流媒体构件库的设计与实现

摘要

本文所讨论的内容基于当前两个热点技术，一是伴随着 Internet 和多媒体技术蓬勃发展应运而生的流媒体技术；二是软件复用技术在软件工程领域不断演进，继结构化程序设计、面向对象的软件设计后新兴的构件及构件库技术。本文着力将构件库技术引入流媒体软件开发，旨在提高流媒体软件开发的效率和质量。

本文首先研究了构件及构件库技术，总结了流媒体软件的基本功能单元和体系结构，在此基础上将构件技术引入流媒体软件开发中，利用构件库的思想和方法设计实现了流媒体构件库系统，首批入库 24 个自主研发和第三方开发的流媒体构件，最后通过流媒体构件库系统搭建了视频会议系统，进一步验证了本文的设计思想。

关键词： 流媒体，构件，构件库，软件复用

DESIGN AND IMPLEMENTATION OF A COMPONENT LIBRARY FOR STREAMING MEDIA

ABSTRACT

The work of this paper is based on two current hot technologies. The first is streaming media, the second is component library. Component and component library technologies are used for improving the efficiency of developing streaming media software.

In this paper, first of all, we studied component and component library technologies. Second, we summed up basic function units and architecture of streaming media software. Then a component library for streaming media is designed and implemented. 24 components developed by our laboratory and the third party are stored in it. After that, a video conference was built up with the help of the library and validated the idea of this paper.

KEY WORDS: Streaming media, Component, Component library, Software reuse

创新性声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：马海涛　　日期：2006.2.28

关于论文使用授权的说明

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在__年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名：马海涛　　日期：2006.2.28

导师签名：　　日期：2006.2.28

第一章 引言

1.1 研究背景

随着计算机制造业发展的突飞猛进，硬件技术及硬件性能的提高速度总是快于软件，著名的摩尔定律（处理器的性能每 18 个月提高一倍）说明了硬件的发展速度，但是考察一下软件开发的速度，可以发现软件开发的速度一直远远地落后于硬件的速度。实际上业内人士一直没有停止过对软件开发思想的探索，只不过快速地对软件技术变革的需求远远高于软件开发技术的进步。

软件工程的目标就是致力于提高软件生产效率和软件质量，如果每个应用软件系统的开发都从头开始，必然存在大量的重复劳动。软件复用是减少重复劳动、提高软件开发效率和质量、实现软件产业工业化生产方式的重要途径，它以已有的工作为基础，充分利用在开发中所积累的知识和经验进行新的开发。而面向对象技术、软件体系结构、软件再工程、领域工程、软件构件技术等相关技术则为软件复用提供了基本的技术支持，并推动其在研究和实践中的迅速发展。

近年来，在中间件技术的基础上，结合软件复用思想和面向对象方法，基于构件的软件开发（component based software development，简称 CBSD）技术受到了高度重视。CBSD 的兴起主要是源于下面 4 个背景：在研究方面，现代软件工程思想，特别是对复用技术的强调；在产业方面，支持用构件来建造 GUI、数据库和应用部件的一些理论上质朴但实际可用的技术的成功；在策略方面，某些主流互操作技术，如 CORBA、COM 和 EJB 的开发者的推动；在软件界，对象技术的广泛使用，提供了建造和使用构件的概念基础和实用工具。

软件构件是软件复用一条切实可行的途径，它改变了软件生产方式，将软件生产划分为构件生产和构件集成两个方面，开发者专心于构件的生产，集成者充分利用构件，专心于应用。但软件构件只有在数量上达到一定的规模才能真正满足软件复用和基于构件的软件开发的需求，因此必须有一个强有力的工具来对这些数量庞大的软件构件进行管理。软件构件库作为一种支持软件复用的基础设施，它提供了对软件构件进行描述、分类、存储、检索和管理等功能，并提供相应的工具支持开发人员在软件开发过程中方便地检索、理解和选取软件构件，以此来提高软件复用程度以及软件开发的质量和效率。

与此同时，在互联网大发展的时代，流媒体技术的产生和发展将给我们的日常生活和工作带来深远的影响，专家预言，流媒体将成为未来因特网上应用的主流，实现沟通和传播的多向性使传播不再受时间和空间的限制。流媒体技术并不

是单一的技术，它融合了多种网络以及音视频技术。在网络中真正要实现流媒体技术，必须完成流媒体的采集、制作、发布、传输、播放等多个环节。这些环节需要解决多项技术问题。例如采集制作过程的主要工作包括：采用高效的压缩算法减小文件的尺寸；向文件中加入流信息。发布传输过程还需要解决：适合流式传输的传输协议，一般采用建立在 UDP 协议之上的 PTR/RTSP 来传输实时的影音数据。而在播放时需要播放器对流媒体的支持，播放器通过流中内建的 MIME（Multipurpose Internet Mail Extensions）来标记繁多的多媒体文件格式，包括各种流媒体文件格式。作为传输和播放的中间过程，还需要流媒体的缓存技术进行强有力的支持。以上所说的几点均为实现流媒体技术的必要条件，它们最终的目的都是为了解决传输带宽、压缩算法以及安全性等问题。

我们看到，流媒体技术就其他一些计算机技术领域而言有一定的独立性，它所涵盖的知识的广度又覆盖着多媒体技术、网络传输技术、协议分析技术、加密与安全技术等众多方面，有着相当的复杂性，已经成为计算机技术中一个特定的领域，并且该领域也已经结合软件复用的思想开始了构件化发展。

1.2 研究目的

北京邮电大学计算机科学与技术学院智能通信软件与多媒体实验室长期从事多媒体与通信软件方面的研究，积累了丰硕的科研成果。本文作者在硕士研究生就读期间有幸参加了视频会议项目和其他一些多媒体或通信软件方面的项目，参与了项目的需求分析、总体设计、编码和测试工作。

在这些项目完成以后，本文作者继续对流媒体应用系统进行分析和研究，结合实验室在该领域的理论研究成果和技术开发成果，进一步完善流媒体应用程序开发的框架模型，致力于探讨这些工作中积累的知识和经验的复用方法。

1.3 论文成果和意义

基于构件的软件复用作为一种提高软件生产率和软件质量的有效途径，是近几年软件工程界研究的重点之一，被认为是继面向对象方法之后的一个新的技术热潮。微软提出的基于 COM 构件技术的 DirectShow 技术便是可复用的流媒体软件框架模型，本文总结抽象了流媒体应用程序设计与开发模型，在微软 DirectShow 流媒体软件框架模型基础之上又提出并实现了流媒体业务集成构件或称流媒体子系统构件的概念，它掌管流媒体功能的内部实现逻辑，向下调用流媒体基础功能构件，向上屏蔽流媒体功能部件的复杂性，为软件系统逻辑提供简单的调用接口，使得应用程序层开发人员专注于业务逻辑而流媒体层开发人员专

注于流媒体功能的内部实现逻辑。

同时，结合软件构件库的理论，设计并实现了一个面向流媒体领域的构件库，将流媒体基础功能构件和业务集成构件组编入库，便于后续的复用。

最后，本文利用所设计实现的构件库，结合实验室项目“基于 SIP 的视频会议系统”，为该项目搭建了流媒体子系统，进一步验证了本文的设计思想。

1.4 论文的总体结构

论文余下的部分共分为五章，第二章着重介绍构件及构件库的基础知识，主要包括构件技术和构件库理论两方面；第三章将流媒体软件开发同构件技术结合起来，进行流媒体领域构件化开发的研究；第四章是本文的重点，讲述了流媒体构件库的设计与实现；第五章讲述了首批入库的构件，并且给出了通过构件库搭建一个流媒体应用系统流媒体子系统的全过程；第六章对本文进行了总结。

第二章 构件及构件库基础知识

2.1 构件

2.1.1 构件的产生背景

软件质量和软件的开发效率一直是开发者关心的问题，用最少的时间开发出高质量的软件，一直是软件开发者追求的目标。随着信息技术在生产、国防等行业应用的深入，相关软件系统复杂性相应提高。在传统的软件开发技术和开发模式下开发这些复杂的软件系统，通常开发周期长，软件质量很难保证。

为了解决“软件危机”问题，早在 20 世纪 60 年代末，人们就提出了软件的工程化，把软件开发过程当成产品生产过程来对待。过去的几十年，软件工程方法和相关技术得到了很好的发展，一系列的优秀的软件技术和软件开发方法不断涌现，影响最深的是面向对象的技术及相关方法。面向对象方法学尽可能模拟人类认识世界解决问题的习惯方式，很好的支持软件复用，提高了软件的可靠性和可维护性，推动着软件技术的发展。但遗憾的是，面向对象也有自己的缺陷，软件复用仅在代码级复用上，真正的代码级软件复用工程实践效果并不好，更严重的问题是软件系统的互操作性差，产生“对象孤岛”现象。

解决面向对象软件设计开发的不足，消除“对象孤岛”，提高软件的互操作性，在面向对象技术的基础上，产生了构件技术。构件技术可以在二进制级上支持软件复用，提高软件质量，同时基于构件技术的软件互操作性好，可以实现软件的“即插即用”，为软件最终的工程化道路开辟了新的捷径。构件技术已成为一个受重视的软件学科分支，正推动软件技术的发展。

2.1.2 构件的定义

20 世纪 60 年代末到 20 世纪 80 年代初，结构化的模块式软件开发思想占主导地位，当时软件组件的含义是指一些定义良好的方法包或功能模块。

20 世纪 80 年代起，面向对象的软件开发思想迅速发展起来，这时软件组件的含义就是类库。类虽然提供了封装性、多态性和继承性，但需要依赖于具体的编程语言，耦合度高，且需要用户对类库的结构和宿主语言有较深入的了解，因此，不能完全达到软件重用目的。

20 世纪 90 年代后，软件组件的内涵进一步加强，聚合性、独立性和重用性

进一步提高。目前，基于对象的组件软件体系结构中的组件是指可方便地插入到语言、工具、操作系统、网络系统中的二进制代码和数据。

随着分布式计算、Internet 等技术以及基于构件的软件开发技术的发展，人们对构件的认识又产生了新的变化。对于软件构件，至今也没有一个严格的定义。从广义上说，构件是可复用的软件组成成份，可被用来构造其他软件，是一种定义良好的独立、可重用的二进制代码，包括功能模块、被封装的对象类、软件框架和软件系统模型等。这里我们采用一个大家普遍接受的定义。

定义 2.1 构件是一个带有规范化接口和显示语境依赖的组装单元，它能被独立发布并且可以被第三方组装[3]。

为了更好地管理和使用软件构件，对其构成成分及特性进行恰如其分的界定是必需的。构件应具有以下属性：(a)有用性：构件必须提供有用的功能；(b)可用性：构件必须易于理解和使用；(c)质量：构件能够保证服务质量；(d)适应性：构件应该易于通过参数化等方式在不同语境中进行配置；(e)可移植性：构件应能在不同硬件运行平台和软件环境中进行工作。

2.1.3 构件的分类

构件按功能层次可分为 3 类：基础层为基本数据类构件和系统支撑构件，中间层为各种通用的中间件，顶层为针对领域的专用构件或子系统构件。构件还可以拥有自己的内部体系结构，这样的构件被称为复合构件。利用复合构件的概念，开发人员可以逐步精化系统的体系结构模型，更好地进行设计与开发。

2.1.4 构件复用技术

依据复用的对象，可以将构件复用分为产品复用和过程服用。产品复用是指复用已有的构件，通过构件集成（组装）得到新系统。过程复用指复用已有的软件开发过程，使用可复用的应用生成器来自动或半自动地生成所需系统。

依据对构件进行复用的方式，可以分为黑盒复用和白盒复用。黑盒复用是指对已有构件不做任何修改，直接进行复用。白盒复用是指已有构件并不能完全符合用户需求，需要根据用户需求进行适应性修改后才可使用。

构件复用包括两个基本过程：可复用构件的开发和基于可复用构件的应用系统的构造。

按照软件工程的原则，对于构件的开发可以分为四个阶段来完成：论域分析、构件设计、构件库的组织与检索和系统集成。其中：如何提取可复用构件以及如何组装成系统并能实现互操作，是构件开发的两个核心问题。

第一、构件开发首先要面对的问题是如何定义一个新的构件，即如何在应用领域的应用模型中找出有共性、可通用的部分做成软件构件。虽然它与软件工程中常规的需求分析有相似之处，但其要求视角更为广泛，对于可复用构件需要考虑：(a)它不仅要服务于当前的应用，而且要从历史项目中发现这些项目之间的共同点和差异点，并考虑同类或相似应用领域的未来软件项目的需求；(b)构件是否依赖于具体的硬件结构；(c)一个不可复用构件是否通过分解产生一组可复用的构件；(d)构件是否通过参数化或少量的修改就能在很多场合下复用。因此，要解决这些问题，光有软件技术是不够的，还要求有应用领域的相关知识，两者结合起来才能够解决应用问题。因此在做构件之前首先要明确它将适用于哪个应用领域，然后再根据这个领域的知识和应用模型抽取出有代表的项目进行分类，把与论域有关的不变部分和可变部分分开，设计出论域软件的框架，抽象出最合理的构件定义及开发项目的分析模型。

第二、为了使所开发的构件能够装配互换形成应用系统，并在新的环境下能表现出更好的健壮性，在设计构件时，必须将构件应用的上下文与其严格分离，并充分利用抽象化、参数化等手段提取公共特征，增强构件对未来不同应用项目的适应能力。为此常采用的抽象方法有：

(a)功能抽象：构件的功能由接口说明确定，而具体的实现细节对构件的使用者隐藏起来。

(b)数据抽象：在功能抽象的基础上进一步隐藏除接口参数外的所有数据。构件的功能、行为由输入参数和构件自身记忆的内部状态决定，构件的内部状态可由构件的内部操作来更新。面向对象程序设计语言中的“类”就是一种典型的基于数据抽象的软件构件。

(c)过程抽象：在数据抽象的基础上，进一步实现同一软件构件上并发执行的多个线程的无关性。软件构件提供端口，以便多个用户同时访问软件构件中的资源时进行同步控制。访问请求首先进入等待队列，软件构件就绪后，从队列中取出请求逐个执行，使用构件的多个线程之间通过全局共享数据或消息传递机制进行信息交换。

第三，对所开发的构件进行组织、分类和存储，对构件的检索、理解和使用有着十分重要的影响。因此，要求可复用构件所形成的构件库要能支持各种可维护操作，便于管理员和用户使用，而且在构件查找时不仅能够支持精确匹配、还能支持模糊匹配，查找到在功能、行为上等价的或相似的构件。同时要对应用论域有较强的描述能力和较好的描述精确度。构件库的组织模式现有主要采用信息科学方法，此外还有人工智能方法和超文本系统。

第四，在所开发的可复用构件组成的可复用构件库中，找到合适构件类，将

其生成实例，用过程控制语言描述出系统中的各子系统也是一个不容忽视的问题，构件的集成目前主要采用：

(a)基于功能的集成技术：它要求软件开发人员必须对目标系统进行自顶向下的功能分解，将系统分解为高内聚、低耦合的功能模块，然后根据各模块的功能需求提取构件，采用子程序调用和参数传递的方式将构件结合起来。

(b)基于数据的集成技术：它要求软件开发人员根据应用问题的核心数据结构设计一个框架，然后根据框架中各节点的需求提取构件，并进行适应性修改，再把构件逐个分配到框架中的适当位置。

最后进行系统集成，配置用户喜爱的操作界面，通过实际运行，不断修改，直到用户完全满意为止。一旦该软件系统形成后，在开发同一种类型的应用软件时需要做的仅仅是第四阶段的工作，它可由用户自己来完成，维护工作也大大减轻。

2.2 常用构件技术

2.2.1 构件模型

为了使构件使用者能够很容易地理解构件的功能及其属性，对构件做一个清晰的描述是非常必要的。一般认为描述构件的最简捷途径是构件模型。比较有代表性的是 Tracz 提出的 3C 模型、REBOOT (Reuse Based on Object Oriented Techniques) 项目中提出的 REBOOT 模型和北京大学提出的青鸟构件模型。这些模型均是学术界提出的指导性模型，抽象层次比较高，用户可以根据不同的问题域对其进行扩展。

3C 模型的命名主要来自该模型描述构件所采用的 3 个 C 特征，即概念 (Concept) 、内容 (Content) 和语境 (Context) 。概念用于描述构件的功能。构件的概念依据它的接口说明以及它所执行操作的语义描述表现出来，使用者可以从概念描述中了解它的功能；内容用来描述构件怎样完成概念所描述的功能，如算法、结构等，它是概念的细化描述；语境或者叫上下文，主要用于描述构件与其他构件的关系，它是构件中最复杂的特征描述。

REBOOT 模型是一个基于刻面的分类模型，它所考虑的刻面包括依赖、抽象、操作及操作对象，是一个被称为是刻面中项的联合。

北京大学青鸟构件模型从三个不同的、相互正交的视角来看待构件，每个具体的构件都是形态、层次和表示构成的三维空间中的一个点。构件形态被分为类、类树、框架、设计模式、体系结构 5 种；构件层次被分为分析件（指系统需求规约和功能规约）、设计件（指系统体系结构和设计方案）、编码件（由具体程序设

计语言编制的源代码构件)、测试件（测试计划和测试案例）4 个层次；构件的表示与层次有关，不同层次的构件具有不同的表示媒介和手段，如图形、复合文档、正文、伪码、编程语言、目标码等。根据上述概念，青鸟构件模型从 9 个方面来描述构件，即概念、操作规约、接口、类型、实现体、构件复合、构件性质、构件注释、构件语境。青鸟构件模型是一个具有面向对象风格的模型。

在产业界，以 CORBA，COM/DCOM/COM+和 JavaBeans/EJB 为代表的基于分布式对象技术的构件实现模型正在向实用化方向快速发展，它们对构件的基本构成及其体系结构的演化产生着十分重要的影响，从而成为实现级的主流构件模型。

2.2.2 OMA/CORBA

对象管理集团 OMG (Object Management Group) 是一个国际性的软件行业协会，主要宗旨是促进面向对象的方法在软件工程中的应用，以及在面向对象的软件工程方法学的基础上，为大规模并行系统（开放式并行计算）软件的开发与应用制定软件体系结构模型和通用接口规范。

OMG 于 1990 年末提出了一个对象管理结构的基准结构 OMA (Object Management Architecture) 。OMA 主要涉及用于面向对象语言、系统、数据库及应用程序框架的统一术语体系，面向对象软件系统的抽象框架和基于面向对象技术的分布式软件系统参考模型等。为了构造上述参考模型，OMA 规定了四个方面的标准：

