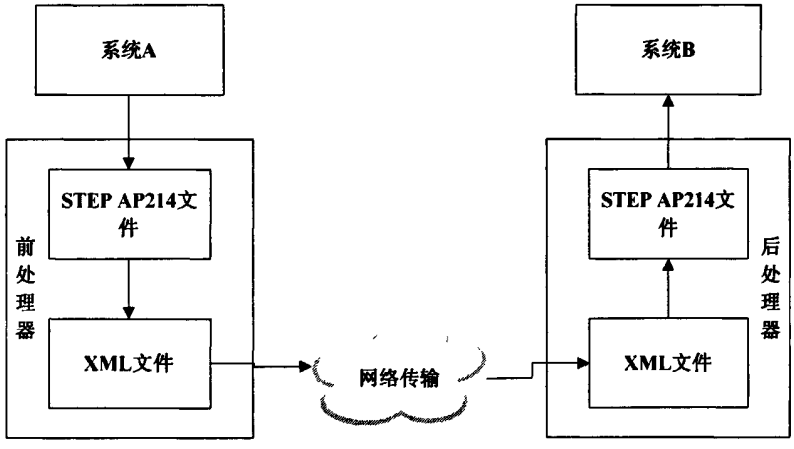

图 2.3 数据交换系统结构图

由图 2.3 可以看出，本系统对支持 STEP AP214 文件接口的软件和系统都是适用的，有广泛的可行性。由于 XML 文件在网络上的适用性，采用 XML 作为信息载体实现异地异构系统网络化数据共享的需求。前处理器主要是完成 STEP AP214 文件向 XML 文件的交换，后处理器主要是实现前者的逆向交换。前、后处理器的具体开发过程会在第四章作以详细介绍。

2.4 本章小结

在本章，对 STEP 标准的相关内容，包括标准的基本原理、基本构成、信息建模语言、中性文件结构、应用协议(主要是 AP214)进行了详细地介绍。

又对 XML 语言的语法规范和核心技术作以细致研究。总结出 STEP 标准的强大产品信息描述功能及 XML 文件的良好网络传输功能，将两者进行组合，既能发挥 STEP 标准强大的描述能力，又可利用 XML 语言适于网络传输的特点。因此，基于 STEP 标准体系和 XML 数据传送机制相结合的 STEP/XML 数据交换技术，逐渐成为异地异构环境下数据交换的关键技术，相信会很快成为制造业信息技术以及网络协同设计中最基本和最有用的基础技术之一。

第3章 EXPRESS与XML映射机制研究

众所周知，EXPRESS语言是STEP标准的数据描述语言，这种语言形式化地规范描述需要交换的数据信息的结构和约束条件，形成STEP在该领域的应用协议(Application Protocol 简称AP)。EXPRESS语言在STEP的第11部分定义，称为Part11。其不仅能描述数据结构，还能表达约束，这些一致性约束条件是表达产品数据的一种显示正确性标准[35]。而STEP Part21中性文件是EXPRESS定义的实例化表达。因此，要实现STEP标准与XML之间的集成，首先就是要建立EXPRESS与XML模式之间的映射关系。

3.1 EXPRESS模式与XML模式映射关系的建立

2000年10月，ISO组织完善了"XML Representation Of Express-Driven Data"为"XML Representation Of EXPRESS Schema & Data",并提交了XML对EXPRESS表述的两种早联编(early binding)方式和一种晚联编(late binding)方式。其中，两种早联编方式分别为ETEB(EXPRESS-Type Early Binding)与OSEB(Object Serialization Early Binding)[36,37]。为二者之间映射关系的建立提供了理论基础。

在早联编中，XML标记与EXPRESS数据模型中的数据类型以及属性直接对应。早联编为所有的EXPRESS schema提供一种方式，并不特别为某一特定schema定义结构。晚联编中，XML标记不直接与EXPRESS模型中数据类型相对应，而是与EXPRESS元数据对象(metadata objects)—包括实体、属性等直接对应。它适用于多个EXPRESS Schema的情形。

二者相比较，晚联编的映射方式映射出的XML文件要比早联编冗长，因此，早联编非常适用于单个EXPRESS数据模型时。但若一个XML应用中包含有多个EXPRESS数据模型，晚联编的映射方式就体现出了它的优势。因为，晚联编是XML标记与EXPRESS数据模型的元数据对象直接对应，而无论应用中有多少个EXPRESS数据模型，其应用元素的元数据对象都来自于同一资源标准。两者各有优缺点，将两者很好地结合，即采用混合联编的方式，便能

充分发挥各自的优势，实现XML向EXPRESS之间地良好映射。

3.1.1 EXPRESS与XML之间映射难点及规则

由于 EXPRESS 语言和 XML 语言的特点不同，描述同一对象的方式也有很大的差异，同时 EXPRESS 语言描述又兼具复杂性，XML 又在不断地扩充与发展，因此，EXPRESS 与 XML 之间的映射存在着一些难点。

（1）EXPRESS 语言中的继承机制，即超类和子类的关系，而 XML 语言不支持继承关系。那么在 XML 语言中将如何表达这种继承关系。

（2）EXPRESS 表达中不仅有模式、实体、类型的表达，还有约束和规则过程的说明，这些在 XML 里应如何表达。

针对上述的难点，为了最大程度地保证映射关系的通用性和标准性，建立的 EXPRESS 语言与 XML 语言的映射关系应遵循如下原则[37]：

（1）若 XML 标准中的关键语法能够表达 EXPRESS 语法表达的要求，则优先采用这部分语法，同时关键语法保证与 EXPRESS 语言的关键字表达一致。

（2）若 XML 标准中的关键语法不能表达 EXPRESS 语法表达的要求，而 XML 通过扩充定义可以满足时，则采用 XML 相关的语义定义，同时关键语法保证与 EXPRESS 语言的关键字表达一致。

（3）在以上规则均不能实现，或难以实现的情况下，则采用原文直译法，即将该 EXPRESS 语法描述直接在 XML 中以原文表达。

（4）在遇到 EXPRESS 中复杂的继承关系时，可以由 XML Schema 中定义属性 base 和 derived 来实现。约束关系表达可以在 XML 的声明中以元素属性的形式来表达。

3.1.2 EXPRESS 模式与 XML 模式映射

将 STEP 标准与 XML 语言相结合，首要的工作是实现描述 STEP 标准的 EXPRESS 模式和 XML 模式之间的映射。

1、EXPRESS 模式

EXPRESS 模式是实体(ENTITY)和类型(TYPE)的集合体。组成 EXPRESS 模式的基本元素有 schema 元素、字符集、注释、符号、保留字、

标识符和文字。基本的语言元素由一系列正文构成，可选择性地截断为若干物理行。下面是截断的一段 AP214 的 EXPRESS Schema 的文件。

```
SCHEMA   automotive_design;
    TYPE axis2_placement = SELECT
            (axis2_placement_2d, axis2_placement_3d);
            ......
     END_TYPE;
END_SCHEMA;
```

上面的 EXPRESS Schema 中，automotive_design 是引用的模式名称，即 AP214 模式名称。TYPE 是关键字，axis2_placement 是实体名称，SELECT 是类型名，axis2_placement_2d 和 axis2_placement_3d 分别为 axis2_placement 的下一级实体名。由上面这段模式文件可清楚明了超类实体与子类实体之间的表达关系。

2、XML 模式

XML 模式(XML Schema)结构如图 3.1 所示，由 Schema 元素、根元素、简单类型元素(SimpleType element)、复杂类型元素(ComplexType element)组成，复杂类型元素下又由若干个简单类型元素，及其属性作为它的子元素。XML 文件主要是由元素和标记构成，每个元素都由开始标记<>和结束标记</>组成。若前面出现了元素的开始标记<>，后面一定会对应出现该元素的结束标记</>，否则 XML 文件为无效。

XML 文件整体看起来像一棵树，由多个结点组成，XML 文件包含的结点有三种类型：元素结点、属性结点和文本结点。每个结点都表示为 XML 文件相应位置处的元素。一个 XML 文件只有一个根结点。在一个根结点下，又有若干个子结点，子结点下又有下一级的子结点。XML 文件满足树状结构信息，不支持交叉嵌套结构。若 XML 文件中出现了交叉嵌套结构表达，XML 的操作系统将会返回错误信息。如果要寻找 XML 树形结构中的一个子结点，则需从根结点开始，找到其中一个分支，寻找是否存在结点，如果该结点下还有子结点，须继续向下找，若没有找到，则返回上一级结点，再向其他分支寻找。

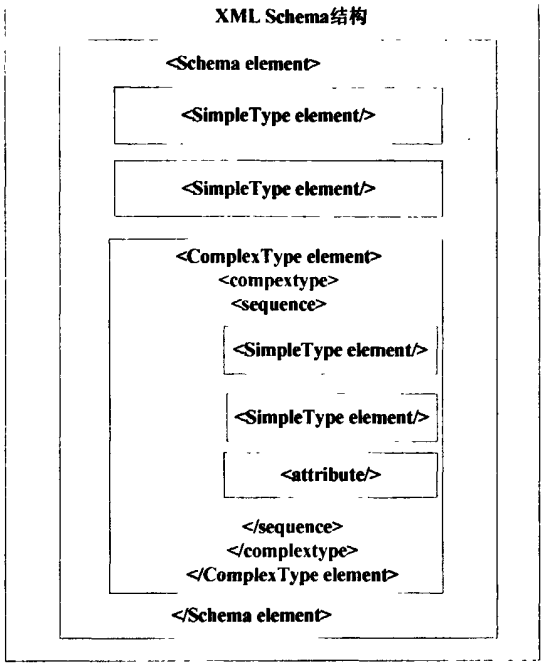

图 3.1 XML 模式结构图

3、EXPRESS 模式与 XML 模式映射

EXPRESS 模式中的 schema 元素映射为 XML 模式中的复合类型元素的根元素 Complex Type，元素名即为模式名。对于模式元素还可附加一些属性。如版本，模式标识和名称等。

EXPRESS 中的实体元素(ENTITY)映射为 XML 模式中复合类型的元素 complexType- element，类型元素(TYPE)映射为 XML 模式中复合类型的类型表达，实体名即为对应的元素名，类型名也为相应的类型名。实体元素内容由子类实体元素和实体属性组成，实体属性名前加上各自的实体名作为前缀(中间以“.”相连)构成相应元素的元素名，从而保证了元素名的唯一性。若实体属性又为此实体元素的下级实体，则可通过将下级实体映射为此实体的子元素(通过 sequence)来实现。若存在子类和超类实体类型的继承关系，则可在 Schema 中定义 derived 属性来实现。

上述的一段 EXPRESS 模式的 XML 模式表达为：

```
<xmlschema>
  < automotive_design >
    ……
```

```
<complextype name=axis2_placement type=select/>
  <sequence>
    <simpletype name=axis2_placement_2d/>
        ……
    <simpletype name=axis2_placement_3d/>
  </sequence>
</complextype>
……
</automotive_design>
</xmlschema>
```

3.1.3 EXPRESS 模式与 XML Schema 数据类型映射

1、EXPRESS 与 XMLSchema 的数据类型

EXPRESS 语言的数据类型主要有简单数据类型、聚合数据类型、命名数据类型、构造数据类型和广义数据类型[35]。若按其应用也可以分为基本数据类型、参数数据类型和基础数据类型。

其中，简单数据类型又分为数(number)、实数(real)、整数(integer)、字串(string)、逻辑(logical)和布尔(boolean)数据类型。聚合数据类型又分为数组(array)、表(list)、包(bag)和集合(set)数据类型。命名数据类型是在形式化描述中说明的数据类型，由实体(entity)数据类型和定义(define)数据类型两种类型组成。构造数据类型分为枚举数据类型(enumeration)和选择数据类型(select)。

XML Schema 主要有两种，其中 Microsoft XML Schema 数据类型有 35 种，常用的主要有布尔型(boolean)、字符型(char)、日期型(date)、日期时间型(datetime)、枚举组值型(enumeration)、浮点型(floats)、整形(int)、字符串型(string)、列表性(list)等。W3C XML Schema 数据类型主要分为简单类型和复杂类型两种，简单类型又分为原子类型、列表类型和联合类型三种。原子类型的值具有不可分割性，内置简单类型 44 种，如字符串型(string)、十进制型(decimal)、整型(integer)、浮点型(float)、布尔型(boolean)、时间型(time)等。本文采用的是 W3C XML Schema。

2、EXPRESS 数据类型的 XML 表达

（1）简单数据类型

简单数据类型定义EXPRESS中的最小数据单元的域[36]，表3.1描述了如何

用XML Schema中的数据类型来表达这些EXPRESS简单数据类型。

表3.1 EXPRESS简单数据类型的XML表达

EXPRESS数据类型	XML数据类型
数值型(number)	decimal
整数型(integer)	integer
实数型(real)	float / double
布尔型(boolean)	boolean
串型(string)	string
二进制型(binary)	base64 Binary / hexBinary

然而，逻辑型没有对应的XML简单数据类型，不过可以通过设置逻辑的真假值来实现[37]。如下定义所示：

```
<SimpleElement type="logical">
  <restriction base="xsd：string">
      <enumeration value="true"/>
      <enumeration value="false"/>
      <enumeration value="unknown"/>
  </restriction>
</SimpleElement>
```

在许多应用中，也可用编程语言来直接定义logical数据类型，并在运算中保持其语义。但本文采用上述方式来定义。

（2）聚合数据类型

EXPRESS聚合数据类型中数组(array)、列表(list)、数袋(bag)和集合(set)数据类型均可用XML语言中的list数据类型及XML中的相关表达方式来表示。下面是一个表达实体cartesian_point的例子：

EXPRESS定义：

```
ENTITY Cartesian_point;
x，y，z：REAL;
END—ENTITY ；
```

用XML表达为：

```
<complexType name="Cartesian_point" type=list>
    <sequence>
        <element name="x" type="float"/>
```

```
        <element name="y" type="float"/>
        <element name="z" type="float"/>
    </sequence>
</complexType>
```

（3）命名数据类型

EXPRESS的命名数据类型是用户定义数据类型，主要表达为实体(entity)数据类型。有了上面基本数据类型映射之后，命名数据类型的映射可在其基础上方便实现。下面是一个关于数据来源于点的direction的相关定义：

EXPRESS定义：

```
ENTITY point;
x, y, z: REAL;
ENDLENTITY ;
ENTITY direction;
derived : point;
END—ENTITY;
```

XML表达：

```
<complexType name= "point" >
    <sequence>
        <element name= "x"   type= "float"/>
        <element name= "y"   type= "float"/>
        <element name= "z"   type= "float"/>
    </sequence>
</complexType>
<complexType name="direction"
                    restriction: derived="point"/>
</complexType>
```

（4）构造数据类型

EXPRESS的构造数据类型有两种：枚举类型和选择类型。这两种类型在XML中可分别由enumeration和choice元素直接描述。

枚举数据类型EXPRESS定义：

```
TYPE drawing_figure = ENUMERATION OF (round, flat, rectangle);
END_TYPE;
```

XML表达为：

```
<complextype name="drawing_figure">
```

```
                <sequence>
                    <enumeration value="round"/>
                    <enumeration value="flat"/>
                    <enumeration value="rectangle"/>
                </sequence>
            </complextype>
```

选择数据类型EXPRESS定义：

```
TYPE area = SELECT (area_length, area_width);
END_TYPE;
```

XML表达为：

```
<complextype>
  <choice>
    <element name="area_length" type="string"/>
    <element name="area_width" type="string"/>
  </choice>
</complextype>
```

3.1.4 EXPRESS模式中约束和规则的XML表达

EXPRESS对实体的约束是通过一系列的规则(RULE)，WHERE子句、INVERSE子句和UNIQUE子句体现的。规则定义可以在一个模式内对一个或多个实体数据对象施加约束定义。在这些规则和子句中还可能包含了对模式中定义的函数引用，函数则可能由复杂的数据操作和控制流组成[38,39]。这就使得EXPRESS具备了数据库和面向对象语言的某些特征。对于WHERE子句、INVERSE子句、UNIQUE子句分别表示值域规则、逆向属性说明和唯一性说明，在XML中用XML复合类型元素中的属性声明(ELEMENT ATTLIST)来表达。属性的名称则根据EXPRESS中WHERE、INVERSE、UNIQUE的标识来定义，并且名称后面要分别紧接"W-"、"I-"、"U-"。下面是WHERE子句用XML表达的实例。

EXPRESS定义：

```
ENTITY action ;
num : INTEGER ;
WHERE
wr1 : num > 0 ;
```

END_ENTITY ；

映射为XML表达为：

<! ELEMENT action EMPTY>

<! ATTLIST action

id= “#REQUIRED”， num=“INTEGER #REQUIRED”