对象请求代理 ORB (Object Request Broker)，即关键通讯单元，在不同的应用程序对象之间以高度协同的方式发送消息；

对象模型，即独立设计的、可移植的抽象模型，可以与其他符合 OMA 规范的面向对象系统进行通讯；

对象服务，利用 ORB 实现基本对象功能，确定对象的逻辑模型和物理存储方式；

通用组件，包含了适用于很多应用领域的基本功能，这些功能可以通过符合 OMA 规范的类接口获得。

OMG 于 1991 年末提出的 CORBA，是 OMA 参考模型中的 ORB 接口技术规范。这一标准规定了如何定义、创建、调度、引用对象，以及对象之间如何通讯。符合 CORBA 规范的 ORB 是典型的中间件，允许 Client 对象向 Service 对象发出请求。

CORBA 规范的基本组成如图所示，主要包括：

对象请求代理 ORB 是 CORBA 的核心。作为一个软件组件，对象可以通过

ORB 发出请求并接收响应。在 CORBA 中，所有的通讯都通过 ORB 进行，因而无论一个对象是本地的还是远程的，对于与之进行通讯的其他对象来说都是等价的。CORBA 并没有规定如何具体实现一个 ORB，几乎每一个不同的产品都有其具体的 ORB 实现；

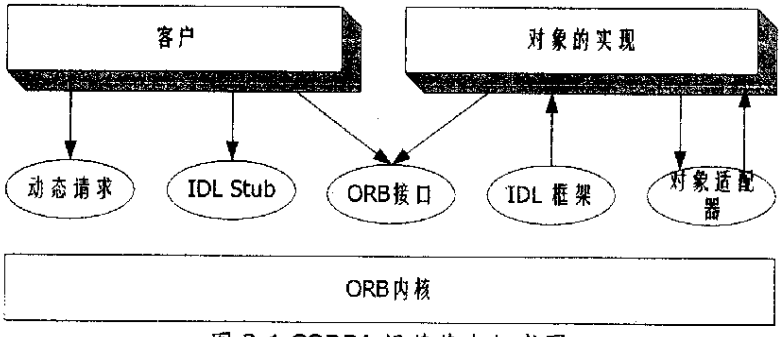

图 2-1 CORBA 规范基本组成图

接口定义语言 IDL (Interface Definition Language)，是用于描述对象接口的一种高级符号语言。IDL 不涉及任何接口的实现细节，所有 COBRA 系统都根据 IDL 用某种特定语言来实现接口。CORBA 为 C、C++、Smalltalk 和 JAVA 制定了规范，ADA95 和 COBOL 的映射规范也正在制定中。IDL 描述通常以接口库 (Interface Repository) 的方式进行存贮；

接口库中包括了所有描述服务对象属性、操作、自定义数据类型及异常处理的 IDL 定义；

基本对象适配器 BOA (Basic Object Adapter)，描述了 ORB 和服务器端应用程序之间的接口。BOA 负责调度服务器端应用程序维护的对象，并与服务对象交换消息；

静态请求接口 SII (Static Invocation Interface)。SII 假设在编译时刻客户对象能够明确了解服务对象的接口，即客户与服务器之间的关系应当是可知的、固定的，服务接口的任何改变都会导致软件系统的重新编译。SII 是由 IDL 描述的；

动态请求接口 DII (Dynamic Invocation Interface)是一组与服务器无关的通用接口。与新服务对象及其行为相关的信息可以通过接口库获得，客户程序根据这些信息可以在运行时刻动态构造请求。DII 使得开发智能化即插即用的软件系统成为可能，但需要涉及大量 CORBA API 调用，增加了软件的复杂度。

总之，CORBA 提供了一个集成框架，应用程序只要给出用接口定义语言 IDL 书写的界面，就可插入框架，与其他对象协同工作，为在分布式环境下实现不同应用程序即插即用的集成提供了有力支持。

2.2.3 JavaBeans/EJB

Java Bean 是基于 Java 环境的，可视的、可操纵的、可重用的组件；JavaBeans 组件模型是 SunSoft 制定的关于 Bean 的软件组件标准，规定设计所有 Bean 所依据的框架，确保 Bean 在具备特定功能的同时，还能被可视化软件构造工具所识别、操纵，并能将这些设计信息保存下来，指导运行时的行为。

作为可视化组件，所有 JavaBean 都具备如下特征：

内省(introspection)机制，能够告诉软件构造工具其所能完成的功能，从而允许软件构造工具在设计时对其加以操纵；

用户定制(customization)机制，允许程序员在软件开发阶段利用软件构造工具改变 Bean 的外观和行为方式；

事件(event)机制，能捕捉事件、引发事件，并将其所能产生和处理的事件告知软件构造工具；

特性(properties)机制，除在软件开发阶段支持用户定制外，还使得软件系统能够在运行时刻对 Bean 进行加工和控制；

保持(persistence)机制，保存程序员开发时利用构造工具对 Bean 所做的修改，并在运行时予以恢复；

设计时刻功能和运行时刻功能分离。

JavaBean 主要用于可视化环境，为软件构造工具所利用，但也能通过程序接口直接操纵，Java 类库中提供了相应的控制类。

EJB 的全称是 Enterprise java bean。是 JAVA 中的商业应用组件技术。EJB 结构中的角色 EJB 组件结构是基于组件的分布式计算结构，是分布式应用系统中的组件。

一个完整的基于 EJB 的分布式计算结构由六个角色组成，这六个角色可以由不同的开发商提供，每个角色所作的工作必须遵循 Sun 公司提供的 EJB 规范，以保证彼此之间的兼容性。这六个角色分别是 EJB 组件开发者(Enterprise Bean Provider)、应用组合者(Application Assembler)、部署者(Deployer)、EJB 服务器提供者(EJB Server Provider)、EJB 容器提供者(EJB Container Provider)、系统管理员(System Administrator)。

EJB 分布式应用程序是基于对象组件模型的，低层的事务服务用了 API 技术。EJB 技术简化了用 JAVA 语言编写的企业应用系统的开发，配置。EJB 技术定义了一组可重用的组件：Enterprise Beans。你可以利用这些组件，象搭积木一样的建立你的分布式应用程序。当你把代码写好之后，这些组件就被组合到特定的文件中去。每个文件有一个或多个 Enterprise Beans，在加上一些配置参数。最

后，这些 Enterprise Beans 被配置到一个装了 EJB 容器的平台上。客户能够通过这些 Beans 的 home 接口，定位到某个 beans，并产生这个 beans 的一个实例。这样，客户就能够调用 Beans 的应用方法和远程接口。

2.2.4 OLE/COM/DCOM/COM+

与 CORBA 一样，Microsoft 的 OLE/COM 是基于分布式对象模型的开放标准，得到多系统软件开发商、独立软件开发(ISV)商和用户的支持。

OLE 实际上是建立在组件对象模型 (COM) 基础上的一组高层次技术。从基本中间件功能视图的角度来说，COM 与 CORBA 非常相似，同样支持对象的定义、创建、调度、引用及对象之间的通讯，同样提供了接口定义语言。

但是两者之间也存在显著的差异。例如，COM 支持由不同程序设计语言或不同编译器实现的对象之间的二进制兼容，以及 OLE 自动操作等。

服务控件管理 SCM (Service Control Manager) 集中负责搜索服务对象和在客户与服务器之间建立信道。当客户向某个特定的对象发出初始化请求时，SCM 判断包含该对象的服务程序是否已经运行，如否，则通过注册表查找服务程序所在的位置。随后，客户端 SCM 建立起与服务器端 SCM 连接的 RPC 信道；服务器端 SCM 也进行同样的处理，并在必要的情况下起动服务程序。最后，信道两端的 SCM 分别将代理程序载入到客户进程和服务器进程中。

OLE/COM 提供了与 CORBA IDL 类似的高级描述语言，用于定义 COM 概念上的接口，即一组逻辑相关的操作或方法。

OLE/COM 结构的另一要素是自动操作，允许客户程序动态构造请求 (包括方法名、相关参数的类型和取值等)，并将请求发送到远端对象。在 CORBA 中，客户程序能够通过接口库获取对象接口的相关信息；而 OLE/COM 则更进一步，任何符合 OLE/COM 规范的对象都能自动提供其所能支持的接口信息。

2.3 构件库理论

2.3.1 构件库系统

构件库系统有效地组织和管理大量的软件构件，并提供相应的工具支持开发者在软件开发过程中方便地检索、理解和选取构件，提高软件开发的效率和质量。构件库系统主要涉及到以下一些内容：

(1) 软件构件的选取；

(2) 软件构件的验证；

(3) 软件构件的分类和表示；

(4) 软件构件的入库存储；

(5) 软件构件库的检索；

(6) 软件构件库的管理和维护。

例如 REBOOT 构件库系统，它是国际上比较著名的构件库系统，包括一个存储可复用构件的构件库和一组产生、认证、插入、提取、评价和适配可复用构件的工具。

在 REBOOT 环境中，采用刻面分类 (Facet) 方法对构件进行分类和表示。刻面分类方法是对构件的分类使用一组{刻面，刻面术语}对，也称为描述符(Descriptors)。它将关键词（术语）置于一定的语境中，并通过从特定的反映构件本质特性的视角（刻面）进行精确的分类。每个刻面具有一组术语（关键词），术语之间具有一般特殊关系而形成结构化的术语空间，允许术语之间有同义词关系。术语仅限在给定的刻面之中取值（受控制的词汇表）。在术语空间中游历可以帮助构件使用者理解相关领域，术语空间可以演变。刻画选择和术语空间的建立依赖于不同构件库的角色和构件组织的需求。

又如北大青鸟工程 JB3 系统作为一个支持复用的软件开发环境，构件的有效管理和检索是关键，其核心是一个构件库系统 JBCL。青鸟构件库系统用于对可复用构件进行描述、管理、存储和检索，以满足基于“构件——构架”复用的软件开发过程的需要。它以青鸟构件模型为基础，建立青鸟构件库数据模型，并与其它 CASE 工具相结合支持构件的生产、描述（使用青鸟构件描述语言）、分类、存储、检索和复合。

2.3.2 构件库管理系统的组成及功能

现有的软构件库管理系统一般都提供对软件构件各类信息的增、删、改、查。而且人们往往把工作的重点放在对于软构件的搜索上。同时，在软构件的应用过程中对实际系统的建模和用已被检索到的软构件自动集成系统对于软件系统的工程化和产业化更具有决定性的意义。如果把这两部分工作包含在软构件库管理系统中不仅能对软构件查询条件的描述提供依据和帮助，也能更好发挥软构件库管理系统的作用。因此构件库的系统设计一般分为 4 部分，软件构件入库系统、建模工具系统、软件构件库维护系统和系统生成器，如图 2-2 所示。

从软构件库管理系统结构图中可以看出，这 4 个子系统的功能以及相互之间的关系：

(1) 软件构件入库系统：从软件构件描述文件中提取出规约信息，要求规约信息全面地反映软构件的各方面信息。然后对各类规约信息进行检查后入库。在

软件构件的分类上，采用刻面分类法和聚类分析相结合的分类方法。

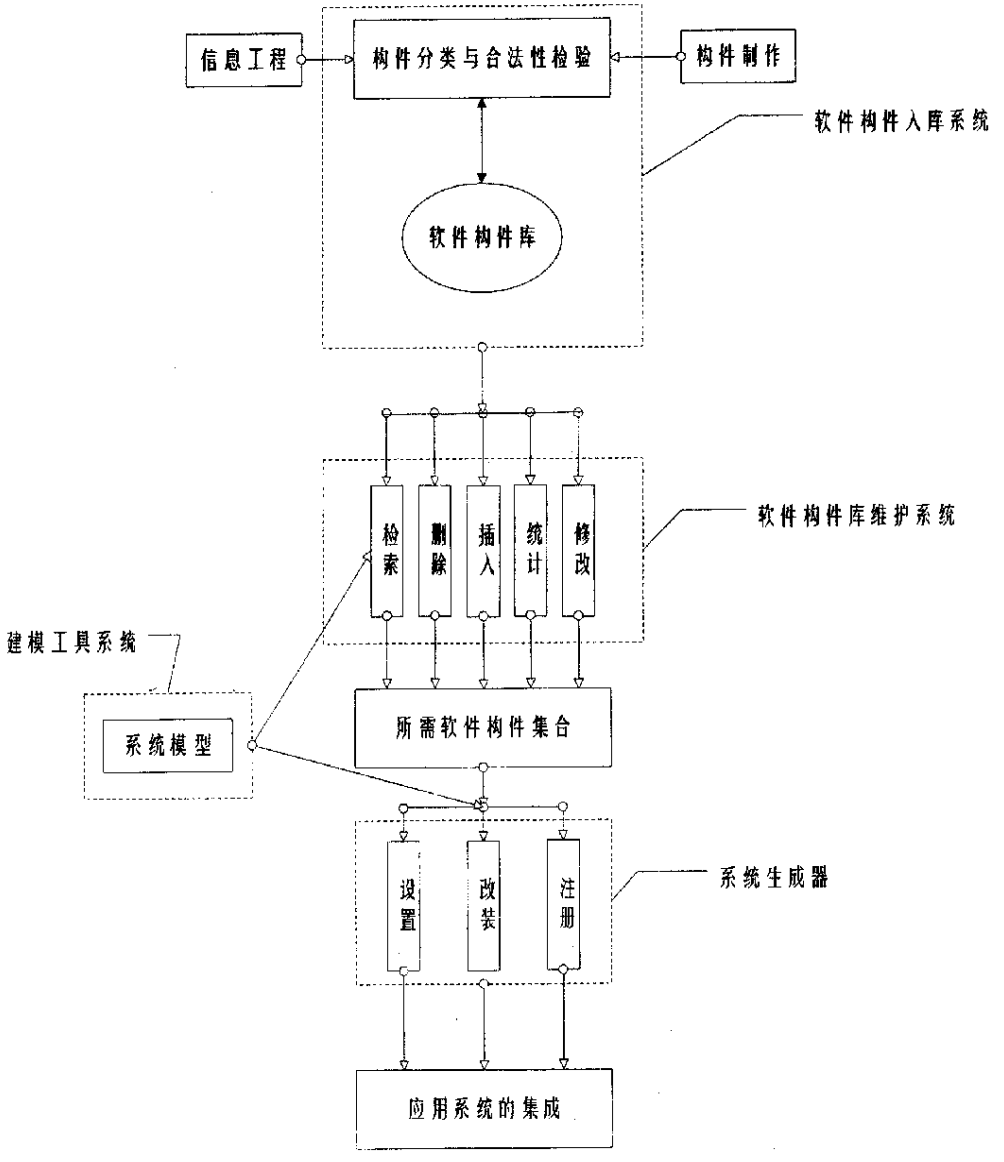

图 2-2 软件构件库管理系统结构

(2) 建模工具系统：根据在工程实践中总结的经验，在借鉴了 IDEF（IDEF 方法是由美国 KBSI 提出一系列建模、分析、仿真方法的统称）系列方法、UML、面向对象的各类设计模式等方法的基础上，形成了一套系统建模方法，并且提供从企业业务逻辑描述到软构件的划分、软构件详细信息描述全过程所需的各个模型的建模工具。利用这套建模方法可以从领域分析的角度分析出每个构件的任务、查询条件以及构件之间的连接关系。

(3) 软件构件库维护系统：包括在软件构件库中检索、修改、删除软件构件，

对软构件的各类信息乃至于编码的修改，对各类软构件的数量、使用频率以及软件成熟度的统计。在软构件的检索方面，不仅包括基于刻面分类的搜索算法，而且还包括基于语义识别的搜索算法，这样可以实现进一步搜索软构件描述文档的功能。

(4) 系统生成器：完成将检索到的软件构件按照建模系统提供的相互关系或软件构件的部署文件中的信息集成为一个新的软件构件或一个完整的系统的功能。可以根据需要自动或半自动地完成软件构件的例化、类型转换等改装工作，以满足系统的需要。改装后的软件构件根据系统模型提供的相互关系既可组装成系统。

2.3.3 构件库中构件的表示

构件在构件库中的表示分为两种：构件功能信息表示和构件描述信息表示。前者关系到如何使用构件和构件的可重用性，后者关系到如何有效地组织、管理构件。一般将构件的功能信息定义为构件体，构件的描述信息定义为描述体。因此，构件组成表示为：

<构件>::=<构件体，描述体>

构件库中的构件是经过分解和优化的一些最基本的功能模块，一般由若干个操作步骤和数据组成。一般将构件体分作三个部分，用三元式表示为：

<构件体>::=<程序项，数据项，构件流程项>

程序项：是构件的可重用部分，它是用某种程序员语言编写的程序源代码和编译后的二进制文件。

数据项：是构件要处理的数据的表示。数据项中每一变量与程序项中每个被处理的数据按出现的先后顺序一一对应。数据项的构成为：

<数据项>::={数据项定义，数据变量名}

构件流程项：是构件体程序项的处理流程。构件的设计者将构件的处理流程用文件的形式记录下来形成构件流程项。

构件描述体是构件的管理信息。构件库能否方便有效地组织、管理和使用在很大程度上依赖于构件描述质量的好坏。构件库中构件描述体用 BNF 形式描述如下：

<构件描述体>::=(构件名<T1>)(构件操作类型<T2>)(功能文字说明<T3>)(作用对象<T4>)(输入类型<T5>)(输出类型<T6>)(构件状态<T7>)(适用语言<T8>)(构造日期<T9>)(最后修改日期<T10>)(子构件集<T11>)(父构件集<T12>)(构造者<T13>)(版本号<T14>)

2.3.4 利用刻面分类方法的形式化定义

刻面分类方法由一组描述构件本质特征的刻面组成，每个刻面从不同角度对构件进行分类，刻面由一组属性构成，属性之间具有一般/特殊关系，形成结构化的属性空间。

分类模式是构件库中构件所拥有的一组共同分类特征的集合。不同构件库适应不同领域特性，在使用刻面分类方法时可能产生不同的刻面分类模式，例如REBOOT 系统定义了 Abstraction、Operations、Operates on、Dependencies 四个刻面[13]，则这四个刻面及相应的术语空间构成一个刻面分类模式。

定义 2.2 刻面是一棵有向树，根节点的值是该刻面的名称，根节点所有子树构成的森林称为术语空间，记为$\{T_i=<V_i, E_i>|$，其中 $V_i=\{v|v$ 是一个节点，v 的值是一个术语$\}$，$E_i=\{<v_1, v_2>|v_1, v_2 \in V_i$，$v_1$ 和 v_2 代表的术语具有一般/特殊关系$\}$，属于空间中任意两个术语都不相同。[21]

术语之间的一般/特殊关系具有如下性质：设某个术语空间 TS 中术语 v_1、v_2 是节点 k_1、k_2 的值，则 TS 中存在一条 k_1 到 k_2 的路径（v_1 和 v_2 存在一般/特殊关系）。