W-Wrl (enumatede value=“false | true | unknown”) #IMPLIED >

3.2 STEP AP214文件的XML模式的制定

STEP AP214 文件主要是通过 Part21 物理文件来显示它的结构。通常所说的 STEP AP214 文件就是符合 AP214 协议的 Part21 物理文件。然而，STEP Part21 物理文件主要是通过 EXPRESS 语言定义地各个实体(entity)与属性类型(type)结合体的实例化，并且允许实体与属性类型之间的模式(Schema)被其它的模式(Schema)所引用。在本文研究的数据交换的具体过程中主要是针对 AP214 协议的 STEP Part21 物理文件进行的。所以，在建立了定义 STEP AP214 文件的 EXPRESS 模式和定义 XML 文件的 XML 模式之间的映射关系之后，制定 STEP AP214 物理文件的 XML Schema 结构是尤为重要的。它保证了数据交换系统中 XML 文件生成的一致性和有效性。下面是针对 Part21 物理文件的头部段和数据段，分别制定的 XML Schema 结构。

3.2.1 头部段的 XML Schema 结构表达

图 3.2 是 STEP AP214 物理文件表达中头段信息的 XML Schema 结构图。其中清楚地说明了头段信息中文件描述、文件名称和文件模式这三部分以及他们内部信息的表达方式和组成关系。

头段信息的XML表达中以part21 schemaxml作为整体的根元素，表达为XML Schema如下：

```
<xsd:part21schema xml>
 <xsd:element name=”Header”>
 <xsd:complexType>
      <xsd:sequence>
       <xsd:simpleType/>
```

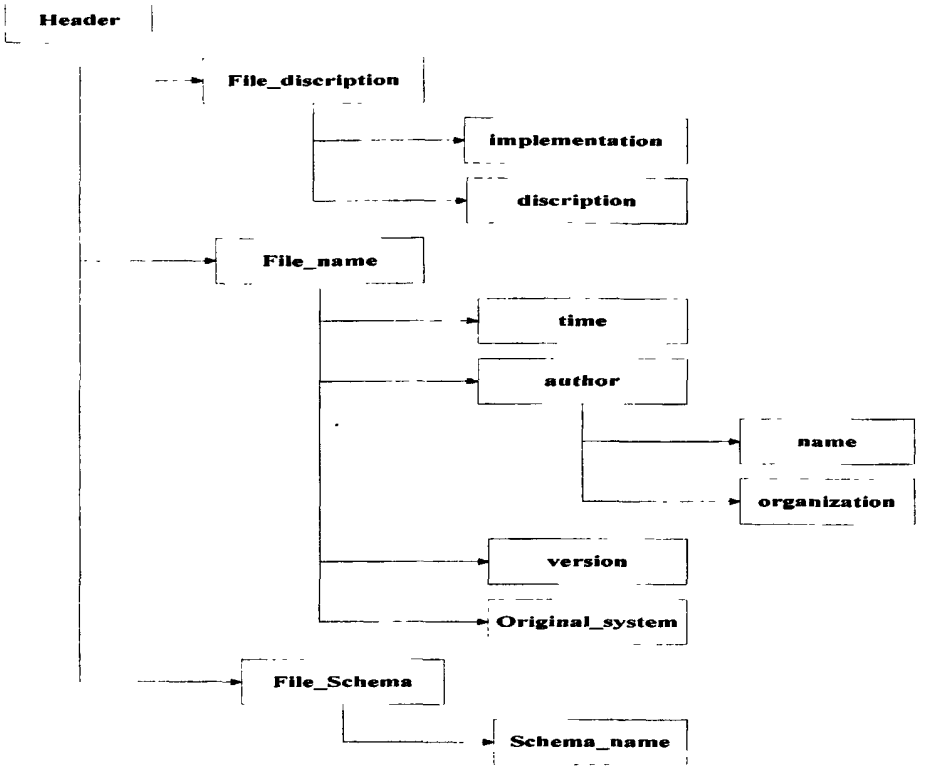

图3.2 STEP Part21头段信息的XML Schema层次结构

```
            <xsd:element name="File_discription", type="xsd:string"/>
            <xsd:element name="File_name",type="xsd:string"/>
            <xsd:element name="File_schema",type="xsd:string"/>
        </xsd:element>
      </xsd:simpleType>
    </xsd:sequence>
<xsd:simpleType>
  <xsd:attribute name="File_name.implementation",type="xsd:string"/>
  <xsd:attribute name="File_name.discription", type="xsd:string"/>
    <xsd:/attribute>
<xsd:/simpleType>
<xsd:complexType>
    <xsd:attribute name="File_name.time", type="xsd:date_time"
```

```
                    restriction="year\month\date\time"/>
    <xsd:attribute name="File_name.author", type="xsd:string" />
    <xsd:attribute name="File_name.version",type="xsd:string"/>
    <xsd:attribute name="File_name.system,type="xsd:string""/>
     <xsd:simpleType>
     <xsd:element name="author_name",type="xsd:string"
                    minoccurs="1",maxoccurs="unbounded"/>
     <xsd:element name="author_organization",type="xsd:string"
                    minoccurs="1",maxoccurs="unbounded">
     </xsd:element>
   </xsd:simpleType>
     <xsd:simpleType attribute="File_name.schema_name",type="xsd:string"
                    minoccurs="1",minoccurs="unbouded"/>
     </xsd:complexType>
  </xsd:element>
</xsd:part21 schemaxml>
```

3.2.2 数据段的XML Schema结构表达

数据段部分是STEP Part21文件的核心部分。数据段的结构可以看成是多个EXPRESS定义的数据实例的集合，比头部段要复杂的多。本文主要采用层次结构的树形组织来定义XML Schema。整个数据段以Data作为根元素。具体的XML Schema层次结构如图3.3所示。数据段部分是由许多个实体实例构成，图3.3只给出了一个具体实例的结构。每个实体实例都有唯一表示的ID属性，数据类型及其子元素构成。子元素下又有ID属性，数据类型等。

根据图3.3表达的数据段实体实例及其组成部分的层次结构，下面是一段定义数据段的XML Schema的表达。

```
<xsd:Data>
   <xsd:ComplexType>
     <xsd:element name="Data Instance",type="xsd:string"/>
      <xsd:element name="Data Type",type="xsd:string"
```

```
                restriction=list[1:?]/>
<xsd:element name="CONTENT",type="xsd:string"/>
<xsd:element name="sub_ENTITY",type="xsd:sring"
                minoccurs="0",maxoccurs="unbound"/>
```

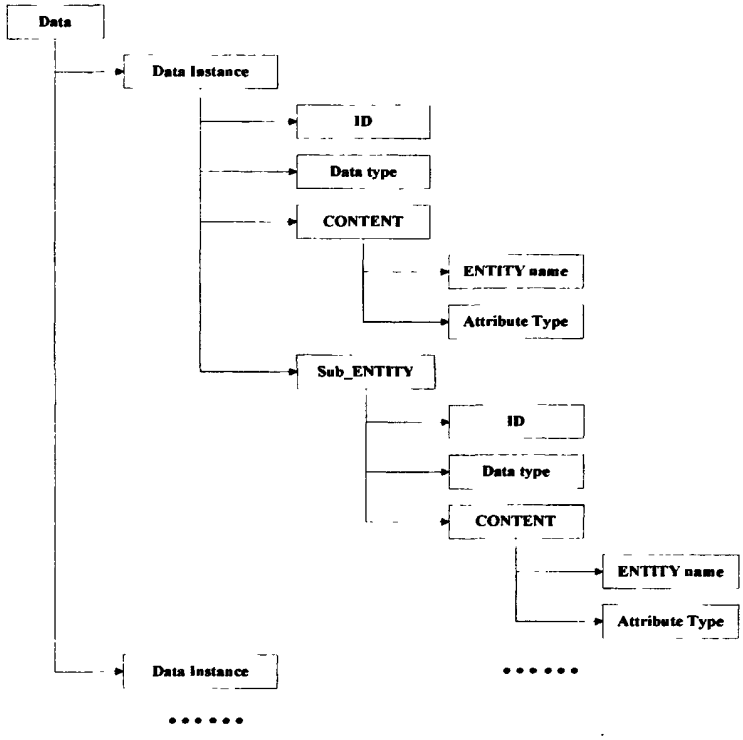

图3.3 STEP Part21文件数据段的XML Schema层次结构

```
</xsd:element>
<xsd:sequence>
  <xsd:SimpleType>
    <xsd:Attribute name="CONTENT.ENTITY name",type="xsd:string"/>
    <xsd:Attribute name="CONTENT.Type", type="xsd:string"
                        Restriction=list[1:?]/>
     </xsd:SimpleType>
  </xsd:sequence>
<xsd:ComplexType>
    <xsd:element name="sub_ENTITY.ID",type="xsd:string"/>
```

```
<xsd:element name="sub_ENTITY. Data Type",type="xsd:string"
                                    restriction=list[1:?]/>
<xsd:element name="sub_ENTITY.CONTENT",type="xsd:string">
 </xsd:element>
<xsd:SimplexType>
   <xsd:Attribute name="CONTENT.ENTITY  name",
                                        type="xsd:string"/>
   <xsd:Attribute name="CONTENT.Type",type="xsd:string"
                                    restriction=list[1:?] />     .....
 </xsd:SimpleType>
</xsd:CompleType>
.....
</xsd:Data>
```

制定了STEP AP214物理文件的XML Schema之后，数据交换过程中映射生成的XML文件就有了统一地定义和表达结构。确保了XML文件的有效性和一致性。

3.3 本章小结

本章首先介绍了 STEP Part28 部分提供的关于 XML 和 EXPRESS 进行映射的两种联编方式，即早联编和晚联编。阐明它们之间的特点及应用的范围。并强调本文进行的两者之间映射研究采用混合联编的方式。随后，通过对 EXPRSS 与 XML 的模式、数据类型、约束和规则的映射说明，建立了 EXPRESS 与 XML 之间的映射关系。并依此建立了 EXPRESS 定义的实例化表达—STEP AP214 物理文件的 XML Schema 结构。

第4章 数据交换系统的实现及其关键技术研究

4.1 数据交换系统的结构和功能

本文所要开发的数据交换系统主要是完成STEP数据到XML数据及其逆向的转换。此数据交换系统是通过借助具体的应用平台，根据EXPRESS语言和XML语言映射关系，将EXPRESS Schema映射为XML Schema，数据类型和约束规则也相应地映射为XML的数据类型和属性声明。再通过具体接口和应用转换程序，最终实现STEP AP214文件和XML文件正向、反向的转换。

本课题所研究的整个数据交换系统正如图4.1所示，在此数据交换系统中，由用户1生成的源文件(可以是三维CAD模型文件)，经CAD系统转换成

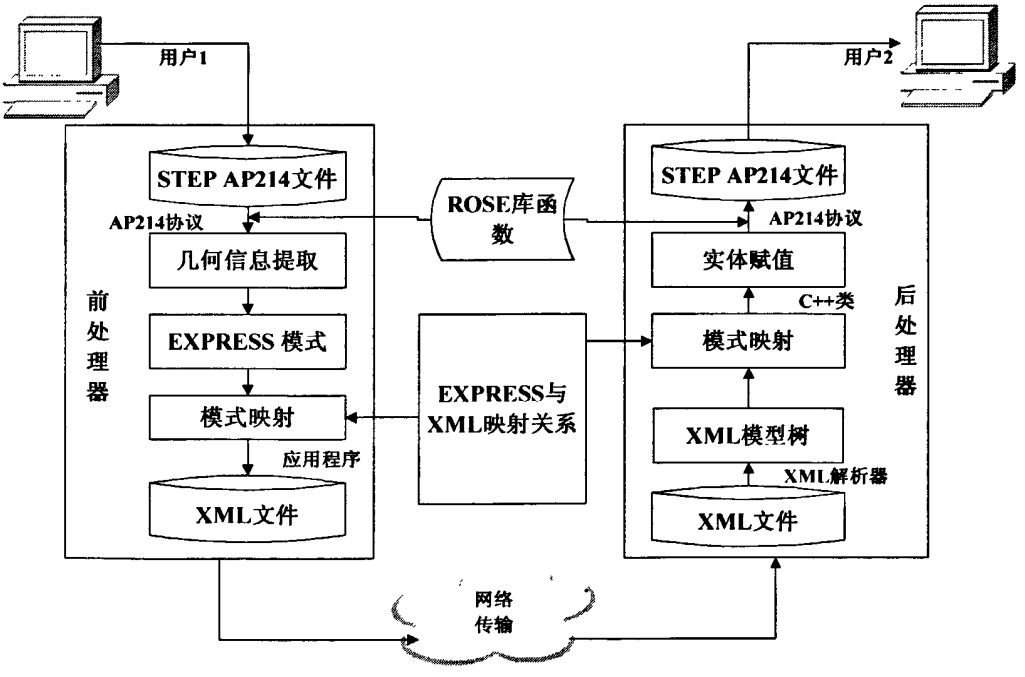

图4.1 数据交换系统整体流程图

为STEP AP214文件，通过STEP TOOLS公司开发的ST-Developer工具下的ROSE库函数的调用，进行AP214文件几何配置信息的提取。根据提取的信息匹配相应的EXPRESS模式。其中EXPRESS模式要符合AP214应用协议的规

范。再根据EXPRESS语言与XML语言之间映射关系的研究，将EXPRESS的模式与XML的模式形成一一对应的关系，用VC++6.0作为平台，通过应用程序的编译，生成XML文件。至此，完成前处理器的功能，即实现STEP AP214文件向XML文件的成功转换。生成的XML文件可以以网页的形式发布在网络服务器上，供其他用户共享使用。

由于XML文件在网络上传输的适用性，使其在网络化趋势的大环境下得到广泛地应用。由图4.1可知用户2从网络上接收到XML文件之后，再通过后处理器的使用，使其重新转换成为STEP AP214文件，可供下游支持STEP AP214格式的系统使用。后处理器的具体过程为：网络上发布的XML文件经过XML解析器的解析，可以得到XML模式的结点树，而结点树中的每个结点都对应着STEP AP214物理文件中一个实体元素或属性。通过对结点树的遍历，再根据EXPRESS模式与XML模式的映射关系，找寻对应实体的EXPRESS模式。再利用EXPRESS模式中应用解释模型的映射规则，通过VC++平台与ROSE库的联编对应用程序的编译，把此EXPRESS模式文件转化成为模式中各实体的C++类，最后进行实体赋值的过程，即可实现最终STEP AP214文件的生成。

4.2 前处理器的设计开发

前处理器是整个数据交换系统的开始。它主要完成 STEP AP214 文件向 XML 文件的交换过程。具体结构流程如图 4.2 所示。首先装载 STEP Part21 文件，载入内存之后，再经过操作系统与应用协议库的联编，对其进行语法分析和语义分析。即从 EXPRESS Schema 中找到与语法分析中的实体类型相匹配的语句单元，并对其进行识别，断定语句的所属类型，从而在语义解析的过程中，通过应用协议库的检索，来解析句子中每个单词的涵义，以及单词之间的层次关系。然后进行 STEP 数据信息提取，利用前文建立好的 STEP Part21 文件的 XML 表达定义结构和 EXPRESS 与 XML 之间模式映射关系，对语句单元进行翻译和转换，直至翻译转换完成 STEP Part21 文件最后一个语句为止，前处理器工作结束。下面将对前处理器的各个组成部分作详细的说明。

4.2.1 STEP数据分析

本文研究的数据交换过程就是对STEP数据进行操作并完成相应交换的过程。因此，首先要对操作的STEP数据进行分析。由图4.2可以清楚看出，在前处理器载入STEP AP214文件之后，需要对其进行词法、语法、语义的分析，即STEP数据分析。

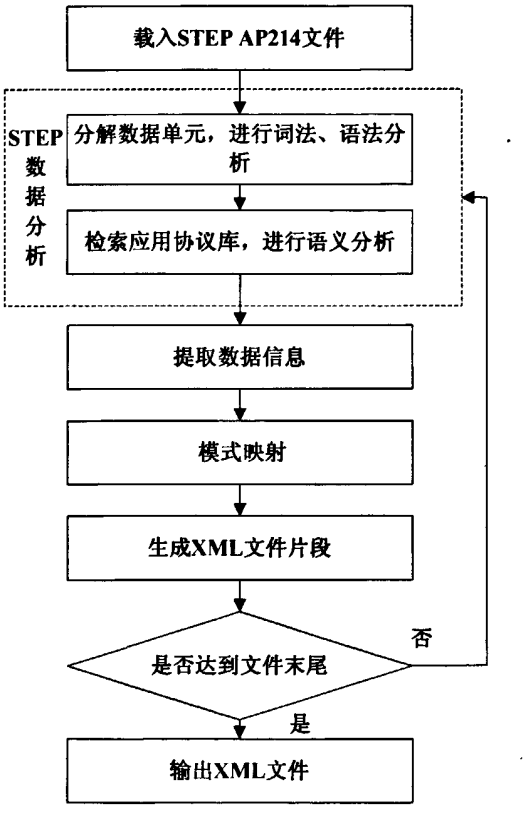

图4.2 前处理器实现流程图

（1）词法分析：主要是在AP214文件中读取字符，将其分隔成一个个的单元，进行单词的识别。根据STEP Part21文件的语法，可以把标识符、变量值、保留字和符号分别分隔成独立单元，便于单词的分析。以实体direction的STEP Part21表达为例：

#25=direction(1.000000,0.000000,2.000000);

可以将其分隔成#25,=,direction，(，1.000000，0.000000，2.000000,),;。便于与EXPRESS中的语义单元匹配。具体表达如表4.1所示：

表4.1 词法分析

单　词	属　性
#25	ENTITY ID
=	ENTITY Start
direction	ENTITY Name
(	List Start
1.000000	Constant 1
,	divide
0.000000	Constant 2
,	divide
2.000000	Constant 3
)	List End
;	ENTITY End

（2）语法分析：根据STEP Part21物理文件的语法规则，进行语法检查的过程。可对层次化结构进行分析。主要是通过针对一些全局变量进行初始化，经过词法分析后，返回值再经过相应的语法分析。语法分析后，识别上一实例：

#25->=->direction->(->1.000000，0.000000，2.000000,->)->;

（3）语义分析：根据STEP AP214协议，匹配direction实体的EXPRESS描述，与语法分析出的层次关系进行匹配。如：

1.000000匹配x_direction_ratio，0.000000匹配y_direction_ratio，2.000000匹配z_direction_ratio等。

STEP数据经上述分析后，每一独立单元的语法和语义都会被系统识别。为后序的STEP应用模块的具体开发做好了充足的准备，也为XML文件的生成提供了一致性保证。

STEP系统应用模块主要是STEP应用协议开发模块，而STEP应用协议的开发离不开ST-Developer开发工具的使用。

4.2.2 ST-Developer

ST-Developer是为了方便STEP应用模块的开发，由STEP TOOLS公司开

发的可以与C、C++、Java语言进行联编的处理EXPRESS定义数据的软件工具。此工具可在复杂的数据库(面向对象数据库、关系数据库等)环境和程序界面下处理EXPRESS信息模型[40]。ST-Developer有几大部分组成，其中包括：ROSE C++库，EXPRESS Complier(EXPRESS编辑器)，STEP一致性编辑和测试，EXPRESS-G工具，标准数据访问接口(SDAI，Standard data access interface)，CAD数据转换器及其他STEP接口。以上模块的相互关系及功能表达如图4.3[41]所示。主要完成的功能有：

（1）编辑STEP数据文件

（2）EXPRESS语言向C++语言转换的编辑器

（3）检查EXPRESS描述的正确性

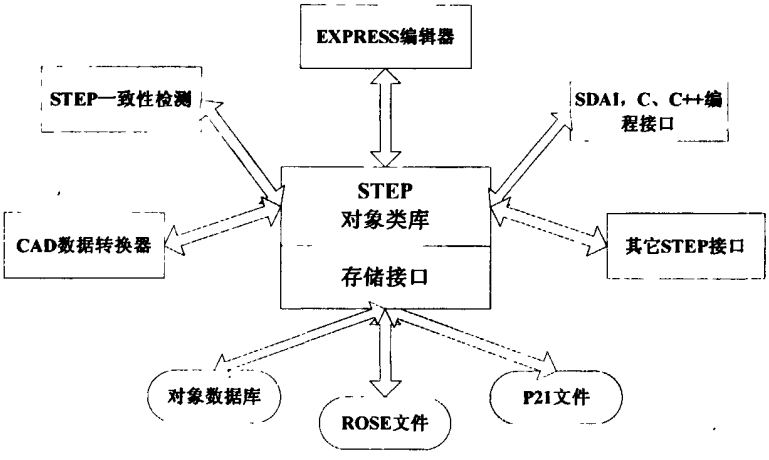

图4.3 ST-Developer工具结构图

（4）开发基于SDAI的 C语言应用程序

（5）开发基于ROSE类库的C++语言的STEP应用程序

（6）STEP数据的一致性测试

（7）处理多种STEP数据存储形式

ST-Developer支持环境为Windows和UNIX两种。本文选用的支持环境为Windows。图4.4所示为其在Windows环境下的控制面板，说明此工具的具体功能。

ST-Developer里最重要的组成部分是ROSE C++类库，此库可方便应用程序对STEP数据信息的读、写操作。EXPRESS编辑器通过构建C++类和数据字典来编辑EXPRESS语言，操纵STEP数据对象。EXPRESS-G图编辑器可浏览、

编辑、输出并打印EXPRESS-G图。STEP数据的一致性测试主要通过STEP应

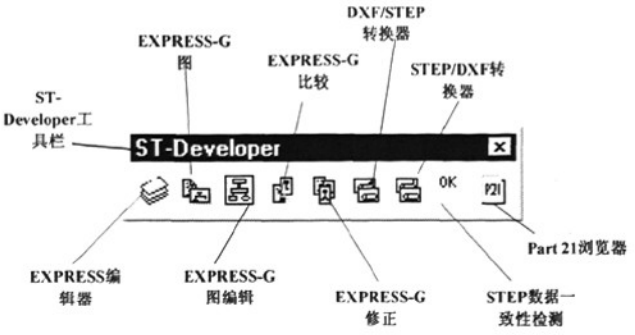

图4.4 ST-Developer工具栏

用协议来约束EXPRESS模型。目前ST-Developer所支持的一致性测试的应用协议有AP203、AP214。

1、ROSE类库

ST-Developer 中最重要的组成部分就是 ROSE C++类库，它是应用系统与 STEP 数据、面向对象数据库、关系数据库和物理文件的接口。ROSE 库是一种基于内存的数据库管理系统，ROSE 工具生成的是针对 STEP 设计对象的数据库，以 .rose 文件的形式存储，并运用 ST-Developer 提供的 EXPRESS Complier 工具把用 EXPRESS 语言定义的 AIM(应用解释模型)中的每一个实体(Entity)转换为一个 C++类。ROSE 库利用这些 C++类提供的函数来操纵 EXPRESS 描述模型中的数据，并且 AIM 中实体之间的继承关系和约束关系都在 C++类中得到很好的定义。这些类可以分为三种，STEP 对象类、ROSE 接口类和支持类。如图 4.5 所示。

ROSE 接口类为数据库系统提供了高层次的编程接口，该类的基本方法可完成大部分的系统功能，如寻找、显示、保存等等；支持类不能够操作 EXPRESS 定义的数据，它主要用来遍历、存储 STEP 数据对象(data object)及其系统内部的管理及运行。STEP 数据对象类表达不同的 EXPRESS 数据结构，这些类的实例在内存中操作 EXPRESS 定义的数据。EXPRESS 编辑器为载入的每一个 EXPRESS 数据创建子类，例如：EXPRESS 语言中定义一个点 Point，编辑器就创建了一个 C++类 Point。再依据 RoseStructure 数据对象类来构建 Point 类实例，此类可被定制为特殊功能用来操作 Point 的数据实例。未被用户化的 C++类也可被写进 ROSEStructure 内，但已被用户化的类操作起来会

更为简便。

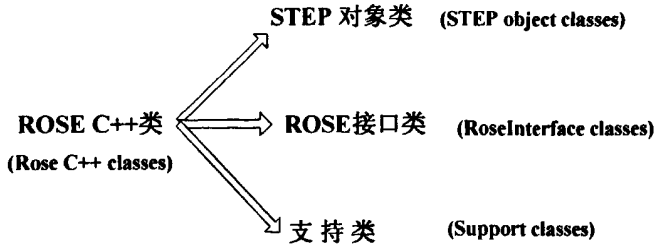

图 4.5 ROSE 类库分类

由上述可知，ROSE 定义的类可以遍历、存储和操纵 STEP 数据对象，ROSE 库对 STEP 数据对象的具体操作如图 4.6 所示。这些类可提供 STEP 数据对象的相关信息。例如：RoseManager 对象类可以存储 STEP 数据对象的外部信息。RoseDesign 类的功能类似于 SDAI 模型，可以读入一系列的 STEP 对象，并将其写入下级子目录中。

清晰表明了 STEP 对象在具体应用中的角色，如果 STEP 对象被包含在 RoseDesign 里，则用户使用的数据将被查找并存储在下级子目录中。未被包含在 RoseDesign 里的 STEP 对象，被系统认为是非永久性查询的，因为当程序执行完毕后，就会自动丢失。如无特殊用途无需使用此类 STEP 对象。在任何环境下，一个 STEP 对象只隶属于一个 RoseDesign。

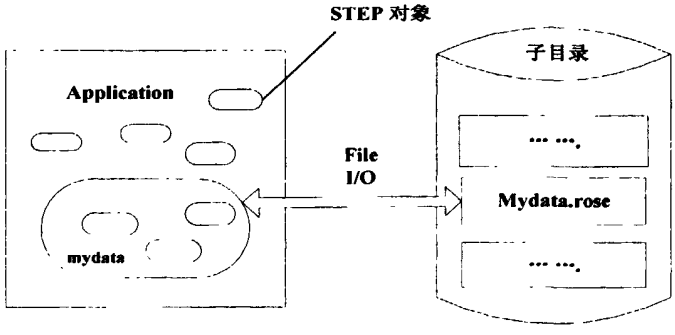

图 4.6 STEP 对象(object)

ROSE 类库联编方式可以分为早联编、晚联编。图 4.7 就是早、晚联编的实现流程图。虽然 STEP 应用协议的 AIM 中包含很多的实体，但在具体的应用中，在处理 STEP 应用协议数据的应用程序时，可能只用到其中一部分实体。此时使用工作集文件来操作 EXPRESS 模式就会非常的方便[42]。编译器

只产生工作集文件中指定实体所对应的 C++类，此时，工作集文件中的实体的 C++类就成为 RoseObject 类下的子类。使用工作集文件，可简略编译过程，提高编译的效率。

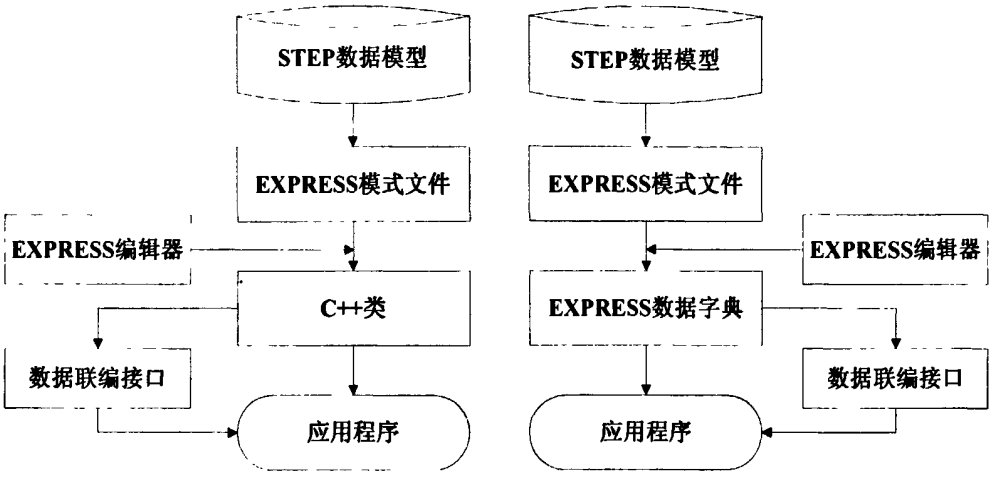

图 4.7 ROSE 库早晚联编的实现

ROSE 库的联编方式是应用协议(如 AP214)的模式文件经由 EXPRESS 编辑器操作 EXPRESS 定义数据的方式。早联编利用 EXPRESS 编译器的编译产生的是 EXPRESS 模式中实体的 C++类。晚联编方式与早联编方式有所不同，它产生的是 EXPRESS 数据字典，并且可针对不同的 EXPRESS 定义来操作不同模式。早联编方法编程代码简单方便、运行效率高，但应用程序只能操作一种特定模式的数据。晚联编的灵活性很强，并且应用程序可操作不同模式的数据。数据类型主要由 EXPRESS 数据字典在程序运行时确定。

2、EXPRESS Complier

EXPRESS Complier(EXPRESS 编辑器)也是 ST-Developer 的组成部分之一，在数据交换过程中有重要的应用。EXPRESS Complier 主要的作用是使 EXPRESS 数据模型生成对应实体的 C++类。根据 ROSE 库把这些类应用于具体应用程序或面向对象数据库。EXPRESS 数据模型定义中有很多实体，当 EXPRESS 文件载入 EXPRESS Complier 后，会将 EXPRESS 模式中的实体一一转化为 C++类，并且所有实体都满足 EXPRESS 模型中定义的继承关系，其属性都继承 RoseObject 和 RoseStructure 类[43]。EXPRESS 模型中的很多实体有超类和子类，如果一个实体没有超类，那么在生成 C++类的过程中就自动成为 RoseObject 的子类。如果在 EXPRESS 模型中定义了很好的子类-超类

关系，在生成 C++类时的继承关系就会被永久的保留并应用。表 4.2 是 EXPRESS 模型中定义的子类-超类在 C++类中继承关系的表达的一个实例。

表 4.2 C++类的继承关系

EXPRESS Specification	C++ classes
ENTITY Point;	RoseObject
ENTITY Point3D;	RoseStructure
SUBTYPE of Point;	Point
	Point3D

4.2.3 提取STEP AP214文件数据信息

在ST-Developer9.0的环境下，以VC++6.0为开发工具，使用ROSE库函数和EXPRESS编辑器，将EXPRESS语言描述的STEP AP214文件信息转换为STEP数据对象和C++类，按AP214数据模型的数据信息属性的层次结构，逐次提取信息。

如图4.8所示，首先通过ROSE接口把AP214文件中的所有实体信息读入

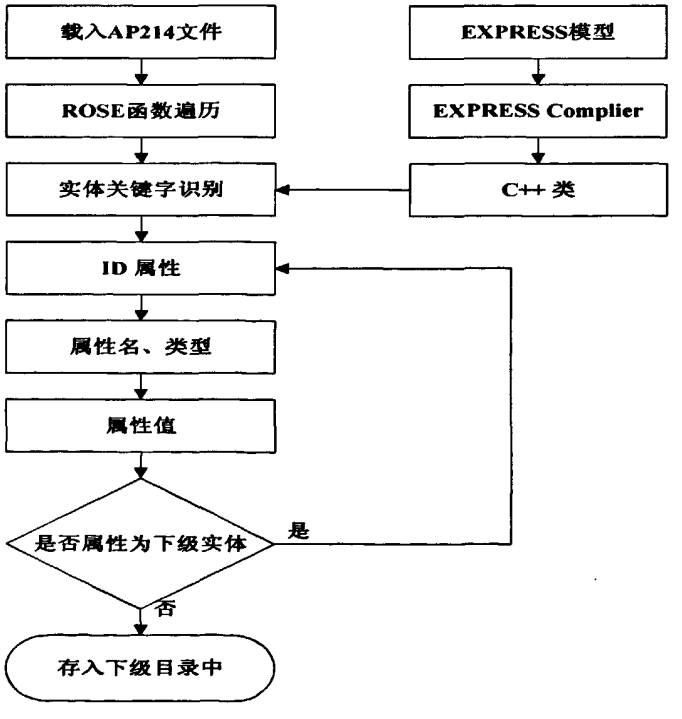

图4.8 AP214文件信息提取过程流程图

系统，以备后序操作。此时，系统应已把EXPRESS编辑器中生成的AP214文件中各实体的C++类添加到系统工程中。然后，利用遍历函数对整个文件进行遍历。对于一个确定实体，通过遍历函数Cursor和ROSE库中STEP对象类从系统中通过关键字的识别读取实体的ID属性值，通过关键字识别的对象可能不只一个，但ID属性值具有唯一性。通过ID属性的标识，在系统中进行属性名称、类型的提取，最终显示此属性值的信息。如果该属性本身又是另外一个实体，其自身还包括属性，则要再一次进行上述操作，进行信息的遍历和提取，直至完成最后一层实体属性信息的提取，存入到下级目录中。

1、头段信息提取及实例

STEP Part21 文件的头段信息主要是整个 STEP 文件的相关信息描述，包括文件描述的名称、注释、作者、作者单位、创建时间等。本文主要是通过利用 ST-Developer 工具下的 ROSE 库函数，以 VC++6.0 为开发平台，对其进行数据信息地提取。以下为利用 ROSE 函数进行头段信息提取的部分程序代码，用以说明头段信息提取的过程。

（1）将 STEP AP214 文件读入

这一过程使用 ROSE 接口类函数 findDesign()实现，函数原形为 RoseDesign * findDesign(char * design_name)[42]。使用此函数在系统内存和磁盘中搜索名字为 design_name 的设计对象，如果找到该设计对象就把它读入内存，否则返回 NULL。具体方法如下：

```
RoseDesign * design;                          /* 定义一个 design 指针 */
automotive_design();                          /*AP214 协议模式根元素*/
design=ROSE.findDesign("E：\\ pro/e\\zhou.stp");   /*指定寻找路径*/
/*根据路径寻找文件*/
  if(!design)
 {
  printf ("STEP file not found\n");
   exit(1);
 }
```

（2）信息提取过程

对头段信息的提取主要是根据RoseDesign函数下的header name( )、size()、

get( )等来实现。其中header_name()的函数原型为：file_name * header_name( )；size( )函数的原型为：unsigned size( )；get( )的函数原型为：get( )(RoseAttribute * att)[42]。

```
 /* 定义一指针 fxj, 变量 i,j */
 for(int i=0，fxj=Design->header_name()->author()
              ->size();i< len1;i++)
     {
/*提取头段中author的信息*/
      Printf("fileheader's author is %s." Design->header_name()
                  ->author()->get(i) );
      }
/*提取头段中orgnization的信息*/
     for(int j=0，fxj=Design->header_name()->organization()
                  ->size();j<len2;j++)
     {
         Printf(fileheader's organization is %s . Design->header_name()
                  ->organization()->get(j));
     }
/*提取头段信息中 schema 的名称*/
     for (int i=0, fxj=schemas-> size(); i<len3; i++)
    {
         Printf("fileheader's schemas is %s". Design->header_name()
                  ->schemas-> get(i)-> name());
    }
    ……
```