从图 2-3 我们可以看到一个简单的刻面。

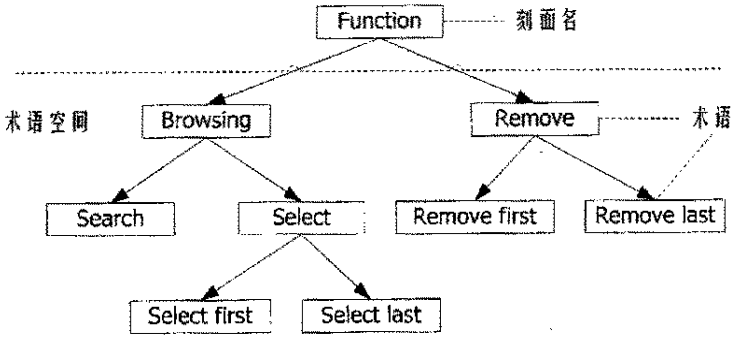

图 2-3 一个简单的刻面

实际应用中即使名称不同的刻面也可能代表同一个分类视角，例如刻面“功能”和“function”、刻面“应用环境”和“使用环境”。若两个刻面 F_1 和 F_2 都是从分类视角 A 对构件进行分类，则称 F_1 和 F_2 是基于分类视角 A 的同视刻面，记为$[A]\ F_1 \approx F_2$。若 F_1 和 F_2 的分类视角完全相同或者一个刻面的分类视角包含另一个，则可以简记为：$F_1 \approx F_2$。

定义 2.3 刻面分类模式 FM 是一组刻面的集合，记为 $FM=\{F|F$ 是一个刻面$\}$，$\forall F_1, F_2\ (F1 \in F_1 \wedge F_2\ (FM \Rightarrow \neg\ F_1 \approx F_2))$。[21]

定义 2.4 刻面分类信息 S 是一个三元组，记为 $S=<m, n, v>$，其中 m 是刻面分类模式的名称，n 为 m 中一个刻面的名称，v 是该刻面的一个术语。[21]

构件一般由分类信息、构件属性、构件关系、构件规约和构件实体组成。为了表达式的简明，仅考虑构件和刻面分类模式，给出构件的定义如下：

定义 2.5 构件 C 是一个二元组，记为 C=<ID, S>，其中 ID 是构件标识，S 是一个刻面分类信息的集合。[21]

同样仅考虑构件和刻面分类模式，给出构件库的定义如下：

定义 2.6 构件库 CL 是一个二元组，CL=<C, M>，其中 C 是构件的集合，M 是一个刻面分类模式，C 和 M 满足以下条件：∀c，s（设构件 c=<ID, S>，分类信息 s=<m, n, v>，则 s（S∧c（C=>m＝刻面 M 的名称），即库中构件只能使用本库的刻面分类模式分类））。

2.3.5 刻面分类的关系模型

当利用刻面化方法对构件进行分类后，结合构件在数据库中规约化的表示形式，就可以得到刻面分类的关系模型。

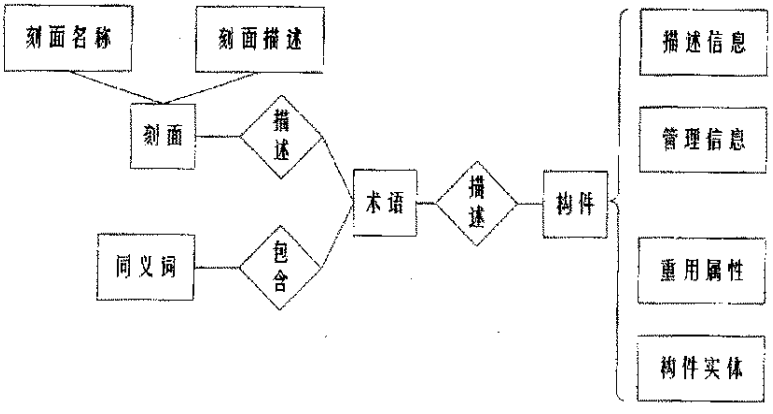

图 2-4 刻面分类的关系模型

第三章 流媒体领域构件化开发研究

3.1 流媒体关键技术

3.1.1 流媒体技术简介

用户需要一种突破带宽限制的新的信息传输方式，使得网络中的多媒体能够实时播放，于是流媒体技术应运而生。流媒体技术的产生使得 Internet 中多媒体的实时播放成为现实，使得在窄带互联网中传播多媒体信息成为可能。

所谓流媒体技术（或称流式媒体技术）就是把连续的影像和声音经压缩处理后放到网络服务器上，让浏览者一边下载一边收看收听，而不需要整个多媒体文件下载完成就可以即时收看的技术。在采用流式方式传输的系统中，声音、影像或动画等多媒体信息由音视频服务器向用户计算机连续、实时传送，用户只需经过几秒或十几秒的启动延时即可进行观看。当多媒体文件在客户机上播放时，文件的剩余部分将在后台从服务器内继续下载。与单纯的下载方式相比，流式传输不仅使启动延时成十倍、百倍地缩短，而且不需要太大的缓存容量。流式传输避免了用户必须等待整个文件全部下载后才能观看的缺点。实际上流媒体技术是网络音视频技术发展到一定阶段的产物，是一种解决多媒体播放时网络带宽问题的一种“软技术”。

流式传输有顺序流式传输 (progressive streaming) 和实时流式传输 (Realtime streaming) 两种方式。顺序流式传输是顺序下载，在下载文件的同时用户可观看在线媒体。由于标准的 HTTP 服务器可发送这种形式的文件，也不需要其他特殊协议，它经常被称作 HTTP 流式传输。顺序流式文件易于管理，但不支持现场直播，严格地说是一种点播技术。

实时流式传输与顺序流式传输不同，他需要专用的流媒体服务器与传输协议。实时流式传输总是实时传送，特别适合现场事件。实时流式传输必须匹配连接带宽。这意味着图像质量会因网络速度降低而变差。实时流式传输允许你对媒体发送进行更多级别的控制，因而系统设置、管理比标准 HTTP 服务器更复杂。

流媒体技术不是单一的技术，它融合了多种网络音视频技术。在网络中要真正实现流媒体技术，必须完成流媒体的制作、发布、传输、播放四个环节。在这四个环节中需要解决多项技术问题。

3.1.2 流媒体文件存储

首先，目前在流媒体领域中，有着众多的文件存储格式，例如 Real Networks 公司推出的 RA (Real Audio)、RM (Real Video)、RP (Real Pix) 和 RT (Real Text) 四类文件。其中 Real Audio 用来传输接近 CD 音质的音频数据；Real Video 用来传输不间断的视频数据。Microsoft 公司推出的 ASF (Advanced Stream Format)。ASF 是一种数据格式，音频、视频、图像以及控制命令脚本等多媒体信息通过这种格式，以网络数据包的形式传输，实现流式多媒体内容发布。另外还有一些其他公司的流媒体文件格式，像 Macromedia 公司的 SWF (Shock Wave Flash)、ViV (Vivo Movie) 以及由 Real Networks 公司与 Macromedia 公司新近联合推出的一种高压缩比动画格式 RF (Real Flash) 等。

3.1.3 流媒体数据压缩与解压技术

多媒体计算机技术是面向以文本、图形、图像、声音视频和动画等多种媒体信息为主的综合处理技术。数字化了的视频和音频信号的数据量是十分惊人的，要减少多媒体数据的时空（传输与存储）量，有两种最简单的方法：其一是减小媒体信息的播放窗口，如把中分辨率 640×480 个象素的窗口改为 100×100 个象素，可使数据量减少为原来的三十分之一；另一种方法是放慢媒体信息的播放速度，如将现在 25 帧/s 或 30 帧/s 的视频播放信息减少到 15 帧/s 或 10 帧/s，也可使数据量减少到原来数据量的三分之一。显然这些方法都是以牺牲媒体信息的播放质量和效果而换得数据所需的时空。

数据压缩的目的是为了在不丢失信息接收效果的前提下，按照一定的数学算法或操作方法对原信号数据进行变换、量化和编码，并通过此过程减少数据量，此即数据压缩或数据编码。与数据压缩过程相反，对已压缩的原信号数据进行量化、变换和还原，称之为数据的解压缩或解码。在多媒体数据压缩和解压缩的方法上，各类不同媒体数据的压缩与解压缩过程和方法没有本质上的区别，我们以视频信号为例。

图 3-1 所示的视频信号的数据压缩是基于 R、G、B 信号，或 Y、U、V 信号，也可以上 H、S、I 信号。原始彩色图像的 R、G、B 信号由发送端进入编码系统，经过坐标空间变换，生成 Y、U、V 信号，再经 A/D 变换（抽样、量化）得到数字信号。对每一个象素的 Y、U、V 分别赋予一组 bit 数，用以量化各个信号的特性参数。一般三个信号各取 8 位，则一个象素共需 24bit。直接对 A/D 变换够的数字信号进行编码是最基本的编码方法，称之为 PCM（脉冲编码调制）。PCM 编码方法可分固定和自适应两种情况，并对每个采样点分配以固定长度的码字。

如果对 A/D 变换后的数字信号再做一次映射变换，可以减少原始数据的相关性和冗余度，便于数据的压缩。采用不同的映射方法，就可得到不同的编码方法（由编码器实现），如预测编码、变换编码、统计编码等。映射编码和编码器一般都是可逆运算，对信号的变换和编码不会产生任何信息损失，而遗憾的是量化器是导致信息损失的根源，因为量化是以视频图像的失真度为代价换取 bit 数的下降，而达到数据压缩的目的。因而，量化的本质就是以质量换空间，得失并存，要在 bit 数下降的限度与图像质量保真度之间折中考虑。解码/解压缩过程正好与编码过程相反。

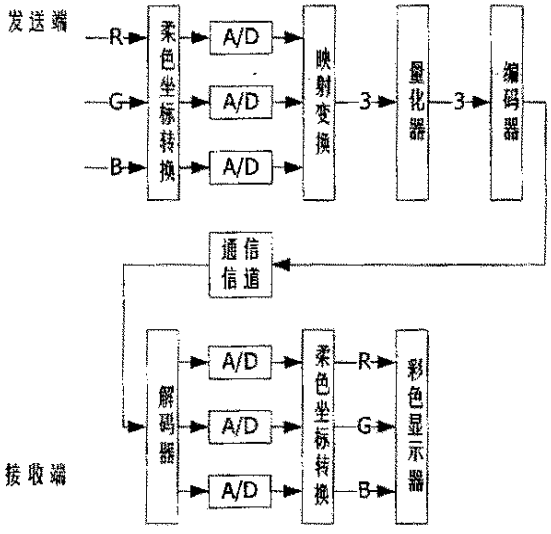

图 3-1 视频信号数据压缩和解压缩的基本过程示意图

图 3-2 给出了多媒体系统常用的数据压缩算法，无失真压缩有：哈夫曼编码、算术编码和行程编码等；有失真压缩有：预测编码、变换编码、子带编码、矢量量化编码、混合编码和小波编码等。图 3-2 仅给出了多媒体系统常用的数据压缩算法，除此以外，还有很多不同类型的数据压缩算法。

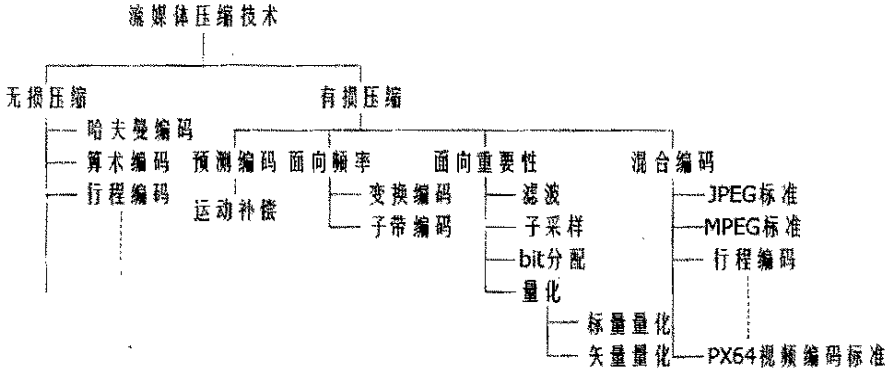

图 3-2 流媒体数据压缩方法

3.1.4 流媒体传输协议

流媒体在网络中传输必然涉及到网络传输协议，这是制约流媒体性能的最终要因素。为了保证对网络拥塞、时延和抖动极其敏感的流媒体业务在面向无连接的 IP 网络中的服务质量，必须采用合适的协议，其中包括 Internet 本身的多媒体传输协议，以及一些实时流式传输协议等。例如，下表列出了一些常用的多媒体传输相关协议。

表 3-1 常用多媒体传输相关协议

协议	描述
ITU-T H.323-2000	用于提供不保证质量的业务本地网上的可视电话系统和终端设备
ITU-T H.225.0-1998	用于不保证质量的业务本地网上的可视电话系统的媒体流的打包与同步
ITU-T H.245-1998	多媒体通信的控制协议
RFC2138-1997	RADIUS 协议
RFC2139-1997	RADIUS 计费协议
ITU-T H.243-1997	使用最高速率为 1920kbit/s 的数字通道三方或多方视/音频终端通信建立的规程
ITU-T H.282-1999	多媒体通信远程设备控制协议
ITU-T H.261-1993	P*64kbit/s 视听服务的视频编码
ITU-T H.263-1998	低速率的视频编码
ITU-T G.711-1988	话音频率的脉冲编码调制
ITU-T G.722-1988	7kHz 的 64kbit/s 音频编码
ITU-T G.728-1992	采用线形预测激励的低时延码在 16 Kbit/s 速率上的语音编码
ITU-T G.723.1-1996	以 5.3Kbit/s 和 6.3Kbit/s 为速率的多媒体通信的双速语音编码器
ITU-T Q.931-1998	ISDN 第三层用户网络基本呼叫的控制协议
ITU-T E.164-1997	用于 ISDN 的编号计划
ITU-T H.235-2000	H 系列多媒体终端的安全和加密

这里讨论一下 RSVP、RTP 和 RTSP，其中着重讨论 RTP 协议。

RSVP（Resource Reserve Protocol）属于 Internet 本身的多媒体传输协议，它预留一部分网络带宽，能在一定程度上为流媒体的传输提供 QoS。该协议的两个重要概念是流与预定。流是从发送者到一个或多个接收者的连续特征，通过 IP 包中“流标记”来认证。发送一个流之前，发送者传输一个路径信息到目的接收方，这个信息包括源 IP 地址、目的 IP 地址和一个流规格。这个流规格是由流的速率和延迟组成的。接收者实现预定后基于接收者的模式能够实现一种分布式解决方案。

RTP 是针对实时多点多媒体会议而设计的实时传输协议，它提供了端到端的实时媒体（交互式音频和视频）传输服务。它是运行在 UDP 上的传输层协议，以便利用 UDP 的多路复用、检查和服务。然而 RTP 也可以和其它适当的下层网络和传输层协议一起使用。RTP 本身不能提供任何保证及时提交的机制，也不提

供其他服务质量保证，而是依靠下层服务完成这些任务。它既不保证提交，也不防止错序提交，还不假设下层基础网络可靠而且按序提交分组。RTP 当中包含的顺序号允许接收方重建发送方分组序列，但是，在视频解码中，顺序号也可能用于确定分组的适当位置，从而不必按序解码分组。

RTP 协议由两个相关的协议组成：

(1) 实时传送协议 RTP 负责传送具有实时属性的数据；

(2) 实时传送控制协议 RTCP(RTP Control Protocol)监控服务质量，传递有关正在进行中的会话的参加方的信息。

RTP 数据包的格式图 3-3 所示：

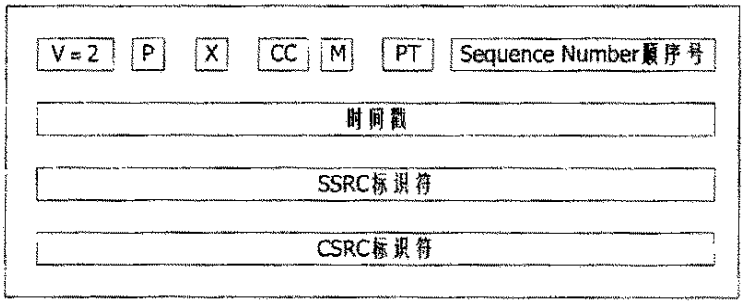

图 3-3 RTP 数据包格式

RTP 数据包由 RTP 头和不定长连续媒体数据组成，其中固定的 RTP 头为 12 字节，媒体数据可以是编码数据。

RTP 头的编码格式：

－V＝2，2 位版本号；

－P，1 位附加标记，加密 RTP 数据包；

－X，X 位长扩展位；

－CC，4 位长 CSRC（贡献源）计数；

－M 标记位；

－PT，负荷类型；

－SN 顺序号，2 字节长的包序号；

－Timestamp，4 字节的时间戳；

－SSRC 标识符，4 字节长的同步源标识符；

－CSRC，贡献源表 0-15 项，每项 4 字节长。

RTCP 的功能是基于向会话的所有参加方周期性地发送控制分组，使用和数据分组相同的分发机制。下层基础协议必须提供数据和控制分组的多路复用。RTCP 提供数据分发质量反馈。反馈的功能通过 RTCP 发送方和接收方报告执行。RTCP 携带用于 RTP 源的持久传送级标识符——规范名 CNAME(Canonical Name)，用于跟踪和关联数据流，可选功能是传送最小会话控制信息。

RTCP 的数据包分如下 5 类：

SR：源报告包，用于发送和接收活动源的统计信息；

RR：接收者报告包，用于接收非活动站的统计信息；

SDES：源描述包，用于报告和站点相关的信息；

BYE：站点离开系统报告包；

APP：特殊应用包；

借助上述控制包，RTCP 可以完成下列控制功能：

(1) QoS 监测和拥塞控制。RTCP 包是多点传送的其中包括监控 QoS 所必须的信息，这些参数包括包丢失率、抖动、延迟、接收到的最大顺序号，因此所有的对话成员都能够大致的了解其它参与者的进展情况；

(2) 媒体间同步。RTCP 发送报告中包含了实际时间和相应的 RTP 时间戳指示器，这两个值允许不同媒体，如音频、视频之间实现“唇同步”；

(3) 识别信息。RTCP 的 SDES 包中为每一个对话成员提供一个全局唯一的标识符称为段名或 CNAME、用户名、位置等；

(4) 会议大小识别和控制信息量调节。由于每个对话成员定期发送 RTCP 包，当对话包含数百个与会者时!必须限制控制流量只占对话带宽的一小部分，通常控制流量限制为对话带宽的 5%；