2、数据段信息提取及实例

数据段信息提取是 STEP AP214 模型信息提取的核心部分，为后面交换系统的实现提供支持。主要是提取 AP214 信息模型中的几何信息及其配置信息(针对 AP214 中 CC1 与 CC2)。一个 STEP AP214 物理件中是由多个 EXPRESS 数据模式的实例构成的。把 AP214 文件读入系统内存之后，ROSE

库函数会将 AP214 文件转换为 STEP 数据对象。在具体数据交换过程中，使用混合联编的方式来操作 EXPRESS 数据格式，具体过程为：

使用 ROSE 库函数将 STEP AP214 文件数据段中的每个实体实例转换为 STEP 数据对象，再利用 EXPRESS Compiler 将 AP214 模式文件中 EXPRESS 定义的每个实体转化为相应 C++类，每个实体的属性都在 C++类实体定义中用属性和规则来表达。这些 C++类是 ROSE 库中 RoseObject 类下的子类，ROSE 库中的 RoseObject 类是所有 STEP 数据对象的父类，它提供了操作任意类型 STEP 数据对象的函数。所以用 ROSE 接口和 C++类可直接操作 STEP 数据对象。

在清楚 STEP 数据对象中各个实体之间的属性层次关系之后，针对每个具体的 STEP 数据对象使用 RoseCursor 函数来遍历，用 RoseDomain 来界定遍历的范围。get<type>( )函数用来获得数据类型，其中，type 可针对不同的类型。getAttribute( )函数是按信息的层次属性顺次提取其属性值。以下是信息提取过程的部分程序代码，用于具体说明提取的整个过程。

在上文，我们已经介绍过了将 STEP AP214 文件读入内存的方法，下面是将文件读入内存之后，如何对 STEP 数据对象进行遍历及属性信息提取的过程。

对于已读入内存中的文件信息，需要对 STEP 对象实体进行遍历，利用 ROSE 类库中 RoseObject 类建立实体的 RoseObject 对象指针。RoseObject 类是所有 STEP 数据对象的父类，它提供了操作任意类型 STEP 数据的函数。通过游标遍历 ROSECursor 类来实现对设计对象的遍历。下例中的 axis2_placement_3d 是实体名称。

```
RoseObject * obj;
RoseCursor objects;
RoseDesign * design;
objects.traverse(design);
objects.domain(ROSE_DOMAIN(axis2_placement_3d));
```

遍历过设计对象之后，RoseObject 指针自动会指向 ROSE_DOMAIN 界定下的实体位置处，再进行下一步的实体 ID 及属性地提取。在提取的过程中主要用到 ROSE 函数库里的 entity_id( )、getAttribute( )、getType( )函数。函

数 entity_id( )的功能是从系统读入的 STEP Part 21 文件中读取 RoseObject 所指向的实体的标识符(即 ID 属性)，并且标识符所代表的仅仅是 STEP 物理文件中的实体实例。函数 getAttribute( )的功能是寻找当前 RoseObject 指针所指向实体的 EXPRESS 定义，并得到具体属性信息。它主要通过利用属性名称和成员数据地址两种方法进行寻找，本论文中采用的是前者。函数 getType( )中的 type 可以是 object、integer、float、double、string、boolean、logical、binary 中的任何一种，例如 getString( )、getFloat( )等等。其主要功能就是返回指针指向实体的具体类型值。如果实体的属性值为数组类型时，还需要用到[nsize]函数，函数[nsize]的功能是返回实体属性的数组值。

下面以实体 shape_definition_representation 为例，介绍 STEP Part 21 文件实体属性提取的具体过程。

shape_definition_representation 的 C++头文件为：

```
ROSE_DECLARE (shape_definition_representation) ;
    virtual public property_definition_representation
{
  private:
  public:
  ROSE_DECLARE_MEMBERS (shape_definition_representation);
    shape_definition_representation( );
    shape_definition_representation(
      property_definition * PARAM_definition,
      representation * PARAM_used_representation );
};
```

shape_definition_representation 由 EXPRESS 编辑器编译的 C++目标文件为：

```
shape_definition_representation:: shape_definition_representation( ) {
  ROSE_CTOR_EXTENSIONS; }
shape_definition_representation::shape_definition_representation(
    property_definition * PARAM_definition,
    representation * PARAM_used_representation   )
```

```
{
    definition (PARAM_definition);
    used_representation (PARAM_used_representation);
    ROSE_CTOR_EXTENSIONS;
}
```

其提取实体属性的具体程序代码如下：

```
.....
objects.domain(ROSE_DOMAIN(shape_definition_representation));
  while(obj=objects.next( ))
  {
   /* 得到 ID 标识*/
   printf("shape_definition_representation entity's id is #%d\n",
        obj->entity_id( ));
   List(RoseAttribute) * atts=obj->attributes( );
    /* 得到属性名称 */
   for(unsigned i=0;i<atts->size( ); i++)
   {
        RoseAttribute * att=atts->get(i);
        printf ("The %s attribute is defined by the %s type\n",att->name( ),
            att->persistentView( )->slotDomain( )->name( ));
   }
    /* 得到各属性类型的属性值 */
   char * str;
   str=obj->getString("name");
   printf("direction's name is %s\n",str);
   RoseObject * definition=obj->getObject("definition");
   printf("shape_def_representation entity's definition attribute is #%d\n",
                                        definition->entity_id( ));
   RoseObject * used_representation;
   used_representation=obj->getObject("used_representation");
```

```
printf("shape_def_rep 's used_representation attribute is #%d\n",
                    used_representation->entity_id( ));
.....
```

根据上述 shape_definition_representation 实体提取的具体过程，再根据实体之间的层次关系，逐次完成对 STEP AP214 文件中其它实体属性的提取。

4.2.4 XML 文件的生成

1、XML 文件结构

XML 文件生成的编译系统主要是最终生成 XML 文件的过程。因此，除了要选择合适的开发平台之外(这里同样选择 VC++6.0)，还要清楚地了解 XML 文件的结构规范。

XML 的语法要求很严格。一般来说，一个 XML 文件可分为三个部分：序言部分、主体部分和尾声部分。

序言部分是 XML 文件的起始部分，它主要包含了文档的相关信息，例如 XML 的版本号、文档的特征信息和文件所遵循的文档类型等。XML 的声明就属于此部分，是可选择的。XML 声明包含三方面的信息，版本信息、编码信息以及文件独立性信息，它们都以属性的形式表示在声明中。下面是一个完整 XML 声明的例子：

<?xml version="1.0" encoding="Unicode" standalone="no"?>

XML 文件的主体部分是由一个或多个元素组成的，其形式为一个包含多层信息的层次树。XML 文件主体包含唯一的根元素(root element)，其他所有元素都是根元素的子元素。

XML 文件还可包含尾声部分，尾声部分的内容主要包括注释、处理指令等，但尾声部分并不是 XML 文件所必需的。

最为关键的是，XML 文件的表达一致性主要是通过 XML DTD(文档类型定义)和 XML Schema 来保证的。由于 EXPRESS 语言的描述定义机制远比 XML 强，提供了实体、属性、类型及其约束规则的定义[44]。但 XML 也有其灵活性，所以制定相关的 XML 文件的定义结构是很必要的。由于 XML DTD 的描述约束的表达远不如 XML Schema。因此，本文建立的是定义 XML 文件一致性的 XML Schema。已在第三章作以说明，这里不再阐述。

2、XML 文件的生成实例

清楚了 XML 文件的结构规范，再根据前文总结的 EXPRESS 与 XML 模式之间的映射关系，及建立好的 STEP Part21 文件的 XML Schema 结构，就可根据具体应用程序的编译实现 XML 文件的生成。

在STEP Part21 文件中，主要是通过EXPRESS语言定义地各个实体(entity)与属性类型(type)的结合体，并且允许实体与属性类型之间的模式(Schema)被其它的模式(Schema)所引用。下面是针对 Part21 中不同实体和类型的情况，应用建立的 XML Schema 定义的 XML 文件的表达。

ENTITY 的定义表达要根据其属性类型的不同划分为几种情况：

（1）element 属性类型实体：其主要是以 ENTITY 作为 XML 中的元素，它的属性作为 XML 对象的元素(element)来处理。ENTITY 的名称就作为 element 元素属性“name”下的内容值，其数据类型的描述直接作为 element 元素属性下的“type”的内容值。

（2）optional 属性类型实体：为选择属性类型，与 element 类型类似，同样作为元素对象来处理，optional 属性的名称作为元素属性“name”下的内容值，其数据类型也作为属性下的“type”的内容值。但 optional 属性类型与 element 类型不同的是 optional 可以通过设置元素属性 minoccurs 和 maxoccurs 来定义，前者用来指定在文档中允许出现的该元素的最小次数，后者为最大次数。如表 4.3 所示：

表 4.3 optional 属性类型的 XML 映射实例

EXPRESS 表达	XML 表达
ENTITY axis2_placement_3d axis : OPTIONAL direction; ref_direction:OPTIONAL direction;	<element name=“axis2_placement_3d”> <elementxtype name=“axis” type=“opitional” Minocuurs=“1”maxocuurs=“unbounded”/> <elementxtype name=“ref_direction” type=“opitional” Minocuurs=“1”maxocuurs=“unbounded”/> <elementxtype/>

（3）enumerated 属性类型实体：为枚举属性类型，同样是作为 XML 文档中的元素对象来处理，enumerated 属性的名称作为元素属性“name”下的内

容值，它的类型为自定义的 simpletype 数据类型，使用 restriction base 指定它的数据限定类型。

（4）derived 属性类型实体：为导出属性类型，同样处理为 XML 文档对象元素，derived 属性的名称作为元素属性“name”的内容值，元素的类型为自定义的 complextype 数据类型，下面以 vertext 实体为例进行说明。如表 4.4：

表 4.4 derived 属性类型的 XML 映射实例

EXPRESS 表达	XML 表达
ENTITY vertext; x: real y: real z: real END_ENTITY	<element name="vertext" type="vertexttype"> <complextype name="vertextttype"> <elementxtype name="x" type="real"/> <elementxtype name="y" type="real"/> <elementxtype name="z" type="real"/> <complextype/>

TYPE 类型定义主要用于数据类型的定义，EXPRESS 语言的数据类型中简单数据类型、聚合数据类型、构造数据类型、命名数据类型都作为基本元素类型“ElementType”来定义。表 4.5 是一个具体的 TYPE 类型的映射实例。

表 4.5 type 类型的 XML 映射表达

EXPRESS 表达	XML 表达
TYPE area_measure="real"; END_TYPE;	<elementType name="area_measure"> <restriction base="real"> </restriction> </elementType>

有了保证 XML 文件一致性的定义结构 XML Schema，不同的实体类型就可以根据 XML Schema 中的定义，实现格式良好的 XML 表达，最终生成 XML 文件。

4.3 后处理器的设计开发

后处理器主要是完成对 XML 文件的解析，及其向 STEP AP214 文件的数据交换过程。后处理器的工作原理如图 4.9 所示。主要组成部分有 XML 文件解析器的设计开发，XML 结点树的属性信息提取，根据 XML 与 EXPRESS 的映射关系进行模式映射获得需要的 EXPRESS 模式，由 EXPRESS 编译器编

辑生成 C++类，通过 ROSE 存取接口操作及实体赋值过程，创建实体实例，最终生成 STEP Part21 文件。

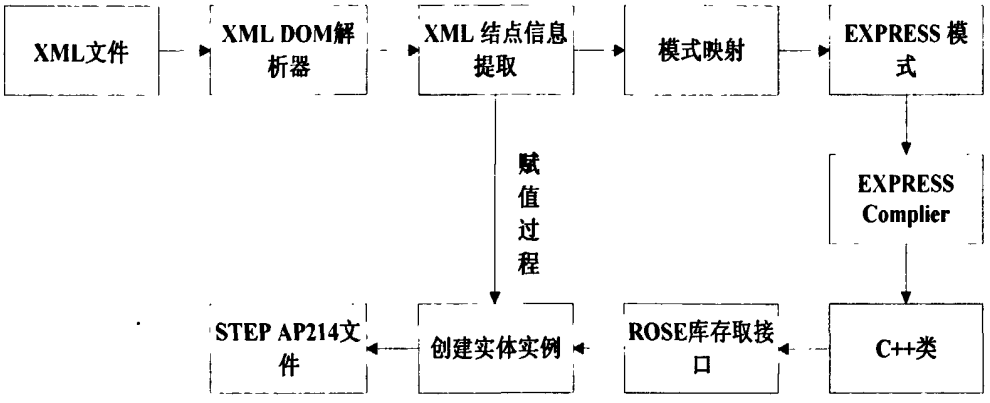

图 4.9 后处理器结构流程图

后处理器的源数据文件是 XML 文件，首先是要通过对 XML 文件的操作来进行的，所以 XML 文件的解析是首先要解决的问题。

4.3.1 XML 的解析接口研究

解析器是处于应用程序和 XML 文档之间的软件组织[45]，在 XML 解析模块的开发过程中，有两种 VC 与 XML 的接口是经常被使用的。一种是文档对象模型 DOM(Document Object Model)，另一种是 XML 应用程序接口 SAX(Simple API for XML)。然而，正确地选择合适的接口是设计 XML 解析器的必要步骤。

1、DOM 接口

DOM 是一组独立于语言和平台的应用程序编程接口[46]，它能够描述如何访问和操纵存储在结构化 XML 文件中的信息。利用 DOM 应用程序接口(DOM API)，应用程序开发人员就能够通过编写特定的代码来实现特定的功能。具体地，开发人员可以通过利用 DOM 中的接口对象对文档进行读取、搜索、修改、添加和删除等操作。DOM 支持树状信息，它以分支的树状结点来维持整个 XML 文档，XML 文档中的每个元素、处理指令、注释等都被表示为结点。DOM 为 XML 文件的内容和结构提供了标准函数。

DOM 的工作方式如下图 4.10 所示，首先将 XML 文档一次性的载入内存，然后对文档进行解析，根据文档中定义的组件(元素、属性、注释、处理指令

等)的不同内容进行分解，以“结点树”的形式在内存中创建 XML 文件的表示形式，也就是一个文档对象模型。然后根据对象提供的编程接口，在应用中访问 XML 文件，进而操作 XML 文件。

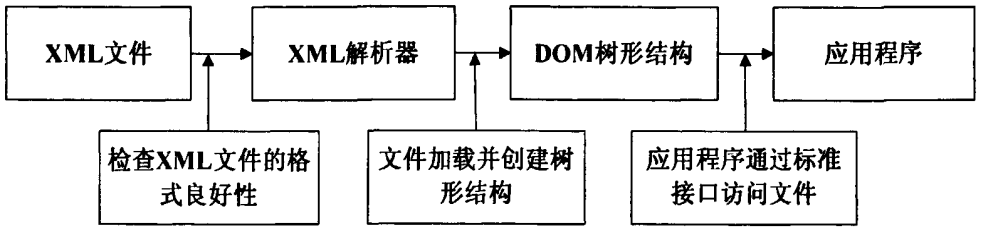

图 4.10 DOM 工作方式图

DOM 在处理具体事件时，是以树形结点顺序进行的。DOM 中的结点包括属性结点、元素结点、文本结点等。如下面 book 的 XML 文档在 DOM 中的处理过程如下：

```
<book type="study">
    XML 解析器
  <number>ZC10086</number>
  <author>wanggang</author>
</book>
```

此处 DOM 加载此 XML 文件之后，把整个文件读入内存，进行结点顺序处理。book 为根元素，它表示为一个元素结点，它包含有四个子元素，这四个子元素是由不同属性类型的结点构成：

(1) “type”为属性结点，“type”又有唯一的文本节点“study”作为其子结点。

(2) “XML 解析器”为文本结点，并且此文本结点下没有任何的子结点。

(3) “number”和“author”都为元素结点，他们下面都分别有一个文本结点 ZC10086 和 wanggang 作为他们的子结点。

2、SAX 接口

SAX 被称为事件驱动解析器，在把 XML 文件加载入内存之后，顺序读取文件，并基于它在文件中遇到的特定标记产生相应的事件，执行应用程序的回调函数[47]。也就是说，SAX 在处理 XML 文件时，并不会顺序执行相应事件，而是在 XML 文件中找到特殊标识的项时，才会发生事件。随后，应用程序就会响应这些事件。SAX 解析器在解析完 XML 文件之后，并不保留 XML 文件原本的树状结构。若从面向对象编程的角度考虑，SAX 的处理程

序就是一系列的事件处理器。其处理事件的具体流程如图 4.11 所示。

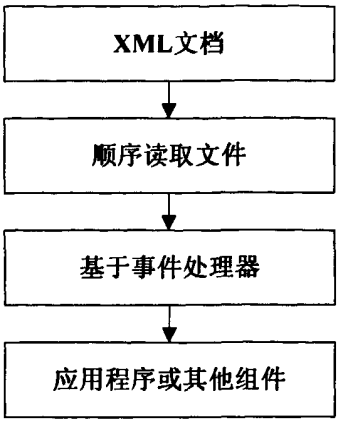

图 4.11 SAX 工作流程图

既然两种接口都可以对 XML 文档进行解析处理，就需要比较两者的优缺点，了解两者的区别，进行综合地选择。DOM 与 SAX 存在许多的不同之处，在不同的应用场合各有利弊。

3、DOM 和 SAX 的比较

DOM 可以一次加载整个文件到内存中，并形成相应的结点树，对于处理树形结构的文件存在很大优势，但是会占用内存很大的资源。SAX 是从开始到结尾一直在读取 XML 文件，并不完全一次性把文件装入内存中，是基于文件标记来形成事件的。因此，SAX 处理事件时所占用的内存资源没有 DOM 高，处理的速度也比 DOM 要快。另外，DOM 提供了很好的结点添加、删除、修改的功能，在此方面，SAX 的功能就远远不如 DOM。

SAX 解析器的指针在文件中的移动顺序是只进的。前一刻从解析器中抛出的事件信息，如果没有缓存下来，就没有办法再次获取。如确实需要前面已抛的数据，就只能重新加载文件。使用 DOM 时，此类事情即可避免。DOM 将整个文件加载到内存中再处理，可使指针在文件模型中往复移动，获取所需的数据，并且可随时修改文档中的内容。

从以上分析，可以得出，在需要处理大型文件、在文件中只需要检索小部分数据，或需要创建大型文件的小部分子集时，使用 SAX 的程序不需占用太大内存，同时又可获得很高的处理速度。在需要修改文件、随机存取文件、综合分析文件时，使用 DOM 会更佳。因此，本论文中，选用 DOM 作为 XML 文件解析器的接口。

4.3.2 DOM 解析接口方法研究

在解析方法的设计过程中，主要是通过 DOM 中的不同接口的应用来实现对 XML 文件的处理的。然而 DOM 接口有符合其自身规范的继承关系，如图 4.12 所示。DOM 核心模块的各个接口是在 DOM 内部处理信息的基本工具。合理地应用 DOM 中的各个接口是实现 XML 文档解析的关键。

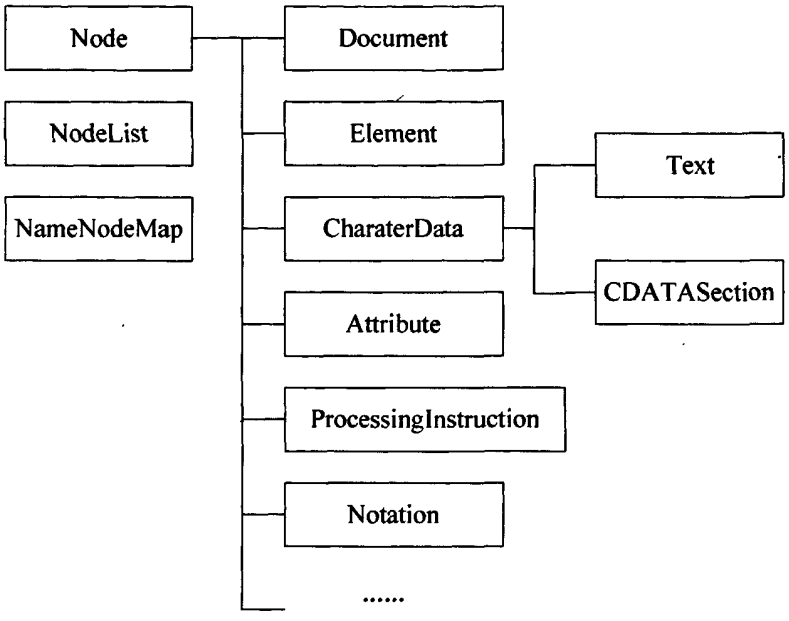

图 4.12　DOM 中主要接口的继承关系

1、Node 接口

Node 接口是 DOM 最基本的接口，其他的接口如 Document、Element 等，都是从 Node 接口中继承它的方法和属性[48]。Node 接口包含了基本结点操作所需要的全部方法和属性，但是在 Node 结点中并不包含创建新结点，因为所有的结点只在文件的上下文中出现，因此创建新结点的方法包含在 Document 接口中。遍历 XML 文件是 DOM 实现中很重要的一种方式，Node 接口中的 firstchild( )、lastchild( )以及 nextsibing( )都可实现对 XML 文档的遍历。下面这段代码就是 Node 接口遍历 XML 文档的代码。

```
Void   proccessNode(Node n)
  { Node c;
    StartProcessing(n);
    for(c=n.firstchild( );  c!=null;  c=c.nextsibiling( ))
```

```
        { processNode( );
          }
        EndProcessing(n);
        }
    Void Startprocessing(Node n)
{
    {void Endprocessing(Node n)}
}
```

上面这段代码为载入内存的 XML 文件提供了一种直接访问各个结点的方式。

XML 文件是由很多的结点集合而成，Node 接口提供的属性和方法可以处理文件中的各个结点，除可以对结点进行读取之外，还可通过 Node 接口下的 appendchild( )和 insertchild( )、removechild( )、replacechild( )进行结点的添加、删除、替换等操作。

Node 接口是通过 NodeList 接口和 NameNodeMap 接口的方法和属性获得对文档结点的对象引用。如图 4.13 是一个典型的 Node 接口。NodeList 接口是一个结点的集合，它提供了一个访问有序结点的方法，在 NodeList 中的结点列表按照结点在文件中原有的顺序进行排列。NodeList 中的第一个元素和最后一个元素分别对应此 Node 接口的 firstchild 和 lastchild。NodeList 接口只有一个只读属性 length，它表示 NodeList 结点列表中结点的数量。NodeList 接口还包含一个方法 item( ),它用于返回 NodeList 结点列表中的结点。下面是用 NodeList 接口获得文本属性(text)对象引用的例子。由此，可获得文档标记名称为 text 的所有元素的 NodeList。

```
NodeList texts=document.getElementsByTagName("text");
```

如果遇到需要获得每个文本属性元素的对象引用时，可以通过下面这个循环语句来实现：

```
for(long i=0，i<texts.length，i++)
  {
   Node text=texts.item(i)；}
```

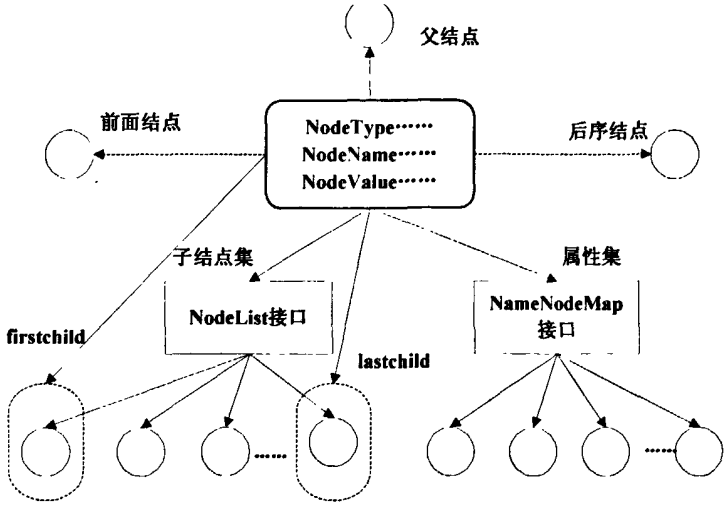

图 4.13 Node 接口表达图

NameNodeMap 接口也表示一系列结点的集合，通过此接口，可以建立结点名称和对应结点之间的一一映射关系，从而利用各结点名称就可以直接访问指定的结点。为后序的 DOM 结点信息提取提供了方法。

2、Document 接口

Document 接口也是 DOM 里的核心接口，它的应用也非常广泛。可包括多个处理指令结点和注释结点。Document 主要继承了 Node 结点的属性和方法，最重要的是，它能够在 XML 文件中创建新的结点。Document 包括了 docType、documentElement 以及 implementation 三个属性。这三个属性都是只读属性。其中，docType 表示文件类型结点，documentElement 直接指向文件的根元素。它提供了访问和操作文件数据的入口。

Document 接口包含了许多方法，都有不同的应用。其中，createElement( ) 用于创建一个提供了标记名称的新元素，它采用的方式为：

Element createElement (DOMString tagName)；

createDocumentFragment( )用于创建一个空白的 DocumentFragment。它采用的方式为：

DocumentFragment createDocumentFragment( )；

createTextNode( )用于创建指定结点内容的文本结点。它采用的方式为：

Text createTextNode(DOMString data)；

createAttribute( )用于创建一个带有名称的属性结点，并且属性结点的初

始值为空。它采用的方式为：

Attr createAttribute(DOMString name)；此处 name 为结点的名称。

3、Element 接口

Element 接口也是 DOM 里的核心接口，同样继承 Node 接口的属性和方法，但也包括了它自己本身的属性和方法。其中，tagName 是 Element 的一个只读属性，表示元素对应的标记名称，与 Node 接口中的 nodeName 相对应。

Element 接口的方法有很多，其中获取属性值的方法主要有 getAttribute( ) 和 getAttributeNode( )两种。前者是通过属性的名称来获取属性值，后者是通过属性的名称来获取属性结点。与之相对应，removeAttribute( )和 removeAttributeNode( )方法则是通过指定属性名称来删除属性，和通过指定属性结点来删除该结点。setAttribute( )和 setAttributeNode( )方法表示的是添加一个新的属性，和添加一个新的属性结点。下面以 getAttribute( )和 getAttributeNode( )为例，说明其语法表达：

DOMString getAttribute (DOMString name);

Attr getAttributeNode (DOMString name);

getAttribute( )和 getAttributeNode( )会在后面提取 DOM 结点信息中有重要的应用。

4、Comment 接口

Comment 接口继承于 CharcterData 接口，而 CharcterData 接口通过访问 DOM 文档内字符数据的方法来扩充 Node 接口。它还包含了自己的属性和方法 data 和 length，分别表示实现接口的结点字符数据和字符数据的长度。Comment 主要表示 XML 文档的注释内容，但不包含注释定界符(<!- 和 ->)，Comment 本身没有任何的扩展属性或者方法。

根据以上对 DOM 核心接口的研究，可以清楚地知道在设计 XML 的 DOM 解析器时，DOM 接口方法的顺序选用是很重要的。在 DOM 读入一个完整的 XML 文件之后，通过模型构建器的设计，在内部会形成 XML 结构树，便于后序的遍历、读写等应用操作。

4.3.3 DOM 解析过程设计

DOM 是一种简单的从 XML 语言到具体的应用程序之间的标准化接口。

DOM 允许用户加载和创建 XML 文件，并适合应用在对 XML 文件的结点随机访问的场合，在处理要求同时访问多个结点时，优势更为明显。因此，在设计 XML 文件的 DOM 解析器时，除了要求对 DOM 中各个接口有很好的研究之外，还要对其进行合理的使用，才能达到理想的解析效果。

本论文中需要解析的 XML 文件是由网络上传输得到的，并且是由 STEP 物理文件经由数据转换系统转换而来。因此，此 XML 文件在加载入 DOM 之后，根结点定为 STEP 物理文件中的根元素。又由于，此处 XML 解析器的作用是为了能够便于在后序部分完成 XML 文件向 STEP AP214 文件的成功转换，而且，XML 文件中的每个标记结点都为 STEP AP214 文件中的实体或属性。结点中的内容为实体标识或属性值，即为源数据中的零件模型的数据信息。因此，DOM 解析器的设计应着重考虑 XML 文件中各实体的层次关系及其属性信息。即同时实现多结点的访问。

DOM 解析器的实现主要以 VC++6.0 为主要的开发平台，根据具体程序流程地编译来实现的，具体过程如下图 4.14 所示。得到 XML 文件之后，首

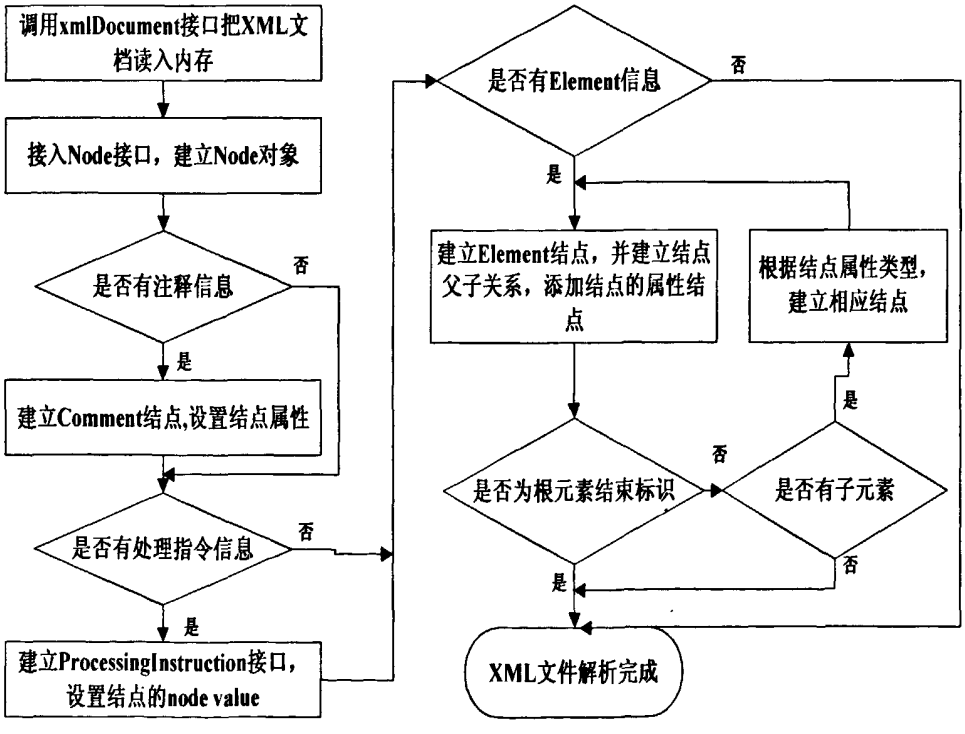

图 4.14 XML 文件解析过程流程图

先调用 DOM 里的 Document 接口把 XML 文件读入系统内存中，自动形

Document 对象。其中，Document 接口里的 creatDocument 接口函数就会以 DOMString 类型变量作为参数，指明 XML 文件的路径和名称。读入 XML 文件后，文件指针会自动指向 XML 文件在内存中的首地址，建立 Document 的 Node 对象，便于后序对文件指针对 XML 文件的遍历。设置 NodeType 属性为 Document—node。然后文件指针指向文件的开始标识处，开始判断 XML 的注释信息(Comment，以<!-开始，以->结束)，包括文件版本和编码信息等。如果有注释信息，则需要建立相应的Comment接口，因为Comment接口主要处理XML文件中的注释信息。然后需设置它的属性为Comment-node, 并把该结点加入到Document Node的子结点单链表的末尾。遇到注释结束符号后，在NodeName里记入结点名字的标记内容。

接下来需要判断XML文件中有无处理指令(ProcessingIn-struction)信息，它是以<?开始，以?>结束。对于此信息，可以不把其结点的名字复制到Node对象中，而只是在Node对象中记录其地址信息，这样做主要是为了提高解析XML文件的速度，以及节省内存空间。如果有处理指令，文件指针则移动到处理指令信息的首地址，建立一个Node对象，设置Node-Type属性为Processing-instructionNode，并建立相关的父子结点关系。然后文件指针从首地址开始顺序检查字符，判断有无语法错误。遇到结束符时，把结点名字的开始地址和结束地址记入NodeName里，并把结点值的开始地址和结束地址填入NodeValue里。便于结点的遍历和属性信息的提取。

在前面工作完成之后，若已没有处理指令信息，那么继续移动文件指针，判断文件中是否有元素结点的信息(Element，以<>开始，以</>结束，或者以<开始，以/>结束)。此处通常是从根元素开始的。如果有元素信息，则建立一个Node对象，设置NodeType属性为Element—Node，并处理后续字符。由于Element结点可能会有ProcessingInstruction、Comment、Text、CDATASection或EntityReference等子结点，因此还需要调用每种类型结点的处理函数，将各种结点加入到DOM中。图4.14仅以Element来表示所有Element下的子结点。若遇到其他类型的结点采用同一方法。Element、ProcessingInstruction、Comment接口前文已经描述过，这里不再阐明。Text接口与Comment接口都是继承于CharcterData接口，它的作用是表示各元素和属性的文本内容。CDATASection接口又继承于Text接口，它代表可解析的文本内容。

EntityReference接口在文档中表示实体的引用，EntityReference结点是一个只读的结点，没有扩展的属性和方法。

如果此时文件指针仍未移动到根元素结束符处，则应继续判断是否仍有元素结点，进行递归调用，反复上述过程，直至文件指针遇到根元素结束标识符为止。至此，完成对XML文件的解析。