(5) 接收到 RTCP RR 包后，发送者处理步骤如下：

(i) RTCP 分析。分析统计包丢失率、包延迟抖动，包的往返时间；

(ii) 网络状态评估。可根据 RTCP 分析结果，对发送方的发送速度进行调整；

(iii) 带宽调整。根据网络状态分析调整多媒体应用的带宽，用户可设置可调整带宽的交换范围，如说明最大和最小的带宽。

实时流协议 RTSP（Real-Time Transport Protocol）定义了一对多的应用程序如何有效地通过 IP 网络传送多媒体数据。它在体系结构上位于 RTP 和 RTCP 之上，使用 TCP 或 RTP 完成数据传输。HTTP 与 RTSP 相比，前者的请求由客户机发出，服务器做出响应；使用后者时，客户机和服务器都可以发出请求，即 RTSP 可以是双向的。RTSP 是应用级协议，控制实时数据的发送，它提供了可扩展框架，使实时数据的受控、点播成为可能。该协议的目的在于控制多个数据发送链接，为选择数据发送通道（如 UDP、组播 UDP 与 TCP）提供途径，并为选择基于 RTP 上发送机制提供方法。

3.1.5 流媒体应用程序协议框架

在流媒体领域，有着众多的协议及其指导性开发框架，图 3-4 显示了一个基于 SIP 的流媒体系统所涵盖的协议栈结构：

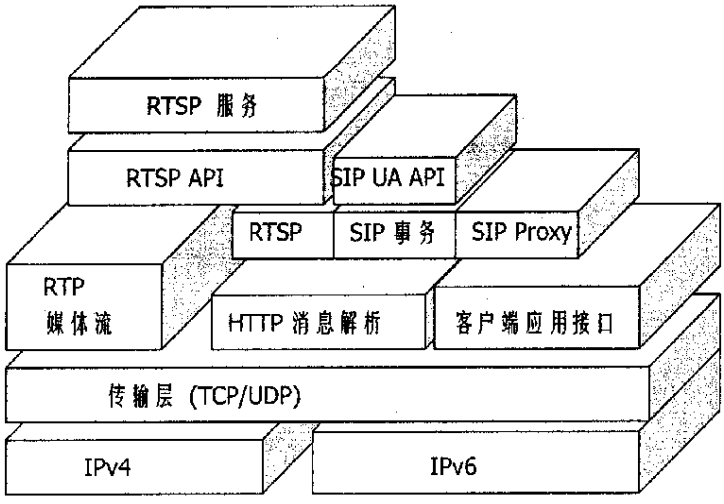

图 3-4 基于 SIP 的流媒体系统协议栈

3.1.6 流媒体应用程序模型

综合研究流媒体基础技术，包括媒体存储技术、数据压缩和解压缩技术、流媒体传输协议和流媒体应用程序协议框架，可以得到流媒体应用程序模型如图 3-5 所示：

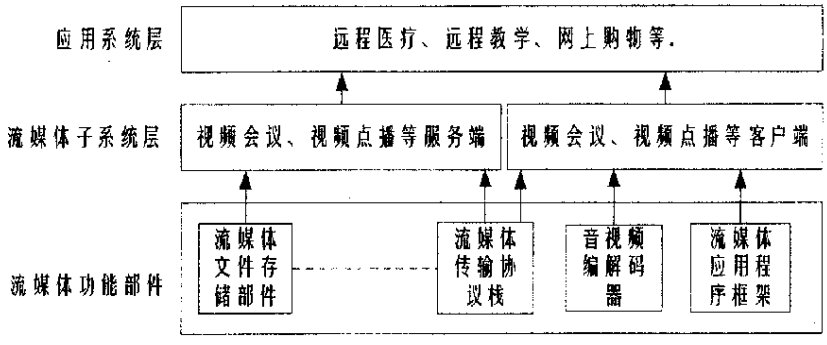

图 3-5 流媒体应用程序模型

3.2 流媒体构件

3.2.1 流媒体构件设计思路

从软件复用的角度考虑，我们提出了流媒体应用构件的设计思路：将构件技术和流媒体处理技术相结合，实现流媒体关键处理技术的构件化，并通过集成组

装构件用这些关键技术构件搭建满足特定需求的多媒体应用开发平台，从而提高系统开发的易用性和编程开发的灵活性，提高多媒体应用软件的开发效率。

首先，通过分析流媒体关键技术，从中抽取不同媒体信息处理的共性，包括：

(1) 输入/输出，含数据采集和数字化；

(2) 数据压缩和还原技术；

(3) 媒体处理；

(4) 存储技术；

(5) 媒体传输技术。

接下来，从所处理的媒体信息类型和角度对流媒体技术进行分析，流媒体技术主要包括以下三方面的媒体处理技术：

(1) 音频数据的获取与处理；

(2) 视频和图像数据的获取与处理；

(3) 流媒体传输技术。

在上述分析结果的基础上，对流媒体应用构件进行划分，分为以下四种类型的构件：

(1) 音频构件。该类构件实现音频数据的采集、数字化、编辑、存储、播放等功能。该构件可以下分为四个子类的构件，即音频系统配置构件，虚拟设备接口读写构件，建立和配置音频文件结构构件及音频编辑构件。

(2) 视频和图像构件。该类构件封装了视频和图像数据的获取、数字化、编辑、存储、显示、播放等功能。该类构件也可分为四个子类的构件，即视频捕捉构件，视频编辑构件，图像处理构件及视频播放构件。

(3) 流媒体传输构件。该构件实现多种媒体信息的复合传输处理功能，包括媒体流的发送、接收、中继等。该构件由基于不同传输协议的传输构件族组成。

(4) 集成构件。该构件是为了对上述三类构件进行理解、浏览、修改和组装而提供的工具，它能够以简单实用的方式组装其他多媒体构件，形成一个简易实用的流媒体子系统。

以上四类构件可以形成这样的层次关系，即音频构件、视频和图像构件、流媒体传输构件均属于基本功能单元构件，而集成构件实现的是子系统级的功能。基础功能单元构件在下层实现较小粒度的独立功能，由集成构件在上层依照一定的流媒体业务配置逻辑根据 XML 配置文件在运行时进行组装。

3.2.2 流媒体构件的实现技术

流媒体应用构件的设计和实现是基于 OLE/COM 的，图 3-6 描述了构件在系统中所处的位置以及各层之间的依赖关系。

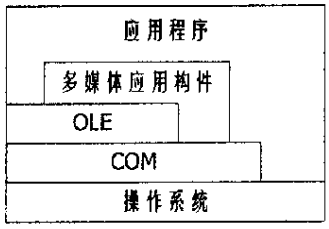

图 3-6 构件在系统中的位置

如图 3-6 所示，流媒体构件在理论上是独立于操作系统的，只要是支持 OLE/COM 技术的平台，构件就能发挥作用。而且对构件的使用并不限制应用程序在系统对操作系统的调用，也不限制应用程序使用其他的COM技术或OLE 技术。

流媒体构件中封装了对象，对象包含一系列可改变其状态的方法，这些方法是通过接口暴露给外界的。当客户程序要使用对象改变其状态，首先应该创建对象实例，然后再通过接口操作对象状态。具体过程如图所示：

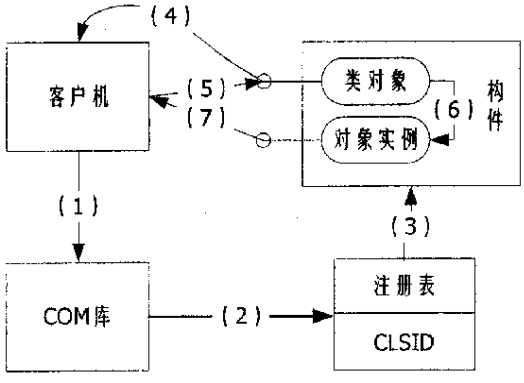

图 3-7 客户和 COM 对象交互图

过程描述如下：

1）客户程序调用 CoGetClassObject()，请求创建对象类工厂；

2）COM 库查找注册表定位类对象所在的构件；

3）COM 库启动构件；

4）对象类工厂创建成功，得到类工厂接口 IClassFactory 指针，通过 COM 库返回给客户程序；

5）客户通过 IClassFactory 创建对象实例；

6）对象实例被创建；

7）对象创建成功，对象接口指针返回给客户。客户可以通过接口操作方法，改变对象状态，实现对对象的操作。

同时，流媒体构件采用 DirectShow 开发框架进行开发，DirectShow 的基本

模块是 Filter 这种软件组件，Filter 通常实现一个对多媒体流的功能操作，比如读文件、从视频捕捉设备上获取视频、解码特殊的流格式、将数据传给图形卡或声卡等。这相当于流媒体基本功能单元构件，由于 Filter 符合 COM 规范，因而一样可以在流媒体构件库系统内使用。

一系列连接的 Filters 被称为 Filters Graph，通过 Filters Graph Manager 控制 Graph 中的数据流，它可以传递事件通知给应用程序，以便程序能对事件做出反应。一个 Filter Graph 本身不是独立的软件组件，但它可以被 COM 对象（Filter Graph Manager）管理，为所有的 Filter 建立参考时钟，在 Filters 和应用程序之间通信。集成构件利用 DirectShow 的框架模型和某些特性，可以更加简单方便地组装基本单元构件，同时向上对应用程序级屏蔽流媒体实现细节，提供功能更加强大的子系统级功能。

3.2.3 流媒体构件设计实现举例

流媒体集成构件 DSMedia 提供复杂的子系统级功能集成。DSMedia 部署在动态链接库 VC4MC.DLL（Video Conference Version 4 Media Client）中。该构件外提供七个接口：一对创建实例（Instance）和销毁实例（Unist）的接口，两个启动媒体发送机接口 AudioStreamer（音频）和 VideoStreamer（视频），两个启动媒体接收机接口 AudioReceiver（音频）和 VideoReceiver（视频），一个关闭流接口 CloseStream。

它采用 DirectShow 框架实现了创建音视频发送和接收所需要的 FilterGraph，并维护一个流的列表。

DSMedia 以单件模式工作，调用者通过 Instance 函数创建实例，并通过所创建的实例调用类所提供的接口，如果调用者创建实例时 DSMedia 对象已经存在，那么会得到已存在的实例。在调用者退出前，应该调用 Unist 函数释放所创建的实例。

(1) 创建实例函数：

static HRESULT CMediaClient::Instance(CMediaClient *pMediaClient)；

返回值 S_OK 表示新创建的实例，S_FALSE 表示得到一个已存在的实例，E_POINTER 表示输入参数 pMediaClient 不为空（输入参数必须已被初始化为空），其他返回值待定。

(2) 销毁实例函数：

static HRESULT CMediaClient::Unist(CMediaClient *pMediaClient)；

实例销毁后实例指针 pMediaClient 被赋空值。由于 CMediaClient 会维护一个实例对象的引用计数，因而当调用 Unist 时如果实例仍然被引用则并不会被真

正销毁掉，只有当引用实例的最后一个指针被释放时才会导致实例被真正销毁。

媒体发送机分为音频发送机 AudioStreamer 和视频发送机 VideoStreamer。

(3) 音频发送器：

UINT CMediaClient::AudioStreamer(char* destinationAddressStr, unsigned short rtpPortNum, unsigned char rtpPayloadFormat);

destinationAddressStr 是以字符串形式的目的 IP 地址，例如"239.255.42.42"；

rtpPortNum 是所使用的 RTP 端口，默认使用 rtpPortNum＋1 为 RTCP 端口；

rtpPayloadFormat 是 RTP 载荷类型；

返回值为所获得的流标记，该值在整个实例的生命期内唯一，调用者需要保存该值，以便提供该值关闭该流。如果调用失败，返回值为 0。

应用程序调用该函数即可启动一个音频捕获并且发送该流。

(4) 视频发送器：

UINT CMediaClient::VideoStreamer(char* destinationAddressStr, unsigned short rtpPortNum, unsigned char rtpPayloadFormat, IVideoWindow *pIVideoWindow);

destinationAddressStr 是以字符串形式的目的 IP 地址，例如"239.255.42.42"；

rtpPortNum 是所使用的 RTP 端口，默认使用 rtpPortNum＋1 为 RTCP 端口；

rtpPayloadFormat 是 RTP 载荷类型；

pIVideoWindow 提供视频窗口的接口，该接口的具体使用方法参见 IRTCClient::get_IVideoWindow 文档，同样，在释放该流之前必须将该视频窗口的所有者赋空值。

返回值为所获得的流标记，该值在整个实例的生命期内唯一，调用者需要保存该值，以便提供该值关闭该流。如果调用失败，返回值为 0。

应用程序调用该函数即可得到一个视频捕获并且发送该流，同时还可以通过 IVideoWindow 接口控制视频的预览。

媒体发送机分为音频接收机 AudioReceiver 和视频接收机 VideoReceiver。

(5) 音频接收器：

UINT CMediaClient::AudioReceiver(char* destinationAddressStr, unsigned short rtpPortNum, unsigned char rtpPayloadFormat);

destinationAddressStr 是以字符串形式的目的 IP 地址，例如"239.255.42.42"；

rtpPortNum 是所使用的 RTP 端口，默认使用 rtpPortNum＋1 为 RTCP 端口；

rtpPayloadFormat 是 RTP 载荷类型；

返回值为所获得的流标记，该值在整个实例的生命期内唯一，调用者需要保存该值，以便提供该值关闭该流。如果调用失败，返回值为 0。

应用程序调用该函数即可接收一个远程的音频流并且自动播放它。

(6) 视频接收器：

UINT CMediaClient::VideoReceiver(char* destinationAddressStr, unsigned short rtpPortNum, unsigned char rtpPayloadFormat, IVideoWindow *pIVideoWindow);

destinationAddressStr 是以字符串形式的目的 IP 地址，例如"239.255.42.42"；

rtpPortNum 是所使用的 RTP 端口，默认使用 rtpPortNum＋1 为 RTCP 端口；

rtpPayloadFormat 是 RTP 载荷类型；

pIVideoWindow 提供视频窗口的接口，该接口的具体使用方法参见 IRTCClient::get_IVideoWindow 文档，同样，在释放该流之前必须将该视频窗口的所有者赋空值。

返回值为所获得的流标记，该值在整个实例的生命期内唯一，调用者需要保存该值，以便提供该值关闭该流。如果调用失败，返回值为 0。

应用程序调用该函数即可接收一个远程的视频流，并且通过 IVideoWindow 接口控制视频的播放。

(7) 关闭流接口

HRESULT CMediaClient::CloseMedia(UINT mediaID);

返回值 S_OK 表示成功关闭所要关闭的流，S_FALSE 表示要关闭的流不存在或已经被关闭，其它返回值待定。

当应用程序需要关闭一条已启动的流时，只需要调用该函数并提供在创建该流时所得到的流标记即可。

DSMedia 需要四条 Graph 链路，如图 3-8 至 3-11 所示，分别为音频发送链路 AudioSreamerGraph、音频接收链路 AudioReceiverGraph、视频发送链路 VideoStreamerGraph、视频接收链路 VideoReceiverGraph。四条链路独立实现，以增强各自的灵活性。

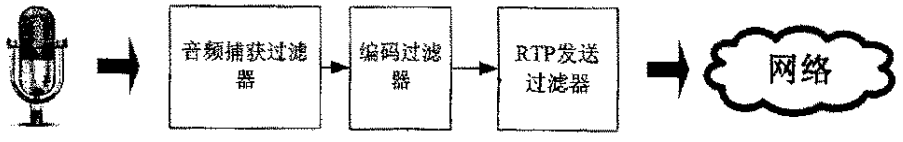

图 3-8 音频发送链路

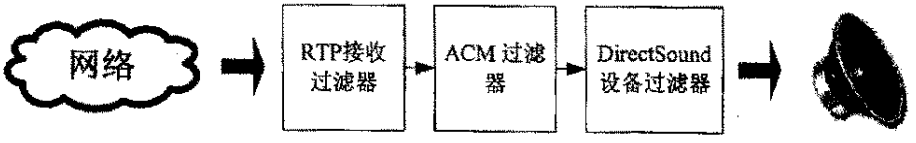

图 3-9 音频接收链路

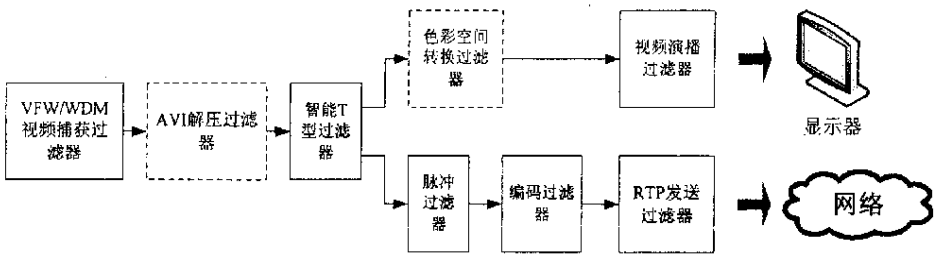

图 3-10 视频发送链路

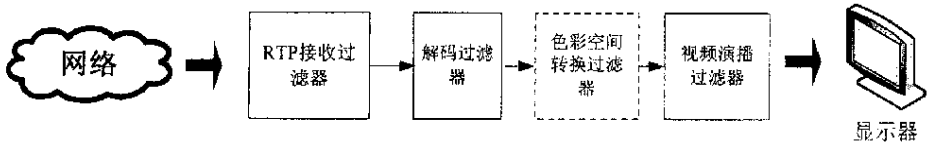

图 3-11 视频接收链路

第四章 流媒体构件库的设计与实现

4.1 流媒体构件库刻面分类设计与实现

4.1.1 流媒体构件库的刻面分类设计

根据第二章对构件理论的探讨，构件库中的构件分类模式是由一组描述软件构件本质属性的刻面所组成，每个刻面从不同的视角对软件构件库中的软件构件进行精确的分类，每个刻面具有一组术语（即关键词），术语之间由同义词关系而形成术语空间。流媒体构件库所采用的便是基于刻面分类的模式，

在流媒体构件库中，一个软件构件可以被每个刻面中的一个或多个术语所刻画，而每个刻面则反映了对流媒体构件库中软件构件的一种划分，因此用户可以直观地从不同的角度指明待检索的软件构件，也有利于用户对软件构件的理解。

流媒体构件库的这种分类模式包括了三个方面的内容：软件构件描述符、同义词词典以及概念距离图。软件构件描述符是一个有序 n 元组，其中的每一个元素对应一个刻面，每个刻面从不同的角度来分类构件库中的软件构件；同义词词典是把表示同一概念的术语放在同一个 n 元组中，用以消除同义词对检索的影响；概念距离图描述了两个术语间的相似程度。

在检索软件构件时，用户根据查询需求从各个刻面中选择术语，组成合法的软件构件描述符。查询时，系统首先将根据同义词词典形成查询，再根据概念距离图计算术语的相似性，形成新的查询，这样，找出的软件构件将根据相似性程度自然排序。

流媒体构件库所采用的刻面分类模式，有两个基本问题需要解决：一是流媒体构件库系统中软件构件刻面的选取；二是流媒体构件库系统中软件构件刻面术语的选取。流媒体软件构件的刻面可以从流媒体软件构件的本质属性中获得，流媒体软件构件有许多属性，但有许多属性使用者在检索软件构件时并不会感兴趣，也不太可能通过它们去检索。根据第三章对流媒体软件构件化开发的研究，经过仔细筛选，流媒体构件名称、流媒体构件媒体性质、所属工程层次、流媒体构件主要功能、流媒体构件采用规格、流媒体构件实现方式，这六个属性作为流媒体构件库的刻面。流媒体构件库的刻面选取好之后，为每个刻面建立相应的术语，刻面的术语是一个逐步改进的过程，它随着流媒体构件库的研制和流媒体构件利用程度的加深而完善。

根据第二章构件库中构件的表示范式，可以做如下描述：<流媒体构件>：=

<流媒体构件名称><流媒体构件媒体性质><所属工程层次><流媒体构件主要功能><流媒体构件采用规格><流媒体构件采用实现方式>。

对流媒体软件构件采用这样的模式分类，大量的软件构件刻面术语将会存在，为了提高软件构件的检索质量和效率，必须进行有效的管理。根据上文刻面分类模式的分析，可以采用层次网络词典的概念对流媒体软件构件刻面术语进行管理。层次网络词典概念的基本结构如图 4-1 所示，该结构体现了术语之间的同义词关系、层次关系以及术语之间距离关系。

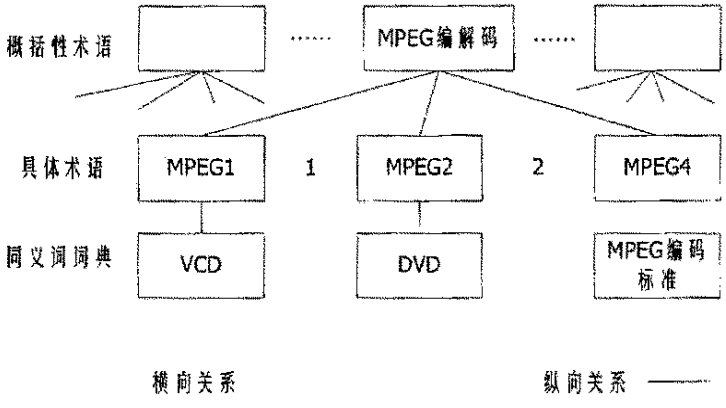

图 4-1 层次网络词典的基本结构

整个图的构造过程是首先用分类树的方法建立概念之间的上下层关系，上层概念是其所有子概念共同术语的概括。子概念则从不同的角度对其父概念加以细化。同一父概念的所有子概念之间形成兄弟关系，在同一层的兄弟结点间，添加横向关系，并标注距离值表示其相似性，距离值越小表明其相似程度越高。

在上图中，最上面一层是概括性术语，例如“MPEG 编解码”，它有一个同义词“MPEG 编码标准”，对于“MPEG 编解码”，它有三个具体术语，分别是“MPEG1”、“MPEG2”、“MPEG4”，其中具体术语“MPEG1”和“MPEG2”分别列出了同义词“VCD”、“DVD”，实际上图中所有的具体术语还有更多的同义词如“MPEG1 编解码”、“MPEG1 编码标准”、“MPEG2 编解码”、“MPEG2 编码标准”、“MPEG4 编解码”、“MPEG4 编码标准”等等，限于篇幅在图中就略去了。

在层次网络词典概念中，定义了两种操作：泛化操作（find-father），由当前概念找到父概念的操作；平级扩展操作（find-brother），由当前概念找到其所有兄弟结点并按距离值排序的操作。

上图也给出了平级扩展操作的距离值，例如“MPEG1”和“MPEG2”距离值为 1，而他们分别和“MPEG4”的距离值为 2，这是因为 MPEG4 是适合网络传输的编解码方式，而 MPEG1 和 MEG2 更多的用在非网络传输的场合。

4.1.2 流媒体构件库刻面分类的数据库表实现

利用 Microsoft SQL Server 2000 建立数据库及其存储过程。在数据库中实现构件信息的数据表，并通过关系型数据库的表达方式来实现上述层次网络词典。

在对流媒体构件、刻面、术语和同义词这些实体进行分析后得到如图 4-2 的实体关系图。

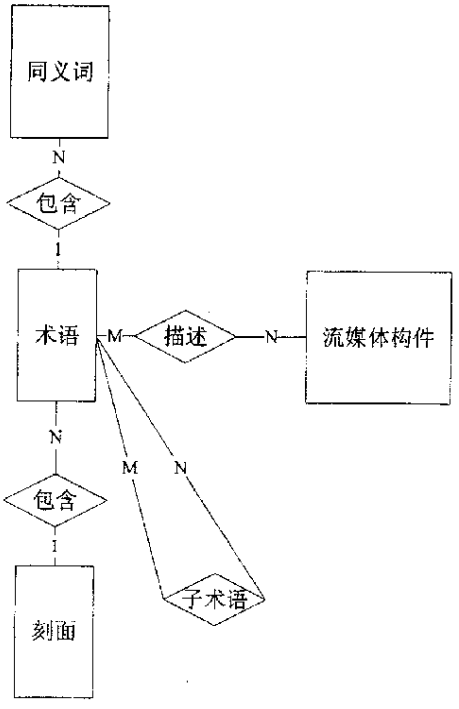

图 4-2 层次网络词典基本结构 ER 图

通过上图，得到这些实体在数据库中的存储表。

表 4-1：刻面描述表

字段名	数据类型	可空标志	说明
Id	Integer	N	刻面 ID
Name	Varchar2(20)	N	刻面名称

表 4-2：术语表

字段名	数据类型	可空标志	说明
Id	Integer	N	术语 ID
Name	Varchar2(20)	N	术语名称
KmId	Integer	N	刻面 ID
GkId	Integer	N	概括性术语 ID
Distance	Integer		距离

术语表中，同时存储了概括性术语和具体术语，通过存储术语的概括性 ID 来实现术语的泛化操作（find-father），如果该条记录本身是一个概括性术语，那么概括性术语 ID 字段即是本身的 ID，此时距离字段值为 0。为了能够在关系数

据库中实现平级扩展操作（find-brother），对距离值进行了如下修饰，规定同一概括性术语下的具体术语之间，任何两个具体术语之间都具有有意义的平级扩展操作，如果存在具体术语之间无法进行有意义的平级扩展，那么就要求将概括性术语进行细分。这样，在任何一个概括性术语下所有的具体术语之间都是可达的。如下图所例：

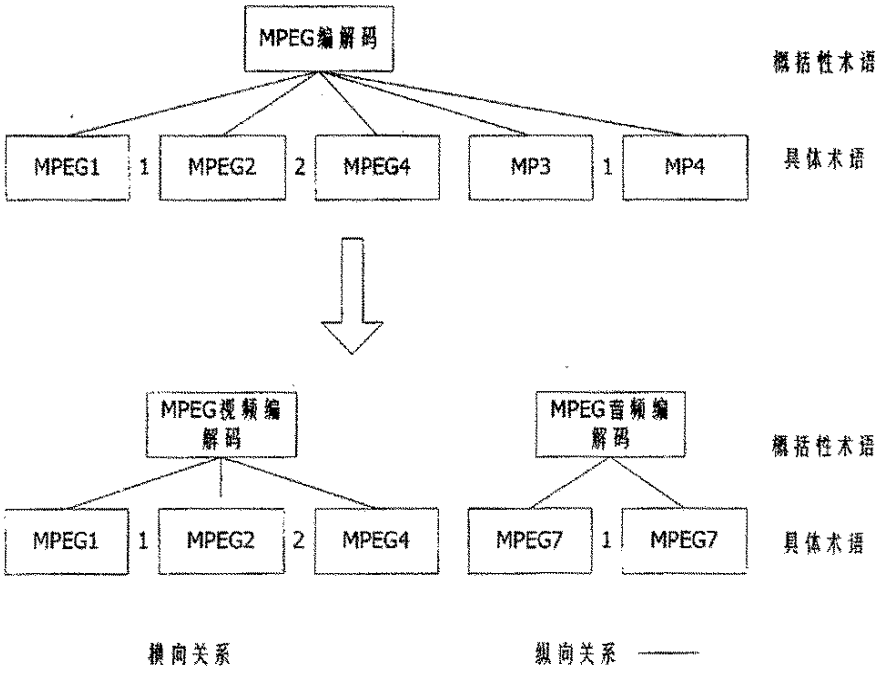

图 4-3 概括性术语细分示意图

任取一个具体术语 S，距离值取值为 1，表示它到概括性术语的距离为 1，那么任何其它具体术语 A 的取值将按如下规则进行： A 的距离值＝层次网络词典图中 A 到 S 的距离值＋1。这样通过简单地加减法并求绝对值就可以得到任意两个同一概括性术语下具体术语之间的距离。

表 4-3: 同义词词典表

字段名	数据类型	可空标志	说明
Id	Integer	N	同义词 ID
Name	Varchar2(20)	N	同义词
SyId	Integer	N	术语 ID

同义词具有相同的术语编号。

表 4-4: 术语描述构件表

字段名	数据类型	可空标志	说明
Cid	Integer	N	构件 ID
Nid	Integer	N	术语 ID

一个流媒体构件由哪些术语描述。

4.2 流媒体构件库构件描述

4.2.1 流媒体构件属性结构的设计

根据流媒体领域构件化开发的研究，可以得到流媒体软件构件的一般构件属性模型：

表 4-5：流媒体软件构件一般构件属性模型

项	分项	说明
流媒体软件构件名称		每个构件都必须有一个确定的名称，该名称必须完整地标识该构件的本质
流媒体软件构件功能	主要功能	该构件在原有或可能的软件系统中所提供的软件功能集
	输入	
	输出	
	媒体性质	该构件所处理媒体类型
基本属性	制作作者	
	制作时间	
	入库时间	
	修改时间	
	修改作者	
	软件构件规格	标准/协议/格式
	实现方式	用来实现软件构件内容的语言形式或技术
	所属工程层次	相对于流媒体系统工程开发的抽象层次
	应用环境	使用该构件时需要提供的软件硬件平台
	应用子领域	原来或者可能应用到的子领域的名称如视频会议、视频点播等等
	修改限制	
	修改影响	
	软件构件尺寸	软件构件的大小，以 KB 为单位度量
	版本号	该构件在一组构件演化系列中相应的版本号
	版权所有	
软件构件接口	输入接口（允许有多个）	名称、参数、功能描述
	输出接口（允许有多个）	名称、参数、功能描述
流媒体软件构件结构	组成单元	名称、功能描述
	组成结构	主要是指该软件构件的体系结构

4.2.2 流媒体构件的 XML 描述

在流媒体构件库中，需要存放各种各样的软件构件，诸如程序数据结构、源代码以及设计文档等，这些数据表现为数据格式多样、数据粒度不可控、数据时间变化、数据关系复杂、数据元素类型多样。对于这种不规则、不符合严格模式

的数据，难以预先定义单一、合适的模式表示，我们称之为半结构化数据，这种数据难以用纯粹的传统关系数据库来进行管理。

XML（Extensible Markup Language，可扩展标记语言）以其良好的可扩展性、自描述性、平台无关性以及数据的内容与其表现相分离等优点，成为企业内部一种数据描述、数据交换与系统集成的新方式它是一种描述数据的元语言，可以由用户自己定义标记、属性以及属性值，并在 DTD（Document Type Declaration）或者 XML Schema 提供的规则下，由 XML 解释器来解析数据。XML 的特长在于描述层次性结构的数据，或赋予原本杂乱的信息一种清晰的结构。

采用 XML 描述流媒体软件构件，需要自己定义软件构件描述信息的内部数据结构 XML Schema。XML Schema 为文档建立了一个模式，规范了文档中使用的标记和文本可能的组合形式，它不仅包含了 DTD 能实现的所有功能，而且它本身就是规范的 XML 文档，特别是支持用户自定义数据类型。

根据上一节分析得到的流媒体构件的属性结构设计，XML Schema 定义如下所示。

```
<?xml version="1.0"encoding="GB2312"?>
<xs:schema id="SMComponent" targetNamespace="http: // lab911.bupt.edu.cn/SMComponent.xsd"element FormDefault="qualified" xmlns="http: // lab911.bupt.edu.cn/SMComponent.xsd" xmlns:mstns=http: // lab911.bupt.edu.cn /SMComponent.xsd xmlns:xs="http: // www.w3.org/2001/XMLSchema">
  <xs:complexType name="流媒体软件构件名称">
  </xs:complexType>
  <xs:complexType name="流媒体构件功能">
    <xs:sequence>
      <xs:element name="主要功能"type="xs:string"/>
      <xs:element name="输入"type ="xs:string"/>
      <xs:element name="输出"type ="xs:string"/>
      <xs:element name="媒体性质"type ="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  ……
  <xs:element name="流媒体构件列表">
    <xs:complexType>
      <xs:sequence>
        <xs:element name ="流媒体构件">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="流媒体构件名称 type ="流媒体构件名称"/>
              <xs:element name="流媒体构件功能"type="流媒体构件功能"/>
              <xs:element name="基本属性"type ="基本属性"/>
              <xs:element name="流媒体构件接口"type="流媒体构件接口"/>
              <xs:element name="流媒体构件结构"type="流媒体构件结构"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
```

```
    </xs:complexType>
  </xs:element>
  </xs :schema>
```

定义了流媒体构件库构件的内部数据结构 XML Schema 之后，就可以对系统的软件构件进行 XML 描述。以“XXXX 软件构件”为例说明其 XML 描述形式如下所示。

```
<?xml version="1.0"encoding="Gb2312"?>
<流媒体构件列表 xmlns="http: // lab911.bupt.edu.cn/SMComponent.xsd">
  <流媒体构件>
    <流媒体构件名称>
      XXXX 流媒体构件
    </流媒体构件名称>
    <流媒体构件功能>
      <主要功能>XXXXXX</主要功能>
      <输入>XXXX<输入/>
      <输出>XXXX<输出/>
      <媒体性质>XXXX<媒体性质/>
    </流媒体构件功能>
    <基本属性>
      <制作作者>XXX<制作作者/>
      <制作时间>XXXX 年 XX 月 XX 日<制作时间/>
      ……
      <版权所有>XXXXXXXXXXXXXXXXXX</版权所有>
    </基本属性>
    <流媒体构件接口>
      <输入接口>
        <名称>XXX XXX INPUT MESSAGE<名称/>
        <参数>XXX XXX<参数/>
        <功能描述>XXXXXX<功能描述/>
      </输入接口>
      <输出接口>
        <名称>XXX XXX OUTPUT MESSAGE<名称/>
        <参数>XXX XXX<参数/>
        <功能描述>XXXXXX<功能描述>
      </输出接口>
    </流媒体构件接口>
    <流媒体构件结构>
      <组成单元>
        <单元名称>XXXX<单元名称/>
        <功能描述>XXXXXX<功能描述/>
      </组成单元>
      ……
      <组成结构>\流媒体构件结构图\XXXX 流媒体构件体系结构.bmp</组成结构>
    </流媒体构件结构>
  </流媒体构件>
</流媒体构件列表>
```

4.3 流媒体构件库的存储策略

一般而言，软件构件库中存储的软件构件包括软件构件的实际拷贝（称为软件构件实体）和软件构件的描述信息，但随着 Web 技术的发展，软件构件库中可以不需要存放软件构件的实际拷贝，采用软件构件描述信息和软件构件实体相分离的存储策略，软件构件描述信息通过数据库管理系统进行存储，每个软件构件的实体就用一个独立文件保存。这种方法的优点是软件构件库系统运行的负荷较低，系统的开放性得到提高。此外，这种策略还有效地保护了软件构件的知识产权。但是，采用这种方法，数据备份工作较为复杂。

针对流媒体构件库中软件规模大、文档多、复杂性高等特点，采用软件构件描述信息和构件实体相分离的存储策略，如图 4-4 所示：

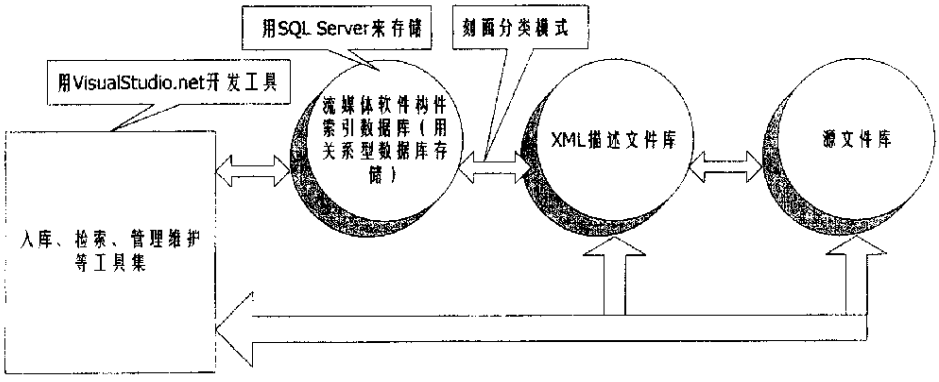

图 4-4 流媒体构件库存储结构

以文件库形式来存储系统的协议、文档、源代码、可执行库文件；然后通过 XML 描述技术来描述这些源文件，使之成为结构清晰、有条理的流媒体软件构件描述文件集，形成 XML 描述文件库；再采用刻面分类模式来对软件构件的 XML 描述文件进行刻面信息提取，使之形成流媒体软件构件索引数据库，该构件索引数据库采用传统的关系数据库来存储。采用这样的存储方式主要是为了发挥数据库提供的复杂、快速的搜索能力，发挥数据库管理数据的能力，发挥 XML 描述文件结构、描述数据的可移植性能力。

4.4 流媒体构件库体系结构

流媒体构件库需要提供两类最基本的工具，一类是流媒体构件库用户使用的工具，其功能应该包含软件构件的描述、分类和检索；另一类是软件构件库管理员使用的工具，其功能应该包含软件构件的管理、用户权限的设置和使用日志等内容。这样根据 Browser/Server 三层体系结构模式（用户界面层、应用逻辑层、

数据库层）得出流媒体软件构件库的体系结构如图 4-5 所示。用户界面层采用 Web 形式，提供一个人机交互的窗口，软件构件入库、软件构件检索和软件构件库的管理功能位于应用逻辑层，用以进行数据的处理，文件库和软件构件索引数据库位于数据层。