XML文件通过该解析器之后，可以创建一个Dom-Implementation类的对象，调用createDocument函数，通过函数的返回值得到一个Node对象。该Node对象类型是Document-Node，对应于DOM的根结点，即为DOM模型树的根结点。对于根结点下的各个子结点和同一父结点下的各个子结点就形成相应的结点列表(也可看作是结点集合)，返回值将会得到一个NodeList对象，来表达整个结点树的信息。如图4.15所示。然后就可以通过设置Node对象和NodeList对象来遍历DOM树中所有的结点。

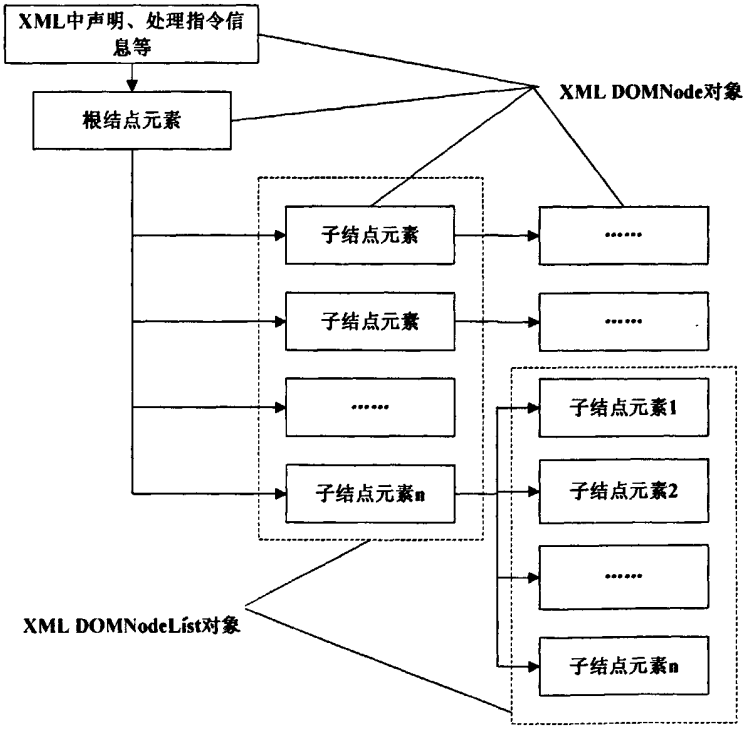

图4.15 XML DOM结点树

4.3.4 提取DOM树结点信息

在前一小节中，我们已经设计完成了XML DOM解析器的具体解析过程

并且在内存中形成相应的结点树，本节主要是通过对DOM结点树的遍历，应用DOM对象来实现结点信息的提取。下面重点研究两种方法。

在XML文件中，可通过属性来标识结点[49]。由于解析器设计的同时，每一个结点处都标识了属性，因此，本文就是利用属性这个特点，来获取指定结点值的。然而，要获取指定的结点值，首先要定位结点，即把文件指针指向你所要获取信息的结点处。然后根据接口函数调用，获取指针指向结点处的结点值。在本文主要研究两种定位结点的方法，一种是SelectSingleNode，另一种是SelectNodes[50]。关于两者的应用由图4.16来表示。

SelectSingleNode是确定单个结点，它的返回值是XML文档里所确定目标结点的第一个结果元素，通过指定形式来输出结果。SelectNodes是确定多个结点，它的返回值是XML文档里所确定目标结点的所有结果元素的列表。得到结点后还需判断结点数目，并对应数目结点返回目标结点值。

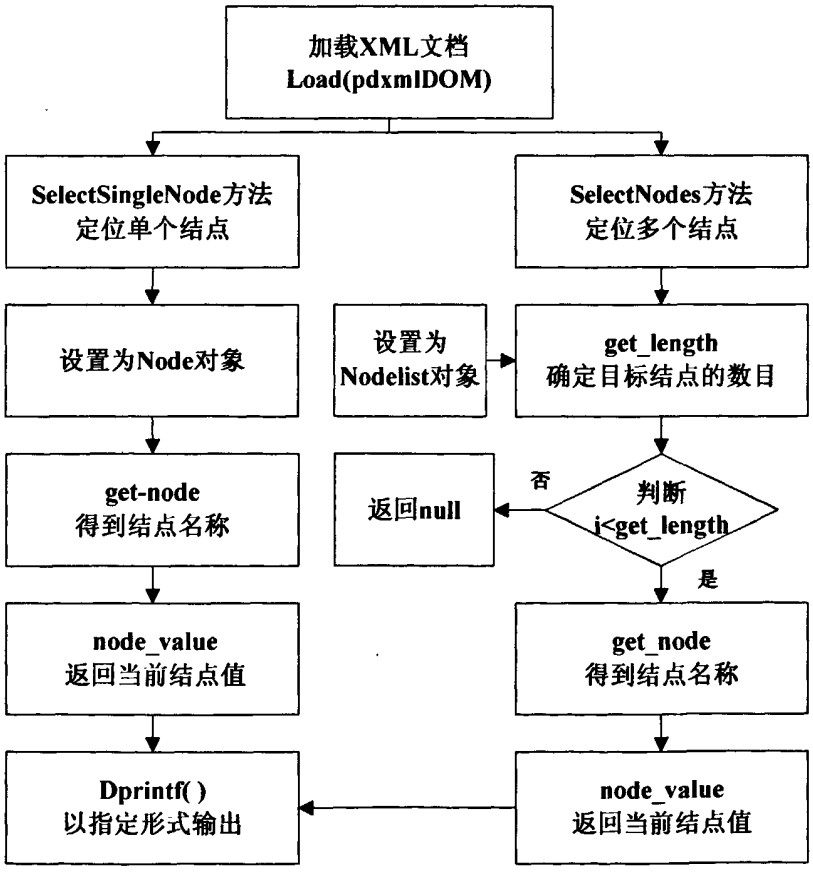

图 4.16 获得结点属性信息流程图

下面的代码是加载XML文件后，读取结点信息的实例。首先需定义一个字符变量，加载XML文件之后，把XML文件的名称赋值给此变量，然后创建一个DOM对象，加载此变量进DOM对象里。

```
VariantInit(&var);
var = VariantString("axis.xml");
HRCALL(pXMLDom->load(var, &status), "dom->load( ): ");
/* 选用 SelectSingleNode 方法来定位结点，获取信息。*/
bstr = SysAllocString("//axis2_placement_3d//");
HRCALL(pXMLDom->selectSingleNode(bstr, &pNode),
    "dom->selectSingleNode: ");
if (!pNode)
  {
  ReportParseError(pXMLDom, "Calling selectSingleNode ");
   }
/* 指定方式输出 */
else {
    dprintf("Result from selectSingleNode:\n");
    if (bstr) SysFreeString(bstr;
    HRCALL(pNode->get_nodeName(&bstr)," get_nodeName ");
    dprintf("Node, <%S>:\n", bstr);
    if (bstr) SysFreeString(bstr);
    HRCALL(pNode->get_xml(&bstr), "get_xml: ");
    dprintf("\t%S\n\n", bstr);
     }
```

如果换用 SelectNodes 来定位结点，需要利用 get_length 来确定目标结点的数量，当变量 i≦get_length 时，执行循环语句，再利用 get_item 逐条列出目标结点的属性值。

```
bstr = SysAllocString( "//axis2_placement_3d// " );
/* 用 SelectNodes 方法来定位结点，获取信息。*/
HRCALL(pXMLDom->selectNodes(bstr, &pNodes), "selectNodes ");
```

```
if (!pNodes)
{
    ReportParseError(pXMLDom, "Null while calling selectNodes ");
 }
else {
    dprintf("Results from selectNodes:\n");
  /* 得到目标结点个数 */
    HRCALL(pNodes->get_length(&length), "get_length: ");
  /* 循环判断结点数目，并按指定格式输出 */
    for (long i=0;   i<length;   i++)
       {
        if (pNode) pNode->Release();
        HRCALL(pNodes->get_item(i, &pNode), "get_item: ");
        if (bstr) SysFreeString(bstr);
        HRCALL(pNode->get_nodeName(&bstr), "get_nodeName: ");
        dprintf("Node (%d), <%S>:\n",i, bstr);
        SysFreeString(bstr);
        HRCALL(pNode->get_xml(&bstr), "get_xml: ");
        dprintf("\t%S\n", bstr);
       }
  ......
 }
```

XML 文件通过 SelectSingleNode 接口和 SelectNode 接口之后，通过设置 Node 对象和 Nodelist 对象，分别得到的是指定的首个目标结点和指定目标结点的多个值。这样，就可以根据 XML 文件中结点的属性类型，及其父结点下子结点数目的不同来选择定位结点的方法提取属性信息。

4.3.5 STEP AP214 文件的生成

STEP AP214 文件的生成是 STEP/XML 数据交换系统中最后的步骤。生成 STEP AP214 文件的过程主要是将 AP214 协议的 AIM 模型中相关几何、

配置信息的实体表达按照 STEP Part21 物理文件的语法语义规则进行映射，及其通过 XML 中提取结点信息的赋值过程。

1、STEP AP214文件生成原理

本文对于STEP对象的许多操作都是基于EXPRESS进行的。STEP Tools公司开发的ST-Developer工具提供很好的对于EXPRESS的处理机制。其中ST-Developer开发环境下的ROSE C++类库提供了丰富的C++类资源，以及面向对象的标准数据访问接口，可以实现对STEP标准的特殊应用。EXPRESS Complier是ST-Developer里的另一个应用工具，通过它可以将EXPRESS 模型中的每一个实体转换生成对应的EXPRESS C++类。应用这些C++类，就可以通过应用程序写入属性信息，完成对STEP对象的操作。

XML文件解析完成之后会在内存中形成一个结点树，便于后续遍历、查询结点。XML文件相关信息提取又是基于此结点树进行的。提取出的每个结点都对应AP214文件中的实体及属性。由EXPRESS编辑器生成的EXPRESS C++类是匹配每一个EXPRESS模型中的实体类型的，并且这些信息都被定义在ROSE库函数里的RoseObject类下。而EXPRESS数据字典是EXPRESS模型定义数据结构的编译设计对象。这些对象被定义为ROSE库函数里RoseDesign下面的设计对象，包括RoseDomain和RoseAttribute[43]。因此，在写入STEP AP214文件的时候，就可以根据提取出的XML文件中的实体结点和属性结点在RoseDesign下遍历寻找相应的应用元素。

由于STEP/XML交换系统中源数据信息是从网络传输得到的XML文件，通过前面对XML文件的解析和属性信息的提取，已经为后序的STEP AP214文件的生成做好了充分的准备。在XML文件的解析处理中，每一个ENTITY结点都代表STEP AP214文件中的一个实体，每一个Attribute结点都表示相应实体的实体属性类型，每一个文本结点都是实体的属性值。因此，在利用EXPRESS Complier对EXPRESS定义数据操作生成C++类和数据字典之后，再利用ROSE库函数就可以找到对应实体C++类，以及在数据字典中的表达。最后ROSE库函数会把实体对象载入内存，通过应用程序的编译，以STEP Part21的形式写出，即.step文件格式。如图4.17所示。

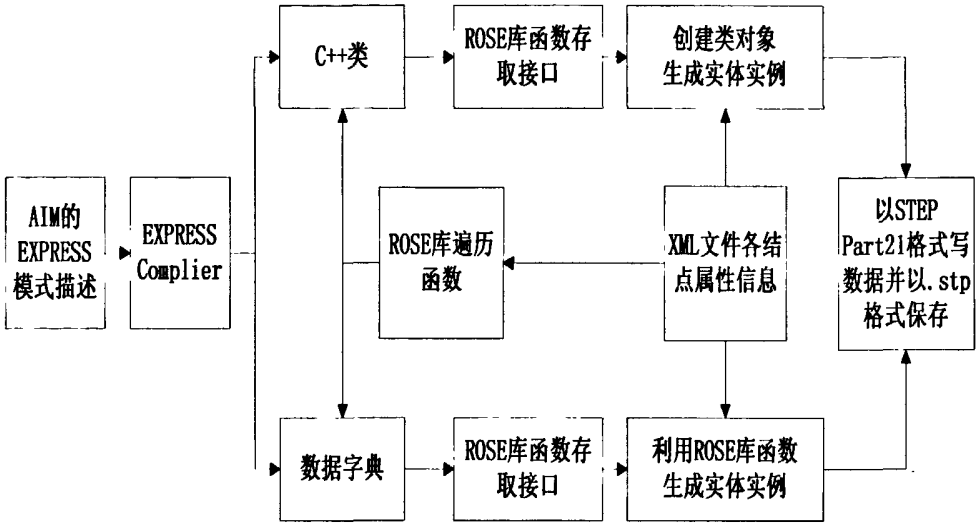

图4.17 STEP AP214文件的生成原理图

2、STEP AIM中的映射表达

STEP 标准中的应用解释模型(AIM)是通过其内部的各个功能单元(UOF)的映射来实现的。应用协议中的映射表是实体到对应模块映射的基础。映射表表示了如何从功能单元和应用对象转换到一个或多个应用解释模型[51]。

在AIM的映射表中，定义了每个确定的信息需求(应用对象和应用声明)是怎样表达的。主要组成部分包括应用元素、AIM元素、源、规则、参考路径，由五部分构成。映射表的格式如表4.6[27]所示。

表4.6 AIM部分映射表信息

应用元素	AIM元素	源	规则	参考路径
plan_nomal	axis2_placement_3d.axis.	1030342		representation representation.items[i]->representation_item representation_item=>geometric_representation_item=>placement=>axis2_placement_3d

其中的符号有其不同的含义：

[i]：表示下行所列元素的聚合属性。

->：表示下行所列实体类型和选择类型的属性参考。

=>：表示当前实体是下行所列实体的超类。

<=：表示当前实体是下行所列实体的子类。

（1）应用元素：显示的是应用元素的名称，它与EXPRESS里定义的对象名称是一致的。若是应用对象的名称则用大写形式显示，属性和声明信息则在应用对象下方用小写形式表达。

（2）AIM元素：显示的是在AIM模型中AIM元素的名称，通常用术语“IDETICAL MAPPING”和“PATH”的大写形式出现，若是AIM模型中的实体元素，则需用<实体名称><属性名称>的顺序来说明。

（3）资源：对于AIM元素而言，是通过集成资源和应用解释结构来解释表达的，此列就是告知AIM元素集成资源的所属部分。

（4）规则：应用在AIM元素里面的规则会有许多条，对于规则本身而言，主要来自于应用对象之间的关系表达。若从整个AIM元素的映射表中来看，会涉及到很多相同的规则。

（5）参考路径：要完整地映射一个应用对象，了解与其相关的AIM元素及其定义的参考路径是非常必要的。此参考路径列出了一系列与之相关的AIM元素，并用符号清楚表达它们之间的关系。对于每一个AIM元素来说，在此部分都创建了一条从集成资源中寻找其超类的途径。

从映射表信息可以看出，它对实体的每一个属性都给了全面的解释，涉及到了属性的来源及属性信息的参考路径。以实体为对象分析这个映射表，它全面的给出了实体本身和它所包含属性之间的关系，并且对各个属性内容给出相应的解释。而映射表中的实体、属性关系的表达符合 AP214 协议的语法语义规则，正是 ROSE 库函数读写数据并最终生成 STEP AP214 文件的基础。

3、STEP AP214文件头段信息的生成

在ROSE库里编译STEP数据有早联编和晚联编两种。早、晚联编各有自己的优势，然而，针对STEP数据描述，每一个STEP描述都至少有一种模式(Schema)，晚联编可以通过数据字典为此提供更多的解决方式，早联编方式

也可以通过产生C++类来实现。本文依旧采用混合联编的方式来完成。

在 STEP 物理文件的头段信息里，主要由三部分组成：文件描述、文件名和文件模式。文件描述主要是文件内容的说明，文件名部分主要包括文件的名称、文件的建立日期、作者、所属单位等，文件模式则是说明数据段实体所采用的应用协议。在创建一个新的设计对象时，利用 RoseDesign 类的 initialize_header( )函数建立头段对象信息，具体信息由 header_description( ) 和 header_name( )来创建，文件模式(file_schema)由 useSchema( )函数写入。相关信息是提取的 XML DOM 结点信息。

首先，建立一个设计对象，用useSchema写入AP214文件的模式:

```
RoseDesign * design = pnew ("zhou", "  automotive_design");
Rose.useSchema("automotive_design ", design);
design-> useSchema("automotive_design ");
/* 写入其他头段相关信息*/
design->initialize_header();
design->header_name()->originating_system("pro/E 2.0");
design->header_description()->description()->add("AP214 file");
design->header_name()->date_time()->add("2008,11,26");
design->header_name()->author()->add("fuxiujuan");
design->header_name()->organization()->add("CIMS lab");
……
// 以STEP Part21格式写入并保存
 design->format("step");
 design->save as(“zhou_out”);
```

4、STEP AP214文件数据段的生成

生成数据段的过程，首先需要用EXPRESS Complier对AP214的EXPRESS模式文件进行编译，使其转化成对应实体的C++类或数据字典。在生成数据段的应用程序中使用这些类创建对应实体的实例，再根据XML解析过程中提取出的实体结点名称，通过ROSE遍历函数找到对应结点的实体C++类，并给实体中的属性赋值。应用协议中每个实体都具有相应语法和约束规则，因此在生成实体实例时必须满足各实体的约束规则和语法要求，这样生成的