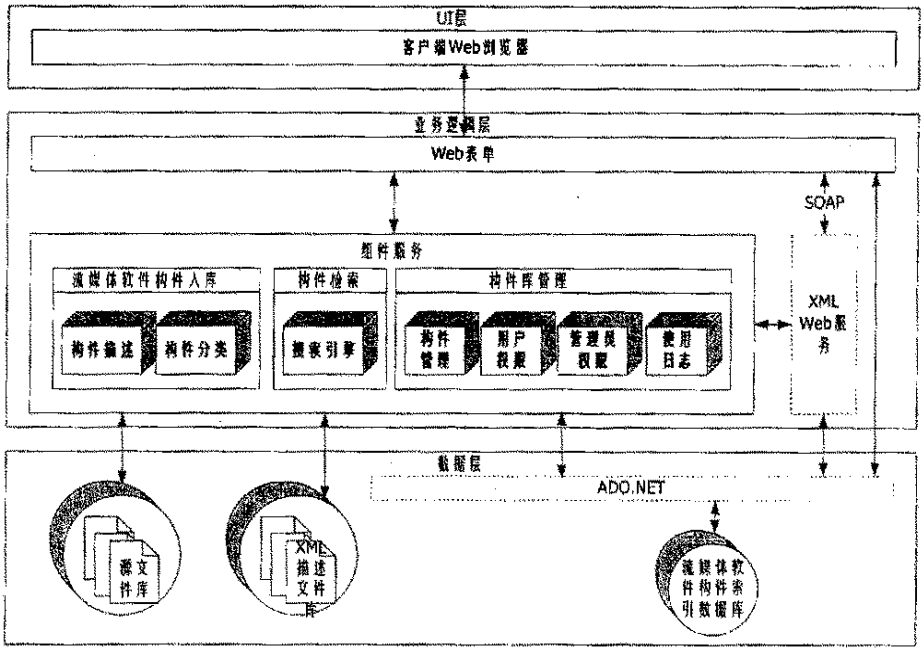

图 4-5 构件库系统体系结构图

4.5 流媒体构件库索引数据库

流媒体构件库系统的构件索引数据库用关系数据库来存储，其 E-R 模型如图 4-6 所示，根据软件构件的一般属性结构，存在着下列实体：刻面、术语、同义词、制造者、使用者、反馈意见、管理员、构件摘要、流媒体构件。

为了给用户在检索流媒体构件时提供方便，可以在检索某个流媒体构件时能够找到其相关的所有流媒体构件，流媒体构件库可以为流媒体构件实体之间建立一些关系，例如，协作关系、相关关系、依赖关系等。

(1) 依赖关系

这是一个一对多的依赖关系。主要是指处于软件生命周期相邻阶段（相邻的抽象层次）的软件构件间的对应关系。例如，一个复杂功能的流媒体子系统构件依赖于一个或多个集成构件的实现，一个集成构件依赖于一个或多个基础功能构件的实现等。

(2) 版本关系

这是一个一对多的关系。主要是指处于同一个软件构件演化系列中的软件构件间的关系。给定一个软件构件，通过版本关系可以找到它的所有版本。

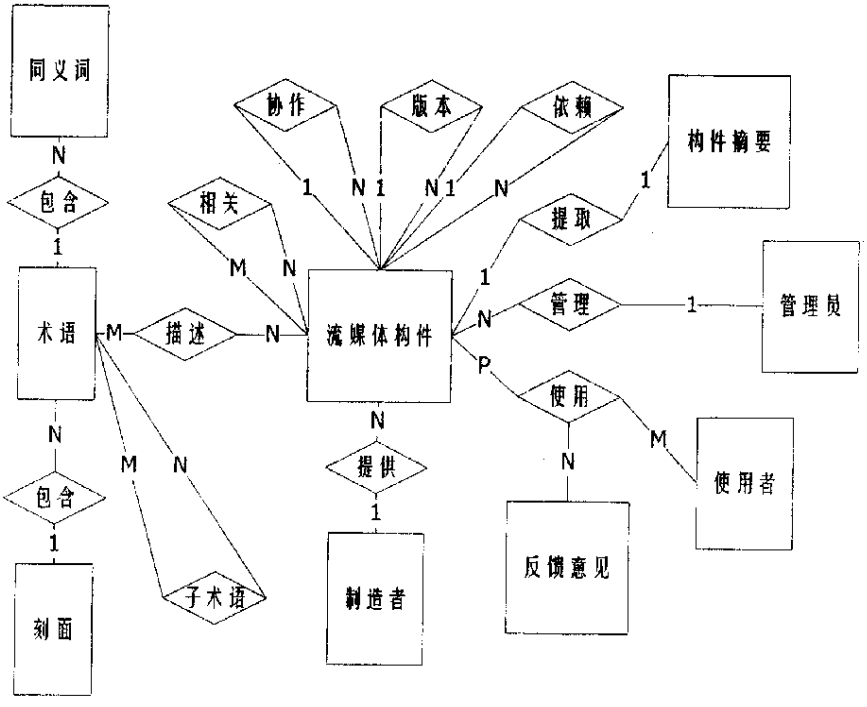

图 4-6 构件索引数据库 E-R 模型

(3) 相关关系

这是一个一对多的关系。与一个软件构件相关的软件构件主要是指与该软件构件具有相同或相似实现技术的软件构件，或具有相关功能的软件构件，或对理解该软件构件有所帮助的软件构件。

(4) 协作关系

这是一个一对多的关系。与一个软件构件具有协作关系的软件构件主要是指与该软件构件相互合作、共同完成一个特定任务的软件构件。

利用 Microsoft SQL Server 2000 建立数据库及其存储过程。在数据库中实现构件信息的数据表如下：

表 4-6: 流媒体构件描述表

字段名	数据类型	可空标志	说明
Id	Integer	N	构件 ID
Abstract	Varchar2(1000)		构件摘要
Version	Varchar2(20)	N	构件版本号
Admin	Integer	N	构件管理员
Producer	Integer	N	构件制造者

表 4-7：流媒体构件依赖关系表

字段名	数据类型	可空标志	说明
Id	Integer	N	构件 ID
Did	Integer	N	被依赖构件 ID

表 4-8：流媒体构件版本关系表

字段名	数据类型	可空标志	说明
Id	Integer	N	构件 ID
Vid	Integer	N	其他版本 ID

表 4-9：流媒体构件协作关系表

字段名	编号	可空标志	说明
Id	Integer	N	构件 ID
Cid	Integer	N	协作构件 ID

表 4-10：流媒体构件相关关系表

字段名	编号	可空标志	说明
Id	Integer	N	构件 ID
Cid	Integer	N	相关构件 ID

表 4-11：流媒体构件库用户表

字段名	编号	可空标志	说明
Id	Integer	N	用户 ID
Pwd	Varchar2(10)		登陆口令
Email	Varchar2(20)		电子邮件帐户
Description	Varchar2(100)		用户描述信息
Degree	Integer	N	用户身份标识

通过用户身份标识来说明用户是管理员、构件制造者、构件使用者或构件库普通用户。不同身份的用户对流媒体构件库具有不同的操作权限，由流媒体构件库管理维护系统来实现。

表 4-12：反馈意见表

字段名	编号	可空标志	说明
Oid	Integer	N	意见 ID
Id	Integer	N	构件 ID
Pid	Integer	N	发表者 ID
Opinion	Varchar2(1000)	N	反馈意见

4.5 流媒体构件库管理维护

4.5.1 流媒体构件库管理维护的内容

流媒体构件库是一个包含人员、工具和过程的组织，提供软件生命周期产品的复用机制，降低流媒体软件的开发成本，提高软件的生产率，并作为可复用流媒体软件构件的开发和基于可复用流媒体软件构件的系统开发这两项工作的基础设施，其管理和维护是相当重要的，包含以下内容：

（1）软件构件库用户管理

包括用户名、登录口令、使用权限、联络地址、本人履历等基本情况的管理，其中使用权限可以分为：一般用户，可以浏览、查询；高级用户（构件制作者和构件使用者），可以浏览、查询、下载；管理员，可以浏览、查询、添加、下载、编辑、删除；

（2）用户登录日志管理

包括登录用户名、登录时间等情况，而且对于超过一定期限的日志，系统能够自动删除；

（3）软件构件使用日志情况

包括谁提取过该软件构件，提取时间，用于什么项目，必要时还应包括用户的联络地址。这样便于发布问题报告，修改声明，需求评价及性能度量；

（4）软件构件的历史

包括用户对某个软件构件的反馈意见、反馈时间，软件构件的修改日志等内容；

（5）问题报告

对每个软件构件记录它较为突出的问题。

4.5.2 流媒体构件库管理维护流程

流媒体构件库系统按照功能可以主要划分成前台用户系统和后台管理系统两大部分。在图 4-7 系统整体流程图中我们可以分别看到这两大部分所包括的功能模块。

4.5.3 流媒体构件库构件入库流程

在众多的管理维护流程中，最重要的一环就是流媒体构件入库管理。图 4-8 给出流媒体构件入库流程图。

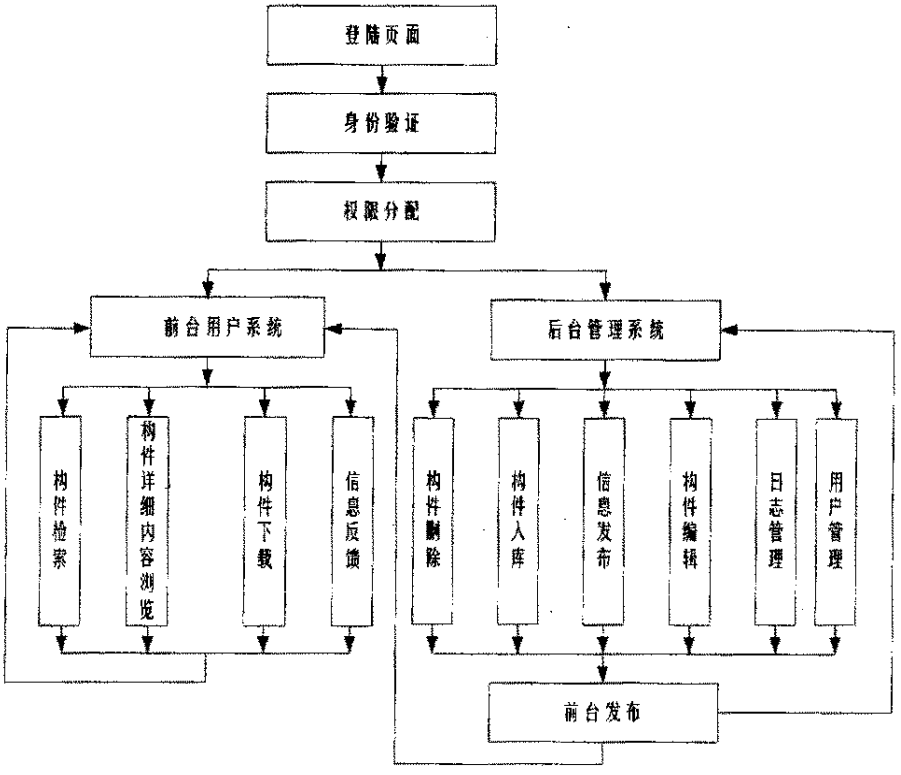

图 4-7 系统整体流程图

如图 4-8 所示，在构件入库的时候需要手工输入构件的详细信息，包括前文所描述的构件各个刻面的分类信息，同时可以输入新的术语和同义词。为了维护构件库中同其他构件的关系，还需要填写相关构件的信息，以便在构件库中建立起同其它构件的关联检索和依赖关系。这些信息，将根据 XML Schema 写入到生成的构件描述 XML 文件中去。同时，构件的文档构成和文件构成也需要在这里进行描述，这些信息主要包括组成构件的源文件都有哪些，二进制文件都有哪些，文档文件包括哪些。这些信息也会写入生成的 XML 文档。上传构件文件及其文档后，构件入库系统将根据构件描述 XML 文件进行校验，主要是 XML 文件中所描述的构件文件和文档文件来检查实际上传的构件文件和文档文件是否正确，有无疏漏和错误文件。当校验无误后，从构件描述 XML 文件中提取出流媒体构件索引数据库所需的信息，设置数据库信息并保存。

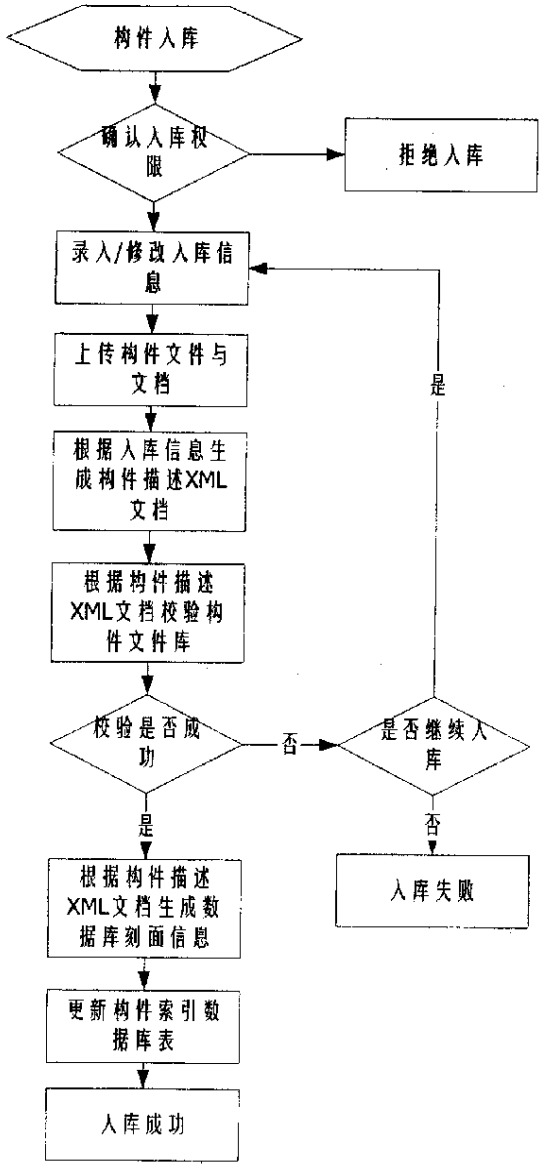

图 4-8 流媒体构件入库流程图

4.6 流媒体构件库的检索策略

要支持基于构件复用的软件开发过程，就必须向应用程序开发者提供流媒体构件库有效的检索提取功能。流媒体软件构件库为用户提供多样化的检索手段和灵活方便的检索界面，以减轻用户的检索负担，尽量缩短检索时间。其检索方式依赖于软件构件的描述方式、分类模式以及软件构件的存储结构，据此有下列三

种软件构件检索方式：

- 基于详细术语关键词的软件构件检索；
- 基于刻面概括性术语的软件构件检索；
- 软件构件 XML 全文检索。

其中关键词检索、刻面检索是针对软件构件索引数据库实施的，其目的是发挥数据库提供的复杂的搜索能力和提供的快速的搜索能力；而全文检索是针对软件构件的XML 描述文件库实施的，只要用户输入的内容在某个软件构件的XML 文件中出现，就一定能够将该软件构件检索出来。

匹配是检索过程的核心，从整体上，流媒体构件库的构件检索遵循这样的检索策略，如图 4-9 所示。

在进行构件检索时，首先通过界面获得用户所选刻面信息，这些刻面信息都是概括性术语，因而是刻面概括性术语查询，在这次查询中，如果用户可以轻易的找到满足需求的构件，那么检索完成。如果此时的检索结果不能满足用户需求，那么检索结果存在这样两种可能，有可能通过概括性术语查询到的构件过多，因而需要一个求精的过程；也有可能概括性术语查询到的构件距离用户需求有一定的差别，这时候需要一个求泛的过程。求精通过进一步查询构件的详细术语关键词实现；而求泛通过距离最小值进行相关术语查询，用户可以选定一个最小距离值，找到同当前最小距离值小于等于用户给定值的具体关键词及其查询结果，如果此时的结果仍不满足用户需求，那么这时候有三种方式可选：前两种方式是对用户当前所选详细关键词进行泛化或者对用户当前所选详细关键词进行同级扩展，这个过程可以反复进行；第三种方式是进行 XML 全文检索。如果泛化和统计扩展后得到的构件仍然不能满足用户需求，那么也可以通过 XML 全文检索再次进行检索。最后，得到一个根据用户需求匹配程度排列出的最终构件列表。用户可以通过这个构件列表选择查看构件的详细信息，首先是可以查看构件 XML 描述文件，其次可以通过文件库查看构件文档。

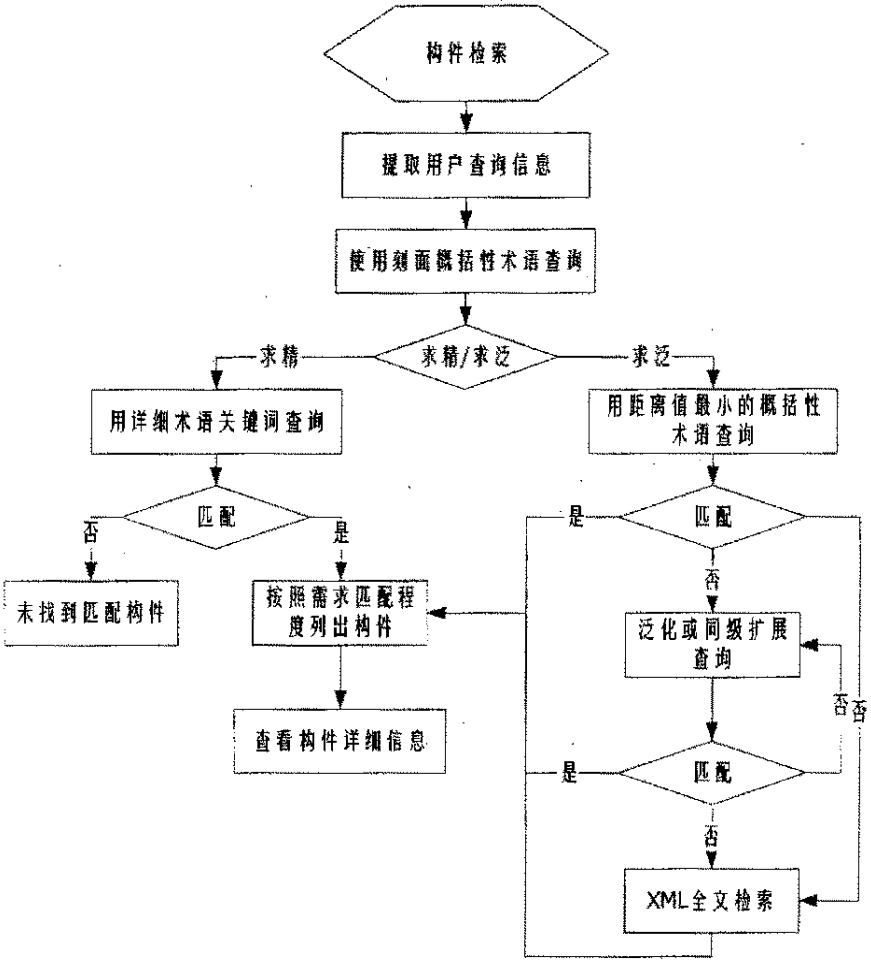

图 4-9 流媒体构件库检索过程流程图

第五章 使用流媒体构件库系统开发举例

5.1 构件入库

登录流媒体构件库，经过身份验证后，进入构件入库页面。下面以 DSMedia 为例，给出构件入库过程。

首先，录入构件基本信息，如图 5-1 所示。

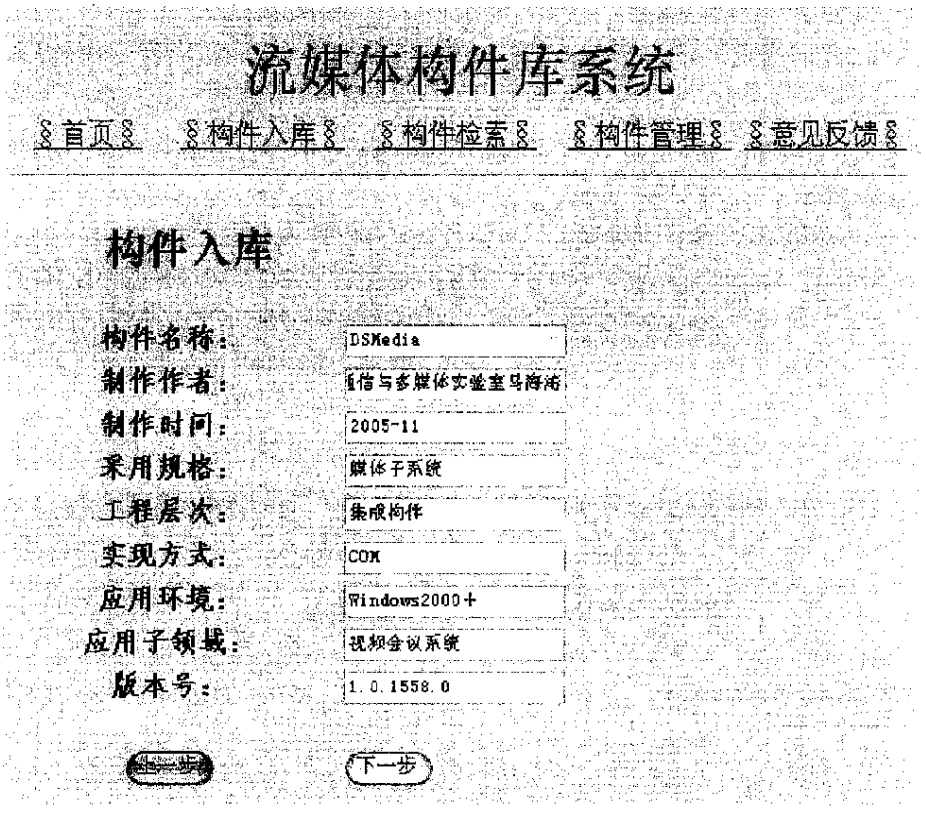

图 5-1 填写构件基本信息

填写信息完毕点击“下一步”，进入构件接口及功能信息描述页面，如图 5-2 所示：

填写完毕，下面顺次进入建立构件相互关系的页面，这些关系分别为相关关系、协作关系和依赖关系，而构件之间的版本关系则由构件库自动维护。图 5-3 示出了相关构件描述页面。

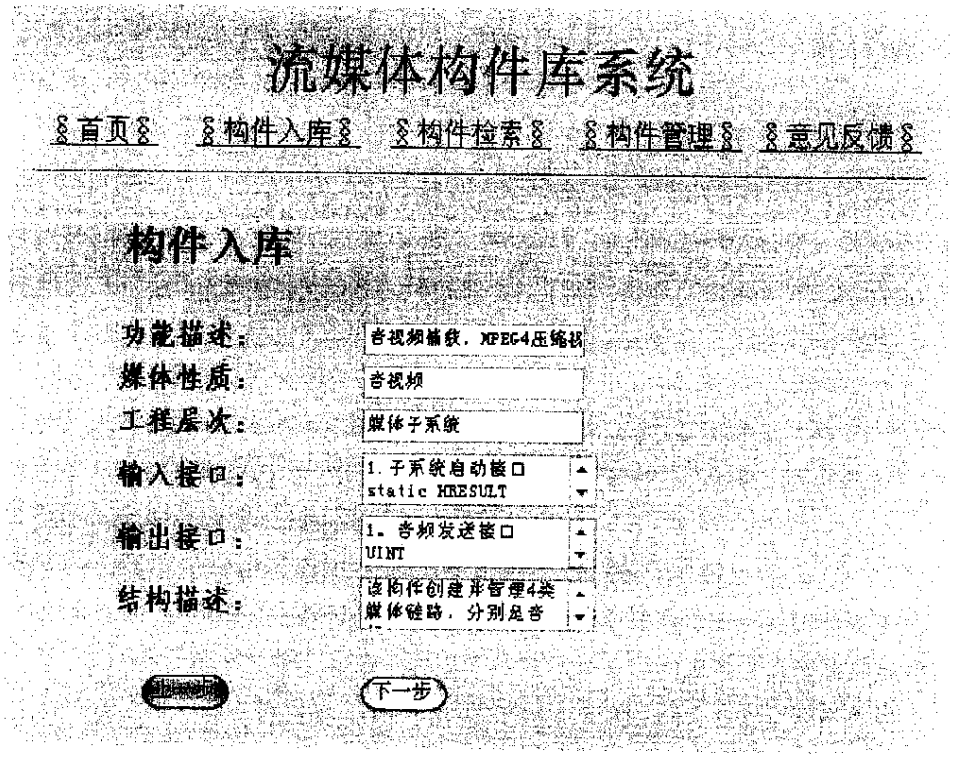

图 5-2 填写构件接口及功能信息

图 5-3 填写相关构件信息

在“相关构件”的下拉列表里选择相关的流媒体构件，点击“添加”按钮可以添加多个相关构件。

构件入库的最后一步是上传相关资源，如图 5-4 所示。

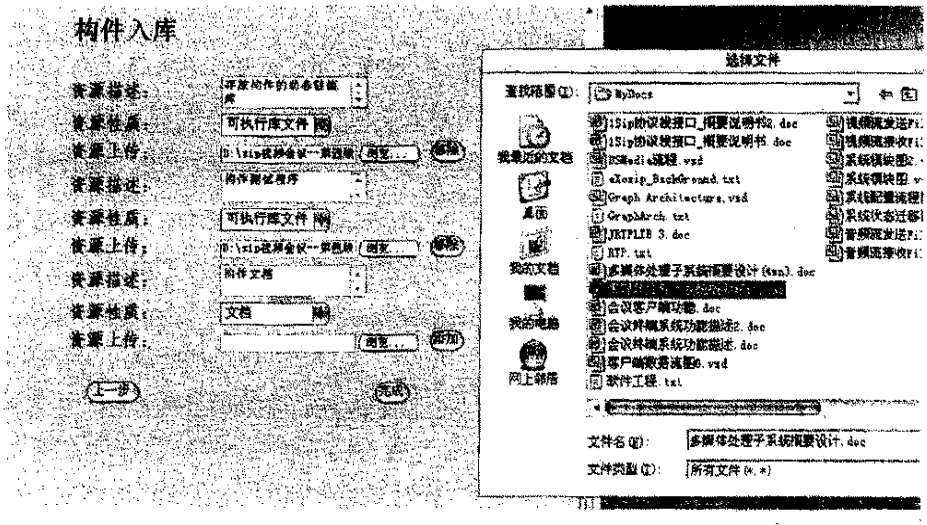

图 5-5 上传相关资源

上传构件文件及其文档等等资源时需要填写资源描述、资源性质，点击“浏览”按钮可以定位上传的文件。最后点击“完成”按钮，即可完成构件入库。

流媒体构件库建立以后，首批入库的构件包括第三方发布的构件和实验室多年来自行研发的流媒体构件，共 24 个。

自行研发的构件如表 5-1 所列：

表 5-1：流媒体构件库自行研发构件列表

构件名称	媒体性质	工程层次	主要功能	采用规格	实现方式
NVDecoder	视频	基础功能构件	解码	MPEG-4	DirectShow Filter
NVEncoder	视频	基础功能构件	编码	MPEG-4	DirectShow Filter
Pulser	视频	基础功能构件	视频流降速器	RAW	DirectShow Filter
StreamSender	音视频	基础功能构件	网络发送	二进制数据	COM
StrmReceiver	音视频	基础功能构件	网络接收	二进制数据	COM
VC4RTPReceiverFilter	视频	基础功能构件	网络接收	RTP	DirectShow Filter
VC4RTPSenderFilter	视频	基础功能构件	网络发送	RTP	DirectShow Filter
DSMedia	音视频	集成构件	多路音视频收发控制器	媒体子系统	COM
MediaRelay	音视频	集成构件	多路音视频中继控制器	媒体子系统	COM

第三方开发的构件如表 5-2 所列：

表 5-2：流媒体构件库第三方开发构件列表

构件名称	媒体性质	工程层次	主要功能	采用规格	实现方式
GSM 6.10	音频	基础功能构件	编解码	GSM	DirectShow Filter
Default DirectSound Device	音频	基础功能构件	Render	DirectSound	DirectShow Filter
PCM	音频	基础功能构件	编解码	PCM	DirectShow Filter
DivX Decoder Filter	视频	基础功能构件	解码	MPEG-4	DirectShow Filter
Microsoft MPEG-4 Video Decompressor	视频	基础功能构件	解码	MPEG-4	DirectShow Filter
Video Renderer	视频	基础功能构件	Render	DirectDraw	DirectShow Filter
XviD MPEG-4 Video Decoder	视频	基础功能构件	解码	MPEG-4	DirectShow Filter
Microsoft H.261 Video Codec	视频	基础功能构件	编解码	H.261	DirectShow Filter
Microstft H.263 Video Codec	视频	基础功能构件	编解码	H.263	DirectShow Filter
XviD MPEG-4 Codec	视频	基础功能构件	编解码	MPEG-4	DirectShow Filter
SmartTee	音视频	基础功能构件	音视频分流器	二进制数据	DirectShow Filter
VFW Capture Filter	视频	基础功能构件	视频捕获	VFW	DirectShow Filter
WDM Video Capture Filter	视频	基础功能部件	视频捕获	WDM	DirectShow Filter
AVI Decompressor Filter	视频	基础功能部件	文件解析	AVI	DirectShow Filter
Color Space Converter Filter	视频	基础功能部件	色彩空间转换	RAW	DirectShow Filter

5.2 基于 SIP 的视频会议系统

基于 SIP 的视频会议系统由 3 个功能实体组成：SIP 服务器（包括注册服务器、代理服务器）、会议服务器、会议客户端，如图 5-6 所示。

整个系统是一个松散的结构。一个会议存在于一个会话服务器中，各个会话服务器相互之间是独立的，不考虑负载平衡问题。系统中的三个功能实体之间也是松耦合的关系。

SIP 服务器是处理 SIP 信令的单元，用来处理用户的定位、呼叫信令的转发等等。它是一个公用的、开放的实体。因为在一个复杂的系统中，SIP 服务器不仅可以应用于视频会议，同时也可以处理系统中的 VoIP、IM、VoiceMail 等服务的 SIP 信令。

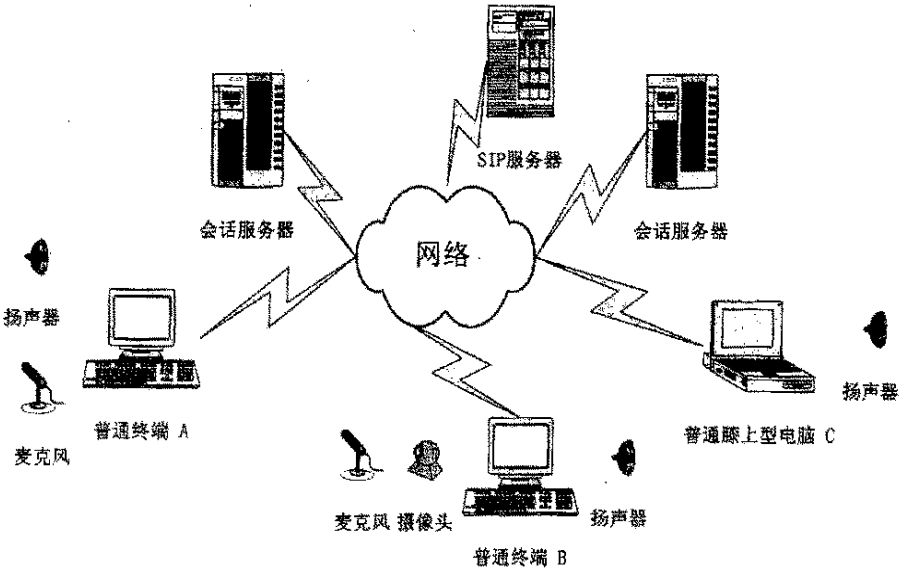

图 5-6 视频会议实体结构

会议客户端是提供给终端用户用来登录系统、与他人进行交流的软终端，也可以称为用户代理。普通的 PC 机一般都可以参加视频会议，在显示器上看到他人的影像、通过扬声器听到他人的发言。当然，客户端需要配备麦克风和摄像头来进行音视频捕捉，才能让对方听到和看到自己。视频会议客户端共分为用户消息界面、媒体界面、会议控制逻辑单元、协议栈和媒体子系统极大功能模块，如图 5-7 所示。

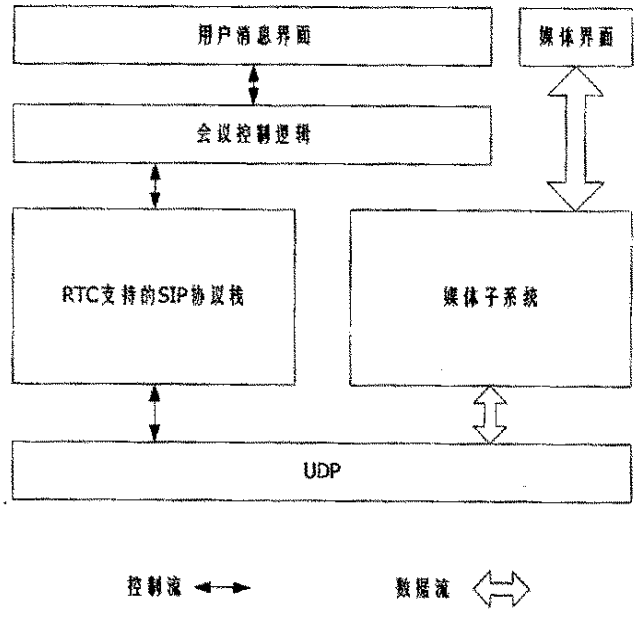

图 5-7 视频会议系统客户端体系结构

而客户端的状态维护如图 5-8 所示。

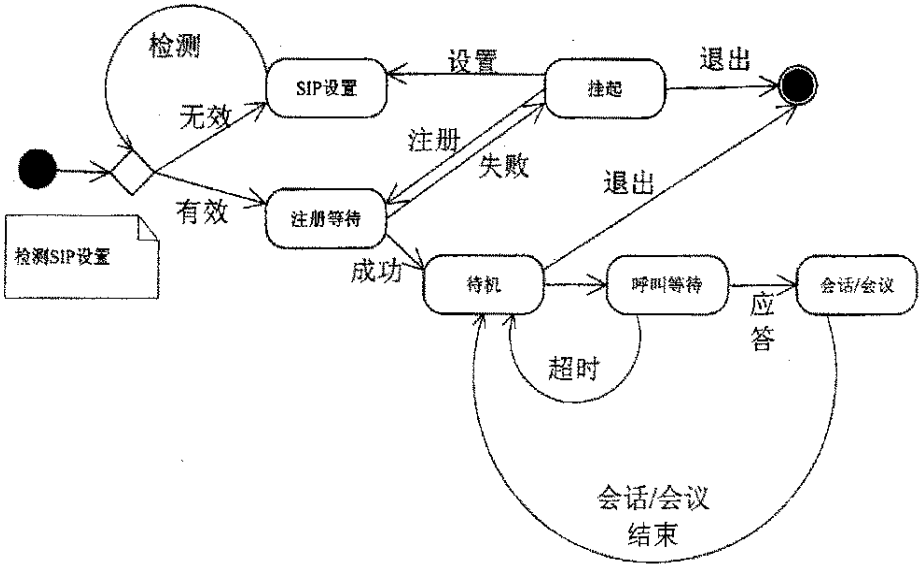

图 5-8 视频会议系统客户端状态机

会话服务器是一个传统意义上视频会议中的 MCU（多点处理单元），所以也可以称为会议服务器。它的主要目的是集中处理会议中多点之间的信令交互和媒体数据转发，因此可以从逻辑上被分为 MC（媒体控制）和 MP（媒体处理）两部分。视频会议服务器端体系结构如图 5-9 所示。

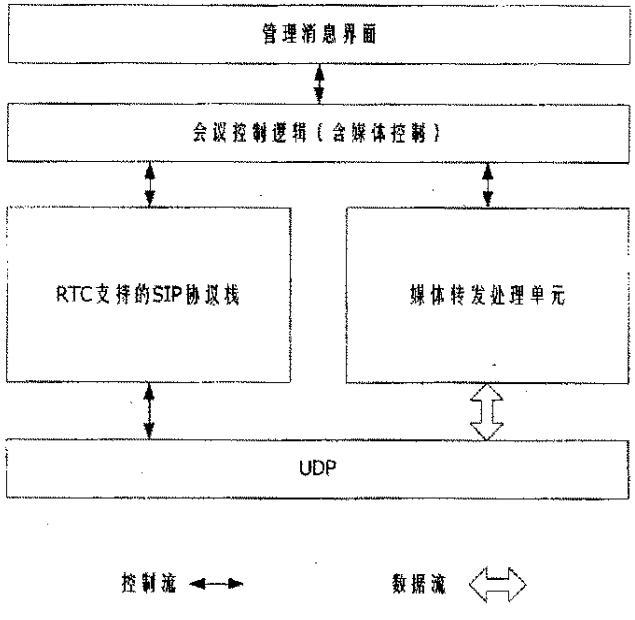

图 5-9 视频会议系统服务器端体系结构

5.3 流媒体构件库在视频会议系统中的应用

在视频会议系统的开发中，客户端和服务器分别都需要各自的流媒体处理子系统级模块。根据这两方面的需求，在流媒体构件库中查找可用构件。首先，登录流媒体构件库系统为服务器端查找满足需求的构件，如图 5-10 所示，输入相关刻面信息。