AP214文件才能满足一致性，主要表现在应用程序中按照实体约束规则的描述，用对应的C++类生成满足约束规则和语法要求的实体实例。

下面以STEP AP214文件中定义的axis2_placement_3d实体为例，讲述应用C++类生成STEP AP214文件的过程。

axis2_placement_3d实体的EXPRESS定义为：

```
ENTITY  axis2_placement_3d
    SUBTYPE OF (placement);
    axis : OPTIONAL direction;
   ref_direction : OPTIONAL direction;
END_ENTITY;
```

由EXPRESS定义可以清楚，实体axis2_placement_3d是实体placement的子类，则实体axis2_placement_3d就继承了实体placement的所有属性。axis和ref_direction 都是实体axis2_placement_3d的属性名称，两者的属性类型均为optional。

通过 EXPRESS Complier 对上述 EXPRESS 描述编译之后，可将axis2_placement_3d 转化为对应实体的 C++类的形式，如下表示：

```
# define axis2_placement_3dOffsets(subClass)
    placementOffsets(subClass)
ROSE_DECLARE (axis2_placement_3d) :
    virtual public placement
{
private:
    direction * PERSISTENT_axis;            /* OPTIONAL */
    direction * PERSISTENT_ref_direction;  /* OPTIONAL */
 public:
    ROSE_DECLARE_MEMBERS(axis2_placement_3d);
    direction * axis()
    {  return ROSE_GET_OBJ (direction,PERSISTENT_axis); }
    void axis (direction * var) {
    ROSE_PUT_OBJ (direction,PERSISTENT_axis,var);
```

```
    }
     direction * ref_direction()
     { return ROSE_GET_OBJ (direction,PERSISTENT_ref_direction);
     }
    void ref_direction (direction * var) {
    ROSE_PUT_OBJ (direction,PERSISTENT_ref_direction,var);
    }
    axis2_placement_3d();
    axis2_placement_3d(
    const char* PARAM_name,
   cartesian_point * PARAM_location,
   direction * PARAM_axis,
   direction * PARAM_ref_direction);
}
```

经编译后的实体 C++类包括了所有实体中的信息及约束，将所生成的C++类添加到应用程序的工程中，作为实体赋值过程的支持类。下面为实体属性赋值并最终生成 STEP AP214 文件的部分代码。

```
    ……
    RoseDesignCursor  cursor;
    cursor.traverse(design);
    cursor.domain(ROSE_DOMAIN(axis2_placement_3d));
    axis2_placement_3d* asmSDR;
    axis2_placement_3d * compSDR;
    asmSDR=ROSE_CAST(axis2_placement_3d, cursor.next());
    compSDR=ROSE_CAST(axis2_placement_3d, cursor.next());
  // 赋值过程，此处 p1、p2 为向量值
    Rosedesign->entity_id()->add(“#25”)
    ListOfDouble* coords = pnew ListOfDouble();
    coords->add(0.000000);
    coords->add(0.000000);
```

```
coords->add(1.000000);
cartesian_point* p1 = pnew cartesian_point("", coords);
axis2_placement_3d* asmAxis = pnew axis2_placement_3d("", p1, 0,
0);
coords = pnew ListOfDouble();
coords->add(1.000000);
coords->add(0.000000);
coords->add(0.000000);
cartesian_point* p2 = pnew cartesian_point("", coords);
axis2_placement_3d* compAxis = pnew axis2_placement_3d("", p2, 1,
0);
……
// 以 STEP Part21 的形式输出
design->format("step");
// 以 zhou_out 的名字保存
design-> save as ("zhou_out");
```

实体 axis2_placement_3d 经程序编译之后，转化成的 AP214 文件为：

```
#25=AXIS2_PLACEMENT_3D('',#42,#43,#44);
#42=CARTESIAN_POINT('',(1.0,0.0,1.0));
#43=DIRECTION('',(0.0,0.0,1.0));
#44=DIRECTION('',(1.0,0.0,0.0));
……
```

根据 axis2_placement_3d 实体的转换实例,可以知道 STEP AP214 文件中其他实体的实例化过程，进而完成整个 STEP AP214 文件的转换。

4.4 本章小结

本章是整篇文章的重点，针对整个数据交换系统进行了开发并作以详细的描述。首先，给出了整个数据交换系统的功能流程图，说明数据信息在整个系统的流向。然后又对整个数据交换系统的开发中所应用到的关键技术和应用工具如 ST-Developer、ROSE 库、EXPRESS Complier 进行了详细介绍。

在具体的 STEP/XML 数据交换系统实现过程中，完成了前处理器和后处理器的功能。首先利用上述工具完成对 STEP AP214 文件数据信息的提取，利用 EXPRESS 与 XML 语言之间的映射关系，通过 XML 编译系统的设计完成向 XML 文件的转换过程，完成前处理器的功能。在后处理器的实现过程中，需要先对网络传输来的 XML 文件进行解析，则本文在对 XML 解析接口 DOM 及其内部提供的诸多应用接口进行研究之后，映射开发出了符合本数据交换系统的 XML DOM 解析器，又通过 STEP AIM 的映射要求，及 ROSE 库函数对 STEP 对象的操作，最终完成 XML 文件向 STEP AP214 文件的转换。进而完成整个数据交换过程。

第 5 章 数据交换系统的交换实例

在前文已经介绍了为实现异地异构系统之间的数据交换，为了适应网络信息共享的需要，本文开发了基于 STEP 标准与 XML 语言的数据交换系统。并对整个数据交换系统作以详细地介绍。为了保证此系统在数据交换过程中的准确性和一致性。本章将以三维 CAD 软件(UG)建立的轴的三维模型的 STEP AP214 文件为源数据，使其完成在本系统中的数据交换过程，验证本系统的可行性。

5.1 STEP/XML 数据交换系统的交换实例

首先，需由 UG 软件建立一个阶梯轴的三维模型。由于轴在机械行业的众多领域都有非常重要的应用，尤其在传动系统中，是非常常见的零、部件，有着非常广泛的用途。因此，本例选用轴的三维模型作为数据交换对象。如图 5.1 为轴的三维模型图。

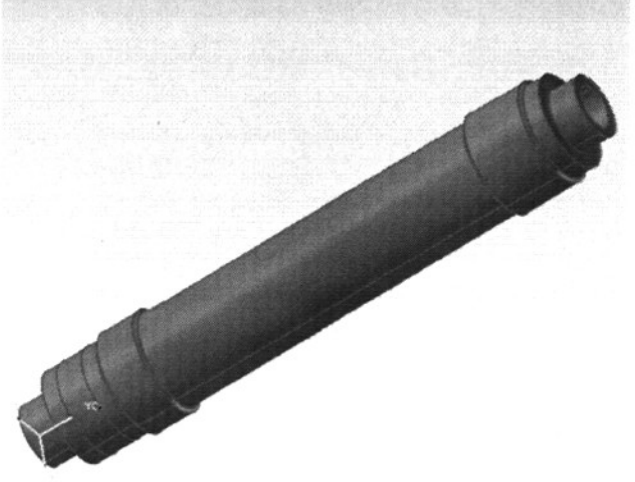

图 5.1 轴的三维模型图

由于 UG 软件支持 STEP AP214 接口，所以，UG 可以把轴的三维模型以 STEP 中性文件的形式来表达，即.stp 文件的格式。下图 5.2 是经 UG 软件输

出的轴的三维模型的 STEP AP214 文件经由 STEP 文件浏览器后的显示结果。

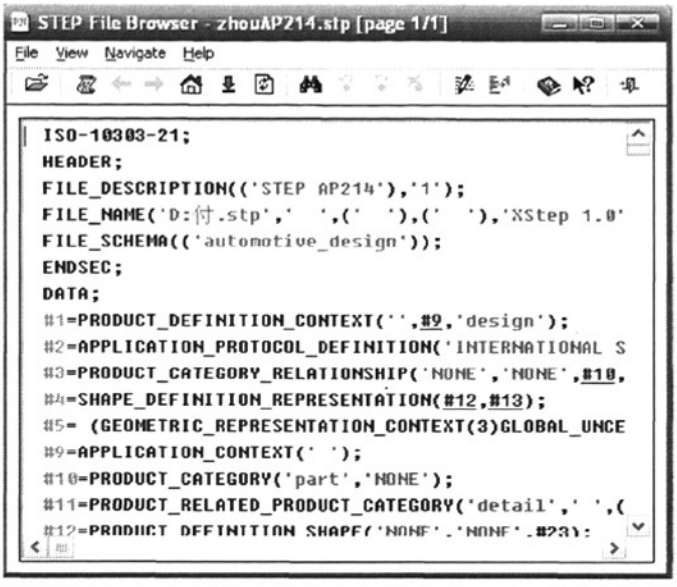

图 5.2 轴的 STEP AP214 文件

接下来，需要经过 STEP/XML 数据交换系统将零件轴三维模型的 STEP AP214 文件转换成适于网络传输的 XML 文件，并需在 Windows2000 Server 系统上发布，图 5.3 为数据转换图。图 5.4 为发布的 XML 网页。

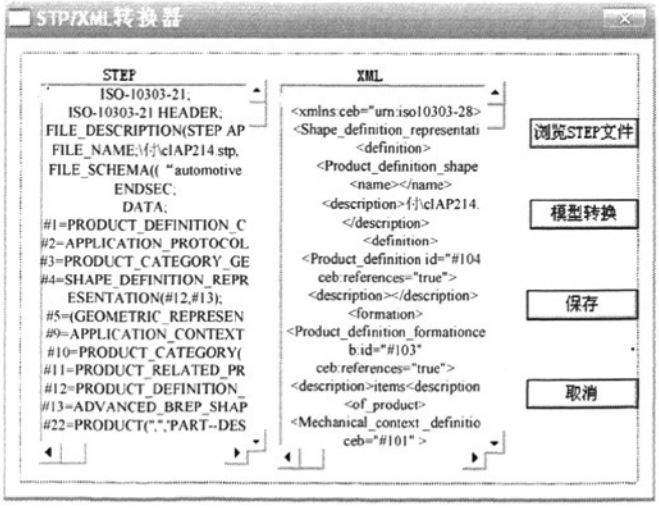

图 5.3 数据交换界面

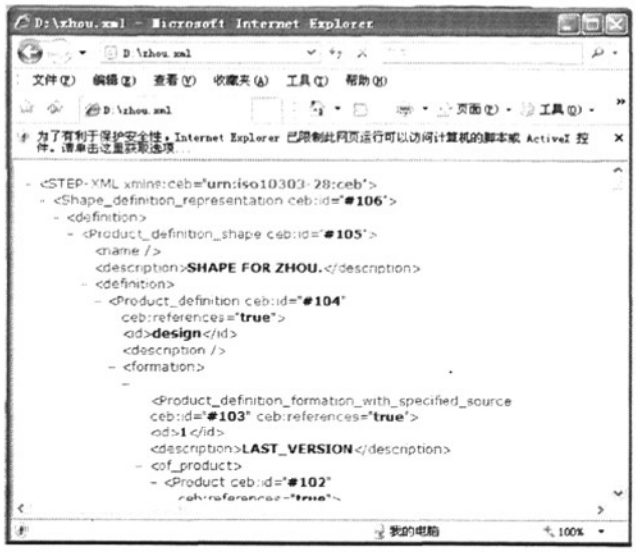

图 5.4 服务器端发布的 XML 文件网页

服务器终端接收到网络服务器上的 XML 信息之后，可在界面上显示，也可以.xml 文件的形式保存。再经过 XML DOM 解析器，执行数据交换的过程。最后映射成为 STEP AP214 文件。如图 5.5 和图 5.6 所示。

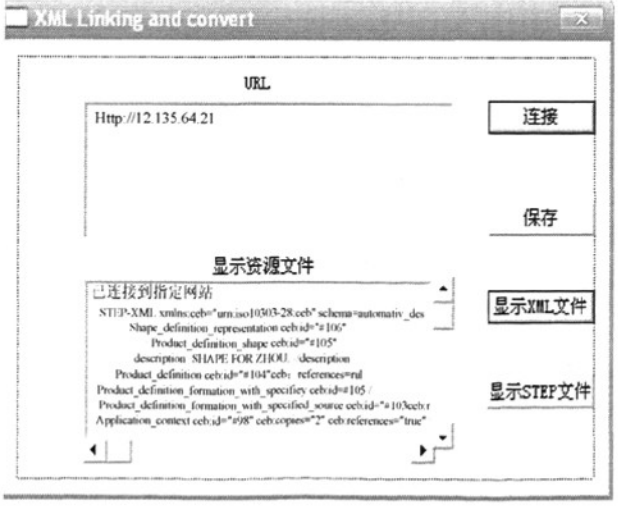

图 5.5 获取网络服务器上的 XML 信息

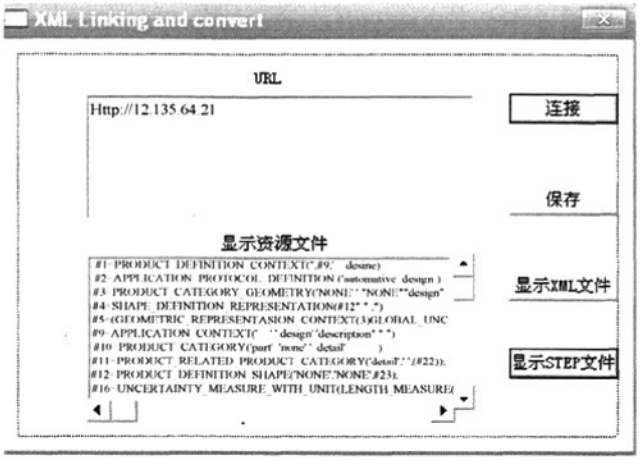

图 5.6 XML 文件转换成 STEP AP214 文件

最后重新生成的 STEP AP214 文件，通过 UG 软件重新加载 STEP 数据，生成的三维模型如图 5.7 所示。

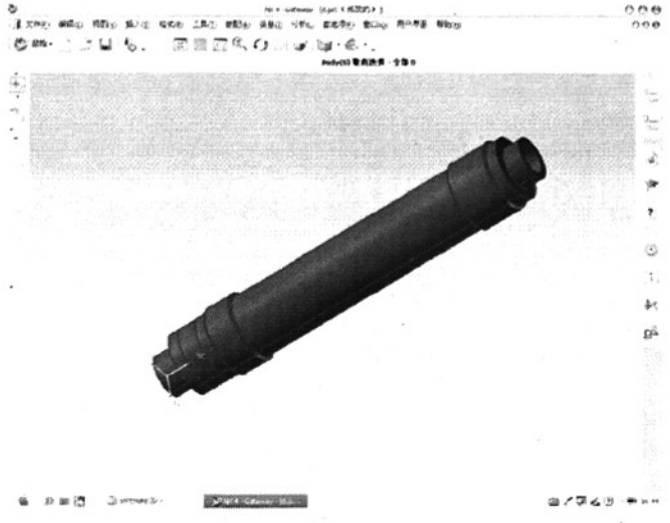

图 5.7 UG 中 STEP 数据模型生成图

由于本文所述的数据交换系统支持异构系统之间的数据交换，所以在其他的 CAD 系统中加载重新生成的 STEP AP214 文件也是可行的。图 5.8 和图 5.9 分别为系统转换生成的 STEP 数据在 pro/E 和 Catia 中加载 STEP AP214 文件后生成的三维模型图。

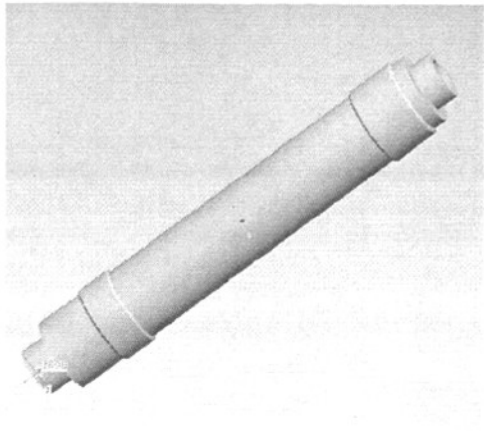

图 5.8 Pro/E 中 STEP 数据模型生成图

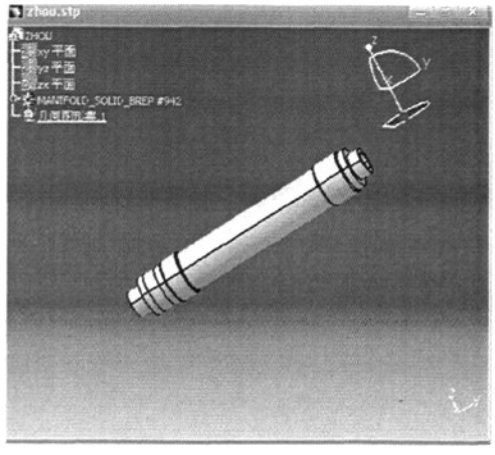

图 5.9 Catia 中 STEP 数据模型生成图

通过上述实例可以知道，STEP/XML 数据交换系统可以把一个支持 STEP 接口的三维的实体模型数据交换成为 XML 数据，经由网络传输之后，接收端经过 XML 文件的解析和映射过程，最终又可重新生成 STEP 格式文件，并可由支持 STEP 接口的所有 CAD 软件进行模型重构，达到数据交换的目的。说明了信息载体 XML 语言在网络共享和数据交换中的作用。验证了此数据交换系统的可行性。