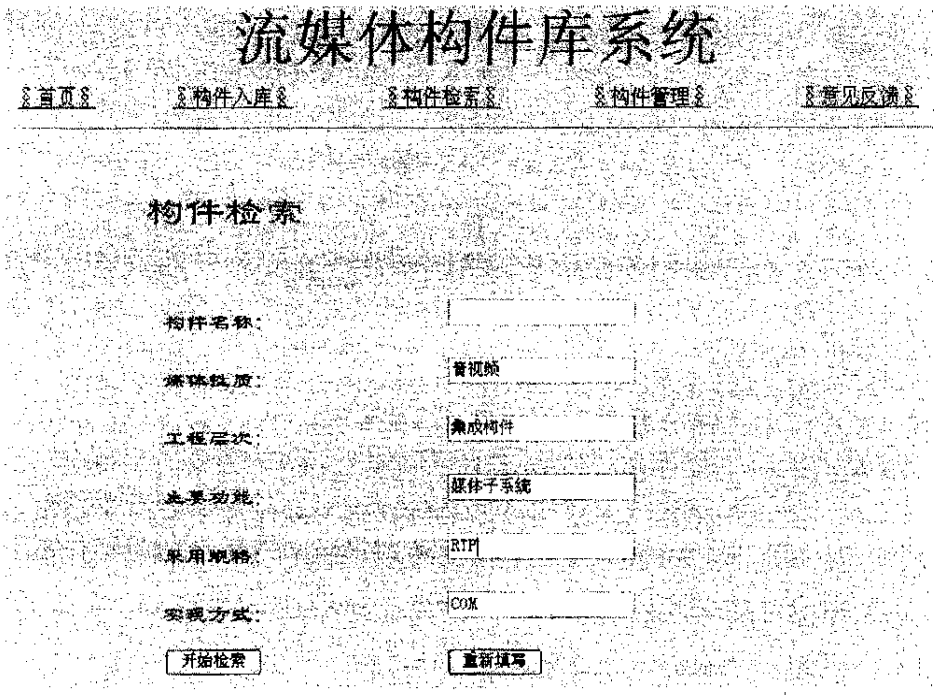

图 5-10 构件检索页面

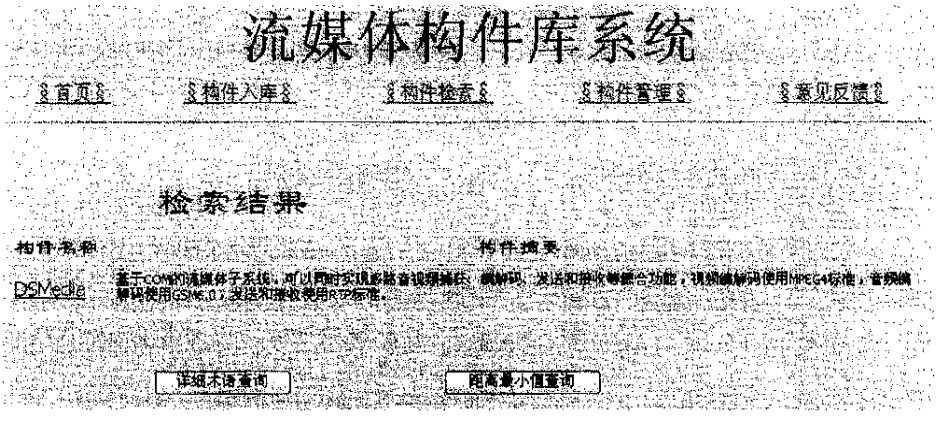

图 5-11 构件检索结果页面

查到构件如图 5-11，点击所列出的构件名可以打开保存构件描述信息的 XML

文件中存储的详细信息，并可以根据详细信息中的地址进入文件库查看构件文档。

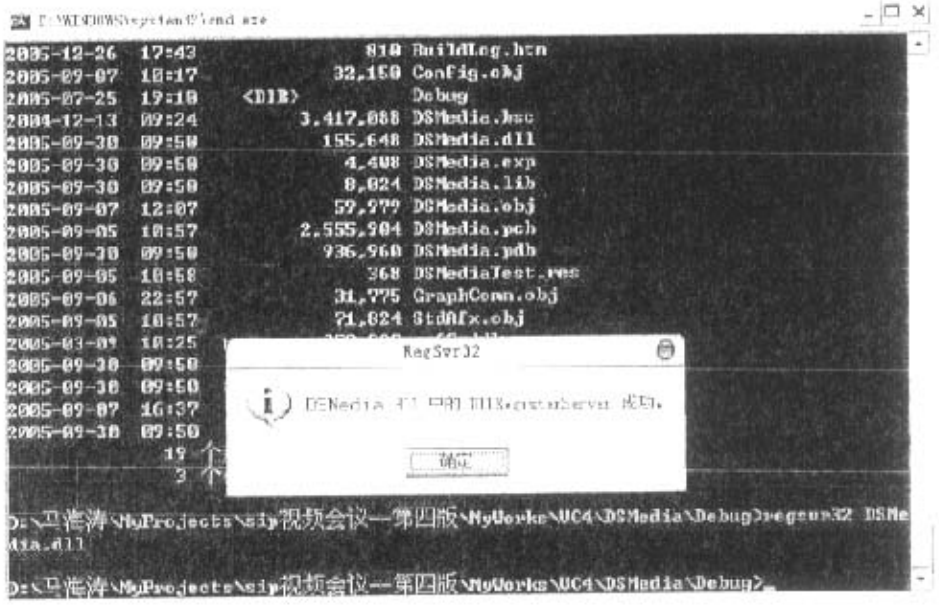

图 5-12 构件注册

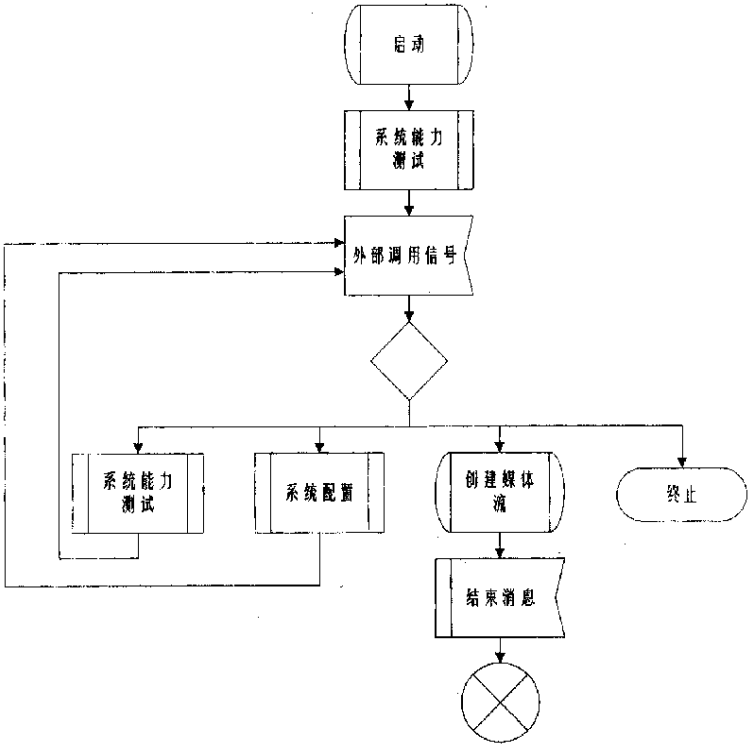

图 5-13 DSMedia 工作流程

现在已经为客户端查找到了满足需求的构件，检索到的构件为 DSMedia，查看详细信息可知，DSMedia 的使用依赖于以下构件：GSM 6.10、Default DirectSound Device、Video Renderer、Color Space Converter Filter、NVDecoder、NVEncoder、Pulser、VC4RTPReceiverFilter 和 VC4RTPSenderFilter。

通过下载这些构件，在操作系统中注册后就可以使用了。图 5-12 以 DSMedia 为例给出注册情况。

DSMedia 工作流程如图 5-13 所示。

其中系统配置过程如图 5-14 所示。

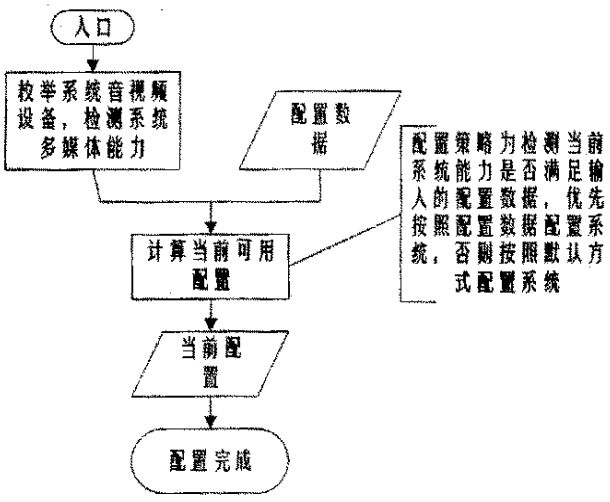

图 5-14 系统配置过程

DSMedia 实现的客户端数据流如图 5-15 所示。

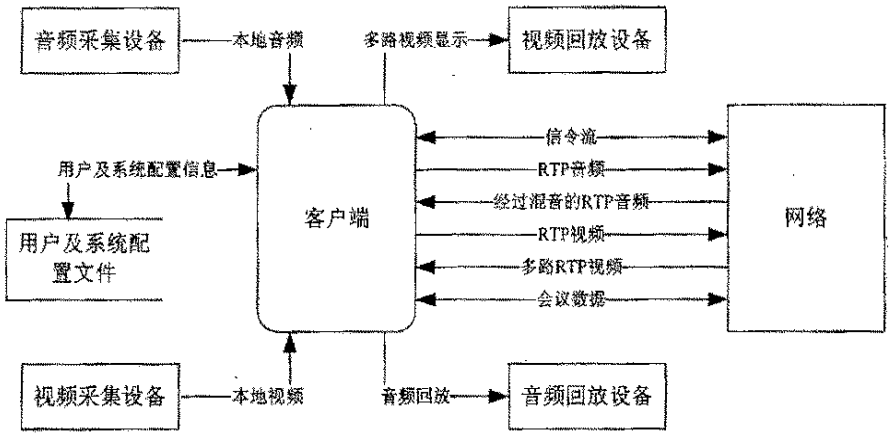

图 5-15 DSMedia 数据流图

最后，利用流媒体构件库及其所含构件开发完成基于 RTC 改进的 SIP 视频会议系统如图 5-16 所示。在图中，可以看到该客户端的一路本地视频（见左下

视频窗口），并且该路视频被放大显示（见中央视频窗口），右侧两个视频窗口是接收自远端的视频窗口，双击小窗口，可以替换中央放大的视频窗口。左侧栏是音视频功能选项，可以看到与会成员列表并且选择是否接收该成员的音视频。在下方中间窗口，是会议聊天室。

图 5-16 视频会议系统运行界面

视频会议系统是一种现代化的办公系统，它可以把不同地点任一会场实时的现场场景和语音互连起来，同时向与会者提供分享听觉和视觉的空间，使各方与会用户有“面对面”交谈的感觉。随着社会的发展，视频会议的应用越来越广泛，同时对其视频音频质量、数据共享、灵活性以及易用性、可靠性和易管理性的要求也越来越严格。

北京邮电大学计算机学院多媒体实验室在多媒体领域有多年的研究经验，拥有雄厚的技术实力，由本实验室自主开发的网络视频会议系统 NETVideo-CS 1.0，在国家经贸委技术创新网和北京邮电大学网络教学平台等应用中取得了较好的效果。本系统是在NETVideo-CS 1.0 在经历了 NETVideo-CS 2.0 和 SVC (SIP Video Conference) 后演进的新版本。其中 SVC 在 NETVideo-CS 2.0 基础上做了较大的改动，使用全新的 SIP 协议作为会议信令；而本次开发的基于 RTC 改进的视频会议系统使用微软新一代通信技术 RTC（RTC 是微软发布的新一代支持 SIP 协议的实时通信开发包），实现了系统在小规模情况下可省去 SIP 服务器，而随时可以引入 SIP 服务器构建大型视频会议服务的架构。本版视频会议系统的完成是

整个项目组共同努力的结果，其中，杨震博士负责了整体系统的技术把关，赵秀芳同学负责了视频会议服务器的开发，肖煜同学负责视频会议客户端的开发，我来实现流媒体相关构件的开发。

第六章 总结

随着流媒体技术的不断成熟和应用范围的不断扩大，流媒体传输技术、流媒体内容存储技术、流媒体内容发布管理技术、流媒体终端、流媒体服务、流媒体网络运营、数字内容版权等技术全面发展。作者以实验室视频会议项目为依托，先后参与了基于 SIP 的视频会议系统、基于 RTC 改进的 SIP 视频会议系统等流媒体项目，积累了较丰富的流媒体开发经验。

本文结合了软件工程中近年来发展比较迅速的构件化开发、领域工程和构件库理论，并在构件库理论的指导下设计实现了一个面向流媒体领域开发的流媒体构件库系统，并以此为辅助工具进行基于 RTC 改进的 SIP 视频会议系统的搭建与开发，从开发时间和所费开发人力的角度来讲取得了较好的效果。综合回顾，本文主要完成了以下工作：

（1）研究了流媒体软件系统的规律和开发模型，结合当前新兴的微软 DirectShow 技术进行了流媒体软件的构件化开发实践，实现了多个流媒体构件，这些流媒体构件在视频会议系统中得到应用；

（2）研究了构件库理论，并结合流媒体领域软件开发的特殊性，设计实现了流媒体构件库；

（3）在基于 RTC 改进的 SIP 视频会议系统中，通过初步使用流媒体构件库搭建了视频会议系统客户端和服务器端的媒体处理子系统，在搭建过程中，体现了利用构件库系统进行开发省时省力的效用。

然而，流媒体构件库仍然处在原型阶段，仍然有很多需要改进和优化的地方，特别是在构件库对构件质量评测方面，现在还是空白，在日后的工作中这也是需要努力的一大方向。

致谢

在本文即将结束之际，我要深深地感谢我的导师马华东教授。在北京邮电大学读研的三年时间里，马老师在学习上帮助我制定详细的读研计划，指导我进行理论和工程领域的研究和实践，工作上对我严格要求，同时生活上也出处处关心。他那深厚的学术造诣和严谨的治学态度使我受益匪浅；他那高尚的品格和为人师表的风范更对我的人生有着深远影响。

感谢北京邮电大学智能通信与多媒体北京市重点实验室的金仙力博士和杨震博士。在我读研期间，是他们分别带领我完成了 IMEC2.2.2 Project PC Client 和实验室视频会议第三、第四版，并在我工作遇到困难的时候，给我建议和帮助。

要感谢同项目组见良、张稷、冼牛、赵秀芳和肖煜等同学，大家在实验室工作学习期间彼此信任、相互鼓励，创造了一个优秀的集体氛围，我的论文中有他们的智慧。

感谢同一年级的吕可达、黄晓冬、王兴梁、牛永民、夏世凤、史刚、黄世兴等，我们的欢声笑语给紧张的求学生活平添了不少生气。

最后，对在百忙之中审阅我的论文的评委老师表示最诚挚的感谢！

参考文献

[1] [英] Ian Sommerville 著, 程成, 陈霞等译. 软件工程[M]. 原书第6版. 机械工业出版社, 2003.

[2] Meyer B, Mingins C. Component-Based development: From buzz to spark. IEEE Computer, 1999, 32(7):35-37.

[3] WANG GJ, UNGAR L, KLAWITTER D. Component assembly for OO distributed systems [J]. IEEE Computer, 1999, 32(7):71-78.

[4] DESMOND F, ALAN C. Objects, components and frameworks with UML[M]. UK: Addison Wesley, 199940-57.

[5] WHIT B, LE T. Models and languages for component description and reuse[J]. Software Engineering Nodes, 1995, 20(2):76—86.

[6] TRACZ W. Implementation working group summary[A]. In : JAMES B. Reuse in Practice Workshop Summary[C]. Alexandria:VA, 1990110-19.

[7] NEC Software Engineering Laboratory. NATO Standard for Management of a Reusable Software Component Library [J]. NATO Communications and Information Systems Agency, 1991, 2:32～43.

[8] 北京大学计算机科学系. 青鸟可复用软件开发指南, 青鸟工程项目组技术报告, 1997.

[9] [美]Fabio Arciniegas 著, 天宏工作室译. XML 开发指南第 1 版[M]. 清华大学出版社, 2003.

[10] NATO Communications and Information Systems Agency. NATO Standard for Management of a Reusable Software Component Library, 1991.

[11] 杨芙清梅宏等. 支持构件复用的青鸟III型系统概述. 计算机科学, 1999VOL. 26NO. 5, 50 – 55.

[12] Ruben Prieto-Diaz, Peter Freeman. Classifying software for reusability [J]. IEEE Software, 1987 1:6-16.

[13] Guttorm Sindre, Reidar Contradi, Even-Andr Karlsson. The REBOOT approach to software reuse [j].J System Software, 1995, 30:201-212

[14] [美]Charles F. Goldfarb, Paul Prescod 等著, 张晓军, 赵伟明等译. XML 手册(第四版) [M]. 电子工业出版社, 2003.

[15] Ma Huadong, Kang G Shin. A new scheduling scheme for multicast true VoD service. Lecture Notes in Computer Science (Proc. PCM2001), Springer, Oct.,

2001, Vol.2195, pp.708-715.

[16] Ji Zhang, Huadong Ma. An Approach to Developing Multimedia Services Based on SIP. ICCC2004, Beijing, Aug, 2004.

[17] Huadong MA, Ji ZHANG. A Unified Framework of Multimedia Service Based on SIP. IEEE ICCE2005, Las Vegas, Jan, 2005.

[18] Zhen YANG, Huadong MA and Ji ZHANG. A Dynamic Scalable Service Model for SIP-based Video Conference. The 9th International Conference on CSCW in Design, Coventry, UK, May 24-26, 2005.

[19] 张辉, 徐玮. 一个支持构件库动态演变的构件检索系统[J]. 计算机工程与应用, 2000; 36 (12):89-91

[20] 张海飞, 袁磊, 夏宽理. 构件库功能集模型[J]. 计算机工程, 2000; 26 (11):87-90

[21] 马亮、谢冰、杨芙清. 多构件库统一刻面检索机制. 电子学报, 2002, Dec, Vol.30, No.12A