5.2 数据交换完成后的对比分析

上述完成了整个数据交换系统的交换过程（以轴的三维模型为源文件）。通过对数据交换前后模型的对比分析，可以得出交换质量的好坏。进而可以

总结此系统的优缺点，便于以后地完善。

从三维 CAD 软件 UG 中生成的零件轴三维模型的 STEP AP214 文件，经数据交换系统后，将会重新生成。此时生成的 STEP AP214 文件又可重新加载进支持 STEP AP214 接口的三维 CAD 软件中，进行模型重构的过程。下面就将重构后的模型与源模型进行比较。可得出以下几点：

（1）转换后的 STEP 文件在进行模型重构的过程中可以实现 STEP AP214 文件中所有实体的转换。图 5.10 和 5.11 分别为 Pro/E 和 CATIA 中导入 STEP AP214 文件后模型重构时生成的实体转换文件。

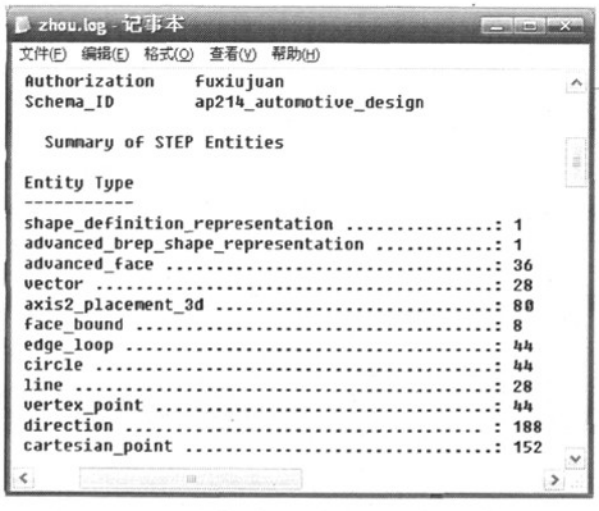

图 5.10 Pro/E 中导入 STEP AP214 文件时实体转换文件

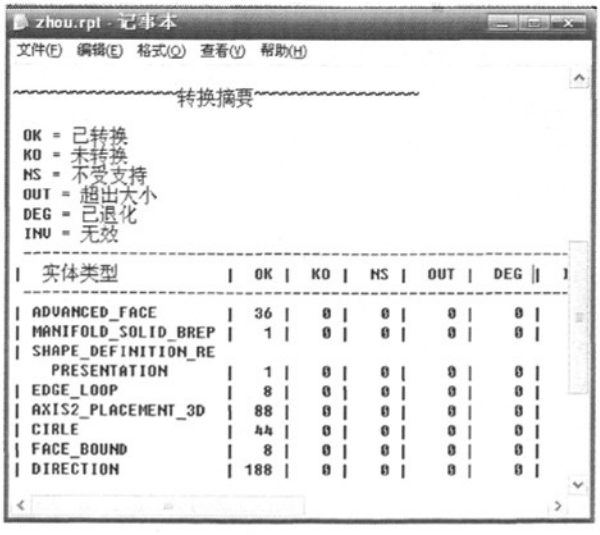

图 5.11 Catia 中导入 STEP AP214 文件时实体转换文件

（2）经转换得到的模型在三维 CAD 软件中显示时，圆周曲面不再是一个整体曲面，被分为两个相交曲面。

（3）转换后的模型显示在三维 CAD 软件中，无法转换模型树和特征构造表。即无法对模型中的尺寸进行显示和修改。

5.3 本章小结

本章主要完成一个机械传动系统中经常用到的轴类零件的模型信息，在异地异构系统的环境下，经过 STEP/XML 数据交换系统之后得到的模型重构的过程。本文主要完成的是 STEP/XML 数据交换系统中前处理器和后处理两个工具的开发。首先把轴类零件的三维模型数据转换成 STEP AP214 文件，经由前处理器后，生成 XML 文件，并可通过服务器发布在 Internet 上，然后其他用户的客户端就可以根据网址直接访问网页，并把 XML 文件直接加载到应用程序内，通过 XML DOM 解析器的解析和映射过程，最后重新生成 STEP AP214 文件，供支持 STEP 接口的其他应用程序和系统软件使用。验证了 STEP/XML 数据交换系统的可行性。同时，也对数据交换后得到的模型与源模型做了对比分析，总结了转换前后的不同之处。

结　论

随着经济全球化及 Internet 技术的不断发展，不同企业之间、异地异构系统之间合作的加剧。对于数据交换和数据共享的要求也越来越高。STEP 中性文件可以通过 SDAI 实现不同 CAD 系统之间的数据交换，但是对于异地异构系统之间的数据交换和信息共享，就一定要借助 Internet 技术的帮助。而描述和定义 STEP 数据的 EXPRESS 语言由于其自身的复杂性无法被浏览器所支持。因此，XML 语言由于其可扩展性和网络适用性等优势成为了当今数据交换的首选。基于 STEP/XML 的数据交换系统的提出和发展是网络化数据交换和信息共享的必然选择。

本文针对 STEP/XML 数据交换系统内部的前处理器和后处理器作了详细地研究，其内部的各个应用环节都进行了设计开发。主要完成的研究工作体现在以下几个方面：

（1）通过对可扩展标记语言 XML 和 STEP 标准的研究，确定可以将 XML 语言作为网络信息传输的载体，又因为 XML 语言对于结构化信息的描述功能和 STEP 中性文件在异构系统之间的通用性。决定将 STEP 标准与 XML 语言结合起来，实现异地异构系统的数据交换。

（2）根据对 ISO10303 28 标准的研究，建立了 EXPRESS 语言与 XML 语言之间的映射关系，包括模式、实体、属性、类型、约束和规则的映射。给出了映射规则。研究了用 XML 语言描述 EXPRESS 中继承关系的方法，为 STEP 数据向 XML 数据的交换提供了理论依据。

（3）完成前处理器将 STEP AP214 文件转换成 XML 文件的功能。以 ST-Developer 工具和 VC++6.0 为开发平台。首先利用 ROSE 库函数将 EXPRESS 语言描述的 STEP AP214 文件转化成 STEP 数据对象，通过词法分解、语法、语义分析，对 STEP 数据进行处理，然后按照 AP214 数据模型的信息层次结构逐次提取数据信息。再利用建立好的 EXPESS 语言与 XML 语言之间的映射关系，把 AP214 文件逐句的转换翻译成 XML 语句。最终生成 XML 格式文件。同时，一定要保证生成的 XML 文件符合提前已经建立好的

STEP Part21 文件的 XML Schema 的结构。这样才会生成格式良好的 XML 文件，保证它的有效性。

（4）完成后处理器将服务器上得到的 XML 文件重新转换生成 STEP AP214 文件的功能。首先选用了 DOM 作为 XML 文件的解析接口，根据 DOM 中各应用接口的研究以及系统后序操作的要求，设计了 XML 的 DOM 解析器对 XML 文件进行解析，并生成 XML 文件的 DOM 结点树。通过对 DOM 结点树的遍历和 SelectNodes(SelectSingleNode)接口的应用，完成对 XML 文件结点信息的提取。利用建立好的 EXPRESS 与 XML 之间的映射关系，找寻对应的 EXPRESS 模式数据。根据 AP214 文件中的实体在 AIM 中的映射表达，以及利用 ROSE 库函数进行实体的赋值，完成写出 STEP AP214 文件的过程。

（5）利用三维 CAD 软件构建的轴类零件的数据模型，通过 STEP/XML 数据交换系统完成数据交换之后，可在其他的 CAD 软件中重新构建模型。证实了此数据交换系统的可行性。

本文也进一步提出对本研究方向后序工作的展望：

（1）STEP AP214 协议覆盖的范围很广，其功能单元可以涉及到机械行业的各个方面。由于时间的关系，本文只针对此协议的一致性类中的 CC1 和 CC2 中的部分实体进行了映射和转换研究。其他的实体转换关系需要后序的工作完成。

（2）利用 XML 语言在网络上传输信息，可以大大提高系统的效率，也为异地异构系统的数据交换提供了条件。但是，考虑到异构系统数据交换的安全性，后续工作还可以对上传的 XML 文件作以密码或数字签名保护。

参考文献

[1] 叶修梓，彭维，何利力. 从工业界的角度看 CAD 技术的研究主题与发展方向. 计算机辅助设计与图形学学报. 2003：1194-1198 页

[2] 殷毅，祖旭，周峰，王永福. XML 在计算机支持的系统设计系统中的应用. 重庆工学院学报. 2005，18(2)：21-22 页

[3] 刘锡冬. 基于 XML 的数据交换的研究与应用. 山东大学硕士学位论文. 2007：2-6 页

[4] 赵秀栩，盛步云. 基于 Web 的产品信息共享交换技术研究. 武汉理工大学学报. 2006，21(4)：418-420 页

[5] 贾素玲，王强，姚琪舒，舒毕磊，柴庆慧. XML 技术应用. 北京：清华大学出版社，2007：217-224 页

[6] 钟巍. 数据交换模型研究与实现. 武汉理工大学硕士学位论文. 2008：3-5 页

[7] Shiau Joe Y, Ratchev S M, George Valtchanov. Distributed Collaborative Design and Manufacturing Assessment for Extended Enterprise in XML-based Agent System. http://www.dsi.unimo.it/WETICE2006/shiau.pdf

[8] Polly M.S.Poon, Tharam S.Dillon, Elzabeth Chang. XML as a basis for Intetoperability in Real Time Distirbuted system. WSTFEUS. 2004: 17-25 P

[9] Kenneth.Chiu, Wei Lu, YinFei.Pan. A parallel approach to XML pasing. Department of Computer Science State University of New York-Binghamton. 2006: 223-227 P

[10] 赵宝翠，张勇. 基于 XML 的 Socket 方式实时数据交换. 计算机工程学报. 2006，32(11)：83-85 页

[11] 黄红明，尹志兵，熊佳喜. 基于 XML 的数据交换技术的研究及其在大型系统中的应用. 计算机工程学报. 2003，18(5)：139-141 页

[12] 齐建军. 基于 Web 制造资源的优化配置及数据交换系统研究. 北京航空航天大学硕士学位论文. 2005：69-80 页

[13] 徐亨钟，蒋海斌，王精业．基于 XML 的数据交换格式及其应用．计算机工程学报．2003，20(4)：77-80 页

[14] 谢莉莉，林春梅，陈家训．基于 XML 的数据交换中心原型系统．计算机工程学报．2002，15(8)：100-102 页

[15] NIKOLA, B.SERBEDZIJA. Web computing framework. Journal of System Architecture. 2000，20(3): 1293-1306 P

[16] APPELT W, BUSBACH. U. A Web based application to support cooperated design. http://www.simwe.com/article/art/2/.

[17] 简峥峰，谭建荣．基于 XML 的 STEP 产品数据网上描述与识别．计算机辅助设计与图形学学报．2002，13(11)：983-990 页

[18] 王敏，吴波．应用 STEP 和 XML 进行产品信息共享．华中科技大学机械与电子学报．2007，25(1)：58-60 页

[19] 仇晓黎，易红，吴锡英，江勇．STEP/XML 产品数据模型转换编译系统的实现．成组技术与生成现代化．2004，21(4)：5-8 页

[20] 仇晓黎，易红，吴锡英，周维．基于 STEP/XML 标准的产品数据模型转换系统的开发．东南大学机械工程学报．2003，19(5)：1-4 页

[21] 殷毅．基于 XML 的协同设计中的异构产品信息集成技术研究．大连理工大学硕士学位论文．2004：12-14 页, 21-23 页

[22] 李洋，钟延修．基于 STEP 和 XML 的产品数据交换与集成在网络化制造中的应用．机床与液压．2002：116-118 页

[23] Z.M.Ma, HUAQing.Wang. STEP implementation of imperfect EXPRESS model in object-oriented databases. http:// www.elsevier.com/locate/fss.

[24] 王建德．基于特征的零件库信息模型及其实现的研究．哈尔滨工程大学硕士学位论文．2006：16-17 页

[25] Z.M.Ma. Extending EXPRESS for imprecise and uncertain engineering information modeling. J.Intell.Manuf. 2006(7): 57-83 P

[26] 阳小良，张葵葵，谭理刚，胡光辉．基于 XML 模式的车身几何模型表达．汽车与船舶工程学报．2007(161)：147-151 页

[27] ISO/DIS 10303-214. Industrila automation systems and integration product data representation and exchange—Part 214: Application protocol: Core

data for automotive mechanical design processes. 2001: 2512-2525P, 3047-3056 P, 2170-3306 P

[28] 刘哲，孙军，王军，聂新刚，王国勋. 基于STEP AP214特征信息模型的研究. 机械设计与制造. 2005(11)：146-148页

[29] Frank P.Coyle. XML, Web Services, and the Data Revolution. Addison Wesley Information Technology Series. 2006: 85-92 P

[30] 左伟明. XML数据标记语言参考手册. 北京：人民邮电出版社，2007：35-49页，51-60页

[31] 黄立冬，王新，施国兴，陆歌皓. 基于模板的XML文档表示技术研究. 云南民族大学自然科学学报. 2008，17(4)：355-358页

[32] J.Gu, B.Xu, X.Chen. An XML query rewriting mechanism with multiple ontologies integration based on complex semantic mapping on Web. Information Fusion. 2008(13): 25 P

[33] Cascini G, Giovani G, Rissone P, and Rotini F. Integrated design of turbomachinery through a XML-SOAP based Web. http://xml. coverpages.org/soap, html.

[34] 董仁义. 基于STEP标准的产品信息模型及零件库系统的研究. 武汉大学硕士学位论文. 2005：3-5页

[35] ISO/TS 10303-11. TC184/SC4:Product Data Representation and Exchange —Part 11: The EXPRESS Language Reference Manual. International Standard. 1995: 6-8 P, 17-42 P

[36] ISO/TS 10303-28. Product data representation and exchange: Implementation methods: XML representations of EXPRESS schemas and data. 2002: 6-10 P, 13-39 P, 81-97 P

[37] Edward J, Barkmeyer, Joshua Lubell. XML Representation of EXPRESS Models and Data. Manufacturing Systems Integration Division National Institute of Standards and Technology. 2004: 1-3 P

[38] 吴卓，董唯光，梁秀娟. 基于STEP和XML的网络数据库的建立和集成. 机床与液压. 2005(18)：145页

[39] Yuh-Jen Chen, Yuh-Min Chen. An XML-based modular system pasing from

EXPRESS model. National Kaohsiung First University of Science and Technology. 2007: 6-8 P

[40] STEP TOOLS, Inc. What's new in ST-Developer v12. http://www. step tools.com/product/index.html

[41] STEP TOOLS, Inc. ROSE Library Referance Manual. http://www. step tools.com .

[42] STEP TOOLS, Inc. SDAI C Library Referance Manual. http://www. step tools.com .

[43] 祁型虹，李永峰. 基于STEP-XML的网络制造资源描述. 湖北工业大学学报. 2006，21(3)：204-205页

[44] 李雯，谢辅雯，邹道明. XML数据交换技术的应用与研究. 计算机与现代化. 2008(149)：92-93页

[45] 刘营，王斯梁，谢跃美. 面向C++的XML解析器的实现与研究. 计算机应用研究. 2007，24(12)：268-270页

[46] Megginson D. The Simple AIP for XML version 2.0. http://www. megginson.com/index.

[47] 黄国信，郭徽. 基于XML数据交换中解析方法研究. 计算机工程与设计学报. 2007，28(24)：6000-6001页

[48] 史西兵. 基于XML的Web信息抽取技研究. 西北大学硕士学位论文. 2008：17-26页

[49] Microsoft XML Core Service 4.0 DOM Referance/DOM Method. http://microsoft/software/DOM.

[50] Edward J, Barkmeyer, Joshua Lubell. XML Representation of EXPRESS Models and Data. Manufacturing Systems Integration Division National Institute of Standards and Technology. 2004: 24-27 P

[51] David Loffredo. Fundamentals of STEP Implementation. STEP Tools, Inc. 2002: 7-11 P

致　谢

攻读硕士研究生的光阴转瞬即逝，留下了许多美好而又难忘的时光。在完成论文之际，对在学习和生活中给予我莫大帮助和悉心关怀的邱长华教授致以衷心的感谢。邱老师在课题的进展过程中不断地给予我指导和启迪，他严谨的治学态度和创新求实的科学精神深深地影响着我，相信会成为我今后的学习和工作中一笔宝贵的财富，我将终身受益。再一次感谢恩师对我的教导和培养。

与此同时，我还要对CIMS实验室的张家泰教授，薛开教授，钟宇光副教授在本文的选题、完成过程中给予我的指导表示由衷的感谢。他们为本文的完成提出了许多宝贵的意见，使我受益非浅。

感谢所有被本文所引用文献的作者，以及实验室帮助过我的所有老师和同学们。

在此，深深地感谢我的父母这么多年来对我的培养和鼓励，在物质和精神上不断地支持着我，使我能顺利完成学业。父母为我付出了很多，我会在以后的生活中回报父母的恩情。